

FflasFpack

Generated by Doxygen 1.9.1



<b>1 FFLAS-FFPACK Documentation.</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Goals . . . . .	1
1.3 Design . . . . .	1
1.4 Using FFLAS-FFPACK. . . . .	1
1.5 Contributing to fflas-ffpack, getting assistance. . . . .	1
<b>2 Configuring and Installing FFLAS-FFPACK</b>	<b>3</b>
<b>3 Copying and Licence</b>	<b>5</b>
<b>4 Tutorial</b>	<b>7</b>
<b>5 Architecture of the library.</b>	<b>9</b>
<b>6 Bug List</b>	<b>11</b>
<b>7 Bibliography</b>	<b>13</b>
<b>8 Todo List</b>	<b>15</b>
<b>9 Module Index</b>	<b>17</b>
9.1 Modules . . . . .	17
<b>10 Namespace Index</b>	<b>19</b>
10.1 Namespace List . . . . .	19
<b>11 Hierarchical Index</b>	<b>21</b>
11.1 Class Hierarchy . . . . .	21
<b>12 Data Structure Index</b>	<b>29</b>
12.1 Data Structures . . . . .	29
<b>13 File Index</b>	<b>37</b>
13.1 File List . . . . .	37
<b>14 Module Documentation</b>	<b>45</b>
14.1 CHECKER . . . . .	45
14.2 FFLAS . . . . .	45
14.3 Matrix Multiplication Algorithms . . . . .	45
14.3.1 Detailed Description . . . . .	45
14.4 SIMD wrapper . . . . .	46
14.5 FFLAS-FFPACK . . . . .	46
14.5.1 Detailed Description . . . . .	46
14.6 FFPACK . . . . .	46
14.7 FFLAS-FFPACK fields . . . . .	46
14.7.1 Detailed Description . . . . .	47

14.8 RNS	47
14.9 Interfaces	47
<b>15 Namespace Documentation</b>	<b>49</b>
15.1 FFLAS Namespace Reference	49
15.1.1 Typedef Documentation	74
15.1.1.1 Checker_fgemm	74
15.1.1.2 Checker_ftrsm	74
15.1.1.3 ForceCheck_fgemm	74
15.1.1.4 ForceCheck_ftrsm	74
15.1.1.5 ZOSparseMatrix	75
15.1.1.6 NotZOSparseMatrix	75
15.1.1.7 SimdSparseMatrix	75
15.1.1.8 NoSimdSparseMatrix	75
15.1.1.9 MKLSparseMatrixFormat	75
15.1.1.10 NotMKLSparseMatrixFormat	75
15.1.1.11 has_plus	75
15.1.1.12 has_minus	76
15.1.1.13 has_equal	76
15.1.1.14 has_plus_eq	76
15.1.1.15 has_minus_eq	76
15.1.1.16 has_mul	76
15.1.1.17 has_mul_eq	76
15.1.1.18 Timer	76
15.1.1.19 BaseTimer	77
15.1.1.20 UserTimer	77
15.1.1.21 SysTimer	77
15.1.2 Enumeration Type Documentation	77
15.1.2.1 FFLAS_ORDER	77
15.1.2.2 FFLAS_TRANSPOSE	77
15.1.2.3 FFLAS_UPLO	78
15.1.2.4 FFLAS_DIAG	78
15.1.2.5 FFLAS_SIDE	78
15.1.2.6 FFLAS_BASE	78
15.1.2.7 number_kind	79
15.1.2.8 SparseMatrix_t	79
15.1.2.9 FFLAS_FORMAT	79
15.1.3 Function Documentation	80
15.1.3.1 InfNorm()	80
15.1.3.2 min3()	80
15.1.3.3 max3()	80
15.1.3.4 min4()	80

15.1.3.5 max4()	81
15.1.3.6 fadd() [1/8]	81
15.1.3.7 faddin() [1/4]	81
15.1.3.8 fsub() [1/4]	81
15.1.3.9 fsubin() [1/3]	82
15.1.3.10 fadd() [2/8]	82
15.1.3.11 pfadd()	82
15.1.3.12 psub()	83
15.1.3.13 pfaddin()	83
15.1.3.14 psubin()	83
15.1.3.15 fadd() [3/8]	83
15.1.3.16 fsub() [2/4]	85
15.1.3.17 faddin() [2/4]	85
15.1.3.18 fsubin() [2/3]	86
15.1.3.19 fadd() [4/8]	86
15.1.3.20 fassign() [1/10]	87
15.1.3.21 fassign() [2/10]	87
15.1.3.22 fassign() [3/10]	87
15.1.3.23 fassign() [4/10]	88
15.1.3.24 fassign() [5/10]	88
15.1.3.25 fassign() [6/10]	88
15.1.3.26 fassign() [7/10]	88
15.1.3.27 fassign() [8/10]	89
15.1.3.28 faxpy() [1/6]	89
15.1.3.29 faxpy() [2/6]	90
15.1.3.30 faxpy() [3/6]	90
15.1.3.31 faxpy() [4/6]	90
15.1.3.32 fdot() [1/11]	91
15.1.3.33 fdot() [2/11]	91
15.1.3.34 fdot() [3/11]	91
15.1.3.35 fdot() [4/11]	91
15.1.3.36 fdot() [5/11]	92
15.1.3.37 fdot() [6/11]	92
15.1.3.38 fdot() [7/11]	92
15.1.3.39 fdot() [8/11]	92
15.1.3.40 fgemm() [1/23]	93
15.1.3.41 fgemm() [2/23]	93
15.1.3.42 fgemm() [3/23]	94
15.1.3.43 fgemm() [4/23]	94
15.1.3.44 fgemm() [5/23]	94
15.1.3.45 fgemm() [6/23]	95
15.1.3.46 fgemm() [7/23]	95

15.1.3.47 fgemm() [8/23]	96
15.1.3.48 fgemm() [9/23]	96
15.1.3.49 fgemm() [10/23]	96
15.1.3.50 fgemm() [11/23]	97
15.1.3.51 fgemm() [12/23]	97
15.1.3.52 fgemm() [13/23]	98
15.1.3.53 fgemm() [14/23]	98
15.1.3.54 fgemm() [15/23]	98
15.1.3.55 fgemm() [16/23]	99
15.1.3.56 fgemm() [17/23]	100
15.1.3.57 fgemm() [18/23]	100
15.1.3.58 fsquare() [1/6]	101
15.1.3.59 fsquare() [2/6]	101
15.1.3.60 fsquare() [3/6]	102
15.1.3.61 fsquare() [4/6]	102
15.1.3.62 fsquare() [5/6]	102
15.1.3.63 fgemv() [1/19]	102
15.1.3.64 fgemv() [2/19]	103
15.1.3.65 fgemv() [3/19]	103
15.1.3.66 fgemv() [4/19]	104
15.1.3.67 fgemv() [5/19]	104
15.1.3.68 fgemv() [6/19]	105
15.1.3.69 fgemv() [7/19]	105
15.1.3.70 fgemv() [8/19]	105
15.1.3.71 fgemv() [9/19]	106
15.1.3.72 fgemv() [10/19]	106
15.1.3.73 fgemv() [11/19]	107
15.1.3.74 fgemv() [12/19]	107
15.1.3.75 fgemv() [13/19]	107
15.1.3.76 fgemv() [14/19]	108
15.1.3.77 fgemv() [15/19]	108
15.1.3.78 fgemv() [16/19]	108
15.1.3.79 fger() [1/12]	109
15.1.3.80 fger() [2/12]	109
15.1.3.81 fger() [3/12]	110
15.1.3.82 fger() [4/12]	110
15.1.3.83 fger() [5/12]	110
15.1.3.84 fger() [6/12]	111
15.1.3.85 fger() [7/12]	111
15.1.3.86 fger() [8/12]	111
15.1.3.87 fger() [9/12]	112
15.1.3.88 fger() [10/12]	112

15.1.3.89 fger()	[11/12]	112
15.1.3.90 freduce()	[1/10]	113
15.1.3.91 freduce()	[2/10]	113
15.1.3.92 freduce_constoverride()	[1/2]	113
15.1.3.93 finit()	[1/8]	114
15.1.3.94 finit()	[2/8]	114
15.1.3.95 freduce()	[3/10]	114
15.1.3.96 pfreduce()		115
15.1.3.97 freduce()	[4/10]	115
15.1.3.98 freduce_constoverride()	[2/2]	115
15.1.3.99 finit()	[3/8]	116
15.1.3.100 finit()	[4/8]	116
15.1.3.101 freduce()	[5/10]	117
15.1.3.102 freduce()	[6/10]	117
15.1.3.103 freivalds()		117
15.1.3.104 fscal()	[1/10]	118
15.1.3.105 fscal()	[1/10]	118
15.1.3.106 fscal()	[2/10]	119
15.1.3.107 fscal()	[3/10]	119
15.1.3.108 fscal()	[2/10]	119
15.1.3.109 fscal()	[3/10]	120
15.1.3.110 fscal()	[4/10]	120
15.1.3.111 fscal()	[4/10]	120
15.1.3.112 fscal()	[5/10]	121
15.1.3.113 fscal()	[5/10]	121
15.1.3.114 fscal()	[6/10]	121
15.1.3.115 fscal()	[6/10]	122
15.1.3.116 fscal()	[7/10]	122
15.1.3.117 fscal()	[7/10]	122
15.1.3.118 fscal()	[8/10]	122
15.1.3.119 fscal()	[8/10]	123
15.1.3.120 fsyr2k()		123
15.1.3.121 fsyrk()	[1/5]	124
15.1.3.122 fsyrk()	[2/5]	124
15.1.3.123 fsyrk()	[3/5]	125
15.1.3.124 fsyrk()	[4/5]	126
15.1.3.125 fsyrk()	[5/5]	126
15.1.3.126 ftrmm()	[1/3]	127
15.1.3.127 ftrmm()	[2/3]	128
15.1.3.128 ftrsm()	[1/9]	129
15.1.3.129 ftrsm()	[2/9]	129
15.1.3.130 ftrsm()	[3/9]	129

15.1.3.131 ftrsm() [4/9]	130
15.1.3.132 ftrsm() [5/9]	130
15.1.3.133 cblas_imptrsm()	130
15.1.3.134 ftrsv() [1/2]	131
15.1.3.135 igemm_()	131
15.1.3.136 finit() [5/8]	132
15.1.3.137 fconvert() [1/3]	132
15.1.3.138 fnegin() [1/4]	133
15.1.3.139 fneg() [1/4]	133
15.1.3.140 fzero() [1/4]	133
15.1.3.141 frand() [1/2]	134
15.1.3.142 fiszero() [1/4]	134
15.1.3.143 fequal() [1/4]	135
15.1.3.144 faxpby() [1/2]	135
15.1.3.145 fdot() [9/11]	136
15.1.3.146 fswap() [1/2]	136
15.1.3.147 fzero() [2/4]	136
15.1.3.148 frand() [2/2]	137
15.1.3.149 fequal() [2/4]	137
15.1.3.150 fiszero() [2/4]	138
15.1.3.151 fidentity() [1/4]	138
15.1.3.152 fidentity() [2/4]	139
15.1.3.153 finit() [6/8]	139
15.1.3.154 fconvert() [2/3]	139
15.1.3.155 fnegin() [2/4]	140
15.1.3.156 fneg() [2/4]	140
15.1.3.157 faxpby() [2/2]	141
15.1.3.158 fmove() [1/2]	141
15.1.3.159 bitsize()	142
15.1.3.160 bitsize< Givaro::ZRing< Givaro::Integer > >()	142
15.1.3.161 ftrmv()	143
15.1.3.162 ftrsm() [6/9]	143
15.1.3.163 pfgemm() [1/7]	144
15.1.3.164 pfgemm_1D_rec()	144
15.1.3.165 pfgemm_2D_rec()	145
15.1.3.166 pfgemm_3D_rec()	145
15.1.3.167 pfgemm_3D_rec2()	146
15.1.3.168 fgemm() [19/23]	146
15.1.3.169 ftrsm() [7/9]	146
15.1.3.170 ftrsm() [8/9]	147
15.1.3.171 sparse_delete() [1/12]	147
15.1.3.172 sparse_delete() [2/12]	147



15.1.3.173 <code>sparse_init()</code> [1/16]	147
15.1.3.174 <code>sparse_init()</code> [2/16]	148
15.1.3.175 <code>sparse_delete()</code> [3/12]	148
15.1.3.176 <code>sparse_delete()</code> [4/12]	148
15.1.3.177 <code>sparse_print()</code> [1/3]	148
15.1.3.178 <code>sparse_init()</code> [3/16]	148
15.1.3.179 <code>sparse_init()</code> [4/16]	149
15.1.3.180 <code>sparse_init()</code> [5/16]	149
15.1.3.181 <code>sparse_init()</code> [6/16]	149
15.1.3.182 <code>sparse_init()</code> [7/16]	149
15.1.3.183 <code>sparse_init()</code> [8/16]	150
15.1.3.184 <code>sparse_delete()</code> [5/12]	150
15.1.3.185 <code>sparse_init()</code> [9/16]	150
15.1.3.186 <code>sparse_delete()</code> [6/12]	150
15.1.3.187 <code>sparse_delete()</code> [7/12]	150
15.1.3.188 <code>sparse_init()</code> [10/16]	151
15.1.3.189 <code>sparse_init()</code> [11/16]	151
15.1.3.190 <code>sparse_delete()</code> [8/12]	151
15.1.3.191 <code>sparse_delete()</code> [9/12]	151
15.1.3.192 <code>sparse_print()</code> [2/3]	151
15.1.3.193 <code>sparse_init()</code> [12/16]	152
15.1.3.194 <code>sparse_init()</code> [13/16]	152
15.1.3.195 <code>sparse_delete()</code> [10/12]	152
15.1.3.196 <code>sparse_init()</code> [14/16]	152
15.1.3.197 <code>operator&lt;&lt;()</code>	153
15.1.3.198 <code>readSmsFormat()</code>	153
15.1.3.199 <code>readSprFormat()</code>	153
15.1.3.200 <code>getDataType()</code> [1/4]	153
15.1.3.201 <code>getDataType()</code> [2/4]	153
15.1.3.202 <code>getDataType()</code> [3/4]	154
15.1.3.203 <code>getDataType()</code> [4/4]	154
15.1.3.204 <code>readMachineType()</code>	154
15.1.3.205 <code>readDnsFormat()</code>	154
15.1.3.206 <code>writeDnsFormat()</code>	154
15.1.3.207 <code>fspmv()</code> [1/2]	155
15.1.3.208 <code>sparse_delete()</code> [11/12]	155
15.1.3.209 <code>sparse_delete()</code> [12/12]	155
15.1.3.210 <code>sparse_print()</code> [3/3]	155
15.1.3.211 <code>sparse_init()</code> [15/16]	155
15.1.3.212 <code>sparse_init()</code> [16/16]	156
15.1.3.213 <code>computeDeviation()</code>	156
15.1.3.214 <code>getStat()</code>	156

15.1.3.215 fspmv()	[ 2/2 ]	156
15.1.3.216 fspmm()		157
15.1.3.217 fflas_delete()	[ 1/3 ]	157
15.1.3.218 fflas_delete()	[ 2/3 ]	157
15.1.3.219 fflas_new()	[ 1/7 ]	157
15.1.3.220 fflas_new()	[ 2/7 ]	157
15.1.3.221 finit_rns()	[ 1/2 ]	158
15.1.3.222 finit_trans_rns()		158
15.1.3.223 fconvert_rns()	[ 1/2 ]	158
15.1.3.224 fconvert_trans_rns()		158
15.1.3.225 fflas_new()	[ 3/7 ]	159
15.1.3.226 fflas_new()	[ 4/7 ]	159
15.1.3.227 finit_rns()	[ 2/2 ]	159
15.1.3.228 fconvert_rns()	[ 2/2 ]	159
15.1.3.229 freduce()	[ 7/10 ]	159
15.1.3.230 freduce()	[ 8/10 ]	160
15.1.3.231 finit()	[ 7/8 ]	160
15.1.3.232 fconvert()	[ 3/3 ]	161
15.1.3.233 fnegin()	[ 3/4 ]	161
15.1.3.234 fneg()	[ 3/4 ]	162
15.1.3.235 fzero()	[ 3/4 ]	162
15.1.3.236 fiszero()	[ 3/4 ]	163
15.1.3.237 fequal()	[ 3/4 ]	163
15.1.3.238 fassign()	[ 9/10 ]	164
15.1.3.239 fscalin()	[ 9/10 ]	164
15.1.3.240 fscal()	[ 9/10 ]	165
15.1.3.241 faxpy()	[ 5/6 ]	165
15.1.3.242 fdot()	[ 10/11 ]	166
15.1.3.243 fswap()	[ 2/2 ]	166
15.1.3.244 fadd()	[ 5/8 ]	167
15.1.3.245 fsub()	[ 3/4 ]	167
15.1.3.246 faddin()	[ 3/4 ]	168
15.1.3.247 fadd()	[ 6/8 ]	168
15.1.3.248 fassign()	[ 10/10 ]	168
15.1.3.249 fzero()	[ 4/4 ]	169
15.1.3.250 fequal()	[ 4/4 ]	169
15.1.3.251 fiszero()	[ 4/4 ]	170
15.1.3.252 fidentity()	[ 3/4 ]	170
15.1.3.253 fidentity()	[ 4/4 ]	170
15.1.3.254 freduce()	[ 9/10 ]	170
15.1.3.255 freduce()	[ 10/10 ]	171
15.1.3.256 finit()	[ 8/8 ]	171

15.1.3.257 fnegin()	[ 4/4 ]	172
15.1.3.258 fneg()	[ 4/4 ]	172
15.1.3.259 fscaln()	[10/10]	173
15.1.3.260 fscal()	[10/10]	173
15.1.3.261 faxpy()	[ 6/6 ]	174
15.1.3.262 fmove()	[ 2/2 ]	174
15.1.3.263 fadd()	[ 7/8 ]	175
15.1.3.264 fsub()	[ 4/4 ]	175
15.1.3.265 fsubin()	[ 3/3 ]	176
15.1.3.266 fadd()	[ 8/8 ]	176
15.1.3.267 faddin()	[ 4/4 ]	177
15.1.3.268 fgemv()	[17/19]	177
15.1.3.269 fger()	[12/12]	179
15.1.3.270 ftrsv()	[ 2/2 ]	180
15.1.3.271 ftrsm()	[ 9/9 ]	180
15.1.3.272 ftrmm()	[ 3/3 ]	181
15.1.3.273 fgemm()	[20/23]	182
15.1.3.274 fgemm()	[21/23]	182
15.1.3.275 fgemm()	[22/23]	183
15.1.3.276 fgemm()	[23/23]	183
15.1.3.277 fsquare()	[ 6/6 ]	184
15.1.3.278 BlockCuts()	[1/2]	184
15.1.3.279 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()		185
15.1.3.280 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()		185
15.1.3.281 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()		185
15.1.3.282 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()		185
15.1.3.283 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()		185
15.1.3.284 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()		186
15.1.3.285 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()		186
15.1.3.286 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()		186
15.1.3.287 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()		186
15.1.3.288 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()		186
15.1.3.289 BlockCuts()	[ 2/2 ]	187
15.1.3.290 pfzero()		187
15.1.3.291 pfrand()		187
15.1.3.292 fdot()	[11/11]	187
15.1.3.293 pfgemm()	[ 2/7 ]	188
15.1.3.294 pfgemm()	[ 3/7 ]	188
15.1.3.295 pfgemm()	[ 4/7 ]	188
15.1.3.296 pfgemm()	[ 5/7 ]	189
15.1.3.297 pfgemm()	[ 6/7 ]	189
15.1.3.298 pfgemm()	[ 7/7 ]	190

15.1.3.299 fgemv() [18/19]	190
15.1.3.300 fgemv() [19/19]	190
15.1.3.301 parseArguments()	191
15.1.3.302 writeCommandString()	191
15.1.3.303 WriteMatrix() [1/2]	191
15.1.3.304 preamble()	192
15.1.3.305 ReadMatrix() [1/2]	192
15.1.3.306 ReadMatrix() [2/2]	192
15.1.3.307 WriteMatrix() [2/2]	193
15.1.3.308 WritePermutation()	193
15.1.3.309 alignable()	193
15.1.3.310 alignable< Givaro::Integer * >()	194
15.1.3.311 fflas_new() [5/7]	194
15.1.3.312 fflas_new() [6/7]	194
15.1.3.313 fflas_new() [7/7]	194
15.1.3.314 fflas_delete() [3/3]	194
15.1.3.315 prefetch()	194
15.1.3.316 getTLBSize()	195
15.1.3.317 queryCacheSizes()	195
15.1.3.318 queryL1CacheSize()	195
15.1.3.319 queryTopLevelCacheSize()	195
15.1.3.320 getSeed()	195
15.2 FFLAS::BLAS3 Namespace Reference	196
15.2.1 Function Documentation	197
15.2.1.1 Bini()	197
15.2.1.2 WinoPar()	198
15.2.1.3 Winograd()	198
15.2.1.4 WinogradAcc_3_23()	198
15.2.1.5 WinogradAcc_3_21()	199
15.2.1.6 WinogradAcc_2_24()	199
15.2.1.7 WinogradAcc_2_27()	200
15.2.1.8 WinogradAcc_LR()	200
15.2.1.9 WinogradAcc_R_S()	200
15.2.1.10 WinogradAcc_L_S()	201
15.2.1.11 Winograd_LR_S()	201
15.2.1.12 Winograd_L_S()	202
15.2.1.13 Winograd_R_S()	202
15.3 FFLAS::csr_hyb_details Namespace Reference	202
15.4 FFLAS::CuttingStrategy Namespace Reference	202
15.4.1 Typedef Documentation	203
15.4.1.1 RNSModulus	203
15.5 FFLAS::details Namespace Reference	203

15.5.1 Function Documentation	204
15.5.1.1 fadd() [1/5]	204
15.5.1.2 fadd() [2/5]	205
15.5.1.3 fadd() [3/5]	205
15.5.1.4 fadd() [4/5]	205
15.5.1.5 fadd() [5/5]	206
15.5.1.6 freduce() [1/4]	206
15.5.1.7 freduce() [2/4]	206
15.5.1.8 freduce() [3/4]	206
15.5.1.9 freduce() [4/4]	207
15.5.1.10 fscaln() [1/2]	207
15.5.1.11 fscal() [1/2]	207
15.5.1.12 fscaln() [2/2]	207
15.5.1.13 fscal() [2/2]	208
15.5.1.14 igebb44()	208
15.5.1.15 igebb24()	208
15.5.1.16 igebb14()	208
15.5.1.17 igebb41()	209
15.5.1.18 igebb21()	209
15.5.1.19 igebb11()	209
15.5.1.20 igebp()	210
15.5.1.21 pack_lhs()	210
15.5.1.22 pack_rhs()	210
15.5.1.23 gebp()	211
15.5.1.24 BlockingFactor()	211
15.6 FFLAS::details_spmv Namespace Reference	211
15.7 FFLAS::ElementCategories Namespace Reference	211
15.8 FFLAS::FieldCategories Namespace Reference	212
15.8.1 Detailed Description	212
15.9 FFLAS::MMHelperAlgo Namespace Reference	212
15.10 FFLAS::ModeCategories Namespace Reference	212
15.10.1 Detailed Description	213
15.11 FFLAS::ParSeqHelper Namespace Reference	213
15.11.1 Detailed Description	213
15.12 FFLAS::Protected Namespace Reference	213
15.12.1 Function Documentation	216
15.12.1.1 computeFactorClassic() [1/3]	216
15.12.1.2 computeFactorClassic() [2/3]	216
15.12.1.3 computeFactorClassic() [3/3]	217
15.12.1.4 DotProdBoundClassic()	217
15.12.1.5 TRSMBound() [1/3]	217
15.12.1.6 TRSMBound() [2/3]	217

15.12.1.7 TRSMBound() [3/3]	217
15.12.1.8 WinogradThreshold() [1/4]	218
15.12.1.9 WinogradThreshold() [2/4]	218
15.12.1.10 WinogradThreshold() [3/4]	218
15.12.1.11 WinogradThreshold() [4/4]	218
15.12.1.12 WinogradSteps()	218
15.12.1.13 DynamicPeeling()	219
15.12.1.14 DynamicPeeling2()	219
15.12.1.15 WinogradCalc()	220
15.12.1.16 fgemm_convert()	220
15.12.1.17 NeedPreAddReduction() [1/2]	220
15.12.1.18 NeedPreAddReduction() [2/2]	221
15.12.1.19 NeedPreSubReduction() [1/2]	221
15.12.1.20 NeedPreSubReduction() [2/2]	221
15.12.1.21 NeedDoublePreAddReduction() [1/2]	221
15.12.1.22 NeedDoublePreAddReduction() [2/2]	222
15.12.1.23 ScalAndReduce() [1/2]	222
15.12.1.24 ScalAndReduce() [2/2]	222
15.12.1.25 fsquareCommon()	222
15.12.1.26 fgemv_convert()	223
15.12.1.27 fger_convert()	223
15.12.1.28 min_types() [1/7]	223
15.12.1.29 min_types() [2/7]	223
15.12.1.30 min_types() [3/7]	223
15.12.1.31 min_types() [4/7]	224
15.12.1.32 min_types() [5/7]	224
15.12.1.33 min_types() [6/7]	224
15.12.1.34 min_types() [7/7]	224
15.12.1.35 unfit() [1/4]	224
15.12.1.36 unfit() [2/4]	224
15.12.1.37 unfit() [3/4]	224
15.12.1.38 unfit() [4/4]	225
15.12.1.39 igemm_colmajor() [1/2]	225
15.12.1.40 igemm_colmajor() [2/2]	225
15.12.1.41 igemm()	225
15.12.1.42 MatF2MatD_Triangular()	226
15.12.1.43 MatF2MatFI_Triangular()	226
15.13 FFLAS::sell_details Namespace Reference	226
15.14 FFLAS::sparse_details Namespace Reference	227
15.14.1 Function Documentation	229
15.14.1.1 init_y() [1/2]	230
15.14.1.2 init_y() [2/2]	230

15.14.1.3 fspmv_dispatch() [1/2]	230
15.14.1.4 fspmv_dispatch() [2/2]	230
15.14.1.5 fspmv() [1/12]	231
15.14.1.6 fspmv() [2/12]	231
15.14.1.7 fspmv() [3/12]	231
15.14.1.8 fspmv() [4/12]	231
15.14.1.9 fspmv() [5/12]	232
15.14.1.10 fspmv() [6/12]	232
15.14.1.11 fspmv() [7/12]	232
15.14.1.12 fspmv() [8/12]	232
15.14.1.13 fspmv() [9/12]	233
15.14.1.14 fspmm_dispatch() [1/2]	233
15.14.1.15 fspmm_dispatch() [2/2]	233
15.14.1.16 fspmm() [1/9]	234
15.14.1.17 fspmm() [2/9]	234
15.14.1.18 fspmm() [3/9]	234
15.14.1.19 fspmm() [4/9]	235
15.14.1.20 fspmm() [5/9]	235
15.14.1.21 fspmm() [6/9]	235
15.14.1.22 fspmm() [7/9]	236
15.14.1.23 fspmm() [8/9]	236
15.14.1.24 fspmm() [9/9]	236
15.14.1.25 pfspmm_dispatch() [1/2]	237
15.14.1.26 pfspmm_dispatch() [2/2]	237
15.14.1.27 pfspmm() [1/9]	237
15.14.1.28 pfspmm() [2/9]	238
15.14.1.29 pfspmm() [3/9]	238
15.14.1.30 pfspmm() [4/9]	238
15.14.1.31 pfspmm() [5/9]	239
15.14.1.32 pfspmm() [6/9]	239
15.14.1.33 pfspmm() [7/9]	239
15.14.1.34 pfspmm() [8/9]	240
15.14.1.35 pfspmm() [9/9]	240
15.14.1.36 pfspmv() [1/6]	240
15.14.1.37 pfspmv() [2/6]	240
15.14.1.38 pfspmv() [3/6]	241
15.14.1.39 pfspmv() [4/6]	241
15.14.1.40 pfspmv() [5/6]	241
15.14.1.41 pfspmv() [6/6]	241
15.14.1.42 fspmv() [10/12]	242
15.14.1.43 fspmv() [11/12]	242
15.14.1.44 fspmv() [12/12]	242

15.15 FFLAS::sparse_details_impl Namespace Reference . . . . .	242
15.15.1 Function Documentation . . . . .	251
15.15.1.1 fspmm() [1/15] . . . . .	251
15.15.1.2 fspmm() [2/15] . . . . .	251
15.15.1.3 fspmm() [3/15] . . . . .	251
15.15.1.4 fspmm_simd_aligned() [1/2] . . . . .	252
15.15.1.5 fspmm_simd_unaligned() [1/2] . . . . .	252
15.15.1.6 fspmm_one() [1/4] . . . . .	252
15.15.1.7 fspmm_mone() [1/4] . . . . .	252
15.15.1.8 fspmm_one_simd_aligned() [1/3] . . . . .	253
15.15.1.9 fspmm_one_simd_unaligned() [1/3] . . . . .	253
15.15.1.10 fspmm_mone_simd_aligned() [1/3] . . . . .	253
15.15.1.11 fspmm_mone_simd_unaligned() [1/3] . . . . .	253
15.15.1.12 fspmv() [1/21] . . . . .	254
15.15.1.13 fspmv() [2/21] . . . . .	254
15.15.1.14 fspmv() [3/21] . . . . .	254
15.15.1.15 fspmv_one() [1/10] . . . . .	254
15.15.1.16 fspmv_mone() [1/10] . . . . .	254
15.15.1.17 fspmv_one() [2/10] . . . . .	255
15.15.1.18 fspmv_mone() [2/10] . . . . .	255
15.15.1.19 pfspmm() [1/18] . . . . .	255
15.15.1.20 pfspmm() [2/18] . . . . .	255
15.15.1.21 pfspmm() [3/18] . . . . .	256
15.15.1.22 pfspmm_one() [1/2] . . . . .	256
15.15.1.23 pfspmm_mone() [1/2] . . . . .	256
15.15.1.24 pfspmm_one() [2/2] . . . . .	256
15.15.1.25 pfspmm_mone() [2/2] . . . . .	257
15.15.1.26 pfspmv() [1/18] . . . . .	257
15.15.1.27 pfspmv_task() . . . . .	257
15.15.1.28 pfspmv() [2/18] . . . . .	257
15.15.1.29 pfspmv() [3/18] . . . . .	258
15.15.1.30 pfspmv_one() [1/8] . . . . .	258
15.15.1.31 pfspmv_mone() [1/8] . . . . .	258
15.15.1.32 pfspmv_one() [2/8] . . . . .	258
15.15.1.33 pfspmv_mone() [2/8] . . . . .	258
15.15.1.34 fspmm() [4/15] . . . . .	259
15.15.1.35 fspmm() [5/15] . . . . .	259
15.15.1.36 fspmm_simd_aligned() [2/2] . . . . .	259
15.15.1.37 fspmm_simd_unaligned() [2/2] . . . . .	259
15.15.1.38 fspmm() [6/15] . . . . .	260
15.15.1.39 fspmm_one() [2/4] . . . . .	260
15.15.1.40 fspmm_mone() [2/4] . . . . .	260



15.15.1.41 fspmm_one_simd_aligned() [2/3]	260
15.15.1.42 fspmm_one_simd_unaligned() [2/3]	261
15.15.1.43 fspmm_mone_simd_aligned() [2/3]	261
15.15.1.44 fspmm_mone_simd_unaligned() [2/3]	261
15.15.1.45 fspmv() [4/21]	261
15.15.1.46 fspmv() [5/21]	262
15.15.1.47 fspmv() [6/21]	262
15.15.1.48 fspmv_one() [3/10]	262
15.15.1.49 fspmv_mone() [3/10]	262
15.15.1.50 fspmv_one() [4/10]	262
15.15.1.51 fspmv_mone() [4/10]	263
15.15.1.52 pfspmm() [4/18]	263
15.15.1.53 pfspmm() [5/18]	263
15.15.1.54 pfspmm() [6/18]	263
15.15.1.55 pfspmm() [7/18]	264
15.15.1.56 pfspmm() [8/18]	264
15.15.1.57 pfspmm() [9/18]	264
15.15.1.58 pfspmv() [4/18]	264
15.15.1.59 pfspmv() [5/18]	265
15.15.1.60 pfspmv() [6/18]	265
15.15.1.61 fspmm() [7/15]	265
15.15.1.62 fspmm() [8/15]	265
15.15.1.63 fspmm() [9/15]	266
15.15.1.64 fspmv() [7/21]	266
15.15.1.65 fspmv() [8/21]	266
15.15.1.66 fspmv() [9/21]	266
15.15.1.67 pfspmm() [10/18]	267
15.15.1.68 pfspmm() [11/18]	267
15.15.1.69 pfspmm() [12/18]	267
15.15.1.70 pfspmm() [13/18]	267
15.15.1.71 pfspmm() [14/18]	268
15.15.1.72 pfspmm() [15/18]	268
15.15.1.73 pfspmm_zo() [1/2]	268
15.15.1.74 pfspmm_zo() [2/2]	268
15.15.1.75 pfspmv() [7/18]	269
15.15.1.76 pfspmv() [8/18]	269
15.15.1.77 pfspmv() [9/18]	269
15.15.1.78 pfspmv_one() [3/8]	269
15.15.1.79 pfspmv_mone() [3/8]	269
15.15.1.80 pfspmv_one() [4/8]	270
15.15.1.81 pfspmv_mone() [4/8]	270
15.15.1.82 fspmm() [10/15]	270

15.15.1.83 fspmm()	[11/15]	270
15.15.1.84 fspmm()	[12/15]	271
15.15.1.85 fspmm_mone()	[3/4]	271
15.15.1.86 fspmm_one()	[3/4]	271
15.15.1.87 fspmm_mone()	[4/4]	271
15.15.1.88 fspmm_one()	[4/4]	272
15.15.1.89 fspmm_one_simd_aligned()	[3/3]	272
15.15.1.90 fspmm_one_simd_unaligned()	[3/3]	272
15.15.1.91 fspmm_mone_simd_aligned()	[3/3]	272
15.15.1.92 fspmm_mone_simd_unaligned()	[3/3]	273
15.15.1.93 fspmv()	[10/21]	273
15.15.1.94 fspmv()	[11/21]	273
15.15.1.95 fspmv()	[12/21]	273
15.15.1.96 fspmv_one()	[5/10]	274
15.15.1.97 fspmv_mone()	[5/10]	274
15.15.1.98 fspmv_one()	[6/10]	274
15.15.1.99 fspmv_mone()	[6/10]	274
15.15.1.100 pfspmv()	[10/18]	274
15.15.1.101 pfspmv()	[11/18]	275
15.15.1.102 pfspmv()	[12/18]	275
15.15.1.103 pfspmv_one()	[5/8]	275
15.15.1.104 pfspmv_mone()	[5/8]	275
15.15.1.105 pfspmv_one()	[6/8]	275
15.15.1.106 pfspmv_mone()	[6/8]	276
15.15.1.107 fspmv()	[13/21]	276
15.15.1.108 fspmv_simd()	[1/4]	276
15.15.1.109 fspmv()	[14/21]	276
15.15.1.110 fspmv_simd()	[2/4]	276
15.15.1.111 fspmv()	[15/21]	277
15.15.1.112 fspmv_one()	[7/10]	277
15.15.1.113 fspmv_mone()	[7/10]	277
15.15.1.114 fspmv_one()	[8/10]	277
15.15.1.115 fspmv_mone()	[8/10]	277
15.15.1.116 fspmv_one_simd()	[1/2]	278
15.15.1.117 fspmv_mone_simd()	[1/2]	278
15.15.1.118 pfspmmm()	[16/18]	278
15.15.1.119 pfspmmm()	[17/18]	278
15.15.1.120 pfspmmm()	[18/18]	279
15.15.1.121 pfspmv()	[13/18]	279
15.15.1.122 pfspmv()	[14/18]	279
15.15.1.123 pfspmv()	[15/18]	279
15.15.1.124 fspmm()	[13/15]	280

15.15.1.125 fspmm()	[14/15]	280
15.15.1.126 fspmm()	[15/15]	280
15.15.1.127 fspmv()	[16/21]	280
15.15.1.128 fspmv()	[17/21]	281
15.15.1.129 fspmv()	[18/21]	281
15.15.1.130 pfspmv()	[16/18]	281
15.15.1.131 pfspmv()	[17/18]	281
15.15.1.132 pfspmv()	[18/18]	281
15.15.1.133 pfspmv_one()	[7/8]	282
15.15.1.134 pfspmv_mone()	[7/8]	282
15.15.1.135 pfspmv_one()	[8/8]	282
15.15.1.136 pfspmv_mone()	[8/8]	282
15.15.1.137 fspmv()	[19/21]	282
15.15.1.138 fspmv_simd()	[3/4]	283
15.15.1.139 fspmv()	[20/21]	283
15.15.1.140 fspmv_simd()	[4/4]	283
15.15.1.141 fspmv()	[21/21]	283
15.15.1.142 fspmv_one()	[9/10]	283
15.15.1.143 fspmv_mone()	[9/10]	284
15.15.1.144 fspmv_one_simd()	[2/2]	284
15.15.1.145 fspmv_mone_simd()	[2/2]	284
15.15.1.146 fspmv_one()	[10/10]	284
15.15.1.147 fspmv_mone()	[10/10]	284
15.16 FFLAS::StrategyParameter Namespace Reference		285
15.17 FFLAS::StructureHelper Namespace Reference		285
15.17.1 Detailed Description		285
15.18 FFLAS::vectorised Namespace Reference		285
15.18.1 Function Documentation		286
15.18.1.1 VEC_ADD()		287
15.18.1.2 addp()		287
15.18.1.3 VEC_SUB()		287
15.18.1.4 subp()		287
15.18.1.5 add()		288
15.18.1.6 sub()		288
15.18.1.7 reduce()	[1/9]	288
15.18.1.8 reduce()	[2/9]	288
15.18.1.9 reduce()	[3/9]	288
15.18.1.10 reduce()	[4/9]	288
15.18.1.11 reduce()	[5/9]	289
15.18.1.12 reduce()	[6/9]	289
15.18.1.13 reduce()	[7/9]	289
15.18.1.14 reduce()	[8/9]	289

15.18.1.15 reduce() [9/9]	289
15.18.1.16 modp() [1/2]	290
15.18.1.17 modp() [2/2]	290
15.18.1.18 scalp() [1/2]	290
15.18.1.19 scalp() [2/2]	290
15.19 FFLAS::vectorised::unswitch Namespace Reference	291
15.19.1 Function Documentation	291
15.19.1.1 modp() [1/2]	291
15.19.1.2 modp() [2/2]	291
15.19.1.3 scalp() [1/2]	292
15.19.1.4 scalp() [2/2]	292
15.20 FFPACK Namespace Reference	292
15.20.1 Detailed Description	307
15.20.2 Typedef Documentation	307
15.20.2.1 Checker_PLUQ	307
15.20.2.2 Checker_Det	307
15.20.2.3 Checker_invert	308
15.20.2.4 Checker_charpoly	308
15.20.2.5 ForceCheck_PLUQ	308
15.20.2.6 ForceCheck_Det	308
15.20.2.7 ForceCheck_invert	308
15.20.2.8 ForceCheck_charpoly	308
15.20.3 Function Documentation	308
15.20.3.1 LAPACKPerm2MathPerm()	308
15.20.3.2 MathPerm2LAPACKPerm()	309
15.20.3.3 applyP() [1/4]	309
15.20.3.4 applyP() [2/4]	310
15.20.3.5 applyP() [3/4]	310
15.20.3.6 MonotonicApplyP()	310
15.20.3.7 fgetrs() [1/4]	311
15.20.3.8 fgetrs() [2/4]	312
15.20.3.9 fgesv() [1/4]	313
15.20.3.10 fgesv() [2/4]	313
15.20.3.11 ftrtri() [1/2]	314
15.20.3.12 trinv_left() [1/2]	314
15.20.3.13 ftrtrm() [1/2]	315
15.20.3.14 ftrstr()	315
15.20.3.15 ftrssyr2k()	316
15.20.3.16 fsytrf() [1/3]	317
15.20.3.17 fsytrf() [2/3]	317
15.20.3.18 fsytrf() [3/3]	317
15.20.3.19 fsytrf_nonunit() [1/3]	318

15.20.3.20 PLUQ()	[1/6]	318
15.20.3.21 pPLUQ()		319
15.20.3.22 PLUQ()	[2/6]	319
15.20.3.23 PLUQ()	[3/6]	320
15.20.3.24 LUdivine()	[1/4]	320
15.20.3.25 ColumnEchelonForm()	[1/3]	321
15.20.3.26 pColumnEchelonForm()		321
15.20.3.27 ColumnEchelonForm()	[2/3]	322
15.20.3.28 RowEchelonForm()	[1/3]	322
15.20.3.29 pRowEchelonForm()		323
15.20.3.30 RowEchelonForm()	[2/3]	323
15.20.3.31 ReducedColumnEchelonForm()	[1/3]	323
15.20.3.32 pReducedColumnEchelonForm()		324
15.20.3.33 ReducedColumnEchelonForm()	[2/3]	324
15.20.3.34 ReducedRowEchelonForm()	[1/3]	325
15.20.3.35 pReducedRowEchelonForm()		325
15.20.3.36 ReducedRowEchelonForm()	[2/3]	326
15.20.3.37 Invert()	[1/4]	326
15.20.3.38 Invert()	[2/4]	326
15.20.3.39 Invert2()	[1/2]	327
15.20.3.40 CharPoly()	[1/8]	328
15.20.3.41 CharPoly()	[2/8]	328
15.20.3.42 CharPoly()	[3/8]	329
15.20.3.43 MinPoly()	[1/4]	329
15.20.3.44 MinPoly()	[2/4]	330
15.20.3.45 MatVecMinPoly()	[1/2]	330
15.20.3.46 Rank()	[1/3]	331
15.20.3.47 pRank()		331
15.20.3.48 Rank()	[2/3]	332
15.20.3.49 IsSingular()	[1/2]	332
15.20.3.50 Det()	[1/6]	332
15.20.3.51 pDet()		333
15.20.3.52 Det()	[2/6]	333
15.20.3.53 Solve()	[1/3]	334
15.20.3.54 Solve()	[2/3]	334
15.20.3.55 pSolve()		334
15.20.3.56 RandomNullSpaceVector()	[1/3]	335
15.20.3.57 NullSpaceBasis()	[1/2]	335
15.20.3.58 RowRankProfile()	[1/3]	336
15.20.3.59 pRowRankProfile()		336
15.20.3.60 RowRankProfile()	[2/3]	337
15.20.3.61 ColumnRankProfile()	[1/3]	337

15.20.3.62 pColumnRankProfile()	338
15.20.3.63 ColumnRankProfile() [2/3]	338
15.20.3.64 RankProfileFromLU()	338
15.20.3.65 LeadingSubmatrixRankProfiles()	339
15.20.3.66 RowRankProfileSubmatrixIndices() [1/2]	339
15.20.3.67 ColRankProfileSubmatrixIndices() [1/2]	340
15.20.3.68 RowRankProfileSubmatrix() [1/2]	341
15.20.3.69 ColRankProfileSubmatrix() [1/2]	341
15.20.3.70 getTriangular() [1/2]	342
15.20.3.71 getTriangular() [2/2]	343
15.20.3.72 getEchelonForm() [1/2]	343
15.20.3.73 getEchelonForm() [2/2]	344
15.20.3.74 getEchelonTransform()	345
15.20.3.75 getReducedEchelonForm() [1/2]	346
15.20.3.76 getReducedEchelonForm() [2/2]	346
15.20.3.77 getReducedEchelonTransform()	347
15.20.3.78 PLUQtoEchelonPermutation()	348
15.20.3.79 LQUPtoInverseOfFullRankMinor() [1/2]	348
15.20.3.80 RandomNullSpaceVector() [2/3]	349
15.20.3.81 solveLB() [1/2]	349
15.20.3.82 solveLB2() [1/2]	350
15.20.3.83 Danilevski()	350
15.20.3.84 buildMatrix()	350
15.20.3.85 CharPoly() [4/8]	350
15.20.3.86 CharPoly() [5/8]	351
15.20.3.87 Det() [3/6]	351
15.20.3.88 Det() [4/6]	351
15.20.3.89 fsytrf_BC_Crout()	351
15.20.3.90 fsytrf_BC_RL()	352
15.20.3.91 fsytrf_UP_RPM_BC_RL()	352
15.20.3.92 fsytrf_LOW_RPM_BC_Crout()	352
15.20.3.93 fsytrf_UP_RPM_BC_Crout()	352
15.20.3.94 fsytrf_UP_RPM()	353
15.20.3.95 fsytrf_nonunit() [2/3]	353
15.20.3.96 fsytrf_nonunit() [3/3]	353
15.20.3.97 fsytrf_RPM()	354
15.20.3.98 getTridiagonal()	354
15.20.3.99 LUdivine_gauss() [1/2]	354
15.20.3.100 LUdivine_small() [1/2]	354
15.20.3.101 LUdivine() [2/4]	355
15.20.3.102 LUdivine() [3/4]	355
15.20.3.103 MonotonicCompress()	355

15.20.3.104 MonotonicCompressMorePivots()	356
15.20.3.105 MonotonicCompressCycles()	356
15.20.3.106 MonotonicExpand()	356
15.20.3.107 applyP_block()	357
15.20.3.108 doApplyS()	357
15.20.3.109 MatrixApplyS() [1/3]	357
15.20.3.110 MatrixApplyS() [2/3]	358
15.20.3.111 MatrixApplyS() [3/3]	358
15.20.3.112 PermApplyS()	358
15.20.3.113 doApplyT()	359
15.20.3.114 MatrixApplyT() [1/3]	359
15.20.3.115 MatrixApplyT() [2/3]	359
15.20.3.116 MatrixApplyT() [3/3]	360
15.20.3.117 PermApplyT()	360
15.20.3.118 composePermutationsLLL()	360
15.20.3.119 composePermutationsLLM()	360
15.20.3.120 composePermutationsMLM()	361
15.20.3.121 cyclic_shift_mathPerm()	361
15.20.3.122 cyclic_shift_row_col() [1/2]	361
15.20.3.123 cyclic_shift_row() [1/3]	362
15.20.3.124 cyclic_shift_row() [2/3]	362
15.20.3.125 cyclic_shift_col() [1/3]	362
15.20.3.126 cyclic_shift_col() [2/3]	362
15.20.3.127 PLUQ_basecaseV3()	362
15.20.3.128 PLUQ_basecaseV2()	363
15.20.3.129 PLUQ_basecaseCrout()	363
15.20.3.130 _PLUQ()	363
15.20.3.131 PLUQ() [4/6]	363
15.20.3.132 threads_fgemm()	364
15.20.3.133 threads_ftrsm()	364
15.20.3.134 PLUQ() [5/6]	364
15.20.3.135 fflas_const_cast() [1/3]	364
15.20.3.136 fflas_const_cast() [2/3]	364
15.20.3.137 cyclic_shift_row_col() [2/2]	365
15.20.3.138 cyclic_shift_row() [3/3]	365
15.20.3.139 cyclic_shift_col() [3/3]	365
15.20.3.140 applyP() [4/4]	365
15.20.3.141 fgetrs() [3/4]	366
15.20.3.142 fgetrs() [4/4]	366
15.20.3.143 fgesv() [3/4]	366
15.20.3.144 fgesv() [4/4]	367
15.20.3.145 ftrtri() [2/2]	367

15.20.3.146 trinv_left() [2/2]	367
15.20.3.147 ftrtrm() [2/2]	367
15.20.3.148 PLUQ() [6/6]	368
15.20.3.149 LUdivine() [4/4]	368
15.20.3.150 LUdivine_small() [2/2]	368
15.20.3.151 LUdivine_gauss() [2/2]	369
15.20.3.152 RowEchelonForm() [3/3]	369
15.20.3.153 ReducedRowEchelonForm() [3/3]	369
15.20.3.154 ColumnEchelonForm() [3/3]	369
15.20.3.155 ReducedColumnEchelonForm() [3/3]	370
15.20.3.156 Invert() [3/4]	370
15.20.3.157 Invert() [4/4]	370
15.20.3.158 Invert2() [2/2]	370
15.20.3.159 CharPoly() [6/8]	371
15.20.3.160 CharPoly() [7/8]	371
15.20.3.161 CharPoly() [8/8]	371
15.20.3.162 MinPoly() [3/4]	371
15.20.3.163 MinPoly() [4/4]	372
15.20.3.164 MatVecMinPoly() [2/2]	372
15.20.3.165 KrylovElim()	372
15.20.3.166 SpecRankProfile()	372
15.20.3.167 Rank() [3/3]	373
15.20.3.168 IsSingular() [2/2]	373
15.20.3.169 Det() [5/6]	373
15.20.3.170 Det() [6/6]	373
15.20.3.171 Solve() [3/3]	374
15.20.3.172 solveLB() [2/2]	374
15.20.3.173 solveLB2() [2/2]	374
15.20.3.174 RandomNullSpaceVector() [3/3]	374
15.20.3.175 NullSpaceBasis() [2/2]	375
15.20.3.176 RowRankProfile() [3/3]	375
15.20.3.177 ColumnRankProfile() [3/3]	375
15.20.3.178 RowRankProfileSubmatrixIndices() [2/2]	375
15.20.3.179 ColRankProfileSubmatrixIndices() [2/2]	376
15.20.3.180 RowRankProfileSubmatrix() [2/2]	376
15.20.3.181 ColRankProfileSubmatrix() [2/2]	376
15.20.3.182 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [1/2]	376
15.20.3.183 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [2/2]	377
15.20.3.184 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2]	377
15.20.3.185 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2]	377
15.20.3.186 getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()	378
15.20.3.187 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2]	378



15.20.3.188	<a href="#">getReducedEchelonForm&lt; FFLAS_FIELD&lt; FFLAS_ELT &gt; &gt;()</a> [2/2]	378
15.20.3.189	<a href="#">getReducedEchelonTransform&lt; FFLAS_FIELD&lt; FFLAS_ELT &gt; &gt;()</a>	379
15.20.3.190	<a href="#">LQUPtoInverseOfFullRankMinor()</a> [2/2]	379
15.20.3.191	<a href="#">fflas_const_cast()</a> [3/3]	379
15.20.3.192	<a href="#">failure()</a>	379
15.20.3.193	<a href="#">isOdd()</a> [1/3]	379
15.20.3.194	<a href="#">isOdd()</a> [2/3]	380
15.20.3.195	<a href="#">isOdd()</a> [3/3]	380
15.20.3.196	<a href="#">NonZeroRandomMatrix()</a> [1/2]	380
15.20.3.197	<a href="#">NonZeroRandomMatrix()</a> [2/2]	380
15.20.3.198	<a href="#">RandomMatrix()</a> [1/2]	381
15.20.3.199	<a href="#">RandomMatrix()</a> [2/2]	382
15.20.3.200	<a href="#">RandomTriangularMatrix()</a> [1/2]	382
15.20.3.201	<a href="#">RandomTriangularMatrix()</a> [2/2]	383
15.20.3.202	<a href="#">RandInt()</a>	383
15.20.3.203	<a href="#">RandomSymmetricMatrix()</a>	384
15.20.3.204	<a href="#">RandomMatrixWithRank()</a> [1/2]	384
15.20.3.205	<a href="#">RandomMatrixWithRank()</a> [2/2]	385
15.20.3.206	<a href="#">RandomIndexSubset()</a>	385
15.20.3.207	<a href="#">RandomPermutation()</a>	386
15.20.3.208	<a href="#">RandomRankProfileMatrix()</a>	386
15.20.3.209	<a href="#">swapval()</a>	386
15.20.3.210	<a href="#">RandomSymmetricRankProfileMatrix()</a>	387
15.20.3.211	<a href="#">RandomMatrixWithRankandRPM()</a> [1/2]	387
15.20.3.212	<a href="#">RandomMatrixWithRankandRPM()</a> [2/2]	388
15.20.3.213	<a href="#">RandomSymmetricMatrixWithRankandRPM()</a> [1/2]	388
15.20.3.214	<a href="#">RandomSymmetricMatrixWithRankandRPM()</a> [2/2]	389
15.20.3.215	<a href="#">RandomMatrixWithRankandRandomRPM()</a> [1/2]	389
15.20.3.216	<a href="#">RandomMatrixWithRankandRandomRPM()</a> [2/2]	390
15.20.3.217	<a href="#">RandomSymmetricMatrixWithRankandRandomRPM()</a> [1/2]	391
15.20.3.218	<a href="#">RandomSymmetricMatrixWithRankandRandomRPM()</a> [2/2]	391
15.20.3.219	<a href="#">RandomMatrixWithDet()</a> [1/2]	392
15.20.3.220	<a href="#">RandomMatrixWithDet()</a> [2/2]	392
15.20.3.221	<a href="#">maxFieldElt()</a>	393
15.20.3.222	<a href="#">maxFieldElt&lt; Givaro::ZRing&lt; Givaro::Integer &gt; &gt;()</a>	393
15.20.3.223	<a href="#">chooseField()</a>	393
15.20.3.224	<a href="#">chooseField&lt; Givaro::ZRing&lt; int32_t &gt; &gt;()</a>	393
15.20.3.225	<a href="#">chooseField&lt; Givaro::ZRing&lt; int64_t &gt; &gt;()</a>	393
15.20.3.226	<a href="#">chooseField&lt; Givaro::ZRing&lt; float &gt; &gt;()</a>	394
15.20.3.227	<a href="#">chooseField&lt; Givaro::ZRing&lt; double &gt; &gt;()</a>	394
15.21	<a href="#">FFPACK::Protected Namespace Reference</a>	394
15.21.1	<a href="#">Function Documentation</a>	395

15.21.1.1 LUdivine_construct() [1/2]	396
15.21.1.2 GaussJordan()	396
15.21.1.3 KellerGehrig()	397
15.21.1.4 KGFast()	397
15.21.1.5 KGFast_generalized()	397
15.21.1.6 fgemv_kgf()	397
15.21.1.7 LUKrylov()	397
15.21.1.8 Danilevski()	398
15.21.1.9 RandomKrylovPrecond()	398
15.21.1.10 ArithProg()	398
15.21.1.11 LUKrylov_KGFast()	398
15.21.1.12 MatVecMinPoly()	399
15.21.1.13 Hybrid_KGF_LUK_MinPoly()	399
15.21.1.14 updateD()	399
15.21.1.15 newD()	399
15.21.1.16 CompressRows()	400
15.21.1.17 CompressRowsQK()	400
15.21.1.18 DeCompressRows()	400
15.21.1.19 DeCompressRowsQK()	400
15.21.1.20 CompressRowsQA()	400
15.21.1.21 DeCompressRowsQA()	401
15.21.1.22 LUdivine_construct() [2/2]	401
15.22 Givaro Namespace Reference	401
15.23 MKL_CONFIG Namespace Reference	401
15.24 Reclnt Namespace Reference	401
<b>16 Data Structure Documentation</b>	<b>403</b>
16.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference	403
16.1.1 Member Typedef Documentation	403
16.1.1.1 value	403
16.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference	403
16.2.1 Member Typedef Documentation	403
16.2.1.1 value	403
16.3 ArbitraryPrecIntTag Struct Reference	403
16.3.1 Detailed Description	404
16.4 AreEqual< X, Y > Class Template Reference	404
16.4.1 Field Documentation	404
16.4.1.1 value	404
16.5 AreEqual< X, X > Class Template Reference	404
16.5.1 Field Documentation	404
16.5.1.1 value	404
16.6 Argument Struct Reference	404

16.6.1 Field Documentation . . . . .	405
16.6.1.1 c . . . . .	405
16.6.1.2 example . . . . .	405
16.6.1.3 helpString . . . . .	405
16.6.1.4 type . . . . .	405
16.6.1.5 data . . . . .	405
16.7 associatedDelayedField< Field > Struct Template Reference . . . . .	405
16.7.1 Member Typedef Documentation . . . . .	405
16.7.1.1 field . . . . .	405
16.7.1.2 type . . . . .	405
16.8 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference . . . . .	405
16.8.1 Member Typedef Documentation . . . . .	406
16.8.1.1 field . . . . .	406
16.8.1.2 type . . . . .	406
16.9 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference . . . . .	406
16.9.1 Member Typedef Documentation . . . . .	406
16.9.1.1 field . . . . .	406
16.9.1.2 type . . . . .	406
16.10 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference . . . . .	406
16.10.1 Member Typedef Documentation . . . . .	407
16.10.1.1 field . . . . .	407
16.10.1.2 type . . . . .	407
16.11 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference . . . . .	407
16.11.1 Member Typedef Documentation . . . . .	407
16.11.1.1 field . . . . .	407
16.11.1.2 type . . . . .	407
16.12 Auto Struct Reference . . . . .	407
16.13 Bini Struct Reference . . . . .	407
16.14 Block Struct Reference . . . . .	407
16.15 callLUdivine_small< Element > Class Template Reference . . . . .	408
16.15.1 Member Function Documentation . . . . .	408
16.15.1.1 operator() . . . . .	408
16.16 callLUdivine_small< double > Class Reference . . . . .	408
16.16.1 Member Function Documentation . . . . .	408
16.16.1.1 operator() . . . . .	408
16.17 callLUdivine_small< float > Class Reference . . . . .	409
16.17.1 Member Function Documentation . . . . .	409
16.17.1.1 operator() . . . . .	409
16.18 CharpolyFailed Class Reference . . . . .	409
16.19 Checker_Empty< Field > Struct Template Reference . . . . .	409
16.19.1 Constructor & Destructor Documentation . . . . .	409
16.19.1.1 Checker_Empty() . . . . .	409

16.19.2 Member Function Documentation	410
16.19.2.1 check()	410
16.20 CheckerImplem_charpoly< Field, Polynomial > Class Template Reference	410
16.20.1 Constructor & Destructor Documentation	410
16.20.1.1 CheckerImplem_charpoly() [1/2]	410
16.20.1.2 CheckerImplem_charpoly() [2/2]	410
16.20.1.3 ~CheckerImplem_charpoly()	410
16.20.2 Member Function Documentation	410
16.20.2.1 check()	410
16.21 CheckerImplem_Det< Field > Class Template Reference	411
16.21.1 Constructor & Destructor Documentation	411
16.21.1.1 CheckerImplem_Det() [1/2]	411
16.21.1.2 CheckerImplem_Det() [2/2]	411
16.21.1.3 ~CheckerImplem_Det()	411
16.21.2 Member Function Documentation	411
16.21.2.1 check()	411
16.22 CheckerImplem_fgemm< Field > Class Template Reference	412
16.22.1 Constructor & Destructor Documentation	412
16.22.1.1 CheckerImplem_fgemm() [1/2]	412
16.22.1.2 CheckerImplem_fgemm() [2/2]	412
16.22.1.3 ~CheckerImplem_fgemm()	412
16.22.2 Member Function Documentation	412
16.22.2.1 check()	413
16.23 CheckerImplem_ftsm< Field > Class Template Reference	413
16.23.1 Constructor & Destructor Documentation	413
16.23.1.1 CheckerImplem_ftsm() [1/2]	413
16.23.1.2 CheckerImplem_ftsm() [2/2]	413
16.23.1.3 ~CheckerImplem_ftsm()	413
16.23.2 Member Function Documentation	414
16.23.2.1 check()	414
16.24 CheckerImplem_invert< Field > Class Template Reference	414
16.24.1 Constructor & Destructor Documentation	414
16.24.1.1 CheckerImplem_invert() [1/2]	414
16.24.1.2 CheckerImplem_invert() [2/2]	414
16.24.1.3 ~CheckerImplem_invert()	414
16.24.2 Member Function Documentation	415
16.24.2.1 check()	415
16.25 CheckerImplem_PLUQ< Field > Class Template Reference	415
16.25.1 Constructor & Destructor Documentation	415
16.25.1.1 CheckerImplem_PLUQ() [1/2]	415
16.25.1.2 CheckerImplem_PLUQ() [2/2]	415
16.25.1.3 ~CheckerImplem_PLUQ()	415

16.25.2 Member Function Documentation	415
16.25.2.1 check()	416
16.26 Classic Struct Reference	416
16.27 Column Struct Reference	416
16.28 CompactElement< Element > Struct Template Reference	416
16.28.1 Member Typedef Documentation	416
16.28.1.1 type	416
16.29 CompactElement< double > Struct Reference	416
16.29.1 Member Typedef Documentation	417
16.29.1.1 type	417
16.30 CompactElement< float > Struct Reference	417
16.30.1 Member Typedef Documentation	417
16.30.1.1 type	417
16.31 CompactElement< int16_t > Struct Reference	417
16.31.1 Member Typedef Documentation	417
16.31.1.1 type	417
16.32 CompactElement< int32_t > Struct Reference	417
16.32.1 Member Typedef Documentation	417
16.32.1.1 type	417
16.33 CompactElement< int64_t > Struct Reference	418
16.33.1 Member Typedef Documentation	418
16.33.1.1 type	418
16.34 compatible_data_type< Field > Struct Template Reference	418
16.34.1 Field Documentation	418
16.34.1.1 value	418
16.35 compatible_data_type< Givaro::ZRing< double > > Struct Reference	418
16.35.1 Field Documentation	418
16.35.1.1 value	418
16.36 compatible_data_type< Givaro::ZRing< float > > Struct Reference	418
16.36.1 Field Documentation	419
16.36.1.1 value	419
16.37 Compose< H1, H2 > Struct Template Reference	419
16.37.1 Constructor & Destructor Documentation	419
16.37.1.1 Compose() [1/5]	419
16.37.1.2 Compose() [2/5]	419
16.37.1.3 Compose() [3/5]	419
16.37.1.4 Compose() [4/5]	419
16.37.1.5 Compose() [5/5]	419
16.37.2 Member Function Documentation	420
16.37.2.1 first_component()	420
16.37.2.2 second_component()	420
16.37.3 Friends And Related Function Documentation	420

16.37.3.1 operator<<	420
16.38 Const_int_t< n > Class Template Reference	420
16.39 Const_uint_t< n > Class Template Reference	420
16.40 Simd128_impl< true, true, false, 2 >::Converter Union Reference	420
16.40.1 Field Documentation	420
16.40.1.1 v	420
16.40.1.2 t	421
16.41 Simd128_impl< true, true, false, 4 >::Converter Union Reference	421
16.41.1 Field Documentation	421
16.41.1.1 v	421
16.41.1.2 t	421
16.42 Simd128_impl< true, true, false, 8 >::Converter Union Reference	421
16.42.1 Field Documentation	421
16.42.1.1 v	421
16.42.1.2 t	421
16.43 Simd128_impl< true, true, true, 2 >::Converter Union Reference	421
16.43.1 Field Documentation	422
16.43.1.1 v	422
16.43.1.2 t	422
16.44 Simd128_impl< true, true, true, 4 >::Converter Union Reference	422
16.44.1 Field Documentation	422
16.44.1.1 v	422
16.44.1.2 t	422
16.45 Simd128_impl< true, true, true, 8 >::Converter Union Reference	422
16.45.1 Field Documentation	422
16.45.1.1 v	422
16.45.1.2 t	422
16.46 Simd256_impl< true, true, false, 2 >::Converter Union Reference	423
16.46.1 Field Documentation	423
16.46.1.1 v	423
16.46.1.2 t	423
16.47 Simd256_impl< true, true, false, 4 >::Converter Union Reference	423
16.47.1 Field Documentation	423
16.47.1.1 v	423
16.47.1.2 t	423
16.48 Simd256_impl< true, true, false, 8 >::Converter Union Reference	423
16.48.1 Field Documentation	423
16.48.1.1 v	424
16.48.1.2 t	424
16.49 Simd256_impl< true, true, true, 2 >::Converter Union Reference	424
16.49.1 Field Documentation	424
16.49.1.1 v	424

16.49.1.2 t . . . . .	424
16.50 Simd256_impl< true, true, true, 4 >::Converter Union Reference . . . . .	424
16.50.1 Field Documentation . . . . .	424
16.50.1.1 v . . . . .	424
16.50.1.2 t . . . . .	424
16.51 Simd256_impl< true, true, true, 8 >::Converter Union Reference . . . . .	425
16.51.1 Field Documentation . . . . .	425
16.51.1.1 v . . . . .	425
16.51.1.2 t . . . . .	425
16.52 Simd512_impl< true, true, false, 8 >::Converter Union Reference . . . . .	425
16.52.1 Field Documentation . . . . .	425
16.52.1.1 v . . . . .	425
16.52.1.2 t . . . . .	425
16.53 Simd512_impl< true, true, true, 8 >::Converter Union Reference . . . . .	425
16.53.1 Field Documentation . . . . .	425
16.53.1.1 v . . . . .	426
16.53.1.2 t . . . . .	426
16.54 ConvertTo< T > Struct Template Reference . . . . .	426
16.54.1 Detailed Description . . . . .	426
16.55 Coo< ValT, IdxT > Struct Template Reference . . . . .	426
16.55.1 Member Typedef Documentation . . . . .	426
16.55.1.1 Self . . . . .	427
16.55.2 Constructor & Destructor Documentation . . . . .	427
16.55.2.1 Coo() [1/4] . . . . .	427
16.55.2.2 Coo() [2/4] . . . . .	427
16.55.2.3 Coo() [3/4] . . . . .	427
16.55.2.4 Coo() [4/4] . . . . .	427
16.55.3 Member Function Documentation . . . . .	427
16.55.3.1 operator=() [1/2] . . . . .	427
16.55.3.2 operator=() [2/2] . . . . .	427
16.55.4 Field Documentation . . . . .	427
16.55.4.1 val . . . . .	427
16.55.4.2 row . . . . .	427
16.55.4.3 col . . . . .	428
16.56 Coo< Field > Struct Template Reference . . . . .	428
16.56.1 Constructor & Destructor Documentation . . . . .	428
16.56.1.1 Coo() [1/4] . . . . .	428
16.56.1.2 Coo() [2/4] . . . . .	428
16.56.1.3 Coo() [3/4] . . . . .	428
16.56.1.4 Coo() [4/4] . . . . .	428
16.56.2 Member Function Documentation . . . . .	428
16.56.2.1 operator=() [1/2] . . . . .	429

16.56.2.2 operator=() [2/2]	429
16.56.3 Field Documentation	429
16.56.3.1 val	429
16.56.3.2 col	429
16.56.3.3 row	429
16.56.3.4 deleted	429
16.57 Coo< ValT, IdxT > Struct Template Reference	429
16.57.1 Member Typedef Documentation	430
16.57.1.1 Self	430
16.57.2 Constructor & Destructor Documentation	430
16.57.2.1 Coo() [1/4]	430
16.57.2.2 Coo() [2/4]	430
16.57.2.3 Coo() [3/4]	430
16.57.2.4 Coo() [4/4]	430
16.57.3 Member Function Documentation	430
16.57.3.1 operator=() [1/2]	430
16.57.3.2 operator=() [2/2]	430
16.57.4 Field Documentation	430
16.57.4.1 val	430
16.57.4.2 row	431
16.57.4.3 col	431
16.58 CooMat< Field > Struct Template Reference	431
16.58.1 Field Documentation	431
16.58.1.1 _coo16	431
16.58.1.2 _coo32	431
16.58.1.3 _coo64	431
16.58.1.4 _coo16_zo	431
16.58.1.5 _coo32_zo	431
16.58.1.6 _coo64_zo	431
16.59 CsrMat< Field > Struct Template Reference	432
16.59.1 Field Documentation	432
16.59.1.1 _csr16	432
16.59.1.2 _csr32	432
16.59.1.3 _csr64	432
16.59.1.4 _csr16_zo	432
16.59.1.5 _csr32_zo	432
16.59.1.6 _csr64_zo	432
16.60 DefaultBoundedTag Struct Reference	432
16.60.1 Detailed Description	432
16.61 DefaultTag Struct Reference	433
16.61.1 Detailed Description	433
16.62 DelayedTag Struct Reference	433



16.62.1 Detailed Description	433
16.63 ElementTraits< Element > Struct Template Reference	433
16.63.1 Detailed Description	433
16.63.2 Member Typedef Documentation	433
16.63.2.1 value	433
16.64 ElementTraits< double > Struct Reference	433
16.64.1 Member Typedef Documentation	434
16.64.1.1 value	434
16.65 ElementTraits< FFPACK::rns_double_elt > Struct Reference	434
16.65.1 Member Typedef Documentation	434
16.65.1.1 value	434
16.66 ElementTraits< float > Struct Reference	434
16.66.1 Member Typedef Documentation	434
16.66.1.1 value	434
16.67 ElementTraits< Givaro::Integer > Struct Reference	434
16.67.1 Member Typedef Documentation	435
16.67.1.1 value	435
16.68 ElementTraits< int16_t > Struct Reference	435
16.68.1 Member Typedef Documentation	435
16.68.1.1 value	435
16.69 ElementTraits< int32_t > Struct Reference	435
16.69.1 Member Typedef Documentation	435
16.69.1.1 value	435
16.70 ElementTraits< int64_t > Struct Reference	435
16.70.1 Member Typedef Documentation	436
16.70.1.1 value	436
16.71 ElementTraits< int8_t > Struct Reference	436
16.71.1 Member Typedef Documentation	436
16.71.1.1 value	436
16.72 ElementTraits< Reclnt::rint< K > > Struct Template Reference	436
16.72.1 Member Typedef Documentation	436
16.72.1.1 value	436
16.73 ElementTraits< Reclnt::rmint< K, MG > > Struct Template Reference	436
16.73.1 Member Typedef Documentation	437
16.73.1.1 value	437
16.74 ElementTraits< Reclnt::ruint< K > > Struct Template Reference	437
16.74.1 Member Typedef Documentation	437
16.74.1.1 value	437
16.75 ElementTraits< uint16_t > Struct Reference	437
16.75.1 Member Typedef Documentation	437
16.75.1.1 value	437
16.76 ElementTraits< uint32_t > Struct Reference	437

16.76.1 Member Typedef Documentation	438
16.76.1.1 value	438
16.77 ElementTraits< uint64_t > Struct Reference	438
16.77.1 Member Typedef Documentation	438
16.77.1.1 value	438
16.78 ElementTraits< uint8_t > Struct Reference	438
16.78.1 Member Typedef Documentation	438
16.78.1.1 value	438
16.79 EllMat< Field > Struct Template Reference	438
16.79.1 Field Documentation	439
16.79.1.1 _ell16	439
16.79.1.2 _ell32	439
16.79.1.3 _ell64	439
16.79.1.4 _ell16_zo	439
16.79.1.5 _ell32_zo	439
16.79.1.6 _ell64_zo	439
16.80 Failure Class Reference	439
16.80.1 Detailed Description	440
16.80.2 Constructor & Destructor Documentation	440
16.80.2.1 Failure()	440
16.80.3 Member Function Documentation	440
16.80.3.1 operator>() [1/2]	440
16.80.3.2 operator>() [2/2]	440
16.80.3.3 setErrorMessage()	441
16.80.3.4 print()	441
16.80.4 Field Documentation	441
16.80.4.1 _errorMessage	441
16.81 FailureCharnpolyCheck Class Reference	441
16.82 FailureDetCheck Class Reference	441
16.83 FailureFgemmCheck Class Reference	441
16.84 FailureInvertCheck Class Reference	441
16.85 FailurePLUQCheck Class Reference	442
16.86 FailureTrsmCheck Class Reference	442
16.87 FieldSimd< _Field > Class Template Reference	442
16.87.1 Member Typedef Documentation	443
16.87.1.1 Field	443
16.87.1.2 Element	443
16.87.1.3 simd	443
16.87.1.4 vect_t	443
16.87.1.5 scalar_t	443
16.87.2 Constructor & Destructor Documentation	443
16.87.2.1 FieldSimd() [1/3]	443

16.87.2.2 FieldSimd() [2/3]	443
16.87.2.3 FieldSimd() [3/3]	443
16.87.3 Member Function Documentation	444
16.87.3.1 operator=() [1/2]	444
16.87.3.2 operator=() [2/2]	444
16.87.3.3 init() [1/2]	444
16.87.3.4 init() [2/2]	444
16.87.3.5 add() [1/2]	444
16.87.3.6 add() [2/2]	444
16.87.3.7 addin()	444
16.87.3.8 add_r() [1/2]	444
16.87.3.9 add_r() [2/2]	445
16.87.3.10 addin_r()	445
16.87.3.11 sub() [1/2]	445
16.87.3.12 sub() [2/2]	445
16.87.3.13 subin()	445
16.87.3.14 sub_r() [1/2]	445
16.87.3.15 sub_r() [2/2]	445
16.87.3.16 subin_r()	445
16.87.3.17 zero() [1/2]	446
16.87.3.18 zero() [2/2]	446
16.87.3.19 mod()	446
16.87.3.20 mul() [1/2]	446
16.87.3.21 mul() [2/2]	446
16.87.3.22 mulin()	446
16.87.3.23 mul_r() [1/2]	446
16.87.3.24 mul_r() [2/2]	446
16.87.3.25 axpy() [1/2]	446
16.87.3.26 axpy() [2/2]	447
16.87.3.27 axpyin()	447
16.87.3.28 axpy_r() [1/2]	447
16.87.3.29 axpy_r() [2/2]	447
16.87.3.30 axpyin_r()	447
16.87.3.31 maxpy() [1/2]	447
16.87.3.32 maxpy() [2/2]	447
16.87.3.33 maxpyin()	448
16.87.4 Field Documentation	448
16.87.4.1 vect_size	448
16.87.4.2 alignment	448
16.88 FieldTraits< Field > Struct Template Reference	448
16.88.1 Detailed Description	448
16.88.2 Member Typedef Documentation	448

16.88.2.1 category	448
16.88.3 Field Documentation	448
16.88.3.1 balanced	449
16.89 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference	449
16.89.1 Member Typedef Documentation	449
16.89.1.1 category	449
16.89.2 Field Documentation	449
16.89.2.1 balanced	449
16.90 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference	449
16.90.1 Member Typedef Documentation	449
16.90.1.1 category	450
16.90.2 Field Documentation	450
16.90.2.1 balanced	450
16.91 FieldTraits< Givaro::Modular< Element > > Struct Template Reference	450
16.91.1 Member Typedef Documentation	450
16.91.1.1 category	450
16.91.2 Field Documentation	450
16.91.2.1 balanced	450
16.92 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference	450
16.92.1 Member Typedef Documentation	451
16.92.1.1 category	451
16.92.2 Field Documentation	451
16.92.2.1 balanced	451
16.93 FieldTraits< Givaro::ZRing< double > > Struct Reference	451
16.93.1 Member Typedef Documentation	451
16.93.1.1 category	451
16.93.2 Field Documentation	451
16.93.2.1 balanced	451
16.94 FieldTraits< Givaro::ZRing< float > > Struct Reference	451
16.94.1 Member Typedef Documentation	452
16.94.1.1 category	452
16.94.2 Field Documentation	452
16.94.2.1 balanced	452
16.95 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference	452
16.95.1 Member Typedef Documentation	452
16.95.1.1 category	452
16.95.2 Field Documentation	452
16.95.2.1 balanced	452
16.96 FieldTraits< Givaro::ZRing< int16_t > > Struct Reference	453
16.96.1 Member Typedef Documentation	453
16.96.1.1 category	453
16.96.2 Field Documentation	453

16.96.2.1 balanced	453
16.97 FieldTraits< Givaro::ZRing< int32_t > > Struct Reference	453
16.97.1 Member Typedef Documentation	453
16.97.1.1 category	453
16.97.2 Field Documentation	453
16.97.2.1 balanced	454
16.98 FieldTraits< Givaro::ZRing< int64_t > > Struct Reference	454
16.98.1 Member Typedef Documentation	454
16.98.1.1 category	454
16.98.2 Field Documentation	454
16.98.2.1 balanced	454
16.99 FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > Struct Template Reference	454
16.99.1 Member Typedef Documentation	454
16.99.1.1 category	454
16.99.2 Field Documentation	455
16.99.2.1 balanced	455
16.100 FieldTraits< Givaro::ZRing< uint16_t > > Struct Reference	455
16.100.1 Member Typedef Documentation	455
16.100.1.1 category	455
16.100.2 Field Documentation	455
16.100.2.1 balanced	455
16.101 FieldTraits< Givaro::ZRing< uint32_t > > Struct Reference	455
16.101.1 Member Typedef Documentation	455
16.101.1.1 category	456
16.101.2 Field Documentation	456
16.101.2.1 balanced	456
16.102 FieldTraits< Givaro::ZRing< uint64_t > > Struct Reference	456
16.102.1 Member Typedef Documentation	456
16.102.1.1 category	456
16.102.2 Field Documentation	456
16.102.2.1 balanced	456
16.103 Fixed Struct Reference	456
16.104 FixedPreclntTag Struct Reference	456
16.104.1 Detailed Description	457
16.105 ForStrategy1D< blocksize_t, Cut, Param > Struct Template Reference	457
16.105.1 Constructor & Destructor Documentation	457
16.105.1.1 ForStrategy1D() [1/2]	457
16.105.1.2 ForStrategy1D() [2/2]	457
16.105.2 Member Function Documentation	457
16.105.2.1 build()	458
16.105.2.2 initialize()	458
16.105.2.3 isTerminated()	458

16.105.2.4 begin()	458
16.105.2.5 end()	458
16.105.2.6 numblocks()	458
16.105.2.7 blockindex()	458
16.105.2.8 operator++()	458
16.105.3 Field Documentation	458
16.105.3.1 ibeg	458
16.105.3.2 iend	458
16.105.3.3 current	458
16.105.3.4 firstBlockSize	459
16.105.3.5 lastBlockSize	459
16.105.3.6 changeBS	459
16.105.3.7 numBlock	459
16.106 ForStrategy2D< blockSize_t, Cut, Param > Struct Template Reference	459
16.106.1 Constructor & Destructor Documentation	460
16.106.1.1 ForStrategy2D()	460
16.106.2 Member Function Documentation	460
16.106.2.1 initialize()	460
16.106.2.2 isTerminated()	460
16.106.2.3 ibegin()	460
16.106.2.4 jbegin()	460
16.106.2.5 iend()	460
16.106.2.6 jend()	460
16.106.2.7 operator++()	460
16.106.2.8 rownumblocks()	460
16.106.2.9 colnumblocks()	461
16.106.2.10 blockindex()	461
16.106.2.11 rowblockindex()	461
16.106.2.12 colblockindex()	461
16.106.3 Friends And Related Function Documentation	461
16.106.3.1 operator<<	461
16.106.4 Field Documentation	461
16.106.4.1 _ibeg	461
16.106.4.2 _iend	461
16.106.4.3 _jbeg	461
16.106.4.4 _jend	461
16.106.4.5 rowBlockSize	461
16.106.4.6 colBlockSize	462
16.106.4.7 current	462
16.106.4.8 lastRBS	462
16.106.4.9 lastCBS	462
16.106.4.10 changeRBS	462

16.106.4.11 changeCBS . . . . .	462
16.106.4.12 numRowsBlock . . . . .	462
16.106.4.13 numColBlock . . . . .	462
16.106.4.14 BLOCKS . . . . .	462
16.107 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference . . . . .	462
16.108 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference . . . . .	462
16.109 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference . . . . .	463
16.110 ftrmmLeftLowerTransUnit< Element > Class Template Reference . . . . .	463
16.111 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference . . . . .	463
16.112 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference . . . . .	463
16.113 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference . . . . .	463
16.114 ftrmmLeftUpperTransUnit< Element > Class Template Reference . . . . .	463
16.115 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference . . . . .	463
16.116 ftrmmRightLowerNoTransUnit< Element > Class Template Reference . . . . .	463
16.117 ftrmmRightLowerTransNonUnit< Element > Class Template Reference . . . . .	464
16.118 ftrmmRightLowerTransUnit< Element > Class Template Reference . . . . .	464
16.119 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference . . . . .	464
16.120 ftrmmRightUpperNoTransUnit< Element > Class Template Reference . . . . .	464
16.121 ftrmmRightUpperTransNonUnit< Element > Class Template Reference . . . . .	464
16.122 ftrmmRightUpperTransUnit< Element > Class Template Reference . . . . .	464
16.123 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference . . . . .	464
16.124 ftrsmLeftLowerNoTransUnit< Element > Class Template Reference . . . . .	464
16.125 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference . . . . .	465
16.126 ftrsmLeftLowerTransUnit< Element > Class Template Reference . . . . .	465
16.127 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference . . . . .	465
16.127.1 Detailed Description . . . . .	465
16.128 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference . . . . .	465
16.129 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference . . . . .	465
16.130 ftrsmLeftUpperTransUnit< Element > Class Template Reference . . . . .	465
16.131 ftrsmRightLowerNoTransNonUnit< Element > Class Template Reference . . . . .	466
16.132 ftrsmRightLowerNoTransUnit< Element > Class Template Reference . . . . .	466
16.133 ftrsmRightLowerTransNonUnit< Element > Class Template Reference . . . . .	466
16.134 ftrsmRightLowerTransUnit< Element > Class Template Reference . . . . .	466
16.135 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference . . . . .	466
16.136 ftrsmRightUpperNoTransUnit< Element > Class Template Reference . . . . .	466
16.137 ftrsmRightUpperTransNonUnit< Element > Class Template Reference . . . . .	466
16.138 ftrsmRightUpperTransUnit< Element > Class Template Reference . . . . .	466
16.139 GenericTag Struct Reference . . . . .	467
16.139.1 Detailed Description . . . . .	467
16.140 GenericTag Struct Reference . . . . .	467
16.140.1 Detailed Description . . . . .	467
16.141 Grain Struct Reference . . . . .	467

16.142 <a href="#">has_minus_eq_impl&lt; C &gt; Struct Template Reference</a>	467
16.142.1 Field Documentation	467
16.142.1.1 value	467
16.143 <a href="#">has_minus_impl&lt; C &gt; Struct Template Reference</a>	467
16.143.1 Field Documentation	468
16.143.1.1 value	468
16.144 <a href="#">has_mul_eq_impl&lt; C &gt; Struct Template Reference</a>	468
16.144.1 Field Documentation	468
16.144.1.1 value	468
16.145 <a href="#">has_mul_impl&lt; C &gt; Struct Template Reference</a>	468
16.145.1 Field Documentation	468
16.145.1.1 value	468
16.146 <a href="#">has_operation&lt; T &gt; Struct Template Reference</a>	468
16.146.1 Field Documentation	469
16.146.1.1 value	469
16.147 <a href="#">has_plus_eq_impl&lt; C &gt; Struct Template Reference</a>	469
16.147.1 Field Documentation	469
16.147.1.1 value	469
16.148 <a href="#">has_plus_impl&lt; C &gt; Struct Template Reference</a>	469
16.148.1 Field Documentation	469
16.148.1.1 value	469
16.149 <a href="#">HelperFlag Struct Reference</a>	469
16.149.1 Field Documentation	470
16.149.1.1 none	470
16.149.1.2 coo	470
16.149.1.3 csr	470
16.149.1.4 ell	470
16.149.1.5 aut	470
16.149.1.6 pm1	470
16.150 <a href="#">HelperMod&lt; Field, ElementTraits &gt; Struct Template Reference</a>	470
16.151 <a href="#">HelperMod&lt; Field, ElementCategories::MachineIntTag &gt; Struct Template Reference</a>	470
16.151.1 Constructor & Destructor Documentation	471
16.151.1.1 <a href="#">HelperMod()</a> [1/2]	471
16.151.1.2 <a href="#">HelperMod()</a> [2/2]	471
16.151.2 Field Documentation	471
16.151.2.1 p	471
16.151.2.2 invp	471
16.151.2.3 min	471
16.151.2.4 max	471
16.151.2.5 pow50rem	471
16.152 <a href="#">HelperMod&lt; Field, FFLAS::ElementCategories::ArbitraryPrecIntTag &gt; Struct Template Reference</a>	472
16.152.1 Constructor & Destructor Documentation	472



16.152.1.1 HelperMod() [1/2]	472
16.152.1.2 HelperMod() [2/2]	472
16.152.2 Field Documentation	472
16.152.2.1 p	472
16.153 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference	472
16.153.1 Constructor & Destructor Documentation	472
16.153.1.1 HelperMod() [1/2]	472
16.153.1.2 HelperMod() [2/2]	473
16.153.2 Field Documentation	473
16.153.2.1 p	473
16.154 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference	473
16.154.1 Constructor & Destructor Documentation	473
16.154.1.1 HelperMod() [1/2]	473
16.154.1.2 HelperMod() [2/2]	473
16.154.2 Field Documentation	473
16.154.2.1 p	473
16.154.2.2 invp	473
16.154.2.3 min	474
16.154.2.4 max	474
16.155 Hybrid Struct Reference	474
16.156 Info Struct Reference	474
16.156.1 Constructor & Destructor Documentation	474
16.156.1.1 Info() [1/4]	474
16.156.1.2 Info() [2/4]	474
16.156.1.3 Info() [3/4]	474
16.156.1.4 Info() [4/4]	475
16.156.2 Member Function Documentation	475
16.156.2.1 operator=() [1/2]	475
16.156.2.2 operator=() [2/2]	475
16.156.3 Field Documentation	475
16.156.3.1 size	475
16.156.3.2 perm	475
16.156.3.3 begin	475
16.157 Info Struct Reference	475
16.157.1 Constructor & Destructor Documentation	476
16.157.1.1 Info() [1/4]	476
16.157.1.2 Info() [2/4]	476
16.157.1.3 Info() [3/4]	476
16.157.1.4 Info() [4/4]	476
16.157.2 Member Function Documentation	476
16.157.2.1 operator=() [1/2]	476
16.157.2.2 operator=() [2/2]	476

16.157.3 Field Documentation	476
16.157.3.1 size	476
16.157.3.2 perm	476
16.157.3.3 begin	477
16.158 is_simd< T > Struct Template Reference	477
16.158.1 Member Typedef Documentation	477
16.158.1.1 type	477
16.158.2 Field Documentation	477
16.158.2.1 value	477
16.159 isSparseMatrix< Field, M > Struct Template Reference	477
16.160 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > Struct Template Reference	478
16.161 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference	478
16.162 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > Struct Template Reference	478
16.163 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > Struct Template Reference	478
16.164 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference	479
16.165 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > Struct Template Reference	479
16.166 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > Struct Template Reference	479
16.167 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference	480
16.168 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference	480
16.169 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > Struct Template Reference	480
16.170 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > Struct Template Reference	481
16.171 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference	481
16.172 isSparseMatrixMKLFormat< F, M > Struct Template Reference	481
16.173 isSparseMatrixSimdFormat< F, M > Struct Template Reference	481
16.174 isZOSparseMatrix< F, M > Struct Template Reference	482
16.175 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference	482
16.176 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference	482
16.177 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference	483
16.178 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference	483
16.179 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference	483
16.180 Iterative Struct Reference	483
16.181 LazyTag Struct Reference	484
16.181.1 Detailed Description	484
16.182 limits< T > Struct Template Reference	484
16.183 limits< char > Struct Reference	484
16.183.1 Member Typedef Documentation	484
16.183.1.1 T	484
16.183.2 Member Function Documentation	484
16.183.2.1 max()	484

16.183.2.2 min()	484
16.183.2.3 digits()	484
16.184 limits< double > Struct Reference	485
16.184.1 Member Typedef Documentation	485
16.184.1.1 T	485
16.184.2 Member Function Documentation	485
16.184.2.1 max()	485
16.184.2.2 min()	485
16.184.2.3 digits()	485
16.185 limits< float > Struct Reference	485
16.185.1 Member Typedef Documentation	486
16.185.1.1 T	486
16.185.2 Member Function Documentation	486
16.185.2.1 max()	486
16.185.2.2 min()	486
16.185.2.3 digits()	486
16.186 limits< Givaro::Integer > Struct Reference	486
16.186.1 Member Typedef Documentation	486
16.186.1.1 T	486
16.186.2 Member Function Documentation	486
16.186.2.1 max()	486
16.186.2.2 min()	487
16.187 limits< int > Struct Reference	487
16.187.1 Member Typedef Documentation	487
16.187.1.1 T	487
16.187.2 Member Function Documentation	487
16.187.2.1 max()	487
16.187.2.2 min()	487
16.187.2.3 digits()	487
16.188 limits< long > Struct Reference	487
16.188.1 Member Typedef Documentation	488
16.188.1.1 T	488
16.188.2 Member Function Documentation	488
16.188.2.1 max()	488
16.188.2.2 min()	488
16.188.2.3 digits()	488
16.189 limits< long long > Struct Reference	488
16.189.1 Member Typedef Documentation	488
16.189.1.1 T	488
16.189.2 Member Function Documentation	489
16.189.2.1 max()	489
16.189.2.2 min()	489

16.189.2.3 digits()	489
16.190 limits< Reclnt::rint< K > > Struct Template Reference	489
16.190.1 Member Typedef Documentation	489
16.190.1.1 T	489
16.190.2 Member Function Documentation	489
16.190.2.1 max()	489
16.190.2.2 min()	489
16.191 limits< Reclnt::ruint< K > > Struct Template Reference	490
16.191.1 Member Typedef Documentation	490
16.191.1.1 T	490
16.191.2 Member Function Documentation	490
16.191.2.1 max()	490
16.191.2.2 min()	490
16.192 limits< short int > Struct Reference	490
16.192.1 Member Typedef Documentation	490
16.192.1.1 T	490
16.192.2 Member Function Documentation	491
16.192.2.1 max()	491
16.192.2.2 min()	491
16.192.2.3 digits()	491
16.193 limits< signed char > Struct Reference	491
16.193.1 Member Typedef Documentation	491
16.193.1.1 T	491
16.193.2 Member Function Documentation	491
16.193.2.1 max()	491
16.193.2.2 min()	491
16.193.2.3 digits()	492
16.194 limits< unsigned char > Struct Reference	492
16.194.1 Member Typedef Documentation	492
16.194.1.1 T	492
16.194.2 Member Function Documentation	492
16.194.2.1 max()	492
16.194.2.2 min()	492
16.194.2.3 digits()	492
16.195 limits< unsigned int > Struct Reference	492
16.195.1 Member Typedef Documentation	493
16.195.1.1 T	493
16.195.2 Member Function Documentation	493
16.195.2.1 max()	493
16.195.2.2 min()	493
16.195.2.3 digits()	493
16.196 limits< unsigned long > Struct Reference	493

16.196.1 Member Typedef Documentation . . . . .	493
16.196.1.1 T . . . . .	493
16.196.2 Member Function Documentation . . . . .	494
16.196.2.1 max() . . . . .	494
16.196.2.2 min() . . . . .	494
16.196.2.3 digits() . . . . .	494
16.197 limits< unsigned long long > Struct Reference . . . . .	494
16.197.1 Member Typedef Documentation . . . . .	494
16.197.1.1 T . . . . .	494
16.197.2 Member Function Documentation . . . . .	494
16.197.2.1 max() . . . . .	494
16.197.2.2 min() . . . . .	494
16.197.2.3 digits() . . . . .	495
16.198 limits< unsigned short int > Struct Reference . . . . .	495
16.198.1 Member Typedef Documentation . . . . .	495
16.198.1.1 T . . . . .	495
16.198.2 Member Function Documentation . . . . .	495
16.198.2.1 max() . . . . .	495
16.198.2.2 min() . . . . .	495
16.198.2.3 digits() . . . . .	495
16.199 MachineFloatTag Struct Reference . . . . .	495
16.199.1 Detailed Description . . . . .	496
16.200 MachineIntTag Struct Reference . . . . .	496
16.200.1 Detailed Description . . . . .	496
16.201 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference . . . . .	496
16.201.1 Member Typedef Documentation . . . . .	497
16.201.1.1 Self_t . . . . .	497
16.201.1.2 DelayedField_t . . . . .	497
16.201.1.3 DelayedField . . . . .	497
16.201.1.4 DFElt . . . . .	497
16.201.2 Constructor & Destructor Documentation . . . . .	497
16.201.2.1 MMHelper() [1/5] . . . . .	497
16.201.2.2 MMHelper() [2/5] . . . . .	498
16.201.2.3 MMHelper() [3/5] . . . . .	498
16.201.2.4 MMHelper() [4/5] . . . . .	498
16.201.2.5 MMHelper() [5/5] . . . . .	498
16.201.3 Member Function Documentation . . . . .	498
16.201.3.1 initC() . . . . .	498
16.201.3.2 initA() . . . . .	498
16.201.3.3 initB() . . . . .	498
16.201.3.4 initOut() . . . . .	498
16.201.3.5 MaxDelayedDim() . . . . .	499

16.201.3.6 Aunfit()	499
16.201.3.7 Bunfit()	499
16.201.3.8 setOutBounds()	499
16.201.3.9 checkA()	499
16.201.3.10 checkB()	499
16.201.3.11 checkOut()	499
16.201.4 Friends And Related Function Documentation	499
16.201.4.1 operator<<	500
16.201.5 Field Documentation	500
16.201.5.1 recLevel	500
16.201.5.2 FieldMin	500
16.201.5.3 FieldMax	500
16.201.5.4 Amin	500
16.201.5.5 Amax	500
16.201.5.6 Bmin	500
16.201.5.7 Bmax	500
16.201.5.8 Cmin	500
16.201.5.9 Cmax	500
16.201.5.10 Outmin	500
16.201.5.11 Outmax	501
16.201.5.12 MaxStorableValue	501
16.201.5.13 delayedField	501
16.201.5.14 parseq	501
16.202 MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference	501
16.202.1 Member Typedef Documentation	501
16.202.1.1 Self_t	502
16.202.2 Constructor & Destructor Documentation	502
16.202.2.1 MMHelper() [1/5]	502
16.202.2.2 MMHelper() [2/5]	502
16.202.2.3 MMHelper() [3/5]	502
16.202.2.4 MMHelper() [4/5]	502
16.202.2.5 MMHelper() [5/5]	502
16.202.3 Member Function Documentation	502
16.202.3.1 setNorm()	502
16.202.4 Friends And Related Function Documentation	502
16.202.4.1 operator<<	503
16.202.5 Field Documentation	503
16.202.5.1 normA	503
16.202.5.2 normB	503
16.202.5.3 recLevel	503
16.202.5.4 parseq	503

16.203 MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeq↔ Trait > Struct Template Reference	503
16.203.1 Member Typedef Documentation	504
16.203.1.1 Self_t	504
16.203.2 Constructor & Destructor Documentation	504
16.203.2.1 MMHelper() [1/5]	504
16.203.2.2 MMHelper() [2/5]	504
16.203.2.3 MMHelper() [3/5]	504
16.203.2.4 MMHelper() [4/5]	504
16.203.2.5 MMHelper() [5/5]	504
16.203.3 Member Function Documentation	504
16.203.3.1 setNorm()	504
16.203.4 Friends And Related Function Documentation	505
16.203.4.1 operator<<	505
16.203.5 Field Documentation	505
16.203.5.1 normA	505
16.203.5.2 normB	505
16.203.5.3 recLevel	505
16.203.5.4 parseq	505
16.204 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > Struct Tem- plate Reference	505
16.204.1 Member Typedef Documentation	506
16.204.1.1 Self_t	506
16.204.2 Constructor & Destructor Documentation	506
16.204.2.1 MMHelper() [1/4]	506
16.204.2.2 MMHelper() [2/4]	506
16.204.2.3 MMHelper() [3/4]	506
16.204.2.4 MMHelper() [4/4]	506
16.204.3 Friends And Related Function Documentation	506
16.204.3.1 operator<<	506
16.204.4 Field Documentation	506
16.204.4.1 recLevel	506
16.204.4.2 parseq	507
16.205 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Struct Template Reference	507
16.205.1 Member Typedef Documentation	507
16.205.1.1 Self_t	507
16.205.2 Constructor & Destructor Documentation	507
16.205.2.1 MMHelper() [1/5]	507
16.205.2.2 MMHelper() [2/5]	508
16.205.2.3 MMHelper() [3/5]	508
16.205.2.4 MMHelper() [4/5]	508
16.205.2.5 MMHelper() [5/5]	508

16.205.3 Member Function Documentation . . . . .	508
16.205.3.1 setNorm() . . . . .	508
16.205.4 Friends And Related Function Documentation . . . . .	508
16.205.4.1 operator<< . . . . .	508
16.205.5 Field Documentation . . . . .	508
16.205.5.1 normA . . . . .	508
16.205.5.2 normB . . . . .	509
16.205.5.3 recLevel . . . . .	509
16.205.5.4 parseq . . . . .	509
16.206 MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference	509
16.206.1 Detailed Description . . . . .	509
16.206.2 Member Typedef Documentation . . . . .	509
16.206.2.1 Self_t . . . . .	509
16.206.3 Constructor & Destructor Documentation . . . . .	510
16.206.3.1 MMHelper() [ 1 / 4 ] . . . . .	510
16.206.3.2 MMHelper() [ 2 / 4 ] . . . . .	510
16.206.3.3 MMHelper() [ 3 / 4 ] . . . . .	510
16.206.3.4 MMHelper() [ 4 / 4 ] . . . . .	510
16.206.4 Friends And Related Function Documentation . . . . .	510
16.206.4.1 operator<< . . . . .	510
16.206.5 Field Documentation . . . . .	510
16.206.5.1 recLevel . . . . .	510
16.206.5.2 parseq . . . . .	510
16.207 ModeTraits< Field > Struct Template Reference . . . . .	511
16.207.1 Detailed Description . . . . .	511
16.207.2 Member Typedef Documentation . . . . .	511
16.207.2.1 value . . . . .	511
16.208 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference . . . . .	511
16.208.1 Member Typedef Documentation . . . . .	511
16.208.1.1 value . . . . .	511
16.209 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference . . . . .	511
16.209.1 Member Typedef Documentation . . . . .	512
16.209.1.1 value . . . . .	512
16.210 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference . . . . .	512
16.210.1 Member Typedef Documentation . . . . .	512
16.210.1.1 value . . . . .	512
16.211 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference . . . . .	512
16.211.1 Member Typedef Documentation . . . . .	512
16.211.1.1 value . . . . .	512
16.212 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference . . . . .	512
16.212.1 Member Typedef Documentation . . . . .	513
16.212.1.1 value . . . . .	513



16.213	ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference	513
16.213.1	Member Typedef Documentation	513
16.213.1.1	value	513
16.214	ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference	513
16.214.1	Member Typedef Documentation	513
16.214.1.1	value	513
16.215	ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference	514
16.215.1	Member Typedef Documentation	514
16.215.1.1	value	514
16.216	ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference	514
16.216.1	Member Typedef Documentation	514
16.216.1.1	value	514
16.217	ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference	514
16.217.1	Member Typedef Documentation	514
16.217.1.1	value	515
16.218	ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference	515
16.218.1	Member Typedef Documentation	515
16.218.1.1	value	515
16.219	ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference	515
16.219.1	Member Typedef Documentation	515
16.219.1.1	value	515
16.220	ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference	515
16.220.1	Member Typedef Documentation	516
16.220.1.1	value	516
16.221	ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference	516
16.221.1	Member Typedef Documentation	516
16.221.1.1	value	516
16.222	ModeTraits< Givaro::Montgomery< T > > Struct Template Reference	516
16.222.1	Member Typedef Documentation	516
16.222.1.1	value	516
16.223	ModeTraits< Givaro::ZRing< double > > Struct Reference	516
16.223.1	Member Typedef Documentation	517
16.223.1.1	value	517
16.224	ModeTraits< Givaro::ZRing< float > > Struct Reference	517
16.224.1	Member Typedef Documentation	517
16.224.1.1	value	517
16.225	ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference	517
16.225.1	Member Typedef Documentation	517
16.225.1.1	value	517
16.226	ModularBalanced< T > Class Template Reference	517
16.227	ModularTag Struct Reference	518
16.227.1	Detailed Description	518

16.228 Montgomery< T > Class Template Reference . . . . .	518
16.229 need_field_characteristic< Field > Struct Template Reference . . . . .	518
16.229.1 Field Documentation . . . . .	518
16.229.1.1 value . . . . .	518
16.230 need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference . . . . .	518
16.230.1 Field Documentation . . . . .	518
16.230.1.1 value . . . . .	518
16.231 need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference . . . . .	518
16.231.1 Field Documentation . . . . .	519
16.231.1.1 value . . . . .	519
16.232 NoSimd< T > Struct Template Reference . . . . .	519
16.232.1 Member Typedef Documentation . . . . .	519
16.232.1.1 vect_t . . . . .	519
16.232.1.2 scalar_t . . . . .	519
16.232.2 Member Function Documentation . . . . .	519
16.232.2.1 type_string() . . . . .	519
16.232.2.2 valid() . . . . .	519
16.232.2.3 compliant() . . . . .	520
16.232.3 Field Documentation . . . . .	520
16.232.3.1 vect_size . . . . .	520
16.233 Parallel< C, P > Struct Template Reference . . . . .	520
16.233.1 Member Typedef Documentation . . . . .	520
16.233.1.1 Cut . . . . .	520
16.233.1.2 Param . . . . .	520
16.233.2 Constructor & Destructor Documentation . . . . .	520
16.233.2.1 Parallel() . . . . .	520
16.233.3 Member Function Documentation . . . . .	520
16.233.3.1 numthreads() . . . . .	521
16.233.3.2 set_numthreads() . . . . .	521
16.233.4 Friends And Related Function Documentation . . . . .	521
16.233.4.1 operator<< . . . . .	521
16.234 RNSInteger< RNS >::RandIter Class Reference . . . . .	521
16.234.1 Constructor & Destructor Documentation . . . . .	521
16.234.1.1 RandIter() . . . . .	521
16.234.2 Member Function Documentation . . . . .	521
16.234.2.1 random() [1/2] . . . . .	522
16.234.2.2 random() [2/2] . . . . .	522
16.234.2.3 operator>() [1/2] . . . . .	522
16.234.2.4 operator>() [2/2] . . . . .	522
16.234.2.5 ring() . . . . .	522
16.235 RNSIntegerMod< RNS >::RandIter Class Reference . . . . .	522
16.235.1 Constructor & Destructor Documentation . . . . .	522

16.235.1.1 RandIter()	523
16.235.2 Member Function Documentation	523
16.235.2.1 random() [1/2]	523
16.235.2.2 random() [2/2]	523
16.235.2.3 operator>() [1/2]	523
16.235.2.4 operator>() [2/2]	523
16.235.2.5 ring()	523
16.236 readMyMachineType< Field, T > Struct Template Reference	523
16.236.1 Member Typedef Documentation	523
16.236.1.1 Element	524
16.236.1.2 Element_ptr	524
16.236.2 Member Function Documentation	524
16.236.2.1 operator>()	524
16.237 readMyMachineType< Field, mpz_t > Struct Template Reference	524
16.237.1 Member Typedef Documentation	524
16.237.1.1 Element	524
16.237.1.2 Element_ptr	524
16.237.2 Member Function Documentation	524
16.237.2.1 operator>()	525
16.238 Recursive Struct Reference	525
16.239 Recursive Struct Reference	525
16.240 rint< K > Class Template Reference	525
16.241 rns_double Struct Reference	525
16.241.1 Member Typedef Documentation	526
16.241.1.1 integer	526
16.241.1.2 ModField	526
16.241.1.3 BasisElement	526
16.241.1.4 Element	526
16.241.1.5 Element_ptr	527
16.241.1.6 ConstElement_ptr	527
16.241.2 Constructor & Destructor Documentation	527
16.241.2.1 rns_double() [1/4]	527
16.241.2.2 rns_double() [2/4]	527
16.241.2.3 rns_double() [3/4]	527
16.241.2.4 rns_double() [4/4]	527
16.241.3 Member Function Documentation	527
16.241.3.1 precompute_cst()	527
16.241.3.2 init() [1/3]	527
16.241.3.3 init() [2/3]	528
16.241.3.4 init_transpose()	528
16.241.3.5 convert() [1/2]	528
16.241.3.6 convert_transpose()	528

16.241.3.7 reduce()	529
16.241.3.8 init() [3/3]	529
16.241.3.9 convert() [2/2]	529
16.241.4 Field Documentation	529
16.241.4.1 _basis	529
16.241.4.2 _basisMax	529
16.241.4.3 _negbasis	529
16.241.4.4 _invbasis	529
16.241.4.5 _field_rns	530
16.241.4.6 _M	530
16.241.4.7 _Mi	530
16.241.4.8 _MMi	530
16.241.4.9 _crt_in	530
16.241.4.10 _crt_out	530
16.241.4.11 _size	530
16.241.4.12 _pbits	530
16.241.4.13 _ldm	530
16.241.4.14 _mi_sum	530
16.242 rns_double_elt Struct Reference	530
16.242.1 Constructor & Destructor Documentation	531
16.242.1.1 rns_double_elt() [1/3]	531
16.242.1.2 ~rns_double_elt()	531
16.242.1.3 rns_double_elt() [2/3]	531
16.242.1.4 rns_double_elt() [3/3]	531
16.242.2 Member Function Documentation	531
16.242.2.1 operator&() [1/2]	531
16.242.2.2 operator&() [2/2]	532
16.242.3 Field Documentation	532
16.242.3.1 _ptr	532
16.242.3.2 _stride	532
16.242.3.3 _alloc	532
16.243 rns_double_elt_cstptr Struct Reference	532
16.243.1 Constructor & Destructor Documentation	533
16.243.1.1 rns_double_elt_cstptr() [1/5]	533
16.243.1.2 rns_double_elt_cstptr() [2/5]	533
16.243.1.3 rns_double_elt_cstptr() [3/5]	533
16.243.1.4 rns_double_elt_cstptr() [4/5]	533
16.243.1.5 rns_double_elt_cstptr() [5/5]	533
16.243.2 Member Function Documentation	533
16.243.2.1 operator&() [1/2]	533
16.243.2.2 operator*()	533
16.243.2.3 operator[]() [1/2]	533

16.243.2.4 operator[]() [2/2]	534
16.243.2.5 operator++()	534
16.243.2.6 operator--()	534
16.243.2.7 operator+()	534
16.243.2.8 operator-()	534
16.243.2.9 operator+=()	534
16.243.2.10 operator-=()	534
16.243.2.11 operator=()	534
16.243.2.12 operator<()	534
16.243.2.13 operator"!=(())	534
16.243.2.14 operator&() [2/2]	534
16.243.3 Field Documentation	535
16.243.3.1 other	535
16.243.3.2 _ptr	535
16.243.3.3 _stride	535
16.243.3.4 _alloc	535
16.244 rns_double_elt_ptr Struct Reference	535
16.244.1 Constructor & Destructor Documentation	536
16.244.1.1 rns_double_elt_ptr() [1/5]	536
16.244.1.2 rns_double_elt_ptr() [2/5]	536
16.244.1.3 rns_double_elt_ptr() [3/5]	536
16.244.1.4 rns_double_elt_ptr() [4/5]	536
16.244.1.5 rns_double_elt_ptr() [5/5]	536
16.244.2 Member Function Documentation	536
16.244.2.1 operator&() [1/2]	536
16.244.2.2 operator*()	536
16.244.2.3 operator[]() [1/2]	536
16.244.2.4 operator[]() [2/2]	537
16.244.2.5 operator++()	537
16.244.2.6 operator--()	537
16.244.2.7 operator+()	537
16.244.2.8 operator-()	537
16.244.2.9 operator+=()	537
16.244.2.10 operator-=()	537
16.244.2.11 operator=()	537
16.244.2.12 operator<()	537
16.244.2.13 operator"!=(())	537
16.244.2.14 operator&() [2/2]	537
16.244.3 Field Documentation	538
16.244.3.1 other	538
16.244.3.2 _ptr	538
16.244.3.3 _stride	538

16.244.3.4 _alloc . . . . .	538
16.245 rns_double_extended Struct Reference . . . . .	538
16.245.1 Member Typedef Documentation . . . . .	539
16.245.1.1 integer . . . . .	539
16.245.1.2 ModField . . . . .	539
16.245.1.3 BasisElement . . . . .	539
16.245.1.4 Element . . . . .	539
16.245.1.5 Element_ptr . . . . .	539
16.245.1.6 ConstElement_ptr . . . . .	539
16.245.2 Constructor & Destructor Documentation . . . . .	539
16.245.2.1 rns_double_extended() [1/3] . . . . .	540
16.245.2.2 rns_double_extended() [2/3] . . . . .	540
16.245.2.3 rns_double_extended() [3/3] . . . . .	540
16.245.3 Member Function Documentation . . . . .	540
16.245.3.1 precompute_cst() . . . . .	540
16.245.3.2 init() [1/3] . . . . .	540
16.245.3.3 init() [2/3] . . . . .	540
16.245.3.4 convert() [1/2] . . . . .	541
16.245.3.5 init() [3/3] . . . . .	541
16.245.3.6 convert() [2/2] . . . . .	541
16.245.3.7 reduce() . . . . .	541
16.245.4 Field Documentation . . . . .	541
16.245.4.1 _basis . . . . .	541
16.245.4.2 _basisMax . . . . .	541
16.245.4.3 _negbasis . . . . .	541
16.245.4.4 _invbasis . . . . .	542
16.245.4.5 _field_rns . . . . .	542
16.245.4.6 _M . . . . .	542
16.245.4.7 _Mi . . . . .	542
16.245.4.8 _MMi . . . . .	542
16.245.4.9 _crt_in . . . . .	542
16.245.4.10 _crt_out . . . . .	542
16.245.4.11 _size . . . . .	542
16.245.4.12 _pbits . . . . .	542
16.245.4.13 _ldm . . . . .	542
16.246 RNSElementTag Struct Reference . . . . .	542
16.246.1 Detailed Description . . . . .	543
16.247 RNSInteger< RNS > Class Template Reference . . . . .	543
16.247.1 Member Typedef Documentation . . . . .	544
16.247.1.1 BasisElement . . . . .	544
16.247.1.2 integer . . . . .	544
16.247.1.3 Element . . . . .	544

16.247.1.4	Element_ptr	544
16.247.1.5	ConstElement_ptr	544
16.247.2	Constructor & Destructor Documentation	544
16.247.2.1	RNSInteger() [1/2]	544
16.247.2.2	RNSInteger() [2/2]	544
16.247.3	Member Function Documentation	544
16.247.3.1	rns()	544
16.247.3.2	size()	544
16.247.3.3	isOne()	545
16.247.3.4	isMOne()	545
16.247.3.5	isZero()	545
16.247.3.6	characteristic()	545
16.247.3.7	cardinality()	545
16.247.3.8	init() [1/2]	545
16.247.3.9	init() [2/2]	545
16.247.3.10	reduce() [1/2]	545
16.247.3.11	reduce() [2/2]	545
16.247.3.12	convert()	546
16.247.3.13	assign()	546
16.247.3.14	write() [1/2]	546
16.247.3.15	write() [2/2]	546
16.247.4	Field Documentation	546
16.247.4.1	_rns	546
16.247.4.2	one	546
16.247.4.3	mOne	546
16.247.4.4	zero	546
16.248	RNSIntegerMod< RNS > Class Template Reference	546
16.248.1	Member Typedef Documentation	548
16.248.1.1	Element	548
16.248.1.2	Element_ptr	548
16.248.1.3	ConstElement_ptr	548
16.248.1.4	BasisElement	548
16.248.1.5	ModField	548
16.248.1.6	integer	548
16.248.2	Constructor & Destructor Documentation	548
16.248.2.1	RNSIntegerMod()	548
16.248.3	Member Function Documentation	548
16.248.3.1	rns()	548
16.248.3.2	delayed()	549
16.248.3.3	size()	549
16.248.3.4	isOne()	549
16.248.3.5	isMOne()	549

16.248.3.6	isZero()	549
16.248.3.7	characteristic() [1/2]	549
16.248.3.8	characteristic() [2/2]	549
16.248.3.9	cardinality() [1/2]	549
16.248.3.10	cardinality() [2/2]	549
16.248.3.11	minElement()	549
16.248.3.12	maxElement()	549
16.248.3.13	init() [1/3]	550
16.248.3.14	init() [2/3]	550
16.248.3.15	reduce() [1/2]	550
16.248.3.16	reduce() [2/2]	550
16.248.3.17	init() [3/3]	550
16.248.3.18	convert()	550
16.248.3.19	assign()	550
16.248.3.20	add()	550
16.248.3.21	sub()	550
16.248.3.22	neg()	551
16.248.3.23	mul()	551
16.248.3.24	axpyin()	551
16.248.3.25	inv()	551
16.248.3.26	areEqual()	551
16.248.3.27	write() [1/2]	551
16.248.3.28	write() [2/2]	551
16.248.3.29	reduce_modp() [1/2]	551
16.248.3.30	write_matrix()	552
16.248.3.31	write_matrix_long()	552
16.248.3.32	reduce_modp() [2/2]	552
16.248.3.33	reduce_modp_rnsmajor()	552
16.248.4	Field Documentation	552
16.248.4.1	_p	552
16.248.4.2	_Mi_modp_rns	552
16.248.4.3	_iM_modp_rns	552
16.248.4.4	_rns	552
16.248.4.5	_F	553
16.248.4.6	_RNSdelayed	553
16.248.4.7	one	553
16.248.4.8	mOne	553
16.248.4.9	zero	553
16.249	rnsRandIter< RNS > Class Template Reference	553
16.249.1	Constructor & Destructor Documentation	553
16.249.1.1	rnsRandIter()	553
16.249.2	Member Function Documentation	554



16.249.2.1 random() [1/2]	554
16.249.2.2 operator>() [1/2]	554
16.249.2.3 operator>() [2/2]	554
16.249.2.4 random() [2/2]	554
16.249.2.5 ring()	554
16.250 Row Struct Reference	554
16.251 rint< K > Class Template Reference	554
16.252 ScalFunctions< Element, Enable > Struct Template Reference	554
16.253 ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type > Struct Template Reference	555
16.253.1 Member Function Documentation	555
16.253.1.1 zero()	555
16.253.1.2 vand()	555
16.253.1.3 vor()	555
16.253.1.4 vxor()	556
16.253.1.5 vandnot()	556
16.253.1.6 ceil()	556
16.253.1.7 floor()	556
16.253.1.8 round()	556
16.253.1.9 add()	556
16.253.1.10 addin()	556
16.253.1.11 sub()	556
16.253.1.12 subin()	556
16.253.1.13 mul()	557
16.253.1.14 mulin()	557
16.253.1.15 div()	557
16.253.1.16 fmadd()	557
16.253.1.17 fmaddin()	557
16.253.1.18 fmsub()	557
16.253.1.19 fmubin()	557
16.253.1.20 fnmadd()	557
16.253.1.21 fnmaddin()	558
16.253.1.22 lesser()	558
16.253.1.23 lesser_eq()	558
16.253.1.24 greater()	558
16.253.1.25 greater_eq()	558
16.253.1.26 eq()	558
16.254 ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type > Struct Template Reference	558
16.254.1 Member Function Documentation	559
16.254.1.1 zero()	559
16.254.1.2 round()	559
16.254.1.3 vand()	559

16.254.1.4 vor()	560
16.254.1.5 vxor()	560
16.254.1.6 vandnot()	560
16.254.1.7 add()	560
16.254.1.8 addin()	560
16.254.1.9 sub()	560
16.254.1.10 subin()	560
16.254.1.11 mul()	560
16.254.1.12 mullo()	561
16.254.1.13 mulhi()	561
16.254.1.14 mulx()	561
16.254.1.15 fmadd()	561
16.254.1.16 fmaddin()	561
16.254.1.17 fmaddx()	561
16.254.1.18 fmaddxin()	561
16.254.1.19 fmsub()	561
16.254.1.20 fmsubin()	562
16.254.1.21 fmsubx()	562
16.254.1.22 fmsubxin()	562
16.254.1.23 fnmadd()	562
16.254.1.24 fnmaddin()	562
16.254.1.25 fnmaddx()	562
16.254.1.26 fnmaddxin()	562
16.254.1.27 sra() [1/2]	562
16.254.1.28 sra() [2/2]	563
16.254.1.29 srl()	563
16.254.1.30 sll()	563
16.254.1.31 lesser()	563
16.254.1.32 lesser_eq()	563
16.254.1.33 greater()	563
16.254.1.34 greater_eq()	563
16.254.1.35 eq()	563
16.255 Sequential Struct Reference	564
16.255.1 Constructor & Destructor Documentation	564
16.255.1.1 Sequential() [1/3]	564
16.255.1.2 Sequential() [2/3]	564
16.255.1.3 Sequential() [3/3]	564
16.255.2 Member Function Documentation	564
16.255.2.1 numthreads()	564
16.255.3 Friends And Related Function Documentation	564
16.255.3.1 operator<<	564
16.256 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference	565

16.257 Simd128_impl< true, false, true, 4 > Struct Reference . . . . .	565
16.257.1 Member Function Documentation . . . . .	565
16.257.1.1 type_string() . . . . .	565
16.258 Simd128_impl< true, false, true, 8 > Struct Reference . . . . .	565
16.258.1 Member Function Documentation . . . . .	565
16.258.1.1 type_string() . . . . .	565
16.259 Simd128_impl< true, true, false, 2 > Struct Reference . . . . .	566
16.259.1 Member Typedef Documentation . . . . .	567
16.259.1.1 scalar_t . . . . .	567
16.259.1.2 vect_t . . . . .	567
16.259.2 Member Function Documentation . . . . .	567
16.259.2.1 set1() . . . . .	568
16.259.2.2 set() . . . . .	568
16.259.2.3 gather() . . . . .	568
16.259.2.4 load() . . . . .	568
16.259.2.5 loadu() . . . . .	568
16.259.2.6 store() . . . . .	568
16.259.2.7 storeu() . . . . .	568
16.259.2.8 stream() . . . . .	568
16.259.2.9 sra() . . . . .	569
16.259.2.10 greater() . . . . .	569
16.259.2.11 lesser() . . . . .	569
16.259.2.12 greater_eq() . . . . .	569
16.259.2.13 lesser_eq() . . . . .	569
16.259.2.14 mulhi() . . . . .	569
16.259.2.15 mulx() . . . . .	569
16.259.2.16 fmaddx() . . . . .	569
16.259.2.17 fmaddxin() . . . . .	569
16.259.2.18 fnmaddx() . . . . .	570
16.259.2.19 fnmaddxin() . . . . .	570
16.259.2.20 fmsubx() . . . . .	570
16.259.2.21 fmsubxin() . . . . .	570
16.259.2.22 hadd_to_scal() . . . . .	570
16.259.2.23 valid() . . . . .	570
16.259.2.24 compliant() . . . . .	570
16.259.2.25 sll() . . . . .	570
16.259.2.26 srl() . . . . .	571
16.259.2.27 shuffle() . . . . .	571
16.259.2.28 unpacklo() . . . . .	571
16.259.2.29 unpackhi() . . . . .	571
16.259.2.30 blend() . . . . .	571
16.259.2.31 add() . . . . .	571

16.259.2.32 addin()	571
16.259.2.33 sub()	571
16.259.2.34 subin()	571
16.259.2.35 mullo()	572
16.259.2.36 mul()	572
16.259.2.37 fmadd()	572
16.259.2.38 fmaddin()	572
16.259.2.39 fnmadd()	572
16.259.2.40 fnmaddin()	572
16.259.2.41 fmsub()	572
16.259.2.42 fmsubin()	572
16.259.2.43 eq()	573
16.259.2.44 round()	573
16.259.2.45 mod()	573
16.259.2.46 type_string()	573
16.259.2.47 zero()	573
16.259.2.48 sll128()	573
16.259.2.49 srl128()	573
16.259.2.50 vand()	573
16.259.2.51 vor()	574
16.259.2.52 vxor()	574
16.259.2.53 vandnot()	574
16.259.3 Field Documentation	574
16.259.3.1 vect_size	574
16.259.3.2 alignment	574
16.260 Simd128_impl< true, true, false, 4 > Struct Reference	574
16.260.1 Member Typedef Documentation	576
16.260.1.1 scalar_t	576
16.260.1.2 vect_t	576
16.260.2 Member Function Documentation	576
16.260.2.1 set1()	576
16.260.2.2 set()	576
16.260.2.3 gather()	576
16.260.2.4 load()	576
16.260.2.5 loadu()	577
16.260.2.6 store()	577
16.260.2.7 storeu()	577
16.260.2.8 stream()	577
16.260.2.9 sra()	577
16.260.2.10 greater()	577
16.260.2.11 lesser()	577
16.260.2.12 greater_eq()	577

16.260.2.13 lesser_eq()	577
16.260.2.14 mulhi()	578
16.260.2.15 mulx()	578
16.260.2.16 fmaddx()	578
16.260.2.17 fmaddxin()	578
16.260.2.18 fnmaddx()	578
16.260.2.19 fnmaddxin()	578
16.260.2.20 fmsubx()	578
16.260.2.21 fmsubxin()	578
16.260.2.22 hadd_to_scal()	579
16.260.2.23 valid()	579
16.260.2.24 compliant()	579
16.260.2.25 sll()	579
16.260.2.26 srl()	579
16.260.2.27 shuffle()	579
16.260.2.28 unpacklo()	579
16.260.2.29 unpackhi()	579
16.260.2.30 blend()	579
16.260.2.31 add()	580
16.260.2.32 addin()	580
16.260.2.33 sub()	580
16.260.2.34 subin()	580
16.260.2.35 mullo()	580
16.260.2.36 mul()	580
16.260.2.37 fmadd()	580
16.260.2.38 fmaddin()	580
16.260.2.39 fnmadd()	581
16.260.2.40 fnmaddin()	581
16.260.2.41 fmsub()	581
16.260.2.42 fmsubin()	581
16.260.2.43 eq()	581
16.260.2.44 round()	581
16.260.2.45 mod()	581
16.260.2.46 type_string()	582
16.260.2.47 zero()	582
16.260.2.48 sll128()	582
16.260.2.49 srl128()	582
16.260.2.50 vand()	582
16.260.2.51 vor()	582
16.260.2.52 vxor()	582
16.260.2.53 vandnot()	582
16.260.3 Field Documentation	582

16.260.3.1 vect_size	582
16.260.3.2 alignment	583
16.261 Simd128_impl< true, true, false, 8 > Struct Reference	583
16.261.1 Member Typedef Documentation	584
16.261.1.1 scalar_t	585
16.261.1.2 vect_t	585
16.261.2 Member Function Documentation	585
16.261.2.1 set1()	585
16.261.2.2 set()	585
16.261.2.3 gather()	585
16.261.2.4 load()	585
16.261.2.5 loadu()	585
16.261.2.6 store()	585
16.261.2.7 storeu()	585
16.261.2.8 stream()	586
16.261.2.9 sra()	586
16.261.2.10 greater()	586
16.261.2.11 lesser()	586
16.261.2.12 greater_eq()	586
16.261.2.13 lesser_eq()	586
16.261.2.14 mullo()	586
16.261.2.15 mulx()	586
16.261.2.16 fmaddx()	586
16.261.2.17 fmaddxin()	587
16.261.2.18 fnmaddx()	587
16.261.2.19 fnmaddxin()	587
16.261.2.20 fmsubx()	587
16.261.2.21 fmsubxin() [1/2]	587
16.261.2.22 hadd_to_scal()	587
16.261.2.23 valid()	587
16.261.2.24 compliant()	588
16.261.2.25 get()	588
16.261.2.26 sll()	588
16.261.2.27 srl()	588
16.261.2.28 shuffle()	588
16.261.2.29 unpacklo()	588
16.261.2.30 unpackhi()	588
16.261.2.31 blend()	588
16.261.2.32 add()	588
16.261.2.33 addin()	589
16.261.2.34 sub()	589
16.261.2.35 subin()	589

16.261.2.36 mul()	589
16.261.2.37 fmadd()	589
16.261.2.38 fmaddin()	589
16.261.2.39 fnmadd()	589
16.261.2.40 fnmaddin()	589
16.261.2.41 fmsub()	590
16.261.2.42 fmsubin()	590
16.261.2.43 fmsubxin() [2/2]	590
16.261.2.44 eq()	590
16.261.2.45 round()	590
16.261.2.46 mask_high()	590
16.261.2.47 mulhi_fast()	590
16.261.2.48 mod()	590
16.261.2.49 signbits()	591
16.261.2.50 type_string()	591
16.261.2.51 zero()	591
16.261.2.52 sll128()	591
16.261.2.53 srl128()	591
16.261.2.54 vand()	591
16.261.2.55 vor()	591
16.261.2.56 vxor()	591
16.261.2.57 vandnot()	591
16.261.3 Field Documentation	592
16.261.3.1 vect_size	592
16.261.3.2 alignment	592
16.262 Simd128_impl< true, true, true, 2 > Struct Reference	592
16.262.1 Member Typedef Documentation	593
16.262.1.1 vect_t	594
16.262.1.2 scalar_t	594
16.262.2 Member Function Documentation	594
16.262.2.1 valid()	594
16.262.2.2 compliant()	594
16.262.2.3 set1()	594
16.262.2.4 set()	594
16.262.2.5 gather()	594
16.262.2.6 load()	594
16.262.2.7 loadu()	595
16.262.2.8 store()	595
16.262.2.9 storeu()	595
16.262.2.10 stream()	595
16.262.2.11 sll()	595
16.262.2.12 srl()	595

16.262.2.13 sra()	595
16.262.2.14 shuffle()	595
16.262.2.15 unpacklo()	595
16.262.2.16 unpackhi()	596
16.262.2.17 blend()	596
16.262.2.18 add()	596
16.262.2.19 addin()	596
16.262.2.20 sub()	596
16.262.2.21 subin()	596
16.262.2.22 mullo()	596
16.262.2.23 mul()	596
16.262.2.24 mulhi()	597
16.262.2.25 mulx()	597
16.262.2.26 fmadd()	597
16.262.2.27 fmaddin()	597
16.262.2.28 fmaddx()	597
16.262.2.29 fmaddxin()	597
16.262.2.30 fnmadd()	597
16.262.2.31 fnmaddin()	597
16.262.2.32 fnmaddx()	598
16.262.2.33 fnmaddxin()	598
16.262.2.34 fmsub()	598
16.262.2.35 fmsubin()	598
16.262.2.36 fmsubx()	598
16.262.2.37 fmsubxin()	598
16.262.2.38 eq()	598
16.262.2.39 greater()	599
16.262.2.40 lesser()	599
16.262.2.41 greater_eq()	599
16.262.2.42 lesser_eq()	599
16.262.2.43 hadd_to_scal()	599
16.262.2.44 round()	599
16.262.2.45 mod()	599
16.262.2.46 type_string()	599
16.262.2.47 zero()	600
16.262.2.48 sll128()	600
16.262.2.49 srl128()	600
16.262.2.50 vand()	600
16.262.2.51 vor()	600
16.262.2.52 vxor()	600
16.262.2.53 vandnot()	600
16.262.3 Field Documentation	600



16.262.3.1 vect_size . . . . .	600
16.262.3.2 alignment . . . . .	600
16.263 Simd128_impl< true, true, true, 4 > Struct Reference . . . . .	601
16.263.1 Member Typedef Documentation . . . . .	602
16.263.1.1 vect_t . . . . .	602
16.263.1.2 scalar_t . . . . .	602
16.263.2 Member Function Documentation . . . . .	602
16.263.2.1 valid() . . . . .	603
16.263.2.2 compliant() . . . . .	603
16.263.2.3 set1() . . . . .	603
16.263.2.4 set() . . . . .	603
16.263.2.5 gather() . . . . .	603
16.263.2.6 load() . . . . .	603
16.263.2.7 loadu() . . . . .	603
16.263.2.8 store() . . . . .	603
16.263.2.9 storeu() . . . . .	603
16.263.2.10 stream() . . . . .	604
16.263.2.11 sll() . . . . .	604
16.263.2.12 srl() . . . . .	604
16.263.2.13 sra() . . . . .	604
16.263.2.14 shuffle() . . . . .	604
16.263.2.15 unpacklo() . . . . .	604
16.263.2.16 unpackhi() . . . . .	604
16.263.2.17 blend() . . . . .	604
16.263.2.18 add() . . . . .	604
16.263.2.19 addin() . . . . .	605
16.263.2.20 sub() . . . . .	605
16.263.2.21 subin() . . . . .	605
16.263.2.22 mullo() . . . . .	605
16.263.2.23 mul() . . . . .	605
16.263.2.24 mulhi() . . . . .	605
16.263.2.25 mulx() . . . . .	605
16.263.2.26 fmadd() . . . . .	605
16.263.2.27 fmaddin() . . . . .	606
16.263.2.28 fmaddx() . . . . .	606
16.263.2.29 fmaddxin() . . . . .	606
16.263.2.30 fnmadd() . . . . .	606
16.263.2.31 fnmaddin() . . . . .	606
16.263.2.32 fnmaddx() . . . . .	606
16.263.2.33 fnmaddxin() . . . . .	606
16.263.2.34 fmsub() . . . . .	606
16.263.2.35 fmsubin() . . . . .	607

16.263.2.36 fmsubx()	607
16.263.2.37 fmsubxin()	607
16.263.2.38 eq()	607
16.263.2.39 greater()	607
16.263.2.40 lesser()	607
16.263.2.41 greater_eq()	607
16.263.2.42 lesser_eq()	608
16.263.2.43 hadd_to_scal()	608
16.263.2.44 round()	608
16.263.2.45 mod()	608
16.263.2.46 type_string()	608
16.263.2.47 zero()	608
16.263.2.48 sll128()	608
16.263.2.49 srl128()	608
16.263.2.50 vand()	608
16.263.2.51 vor()	609
16.263.2.52 vxor()	609
16.263.2.53 vandnot()	609
16.263.3 Field Documentation	609
16.263.3.1 vect_size	609
16.263.3.2 alignment	609
16.264 Simd128_impl< true, true, true, 8 > Struct Reference	609
16.264.1 Member Typedef Documentation	611
16.264.1.1 vect_t	611
16.264.1.2 scalar_t	611
16.264.2 Member Function Documentation	611
16.264.2.1 valid()	611
16.264.2.2 compliant()	611
16.264.2.3 set1()	611
16.264.2.4 set()	612
16.264.2.5 gather()	612
16.264.2.6 get()	612
16.264.2.7 load()	612
16.264.2.8 loadu()	612
16.264.2.9 store()	612
16.264.2.10 storeu()	612
16.264.2.11 stream()	612
16.264.2.12 sll()	612
16.264.2.13 srl()	613
16.264.2.14 sra()	613
16.264.2.15 shuffle()	613
16.264.2.16 unpacklo()	613

16.264.2.17 unpackhi()	613
16.264.2.18 blend()	613
16.264.2.19 add()	613
16.264.2.20 addin()	613
16.264.2.21 sub()	613
16.264.2.22 subin()	614
16.264.2.23 mullo()	614
16.264.2.24 mul()	614
16.264.2.25 mulx()	614
16.264.2.26 fmadd()	614
16.264.2.27 fmaddin()	614
16.264.2.28 fmaddx()	614
16.264.2.29 fmaddxin()	614
16.264.2.30 fnmadd()	615
16.264.2.31 fnmaddin()	615
16.264.2.32 fnmaddx()	615
16.264.2.33 fnmaddxin()	615
16.264.2.34 fmsub()	615
16.264.2.35 fmsubin()	615
16.264.2.36 fmsubx()	615
16.264.2.37 fmsubxin()	615
16.264.2.38 eq()	616
16.264.2.39 greater()	616
16.264.2.40 lesser()	616
16.264.2.41 greater_eq()	616
16.264.2.42 lesser_eq()	616
16.264.2.43 hadd_to_scal()	616
16.264.2.44 round()	616
16.264.2.45 mask_high()	616
16.264.2.46 mulhi_fast()	617
16.264.2.47 mod()	617
16.264.2.48 signbits()	617
16.264.2.49 type_string()	617
16.264.2.50 zero()	617
16.264.2.51 sll128()	617
16.264.2.52 srl128()	617
16.264.2.53 vand()	617
16.264.2.54 vor()	618
16.264.2.55 vxor()	618
16.264.2.56 vandnot()	618
16.264.3 Field Documentation	618
16.264.3.1 vect_size	618

16.264.3.2 alignment	618
16.265 Simd128fp_base Struct Reference	618
16.265.1 Member Function Documentation	618
16.265.1.1 type_string()	618
16.266 Simd128i_base Struct Reference	619
16.266.1 Member Typedef Documentation	619
16.266.1.1 vect_t	619
16.266.2 Member Function Documentation	619
16.266.2.1 type_string()	619
16.266.2.2 zero()	619
16.266.2.3 sll128()	619
16.266.2.4 srl128()	620
16.266.2.5 vand()	620
16.266.2.6 vor()	620
16.266.2.7 vxor()	620
16.266.2.8 vandnot()	620
16.267 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference	620
16.268 Simd256_impl< true, false, true, 4 > Struct Reference	620
16.269 Simd256_impl< true, false, true, 8 > Struct Reference	621
16.269.1 Member Typedef Documentation	622
16.269.1.1 vect_t	622
16.269.1.2 scalar_t	622
16.269.2 Member Function Documentation	622
16.269.2.1 valid()	622
16.269.2.2 compliant()	622
16.269.2.3 zero()	622
16.269.2.4 set1()	622
16.269.2.5 set()	623
16.269.2.6 gather()	623
16.269.2.7 load()	623
16.269.2.8 loadu()	623
16.269.2.9 store()	623
16.269.2.10 storeu()	623
16.269.2.11 stream()	623
16.269.2.12 unpacklo_twice()	623
16.269.2.13 unpackhi_twice()	624
16.269.2.14 blend()	624
16.269.2.15 blendv()	624
16.269.2.16 add()	624
16.269.2.17 addin()	624
16.269.2.18 sub()	624
16.269.2.19 subin()	624

16.269.2.20 mul()	624
16.269.2.21 mulin()	625
16.269.2.22 div()	625
16.269.2.23 fmadd()	625
16.269.2.24 fmaddin()	625
16.269.2.25 fnmadd()	625
16.269.2.26 fnmaddin()	625
16.269.2.27 fmsub()	625
16.269.2.28 fmsubin()	625
16.269.2.29 eq()	626
16.269.2.30 lesser()	626
16.269.2.31 lesser_eq()	626
16.269.2.32 greater()	626
16.269.2.33 greater_eq()	626
16.269.2.34 vand()	626
16.269.2.35 vor()	626
16.269.2.36 vxor()	626
16.269.2.37 vandnot()	627
16.269.2.38 floor()	627
16.269.2.39 ceil()	627
16.269.2.40 round()	627
16.269.2.41 hadd()	627
16.269.2.42 hadd_to_scal()	627
16.269.2.43 mod()	627
16.269.3 Field Documentation	627
16.269.3.1 vect_size	627
16.269.3.2 alignment	628
16.270 Simd256_impl< true, true, false, 2 > Struct Reference	628
16.270.1 Member Typedef Documentation	629
16.270.1.1 scalar_t	629
16.270.1.2 simdHalf	629
16.270.1.3 vect_t	630
16.270.1.4 half_t	630
16.270.2 Member Function Documentation	630
16.270.2.1 set1()	630
16.270.2.2 set()	630
16.270.2.3 gather()	630
16.270.2.4 load()	630
16.270.2.5 loadu()	630
16.270.2.6 store()	631
16.270.2.7 storeu()	631
16.270.2.8 stream()	631

16.270.2.9 sra()	631
16.270.2.10 greater()	631
16.270.2.11 lesser()	631
16.270.2.12 greater_eq()	631
16.270.2.13 lesser_eq()	631
16.270.2.14 mulhi()	631
16.270.2.15 mulx()	632
16.270.2.16 fmaddx()	632
16.270.2.17 fmaddxin()	632
16.270.2.18 fnmaddx()	632
16.270.2.19 fnmaddxin()	632
16.270.2.20 fmsubx()	632
16.270.2.21 fmsubxin()	632
16.270.2.22 hadd_to_scal()	633
16.270.2.23 valid()	633
16.270.2.24 compliant()	633
16.270.2.25 sll()	633
16.270.2.26 srl()	633
16.270.2.27 shuffle()	633
16.270.2.28 unpacklo_twice()	633
16.270.2.29 unpackhi_twice()	633
16.270.2.30 unpacklo()	633
16.270.2.31 unpackhi()	634
16.270.2.32 unpacklohi()	634
16.270.2.33 blend_twice()	634
16.270.2.34 add()	634
16.270.2.35 addin()	634
16.270.2.36 sub()	634
16.270.2.37 subin()	634
16.270.2.38 mullo()	634
16.270.2.39 mul()	635
16.270.2.40 fmadd()	635
16.270.2.41 fmaddin()	635
16.270.2.42 fnmadd()	635
16.270.2.43 fnmaddin()	635
16.270.2.44 fmsub()	635
16.270.2.45 fmsubin()	635
16.270.2.46 eq()	635
16.270.2.47 round()	636
16.270.2.48 mod()	636
16.270.2.49 type_string()	636
16.270.2.50 zero()	636

16.270.3 Field Documentation	636
16.270.3.1 vect_size	636
16.270.3.2 alignment	636
16.271 Simd256_impl< true, true, false, 4 > Struct Reference	636
16.271.1 Member Typedef Documentation	639
16.271.1.1 scalar_t [1/2]	639
16.271.1.2 simdHalf [1/2]	639
16.271.1.3 scalar_t [2/2]	639
16.271.1.4 simdHalf [2/2]	639
16.271.1.5 vect_t [1/2]	640
16.271.1.6 vect_t [2/2]	640
16.271.1.7 half_t [1/2]	640
16.271.1.8 half_t [2/2]	640
16.271.2 Member Function Documentation	640
16.271.2.1 set1() [1/2]	640
16.271.2.2 set() [1/3]	640
16.271.2.3 gather() [1/2]	640
16.271.2.4 load() [1/2]	640
16.271.2.5 loadu() [1/2]	640
16.271.2.6 store() [1/2]	641
16.271.2.7 storeu() [1/2]	641
16.271.2.8 stream() [1/2]	641
16.271.2.9 sra() [1/2]	641
16.271.2.10 greater() [1/2]	641
16.271.2.11 lesser() [1/2]	641
16.271.2.12 greater_eq() [1/2]	641
16.271.2.13 lesser_eq() [1/2]	641
16.271.2.14 mulhi() [1/2]	641
16.271.2.15 mulx() [1/2]	642
16.271.2.16 fmaddx() [1/2]	642
16.271.2.17 fmaddxin() [1/2]	642
16.271.2.18 fnmaddx() [1/2]	642
16.271.2.19 fnmaddxin() [1/2]	642
16.271.2.20 fmsubx() [1/2]	642
16.271.2.21 fmsubxin() [1/2]	642
16.271.2.22 hadd_to_scal() [1/2]	643
16.271.2.23 set1() [2/2]	643
16.271.2.24 set() [2/3]	643
16.271.2.25 gather() [2/2]	643
16.271.2.26 load() [2/2]	643
16.271.2.27 loadu() [2/2]	643
16.271.2.28 store() [2/2]	643

16.271.2.29 storeu() [2/2]	643
16.271.2.30 stream() [2/2]	644
16.271.2.31 sra() [2/2]	644
16.271.2.32 greater() [2/2]	644
16.271.2.33 lesser() [2/2]	644
16.271.2.34 greater_eq() [2/2]	644
16.271.2.35 lesser_eq() [2/2]	644
16.271.2.36 mulhi() [2/2]	644
16.271.2.37 mulx() [2/2]	644
16.271.2.38 fmaddx() [2/2]	644
16.271.2.39 fmaddxin() [2/2]	645
16.271.2.40 fnmaddx() [2/2]	645
16.271.2.41 fnmaddxin() [2/2]	645
16.271.2.42 fmsubx() [2/2]	645
16.271.2.43 fmsubxin() [2/2]	645
16.271.2.44 hadd_to_scal() [2/2]	645
16.271.2.45 valid() [1/2]	645
16.271.2.46 valid() [2/2]	646
16.271.2.47 compliant() [1/2]	646
16.271.2.48 compliant() [2/2]	646
16.271.2.49 set() [3/3]	646
16.271.2.50 sll() [1/2]	646
16.271.2.51 sll() [2/2]	646
16.271.2.52 srl() [1/2]	646
16.271.2.53 srl() [2/2]	647
16.271.2.54 shuffle_twice() [1/2]	647
16.271.2.55 shuffle_twice() [2/2]	647
16.271.2.56 shuffle() [1/2]	647
16.271.2.57 shuffle() [2/2]	647
16.271.2.58 unpacklo_twice()	647
16.271.2.59 unpackhi_twice()	647
16.271.2.60 unpacklo()	647
16.271.2.61 unpackhi()	647
16.271.2.62 unpacklohi()	648
16.271.2.63 blend()	648
16.271.2.64 add() [1/2]	648
16.271.2.65 add() [2/2]	648
16.271.2.66 addin() [1/2]	648
16.271.2.67 addin() [2/2]	648
16.271.2.68 sub() [1/2]	648
16.271.2.69 sub() [2/2]	648
16.271.2.70 subin() [1/2]	649



16.271.2.71 subin() [2/2]	649
16.271.2.72 mullo() [1/2]	649
16.271.2.73 mullo() [2/2]	649
16.271.2.74 mul() [1/2]	649
16.271.2.75 mul() [2/2]	649
16.271.2.76 fmadd() [1/2]	649
16.271.2.77 fmadd() [2/2]	649
16.271.2.78 fmaddin() [1/2]	650
16.271.2.79 fmaddin() [2/2]	650
16.271.2.80 fnmadd() [1/2]	650
16.271.2.81 fnmadd() [2/2]	650
16.271.2.82 fnmaddin() [1/2]	650
16.271.2.83 fnmaddin() [2/2]	650
16.271.2.84 fmsub() [1/2]	650
16.271.2.85 fmsub() [2/2]	650
16.271.2.86 fmsubin() [1/2]	651
16.271.2.87 fmsubin() [2/2]	651
16.271.2.88 eq() [1/2]	651
16.271.2.89 eq() [2/2]	651
16.271.2.90 round() [1/2]	651
16.271.2.91 round() [2/2]	651
16.271.2.92 mod() [1/2]	651
16.271.2.93 mod() [2/2]	652
16.271.2.94 type_string() [1/2]	652
16.271.2.95 type_string() [2/2]	652
16.271.2.96 zero() [1/2]	652
16.271.2.97 zero() [2/2]	652
16.271.2.98 vor()	652
16.271.2.99 vxor()	652
16.271.2.100 vand()	652
16.271.2.101 vandnot()	652
16.271.3 Field Documentation	653
16.271.3.1 vect_size	653
16.271.3.2 alignment	653
16.272 Simd256_impl< true, true, false, 8 > Struct Reference	653
16.272.1 Member Typedef Documentation	655
16.272.1.1 scalar_t	655
16.272.1.2 simdHalf	655
16.272.1.3 vect_t	655
16.272.1.4 half_t	655
16.272.2 Member Function Documentation	655
16.272.2.1 set1()	655

16.272.2.2 set()	655
16.272.2.3 gather()	655
16.272.2.4 load()	655
16.272.2.5 loadu()	656
16.272.2.6 store()	656
16.272.2.7 storeu()	656
16.272.2.8 stream()	656
16.272.2.9 sra()	656
16.272.2.10 greater()	656
16.272.2.11 lesser()	656
16.272.2.12 greater_eq()	656
16.272.2.13 lesser_eq()	656
16.272.2.14 mullo()	657
16.272.2.15 mulx()	657
16.272.2.16 fmaddx()	657
16.272.2.17 fmaddxin()	657
16.272.2.18 fnmaddx()	657
16.272.2.19 fnmaddxin()	657
16.272.2.20 fmsubx()	657
16.272.2.21 fmsubxin()	657
16.272.2.22 hadd_to_scal()	658
16.272.2.23 valid()	658
16.272.2.24 compliant()	658
16.272.2.25 get()	658
16.272.2.26 sll()	658
16.272.2.27 srl()	658
16.272.2.28 shuffle()	658
16.272.2.29 unpacklo_twice()	658
16.272.2.30 unpackhi_twice()	658
16.272.2.31 unpacklo()	659
16.272.2.32 unpackhi()	659
16.272.2.33 unpacklohi()	659
16.272.2.34 blend()	659
16.272.2.35 add()	659
16.272.2.36 addin()	659
16.272.2.37 sub()	659
16.272.2.38 subin()	659
16.272.2.39 mul()	660
16.272.2.40 fmadd()	660
16.272.2.41 fmaddin()	660
16.272.2.42 fnmadd()	660
16.272.2.43 fnmaddin()	660

16.272.2.44 fmsub()	660
16.272.2.45 fmsubin()	660
16.272.2.46 eq()	660
16.272.2.47 round()	661
16.272.2.48 mask_high()	661
16.272.2.49 mulhi_fast()	661
16.272.2.50 mod()	661
16.272.2.51 signbits()	661
16.272.2.52 type_string()	661
16.272.2.53 zero()	661
16.272.3 Field Documentation	661
16.272.3.1 vect_size	661
16.272.3.2 alignment	662
16.273 Simd256_impl< true, true, true, 2 > Struct Reference	662
16.273.1 Member Typedef Documentation	663
16.273.1.1 vect_t	663
16.273.1.2 half_t	663
16.273.1.3 scalar_t	664
16.273.1.4 simdHalf	664
16.273.2 Member Function Documentation	664
16.273.2.1 valid()	664
16.273.2.2 compliant()	664
16.273.2.3 set1()	664
16.273.2.4 set()	664
16.273.2.5 gather()	664
16.273.2.6 load()	665
16.273.2.7 loadu()	665
16.273.2.8 store()	665
16.273.2.9 storeu()	665
16.273.2.10 stream()	665
16.273.2.11 sll()	665
16.273.2.12 srl()	665
16.273.2.13 sra()	665
16.273.2.14 shuffle()	665
16.273.2.15 unpacklo_twice()	666
16.273.2.16 unpackhi_twice()	666
16.273.2.17 unpacklo()	666
16.273.2.18 unpackhi()	666
16.273.2.19 unpacklohi()	666
16.273.2.20 blend_twice()	666
16.273.2.21 add()	666
16.273.2.22 addin()	666

16.273.2.23 sub()	667
16.273.2.24 subin()	667
16.273.2.25 mullo()	667
16.273.2.26 mul()	667
16.273.2.27 mulhi()	667
16.273.2.28 mulx()	667
16.273.2.29 fmadd()	667
16.273.2.30 fmaddin()	667
16.273.2.31 fmaddx()	668
16.273.2.32 fmaddxin()	668
16.273.2.33 fnmadd()	668
16.273.2.34 fnmaddin()	668
16.273.2.35 fnmaddx()	668
16.273.2.36 fnmaddxin()	668
16.273.2.37 fmsub()	668
16.273.2.38 fmsubin()	668
16.273.2.39 fmsubx()	669
16.273.2.40 fmsubxin()	669
16.273.2.41 eq()	669
16.273.2.42 greater()	669
16.273.2.43 lesser()	669
16.273.2.44 greater_eq()	669
16.273.2.45 lesser_eq()	669
16.273.2.46 hadd_to_scal()	669
16.273.2.47 round()	670
16.273.2.48 mod()	670
16.273.2.49 type_string()	670
16.273.2.50 zero()	670
16.273.3 Field Documentation	670
16.273.3.1 vect_size	670
16.273.3.2 alignment	670
16.274 Simd256_impl< true, true, true, 4 > Struct Reference	670
16.274.1 Member Typedef Documentation	673
16.274.1.1 vect_t [1/2]	673
16.274.1.2 half_t [1/2]	673
16.274.1.3 scalar_t [1/2]	673
16.274.1.4 simdHalf [1/2]	673
16.274.1.5 vect_t [2/2]	674
16.274.1.6 half_t [2/2]	674
16.274.1.7 scalar_t [2/2]	674
16.274.1.8 simdHalf [2/2]	674
16.274.2 Member Function Documentation	674

16.274.2.1 valid() [1/2]	674
16.274.2.2 compliant() [1/2]	674
16.274.2.3 set1() [1/2]	674
16.274.2.4 set() [1/2]	674
16.274.2.5 gather() [1/2]	674
16.274.2.6 load() [1/2]	675
16.274.2.7 loadu() [1/2]	675
16.274.2.8 store() [1/2]	675
16.274.2.9 storeu() [1/2]	675
16.274.2.10 stream() [1/2]	675
16.274.2.11 sll() [1/2]	675
16.274.2.12 srl() [1/2]	675
16.274.2.13 sra() [1/2]	675
16.274.2.14 shuffle_twice() [1/2]	675
16.274.2.15 shuffle() [1/2]	676
16.274.2.16 unpacklo_twice()	676
16.274.2.17 unpackhi_twice()	676
16.274.2.18 unpacklo()	676
16.274.2.19 unpackhi()	676
16.274.2.20 unpacklohi()	676
16.274.2.21 blend()	676
16.274.2.22 add() [1/2]	676
16.274.2.23 addin() [1/2]	677
16.274.2.24 sub() [1/2]	677
16.274.2.25 subin() [1/2]	677
16.274.2.26 mullo()	677
16.274.2.27 mul() [1/2]	677
16.274.2.28 mulhi() [1/2]	677
16.274.2.29 mulx() [1/2]	677
16.274.2.30 fmadd() [1/2]	677
16.274.2.31 fmaddin() [1/2]	678
16.274.2.32 fmaddx() [1/2]	678
16.274.2.33 fmaddxin() [1/2]	678
16.274.2.34 fnmadd() [1/2]	678
16.274.2.35 fnmaddin() [1/2]	678
16.274.2.36 fnmaddx() [1/2]	678
16.274.2.37 fnmaddxin() [1/2]	678
16.274.2.38 fmsub() [1/2]	678
16.274.2.39 fmsubin() [1/2]	679
16.274.2.40 fmsubx() [1/2]	679
16.274.2.41 fmsubxin() [1/2]	679
16.274.2.42 eq() [1/2]	679

16.274.2.43 greater() [1/2]	679
16.274.2.44 lesser() [1/2]	679
16.274.2.45 greater_eq() [1/2]	679
16.274.2.46 lesser_eq() [1/2]	680
16.274.2.47 hadd_to_scal() [1/2]	680
16.274.2.48 round() [1/2]	680
16.274.2.49 mod() [1/2]	680
16.274.2.50 valid() [2/2]	680
16.274.2.51 compliant() [2/2]	680
16.274.2.52 set1() [2/2]	680
16.274.2.53 set() [2/2]	680
16.274.2.54 gather() [2/2]	681
16.274.2.55 load() [2/2]	681
16.274.2.56 loadu() [2/2]	681
16.274.2.57 store() [2/2]	681
16.274.2.58 storeu() [2/2]	681
16.274.2.59 stream() [2/2]	681
16.274.2.60 sll() [2/2]	681
16.274.2.61 srl() [2/2]	682
16.274.2.62 sra() [2/2]	682
16.274.2.63 shuffle_twice() [2/2]	682
16.274.2.64 shuffle() [2/2]	682
16.274.2.65 add() [2/2]	682
16.274.2.66 addin() [2/2]	682
16.274.2.67 sub() [2/2]	682
16.274.2.68 subin() [2/2]	682
16.274.2.69 mullo() [2/2]	682
16.274.2.70 mul() [2/2]	683
16.274.2.71 mulhi() [2/2]	683
16.274.2.72 mulx() [2/2]	683
16.274.2.73 fmadd() [2/2]	683
16.274.2.74 fmaddin() [2/2]	683
16.274.2.75 fmaddx() [2/2]	683
16.274.2.76 fmaddxin() [2/2]	683
16.274.2.77 fnmadd() [2/2]	683
16.274.2.78 fnmaddin() [2/2]	684
16.274.2.79 fnmaddx() [2/2]	684
16.274.2.80 fnmaddxin() [2/2]	684
16.274.2.81 fmsub() [2/2]	684
16.274.2.82 fmsubin() [2/2]	684
16.274.2.83 fmsubx() [2/2]	684
16.274.2.84 fmsubxin() [2/2]	684

16.274.2.85 eq() [2/2]	684
16.274.2.86 greater() [2/2]	685
16.274.2.87 lesser() [2/2]	685
16.274.2.88 greater_eq() [2/2]	685
16.274.2.89 lesser_eq() [2/2]	685
16.274.2.90 hadd_to_scal() [2/2]	685
16.274.2.91 round() [2/2]	685
16.274.2.92 mod() [2/2]	685
16.274.2.93 type_string() [1/2]	685
16.274.2.94 zero() [1/2]	686
16.274.2.95 type_string() [2/2]	686
16.274.2.96 zero() [2/2]	686
16.274.2.97 vor()	686
16.274.2.98 vxor()	686
16.274.2.99 vand()	686
16.274.2.100 vandnot()	686
16.274.3 Field Documentation	686
16.274.3.1 vect_size	686
16.274.3.2 alignment	686
16.275 Simd256_impl< true, true, true, 8 > Struct Reference	687
16.275.1 Member Typedef Documentation	688
16.275.1.1 vect_t	688
16.275.1.2 half_t	688
16.275.1.3 scalar_t	689
16.275.1.4 simdHalf	689
16.275.2 Member Function Documentation	689
16.275.2.1 valid()	689
16.275.2.2 compliant()	689
16.275.2.3 set1()	689
16.275.2.4 set()	689
16.275.2.5 gather()	689
16.275.2.6 get()	689
16.275.2.7 load()	689
16.275.2.8 loadu()	690
16.275.2.9 store()	690
16.275.2.10 storeu()	690
16.275.2.11 stream()	690
16.275.2.12 sll()	690
16.275.2.13 srl()	690
16.275.2.14 sra()	690
16.275.2.15 shuffle()	690
16.275.2.16 unpacklo_twice()	690

16.275.2.17 unpackhi_twice()	691
16.275.2.18 unpacklo()	691
16.275.2.19 unpackhi()	691
16.275.2.20 unpacklohi()	691
16.275.2.21 blend()	691
16.275.2.22 add()	691
16.275.2.23 addin()	691
16.275.2.24 sub()	691
16.275.2.25 subin()	692
16.275.2.26 mullo()	692
16.275.2.27 mul()	692
16.275.2.28 mulx()	692
16.275.2.29 fmadd()	692
16.275.2.30 fmaddin()	692
16.275.2.31 fmaddx()	692
16.275.2.32 fmaddxin()	692
16.275.2.33 fnmadd()	693
16.275.2.34 fnmaddin()	693
16.275.2.35 fnmaddx()	693
16.275.2.36 fnmaddxin()	693
16.275.2.37 fmsub()	693
16.275.2.38 fmsubin()	693
16.275.2.39 fmsubx()	693
16.275.2.40 fmsubxin()	693
16.275.2.41 eq()	694
16.275.2.42 greater()	694
16.275.2.43 lesser()	694
16.275.2.44 greater_eq()	694
16.275.2.45 lesser_eq()	694
16.275.2.46 hadd_to_scal()	694
16.275.2.47 round()	694
16.275.2.48 mask_high()	694
16.275.2.49 mulhi_fast()	695
16.275.2.50 mod()	695
16.275.2.51 signbits()	695
16.275.2.52 type_string()	695
16.275.2.53 zero()	695
16.275.3 Field Documentation	695
16.275.3.1 vect_size	695
16.275.3.2 alignment	695
16.276 Simd256fp_base Struct Reference	695
16.277 Simd256i_base Struct Reference	696



16.277.1 Member Typedef Documentation . . . . .	696
16.277.1.1 vect_t . . . . .	696
16.277.2 Member Function Documentation . . . . .	696
16.277.2.1 type_string() . . . . .	696
16.277.2.2 zero() . . . . .	696
16.278 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference . . . . .	697
16.279 Simd512_impl< true, false, true, 4 > Struct Reference . . . . .	697
16.279.1 Member Function Documentation . . . . .	697
16.279.1.1 type_string() . . . . .	697
16.280 Simd512_impl< true, false, true, 8 > Struct Reference . . . . .	697
16.280.1 Member Typedef Documentation . . . . .	698
16.280.1.1 vect_t . . . . .	698
16.280.1.2 scalar_t . . . . .	699
16.280.2 Member Function Documentation . . . . .	699
16.280.2.1 valid() . . . . .	699
16.280.2.2 compliant() . . . . .	699
16.280.2.3 zero() . . . . .	699
16.280.2.4 set1() . . . . .	699
16.280.2.5 set() . . . . .	699
16.280.2.6 gather() . . . . .	699
16.280.2.7 load() . . . . .	699
16.280.2.8 loadu() . . . . .	700
16.280.2.9 store() . . . . .	700
16.280.2.10 storeu() . . . . .	700
16.280.2.11 stream() . . . . .	700
16.280.2.12 shuffle() . . . . .	700
16.280.2.13 unpacklo_twice() . . . . .	700
16.280.2.14 unpackhi_twice() . . . . .	700
16.280.2.15 blend() . . . . .	700
16.280.2.16 blendv() . . . . .	700
16.280.2.17 add() . . . . .	701
16.280.2.18 addin() . . . . .	701
16.280.2.19 sub() . . . . .	701
16.280.2.20 subin() . . . . .	701
16.280.2.21 mul() . . . . .	701
16.280.2.22 mulin() . . . . .	701
16.280.2.23 div() . . . . .	701
16.280.2.24 fmadd() . . . . .	701
16.280.2.25 fmaddin() . . . . .	702
16.280.2.26 fnmadd() . . . . .	702
16.280.2.27 fnmaddin() . . . . .	702
16.280.2.28 fmsub() . . . . .	702

16.280.2.29 fmsubin()	702
16.280.2.30 eq()	702
16.280.2.31 lesser()	702
16.280.2.32 lesser_eq()	702
16.280.2.33 greater()	703
16.280.2.34 greater_eq()	703
16.280.2.35 floor()	703
16.280.2.36 ceil()	703
16.280.2.37 round()	703
16.280.2.38 hadd()	703
16.280.2.39 hadd_to_scal()	703
16.280.2.40 type_string()	703
16.280.3 Field Documentation	703
16.280.3.1 vect_size	703
16.280.3.2 alignment	704
16.281 Simd512_impl< true, true, false, 8 > Struct Reference	704
16.281.1 Member Typedef Documentation	706
16.281.1.1 scalar_t	706
16.281.1.2 simdHalf	706
16.281.1.3 vect_t	706
16.281.1.4 half_t	706
16.281.2 Member Function Documentation	706
16.281.2.1 set1()	706
16.281.2.2 set() [1/2]	706
16.281.2.3 gather()	706
16.281.2.4 load()	706
16.281.2.5 loadu()	707
16.281.2.6 store()	707
16.281.2.7 maskstore()	707
16.281.2.8 storeu()	707
16.281.2.9 stream()	707
16.281.2.10 sra()	707
16.281.2.11 greater()	707
16.281.2.12 lesser()	707
16.281.2.13 greater_eq()	707
16.281.2.14 lesser_eq()	708
16.281.2.15 mullo()	708
16.281.2.16 mulx()	708
16.281.2.17 fmaddx()	708
16.281.2.18 fmaddxin()	708
16.281.2.19 fnmaddx()	708
16.281.2.20 fnmaddxin()	708

16.281.2.21 fmsubx()	708
16.281.2.22 fmsubxin()	709
16.281.2.23 hadd_to_scal()	709
16.281.2.24 valid()	709
16.281.2.25 compliant()	709
16.281.2.26 set() [2/2]	709
16.281.2.27 sll()	709
16.281.2.28 srl()	709
16.281.2.29 shuffle()	709
16.281.2.30 unpacklo_twice()	709
16.281.2.31 unpackhi_twice()	710
16.281.2.32 unpacklo()	710
16.281.2.33 unpackhi()	710
16.281.2.34 unpacklohi()	710
16.281.2.35 blend()	710
16.281.2.36 add()	710
16.281.2.37 addin()	710
16.281.2.38 sub()	710
16.281.2.39 subin()	711
16.281.2.40 mul()	711
16.281.2.41 fmadd()	711
16.281.2.42 fmaddin()	711
16.281.2.43 fnmadd()	711
16.281.2.44 fnmaddin()	711
16.281.2.45 fmsub()	711
16.281.2.46 fmsubin()	711
16.281.2.47 eq()	712
16.281.2.48 round()	712
16.281.2.49 mask_high()	712
16.281.2.50 mulhi_fast()	712
16.281.2.51 mod()	712
16.281.2.52 signbits()	712
16.281.2.53 type_string()	712
16.281.2.54 zero()	712
16.281.2.55 vor()	713
16.281.2.56 vxor()	713
16.281.2.57 vand()	713
16.281.2.58 vandnot()	713
16.281.3 Field Documentation	713
16.281.3.1 vect_size	713
16.281.3.2 alignment	713
16.282 Simd512_impl< true, true, true, 8 > Struct Reference	713

16.282.1 Member Typedef Documentation	715
16.282.1.1 vect_t	715
16.282.1.2 half_t	715
16.282.1.3 scalar_t	715
16.282.1.4 simdHalf	715
16.282.2 Member Function Documentation	715
16.282.2.1 valid()	716
16.282.2.2 compliant()	716
16.282.2.3 set1()	716
16.282.2.4 set() [1/2]	716
16.282.2.5 set() [2/2]	716
16.282.2.6 gather()	716
16.282.2.7 load()	716
16.282.2.8 loadu()	716
16.282.2.9 store()	717
16.282.2.10 maskstore()	717
16.282.2.11 storeu()	717
16.282.2.12 stream()	717
16.282.2.13 sll()	717
16.282.2.14 srl()	717
16.282.2.15 sra()	717
16.282.2.16 shuffle()	717
16.282.2.17 unpacklo_twice()	717
16.282.2.18 unpackhi_twice()	718
16.282.2.19 unpacklo()	718
16.282.2.20 unpackhi()	718
16.282.2.21 unpacklohi()	718
16.282.2.22 blend()	718
16.282.2.23 add()	718
16.282.2.24 addin()	718
16.282.2.25 sub()	718
16.282.2.26 subin()	719
16.282.2.27 mullo()	719
16.282.2.28 mul()	719
16.282.2.29 mulx()	719
16.282.2.30 fmadd()	719
16.282.2.31 fmaddin()	719
16.282.2.32 fmaddx()	719
16.282.2.33 fmaddxin()	719
16.282.2.34 fnmadd()	720
16.282.2.35 fnmaddin()	720
16.282.2.36 fnmaddx()	720

16.282.2.37 fnmaddxin()	720
16.282.2.38 fmsub()	720
16.282.2.39 fmsubin()	720
16.282.2.40 fmsubx()	720
16.282.2.41 fmsubxin()	720
16.282.2.42 eq()	721
16.282.2.43 greater()	721
16.282.2.44 lesser()	721
16.282.2.45 greater_eq()	721
16.282.2.46 lesser_eq()	721
16.282.2.47 hadd_to_scal()	721
16.282.2.48 round()	721
16.282.2.49 mask_high()	721
16.282.2.50 mulhi_fast()	722
16.282.2.51 mod()	722
16.282.2.52 signbits()	722
16.282.2.53 type_string()	722
16.282.2.54 zero()	722
16.282.2.55 vor()	722
16.282.2.56 vxor()	722
16.282.2.57 vand()	722
16.282.2.58 vandnot()	723
16.282.3 Field Documentation	723
16.282.3.1 vect_size	723
16.282.3.2 alignment	723
16.283 Simd512fp_base Struct Reference	723
16.283.1 Member Function Documentation	723
16.283.1.1 type_string()	723
16.284 Simd512i_base Struct Reference	723
16.284.1 Member Typedef Documentation	724
16.284.1.1 vect_t	724
16.284.2 Member Function Documentation	724
16.284.2.1 type_string()	724
16.284.2.2 zero()	724
16.284.2.3 vor()	724
16.284.2.4 vxor()	724
16.284.2.5 vand()	724
16.284.2.6 vandnot()	725
16.285 SimdChooser< T, bool, bool > Struct Template Reference	725
16.286 SimdChooser< T, false, b > Struct Template Reference	725
16.286.1 Member Typedef Documentation	725
16.286.1.1 value	725

16.287 SimdChooser< T, true, false > Struct Template Reference	725
16.287.1 Member Typedef Documentation	725
16.287.1.1 value	725
16.288 SimdChooser< T, true, true > Struct Template Reference	725
16.288.1 Member Typedef Documentation	726
16.288.1.1 value	726
16.289 simdToType< T > Struct Template Reference	726
16.290 Single Struct Reference	726
16.291 Sparse< Field, SparseMatrix_t, IdxT, PtrT > Struct Template Reference	726
16.292 Sparse< _Field, SparseMatrix_t::COO > Struct Template Reference	726
16.292.1 Member Typedef Documentation	727
16.292.1.1 Field	727
16.292.2 Field Documentation	727
16.292.2.1 col	727
16.292.2.2 row	727
16.292.2.3 dat	727
16.292.2.4 delayed	727
16.292.2.5 kmax	727
16.292.2.6 m	727
16.292.2.7 n	727
16.292.2.8 nnz	727
16.292.2.9 nElements	728
16.292.2.10 maxrow	728
16.293 Sparse< _Field, SparseMatrix_t::COO_ZO > Struct Template Reference	728
16.293.1 Member Typedef Documentation	728
16.293.1.1 Field	728
16.293.2 Field Documentation	728
16.293.2.1 cst	728
16.293.2.2 col	729
16.293.2.3 row	729
16.293.2.4 dat	729
16.293.2.5 delayed	729
16.293.2.6 kmax	729
16.293.2.7 m	729
16.293.2.8 n	729
16.293.2.9 nnz	729
16.293.2.10 nElements	729
16.293.2.11 maxrow	729
16.294 Sparse< _Field, SparseMatrix_t::CSR > Struct Template Reference	729
16.294.1 Member Typedef Documentation	730
16.294.1.1 Field	730
16.294.2 Field Documentation	730

16.294.2.1 delayed	730
16.294.2.2 kmax	730
16.294.2.3 m	730
16.294.2.4 n	730
16.294.2.5 nnz	731
16.294.2.6 nElements	731
16.294.2.7 maxrow	731
16.294.2.8 col	731
16.294.2.9 st	731
16.294.2.10 stend	731
16.294.2.11 dat	731
16.295 Sparse<_Field, SparseMatrix_t::CSR_HYB> Struct Template Reference	731
16.295.1 Member Typedef Documentation	732
16.295.1.1 Field	732
16.295.2 Field Documentation	732
16.295.2.1 delayed	732
16.295.2.2 col	732
16.295.2.3 st	732
16.295.2.4 dat	732
16.295.2.5 kmax	732
16.295.2.6 m	732
16.295.2.7 n	732
16.295.2.8 nnz	732
16.295.2.9 nElements	732
16.295.2.10 maxrow	733
16.295.2.11 nOnes	733
16.295.2.12 nMOnes	733
16.295.2.13 nOthers	733
16.296 Sparse<_Field, SparseMatrix_t::CSR_ZO> Struct Template Reference	733
16.296.1 Member Typedef Documentation	733
16.296.1.1 Field	734
16.296.2 Field Documentation	734
16.296.2.1 cst	734
16.296.2.2 delayed	734
16.296.2.3 kmax	734
16.296.2.4 m	734
16.296.2.5 n	734
16.296.2.6 nnz	734
16.296.2.7 nElements	734
16.296.2.8 maxrow	734
16.296.2.9 col	734
16.296.2.10 st	734

16.296.2.11 stend . . . . .	735
16.296.2.12 dat . . . . .	735
16.297 Sparse< _Field, SparseMatrix_t::ELL > Struct Template Reference . . . . .	735
16.297.1 Member Typedef Documentation . . . . .	735
16.297.1.1 Field . . . . .	735
16.297.2 Field Documentation . . . . .	735
16.297.2.1 delayed . . . . .	735
16.297.2.2 kmax . . . . .	736
16.297.2.3 m . . . . .	736
16.297.2.4 n . . . . .	736
16.297.2.5 ld . . . . .	736
16.297.2.6 nnz . . . . .	736
16.297.2.7 nElements . . . . .	736
16.297.2.8 maxrow . . . . .	736
16.297.2.9 col . . . . .	736
16.297.2.10 dat . . . . .	736
16.298 Sparse< _Field, SparseMatrix_t::ELL_simd > Struct Template Reference . . . . .	736
16.298.1 Field Documentation . . . . .	737
16.298.1.1 delayed . . . . .	737
16.298.1.2 chunk . . . . .	737
16.298.1.3 m . . . . .	737
16.298.1.4 n . . . . .	737
16.298.1.5 ld . . . . .	737
16.298.1.6 kmax . . . . .	737
16.298.1.7 nnz . . . . .	737
16.298.1.8 nElements . . . . .	737
16.298.1.9 maxrow . . . . .	738
16.298.1.10 nChunks . . . . .	738
16.298.1.11 col . . . . .	738
16.298.1.12 dat . . . . .	738
16.299 Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > Struct Template Reference . . . . .	738
16.299.1 Field Documentation . . . . .	738
16.299.1.1 cst . . . . .	738
16.299.1.2 delayed . . . . .	739
16.299.1.3 chunk . . . . .	739
16.299.1.4 m . . . . .	739
16.299.1.5 n . . . . .	739
16.299.1.6 ld . . . . .	739
16.299.1.7 kmax . . . . .	739
16.299.1.8 nnz . . . . .	739
16.299.1.9 nElements . . . . .	739
16.299.1.10 maxrow . . . . .	739



16.299.1.11 nChunks . . . . .	739
16.299.1.12 col . . . . .	739
16.299.1.13 dat . . . . .	739
16.300 Sparse< _Field, SparseMatrix_t::ELL_ZO > Struct Template Reference . . . . .	740
16.300.1 Member Typedef Documentation . . . . .	740
16.300.1.1 Field . . . . .	740
16.300.2 Field Documentation . . . . .	740
16.300.2.1 cst . . . . .	740
16.300.2.2 delayed . . . . .	740
16.300.2.3 kmax . . . . .	740
16.300.2.4 m . . . . .	741
16.300.2.5 n . . . . .	741
16.300.2.6 ld . . . . .	741
16.300.2.7 nnz . . . . .	741
16.300.2.8 nElements . . . . .	741
16.300.2.9 maxrow . . . . .	741
16.300.2.10 col . . . . .	741
16.300.2.11 dat . . . . .	741
16.301 Sparse< _Field, SparseMatrix_t::HYB_ZO > Struct Template Reference . . . . .	741
16.301.1 Member Typedef Documentation . . . . .	742
16.301.1.1 Field . . . . .	742
16.301.1.2 Self_t . . . . .	742
16.301.2 Field Documentation . . . . .	742
16.301.2.1 delayed . . . . .	742
16.301.2.2 kmax . . . . .	742
16.301.2.3 m . . . . .	742
16.301.2.4 n . . . . .	742
16.301.2.5 nnz . . . . .	742
16.301.2.6 maxrow . . . . .	742
16.301.2.7 nElements . . . . .	742
16.301.2.8 dat . . . . .	743
16.301.2.9 one . . . . .	743
16.301.2.10 mone . . . . .	743
16.302 Sparse< _Field, SparseMatrix_t::SELL > Struct Template Reference . . . . .	743
16.302.1 Member Typedef Documentation . . . . .	743
16.302.1.1 Field . . . . .	744
16.302.2 Field Documentation . . . . .	744
16.302.2.1 delayed . . . . .	744
16.302.2.2 chunk . . . . .	744
16.302.2.3 kmax . . . . .	744
16.302.2.4 m . . . . .	744
16.302.2.5 n . . . . .	744

16.302.2.6 maxrow	744
16.302.2.7 sigma	744
16.302.2.8 nChunks	744
16.302.2.9 nnz	744
16.302.2.10 nElements	744
16.302.2.11 perm	745
16.302.2.12 st	745
16.302.2.13 chunkSize	745
16.302.2.14 col	745
16.302.2.15 dat	745
16.303 Sparse< _Field, SparseMatrix_t::SELL_ZO > Struct Template Reference	745
16.303.1 Member Typedef Documentation	746
16.303.1.1 Field	746
16.303.2 Field Documentation	746
16.303.2.1 cst	746
16.303.2.2 delayed	746
16.303.2.3 chunk	746
16.303.2.4 kmax	746
16.303.2.5 m	746
16.303.2.6 n	746
16.303.2.7 maxrow	746
16.303.2.8 sigma	746
16.303.2.9 nChunks	747
16.303.2.10 nnz	747
16.303.2.11 nElements	747
16.303.2.12 perm	747
16.303.2.13 st	747
16.303.2.14 chunkSize	747
16.303.2.15 col	747
16.303.2.16 dat	747
16.304 SpMat< Field, flag > Struct Template Reference	747
16.304.1 Field Documentation	747
16.304.1.1 _coo	747
16.304.1.2 _csr	748
16.304.1.3 _ell	748
16.305 Static_error_check< bool > Class Template Reference	748
16.305.1 Constructor & Destructor Documentation	748
16.305.1.1 Static_error_check()	748
16.306 Static_error_check< false > Class Reference	748
16.307 StatsMatrix Struct Reference	748
16.307.1 Field Documentation	749
16.307.1.1 rowdim	749

16.307.1.2 coldim	749
16.307.1.3 nOnes	749
16.307.1.4 nMOnes	749
16.307.1.5 nOthers	749
16.307.1.6 nnz	749
16.307.1.7 maxRow	749
16.307.1.8 minRow	750
16.307.1.9 averageRow	750
16.307.1.10 deviationRow	750
16.307.1.11 maxCol	750
16.307.1.12 minCol	750
16.307.1.13 averageCol	750
16.307.1.14 deviationCol	750
16.307.1.15 minColDifference	750
16.307.1.16 maxColDifference	750
16.307.1.17 averageColDifference	750
16.307.1.18 deviationColDifference	750
16.307.1.19 minRowDifference	750
16.307.1.20 maxRowDifference	751
16.307.1.21 averageRowDifference	751
16.307.1.22 deviationRowDifference	751
16.307.1.23 nDenseRows	751
16.307.1.24 nDenseCols	751
16.307.1.25 nEmptyRows	751
16.307.1.26 nEmptyCols	751
16.307.1.27 nEmptyColsEnd	751
16.307.1.28 denseRows	751
16.307.1.29 denseCols	751
16.308 support_fast_mod< T > Struct Template Reference	751
16.309 support_fast_mod< double > Struct Reference	752
16.310 support_fast_mod< float > Struct Reference	752
16.311 support_fast_mod< int64_t > Struct Reference	752
16.312 support_simd< T > Struct Template Reference	753
16.313 support_simd_add< T > Struct Template Reference	753
16.314 support_simd_mod< T > Struct Template Reference	753
16.315 tfn_minus Struct Reference	753
16.315.1 Member Function Documentation	754
16.315.1.1 operator>()()	754
16.316 tfn_minus_eq Struct Reference	754
16.316.1 Member Function Documentation	754
16.316.1.1 operator>()()	754
16.317 tfn_mul Struct Reference	754

16.317.1 Member Function Documentation	754
16.317.1.1 operator>()	754
16.318 tfn_mul_eq Struct Reference	755
16.318.1 Member Function Documentation	755
16.318.1.1 operator>()	755
16.319 tfn_plus Struct Reference	755
16.319.1 Member Function Documentation	755
16.319.1.1 operator>()	755
16.320 tfn_plus_eq Struct Reference	755
16.320.1 Member Function Documentation	755
16.320.1.1 operator>()	756
16.321 Threads Struct Reference	756
16.322 ThreeD Struct Reference	756
16.323 ThreeDAdaptive Struct Reference	756
16.324 ThreeDInPlace Struct Reference	756
16.325 TRSMHelper< ReclterTrait, ParSeqTrait > Struct Template Reference	756
16.325.1 Detailed Description	757
16.325.2 Constructor & Destructor Documentation	757
16.325.2.1 TRSMHelper() [1/3]	757
16.325.2.2 TRSMHelper() [2/3]	757
16.325.2.3 TRSMHelper() [3/3]	757
16.325.3 Member Function Documentation	757
16.325.3.1 pMMH() [1/2]	757
16.325.3.2 pMMH() [2/2]	757
16.325.4 Field Documentation	757
16.325.4.1 parseq	757
16.326 TwoD Struct Reference	758
16.327 TwoDAdaptive Struct Reference	758
16.328 UnparametricTag Struct Reference	758
16.328.1 Detailed Description	758
16.329 Winograd Struct Reference	758
16.330 WinogradPar Struct Reference	758
<b>17 File Documentation</b>	<b>759</b>
17.1 101-fgemv.C File Reference	759
17.1.1 Function Documentation	759
17.1.1.1 main()	759
17.2 2x2-fgemv.C File Reference	759
17.2.1 Function Documentation	759
17.2.1.1 main()	760
17.3 2x2-fts.C File Reference	760
17.3.1 Function Documentation	760

17.3.1.1 main()	760
17.4 2x2-pluq.C File Reference	760
17.4.1 Function Documentation	760
17.4.1.1 main()	760
17.5 align-allocator.h File Reference	761
17.6 args-parser.h File Reference	761
17.6.1 Macro Definition Documentation	761
17.6.1.1 TYPE_BOOL	761
17.6.1.2 END_OF_ARGUMENTS	762
17.6.1.3 type_integer	762
17.6.2 Enumeration Type Documentation	762
17.6.2.1 ArgumentType	762
17.6.3 Function Documentation	762
17.6.3.1 printHelpMessage()	762
17.6.3.2 findArgument()	762
17.6.3.3 getListArgs()	762
17.7 arithprog.C File Reference	763
17.7.1 Macro Definition Documentation	763
17.7.1.1 CUBE	763
17.7.1.2 GFOPS	763
17.7.2 Typedef Documentation	763
17.7.2.1 TTimer	763
17.7.3 Function Documentation	764
17.7.3.1 main()	764
17.8 benchmark-charpoly-mp.C File Reference	764
17.8.1 Macro Definition Documentation	764
17.8.1.1 __FFLASFFPACK_FORCE_SEQ	764
17.8.2 Function Documentation	764
17.8.2.1 main()	764
17.9 benchmark-charpoly.C File Reference	764
17.9.1 Macro Definition Documentation	765
17.9.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	765
17.9.2 Function Documentation	765
17.9.2.1 run_with_field()	765
17.9.2.2 main()	765
17.10 benchmark-checkers.C File Reference	765
17.10.1 Macro Definition Documentation	766
17.10.1.1 ENABLE_ALL_CHECKINGS	766
17.10.1.2 _NR_TESTS	766
17.10.1.3 _MAX_SIZE_MATRICES	766
17.10.1.4 CUBE	766
17.10.2 Function Documentation	766

17.10.2.1 main()	766
17.11 benchmark-dgemm.C File Reference	766
17.11.1 Macro Definition Documentation	767
17.11.1.1 CBLAS_GEMM	767
17.11.2 Typedef Documentation	767
17.11.2.1 TTimer	767
17.11.2.2 Floats	767
17.11.3 Function Documentation	767
17.11.3.1 main()	767
17.12 benchmark-dgetrf.C File Reference	767
17.12.1 Macro Definition Documentation	768
17.12.1.1 __FFLASFFPACK_HAVE_DGETRF	768
17.12.2 Typedef Documentation	768
17.12.2.1 TTimer	768
17.12.3 Function Documentation	768
17.12.3.1 main()	768
17.13 benchmark-dgetri.C File Reference	768
17.13.1 Typedef Documentation	768
17.13.1.1 TTimer	769
17.13.2 Function Documentation	769
17.13.2.1 main()	769
17.14 benchmark-dsytrf.C File Reference	769
17.14.1 Macro Definition Documentation	769
17.14.1.1 EFFGFF	769
17.14.2 Typedef Documentation	769
17.14.2.1 TTimer	769
17.14.3 Function Documentation	770
17.14.3.1 main()	770
17.15 benchmark-dtrsm.C File Reference	770
17.15.1 Typedef Documentation	770
17.15.1.1 TTimer	770
17.15.2 Function Documentation	770
17.15.2.1 main()	770
17.16 benchmark-dtrtri.C File Reference	770
17.16.1 Macro Definition Documentation	771
17.16.1.1 __FFLASFFPACK_HAVE_DTRTRI	771
17.16.2 Typedef Documentation	771
17.16.2.1 TTimer	771
17.16.3 Function Documentation	771
17.16.3.1 main()	771
17.17 benchmark-fadd-lvl2.C File Reference	771
17.17.1 Macro Definition Documentation	772

17.17.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	772
17.17.2 Function Documentation . . . . .	772
17.17.2.1 main() . . . . .	772
17.18 benchmark-fdot.C File Reference . . . . .	772
17.18.1 Macro Definition Documentation . . . . .	772
17.18.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	772
17.18.2 Function Documentation . . . . .	772
17.18.2.1 run_with_field() . . . . .	773
17.18.2.2 main() . . . . .	773
17.19 benchmark-fgemm-mp.C File Reference . . . . .	773
17.19.1 Macro Definition Documentation . . . . .	773
17.19.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	773
17.19.2 Function Documentation . . . . .	773
17.19.2.1 tmain() . . . . .	773
17.19.2.2 main() . . . . .	774
17.20 benchmark-fgemm-rns.C File Reference . . . . .	774
17.20.1 Macro Definition Documentation . . . . .	774
17.20.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	774
17.20.2 Typedef Documentation . . . . .	774
17.20.2.1 RNS . . . . .	774
17.20.2.2 Field . . . . .	774
17.20.2.3 Element_ptr . . . . .	775
17.20.2.4 ConstElement_ptr . . . . .	775
17.20.2.5 THREADS . . . . .	775
17.20.2.6 GRAIN . . . . .	775
17.20.2.7 TWOD . . . . .	775
17.20.2.8 TWODA . . . . .	775
17.20.2.9 THREED . . . . .	775
17.20.2.10 THREEDA . . . . .	775
17.20.2.11 THREEDIP . . . . .	775
17.20.2.12 PSeq . . . . .	775
17.20.3 Function Documentation . . . . .	775
17.20.3.1 main() . . . . .	775
17.21 benchmark-fgemm.C File Reference . . . . .	776
17.21.1 Macro Definition Documentation . . . . .	776
17.21.1.1 CLASSIC_HYBRID . . . . .	776
17.21.2 Function Documentation . . . . .	776
17.21.2.1 main() . . . . .	776
17.22 benchmark-fgemv-mp.C File Reference . . . . .	776
17.22.1 Macro Definition Documentation . . . . .	777
17.22.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . .	777
17.22.2 Function Documentation . . . . .	777

17.22.2.1 write_matrix()	777
17.23 benchmark-fgemv.C File Reference	777
17.23.1 Macro Definition Documentation	778
17.23.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	778
17.23.2 Function Documentation	778
17.23.2.1 fill_value()	778
17.23.2.2 genData()	778
17.23.2.3 check_result()	779
17.23.2.4 benchmark_with_timer()	779
17.23.2.5 benchmark_disp()	779
17.23.2.6 benchmark_in_Field()	780
17.23.2.7 benchmark_with_field() [1/2]	780
17.23.2.8 benchmark_with_field() [2/2]	780
17.23.2.9 main()	780
17.24 benchmark-fgesv.C File Reference	780
17.24.1 Macro Definition Documentation	781
17.24.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	781
17.24.2 Function Documentation	781
17.24.2.1 main()	781
17.25 benchmark-fsyrf.C File Reference	781
17.25.1 Macro Definition Documentation	781
17.25.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	782
17.25.1.2 CUBE	782
17.25.2 Function Documentation	782
17.25.2.1 main()	782
17.26 benchmark-fsytrf.C File Reference	782
17.26.1 Macro Definition Documentation	782
17.26.1.1 __FFPACK_FSYTRF_BC_CROUT	782
17.26.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	782
17.26.1.3 CUBE	782
17.26.2 Function Documentation	783
17.26.2.1 main()	783
17.27 benchmark-ftsrm-mp.C File Reference	783
17.27.1 Macro Definition Documentation	783
17.27.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	783
17.27.2 Function Documentation	783
17.27.2.1 main()	783
17.28 benchmark-ftsrm.C File Reference	783
17.28.1 Macro Definition Documentation	784
17.28.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	784
17.28.2 Function Documentation	784
17.28.2.1 main()	784



17.29 benchmark-ftsrv.C File Reference	784
17.29.1 Macro Definition Documentation	784
17.29.1.1 __Fflasffpack_Openblas_Nt_Already_Set	784
17.29.2 Function Documentation	784
17.29.2.1 main()	785
17.30 benchmark-ftsrti.C File Reference	785
17.30.1 Macro Definition Documentation	785
17.30.1.1 __Fflasffpack_Openblas_Nt_Already_Set	785
17.30.1.2 CUBE	785
17.30.2 Function Documentation	785
17.30.2.1 main()	785
17.31 benchmark-inverse.C File Reference	785
17.31.1 Macro Definition Documentation	786
17.31.1.1 CUBE	786
17.31.2 Function Documentation	786
17.31.2.1 main()	786
17.32 benchmark-lqmp.C File Reference	786
17.32.1 Function Documentation	786
17.32.1.1 main()	786
17.33 benchmark-lqp.C File Reference	787
17.33.1 Macro Definition Documentation	787
17.33.1.1 CUBE	787
17.33.2 Function Documentation	787
17.33.2.1 main()	787
17.34 benchmark-pluq.C File Reference	787
17.34.1 Macro Definition Documentation	788
17.34.1.1 __Fflasffpack_Openblas_Nt_Already_Set	788
17.34.1.2 CUBE	788
17.34.2 Typedef Documentation	788
17.34.2.1 Field	788
17.34.3 Function Documentation	788
17.34.3.1 verification_PLUQ()	788
17.34.3.2 Rec_Initialize()	788
17.34.3.3 main()	789
17.35 benchmark-wino.C File Reference	789
17.35.1 Macro Definition Documentation	789
17.35.1.1 CUBE	789
17.35.2 Function Documentation	789
17.35.2.1 launch_wino()	789
17.35.2.2 main()	789
17.36 bit_manipulation.h File Reference	790
17.36.1 Macro Definition Documentation	790

17.36.1.1	<a href="#">__has_builtin</a>	790
17.36.2	Function Documentation	790
17.36.2.1	<a href="#">clz()</a> [1/2]	790
17.36.2.2	<a href="#">clz()</a> [2/2]	790
17.36.2.3	<a href="#">ctz()</a> [1/2]	790
17.36.2.4	<a href="#">ctz()</a> [2/2]	790
17.37	blockcuts.inl File Reference	790
17.37.1	Macro Definition Documentation	792
17.37.1.1	<a href="#">__FFLASFFPACK_fflas_blockcuts_INL</a>	792
17.37.1.2	<a href="#">__FFLASFFPACK_MINBLOCKCUTS</a>	792
17.38	cast.h File Reference	792
17.39	cblas.C File Reference	792
17.39.1	Macro Definition Documentation	792
17.39.1.1	<a href="#">__FFLASFFPACK_CONFIGURATION</a>	792
17.39.1.2	<a href="#">__FFLASFFPACK_HAVE_CBLAS</a>	793
17.39.2	Function Documentation	793
17.39.2.1	<a href="#">main()</a>	793
17.40	charpoly.C File Reference	793
17.40.1	Macro Definition Documentation	793
17.40.1.1	<a href="#">CUBE</a>	793
17.40.1.2	<a href="#">GFOPS</a>	793
17.40.2	Typedef Documentation	793
17.40.2.1	<a href="#">TTimer</a>	794
17.40.3	Function Documentation	794
17.40.3.1	<a href="#">main()</a>	794
17.41	charpoly.C File Reference	794
17.41.1	Function Documentation	794
17.41.1.1	<a href="#">main()</a>	794
17.42	checker_charpoly.inl File Reference	794
17.42.1	Macro Definition Documentation	794
17.42.1.1	<a href="#">__FFLASFFPACK_checker_charpoly_INL</a>	795
17.43	checker_det.inl File Reference	795
17.43.1	Macro Definition Documentation	795
17.43.1.1	<a href="#">__FFLASFFPACK_checker_det_INL</a>	795
17.44	checker_empty.h File Reference	795
17.45	checker_fgemm.inl File Reference	795
17.45.1	Macro Definition Documentation	796
17.45.1.1	<a href="#">__FFLASFFPACK_checker_fgemm_INL</a>	796
17.46	checker_ftsm.inl File Reference	796
17.46.1	Macro Definition Documentation	796
17.46.1.1	<a href="#">__FFLASFFPACK_checker_ftsm_INL</a>	796
17.47	checker_invert.inl File Reference	796

17.47.1 Macro Definition Documentation	796
17.47.1.1 __FFLASFFPACK_checker_invert_INL	796
17.48 checker_pluq.inl File Reference	796
17.48.1 Macro Definition Documentation	797
17.48.1.1 __FFLASFFPACK_checker_pluq_INL	797
17.49 checkers.doxy File Reference	797
17.50 checkers_fflas.h File Reference	797
17.51 checkers_fflas.inl File Reference	797
17.51.1 Macro Definition Documentation	798
17.51.1.1 FFLASFFPACK_checkers_fflas_inl_H	798
17.52 checkers_ffpack.h File Reference	798
17.53 checkers_ffpack.inl File Reference	798
17.53.1 Macro Definition Documentation	799
17.53.1.1 FFLASFFPACK_checkers_ffpack_inl_H	799
17.54 clapack.C File Reference	799
17.54.1 Macro Definition Documentation	799
17.54.1.1 __FFLASFFPACK_CONFIGURATION	799
17.54.1.2 __FFLASFFPACK_HAVE_LAPACK	799
17.54.1.3 __FFLASFFPACK_HAVE_CLAPACK	800
17.54.2 Function Documentation	800
17.54.2.1 main()	800
17.55 config-blas.h File Reference	800
17.55.1 Macro Definition Documentation	801
17.55.1.1 CBLAS_INT	801
17.55.1.2 CBLAS_ENUM_DEFINED_H	801
17.55.1.3 CBLAS_EXTERNALS	801
17.55.1.4 blas_enum	801
17.55.2 Enumeration Type Documentation	801
17.55.2.1 CBLAS_ORDER	801
17.55.2.2 CBLAS_TRANSPOSE	801
17.55.2.3 CBLAS_UPLO	802
17.55.2.4 CBLAS_DIAG	802
17.55.2.5 CBLAS_SIDE	802
17.55.3 Function Documentation	802
17.55.3.1 daxpy_()	802
17.55.3.2 saxpy_()	802
17.55.3.3 ddot_()	803
17.55.3.4 sdot_()	803
17.55.3.5 dasum_()	803
17.55.3.6 idamax_()	803
17.55.3.7 dnorm2_()	803
17.55.3.8 dgemv_()	803

17.55.3.9 sgemv_()	804
17.55.3.10 dger_()	804
17.55.3.11 sger_()	804
17.55.3.12 dcopy_()	804
17.55.3.13 scopy_()	805
17.55.3.14 dscal_()	805
17.55.3.15 sscal_()	805
17.55.3.16 dtrsm_()	805
17.55.3.17 strsm_()	805
17.55.3.18 dtrmm_()	806
17.55.3.19 strmm_()	806
17.55.3.20 sgemm_()	806
17.55.3.21 dgemm_()	806
17.56 config.h File Reference	807
17.56.1 Macro Definition Documentation	807
17.56.1.1 HAVE_BLAS	808
17.56.1.2 HAVE_CBLAS	808
17.56.1.3 HAVE_CXX11	808
17.56.1.4 HAVE_DLFCN_H	808
17.56.1.5 HAVE_FLOAT_H	808
17.56.1.6 HAVE_INT128	808
17.56.1.7 HAVE_INTPTR_T	808
17.56.1.8 HAVE_LAPACK	808
17.56.1.9 HAVE_LIMITS_H	808
17.56.1.10 HAVE_LITTLE_ENDIAN	808
17.56.1.11 HAVE_PTHREAD_H	808
17.56.1.12 HAVE_STDDEF_H	808
17.56.1.13 HAVE_STDINT_H	809
17.56.1.14 HAVE_STDIO_H	809
17.56.1.15 HAVE_STDLIB_H	809
17.56.1.16 HAVE_STRINGS_H	809
17.56.1.17 HAVE_STRING_H	809
17.56.1.18 HAVE_SYS_STAT_H	809
17.56.1.19 HAVE_SYS_TIME_H	809
17.56.1.20 HAVE_SYS_TYPES_H	809
17.56.1.21 HAVE_UNISTD_H	809
17.56.1.22 LT_OBJDIR	809
17.56.1.23 OPENBLAS_NUM_THREADS	809
17.56.1.24 PACKAGE	809
17.56.1.25 PACKAGE_BUGREPORT	810
17.56.1.26 PACKAGE_NAME	810
17.56.1.27 PACKAGE_STRING	810

17.56.1.28 PACKAGE_TARNAME . . . . .	810
17.56.1.29 PACKAGE_URL . . . . .	810
17.56.1.30 PACKAGE_VERSION . . . . .	810
17.56.1.31 SIZEOF_CHAR . . . . .	810
17.56.1.32 SIZEOF_INT . . . . .	810
17.56.1.33 SIZEOF_LONG . . . . .	810
17.56.1.34 SIZEOF_LONG_LONG . . . . .	810
17.56.1.35 SIZEOF_SHORT . . . . .	810
17.56.1.36 SIZEOF___INT64 . . . . .	810
17.56.1.37 STDC_HEADERS . . . . .	811
17.56.1.38 USE_OPENMP . . . . .	811
17.56.1.39 VERSION . . . . .	811
17.57 config.h File Reference . . . . .	811
17.57.1 Macro Definition Documentation . . . . .	812
17.57.1.1 __FFLASFFPACK_HAVE_BLAS . . . . .	812
17.57.1.2 __FFLASFFPACK_HAVE_CBLAS . . . . .	812
17.57.1.3 __FFLASFFPACK_HAVE_CXX11 . . . . .	812
17.57.1.4 __FFLASFFPACK_HAVE_DLFCN_H . . . . .	812
17.57.1.5 __FFLASFFPACK_HAVE_FLOAT_H . . . . .	812
17.57.1.6 __FFLASFFPACK_HAVE_INT128 . . . . .	812
17.57.1.7 __FFLASFFPACK_HAVE_INTPATHS_H . . . . .	812
17.57.1.8 __FFLASFFPACK_HAVE_LAPACK . . . . .	812
17.57.1.9 __FFLASFFPACK_HAVE_LIMITS_H . . . . .	812
17.57.1.10 __FFLASFFPACK_HAVE_LITTLE_ENDIAN . . . . .	812
17.57.1.11 __FFLASFFPACK_HAVE_PTHREAD_H . . . . .	812
17.57.1.12 __FFLASFFPACK_HAVE_STDDEF_H . . . . .	813
17.57.1.13 __FFLASFFPACK_HAVE_STDINT_H . . . . .	813
17.57.1.14 __FFLASFFPACK_HAVE_STDIO_H . . . . .	813
17.57.1.15 __FFLASFFPACK_HAVE_STDLIB_H . . . . .	813
17.57.1.16 __FFLASFFPACK_HAVE_STRINGS_H . . . . .	813
17.57.1.17 __FFLASFFPACK_HAVE_STRING_H . . . . .	813
17.57.1.18 __FFLASFFPACK_HAVE_SYS_STAT_H . . . . .	813
17.57.1.19 __FFLASFFPACK_HAVE_SYS_TIME_H . . . . .	813
17.57.1.20 __FFLASFFPACK_HAVE_SYS_TYPES_H . . . . .	813
17.57.1.21 __FFLASFFPACK_HAVE_UNISTD_H . . . . .	813
17.57.1.22 __FFLASFFPACK_LT_OBJDIR . . . . .	813
17.57.1.23 __FFLASFFPACK_OPENBLAS_NUM_THREADS . . . . .	813
17.57.1.24 __FFLASFFPACK_PACKAGE . . . . .	814
17.57.1.25 __FFLASFFPACK_PACKAGE_BUGREPORT . . . . .	814
17.57.1.26 __FFLASFFPACK_PACKAGE_NAME . . . . .	814
17.57.1.27 __FFLASFFPACK_PACKAGE_STRING . . . . .	814
17.57.1.28 __FFLASFFPACK_PACKAGE_TARNAME . . . . .	814

17.57.1.29	<a href="#">__FFLASFFPACK_PACKAGE_URL</a>	814
17.57.1.30	<a href="#">__FFLASFFPACK_PACKAGE_VERSION</a>	814
17.57.1.31	<a href="#">__FFLASFFPACK_SIZEOF_CHAR</a>	814
17.57.1.32	<a href="#">__FFLASFFPACK_SIZEOF_INT</a>	814
17.57.1.33	<a href="#">__FFLASFFPACK_SIZEOF_LONG</a>	814
17.57.1.34	<a href="#">__FFLASFFPACK_SIZEOF_LONG_LONG</a>	814
17.57.1.35	<a href="#">__FFLASFFPACK_SIZEOF_SHORT</a>	814
17.57.1.36	<a href="#">__FFLASFFPACK_SIZEOF__INT64</a>	815
17.57.1.37	<a href="#">__FFLASFFPACK_STDC_HEADERS</a>	815
17.57.1.38	<a href="#">__FFLASFFPACK_USE_OPENMP</a>	815
17.57.1.39	<a href="#">__FFLASFFPACK_VERSION</a>	815
17.58	<a href="#">coo.h File Reference</a>	815
17.59	<a href="#">coo_spmv.inl File Reference</a>	815
17.59.1	<a href="#">Macro Definition Documentation</a>	816
17.59.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_coo_spmv_INL</a>	816
17.60	<a href="#">coo_spmv.inl File Reference</a>	816
17.60.1	<a href="#">Macro Definition Documentation</a>	817
17.60.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_coo_spmv_INL</a>	817
17.61	<a href="#">coo_utils.inl File Reference</a>	817
17.61.1	<a href="#">Macro Definition Documentation</a>	818
17.61.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_coo_utils_INL</a>	818
17.62	<a href="#">csr.h File Reference</a>	818
17.63	<a href="#">csr_hyb.h File Reference</a>	818
17.64	<a href="#">csr_hyb_pspmm.inl File Reference</a>	819
17.64.1	<a href="#">Macro Definition Documentation</a>	819
17.64.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL</a>	819
17.65	<a href="#">csr_hyb_pspmv.inl File Reference</a>	819
17.65.1	<a href="#">Macro Definition Documentation</a>	820
17.65.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL</a>	820
17.66	<a href="#">csr_hyb_spmv.inl File Reference</a>	820
17.66.1	<a href="#">Macro Definition Documentation</a>	820
17.66.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL</a>	820
17.67	<a href="#">csr_hyb_spmv.inl File Reference</a>	821
17.67.1	<a href="#">Macro Definition Documentation</a>	821
17.67.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL</a>	821
17.68	<a href="#">csr_hyb_utils.inl File Reference</a>	821
17.68.1	<a href="#">Macro Definition Documentation</a>	821
17.68.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL</a>	822
17.69	<a href="#">csr_pspmm.inl File Reference</a>	822
17.69.1	<a href="#">Macro Definition Documentation</a>	822
17.69.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_CSR_pspmm_INL</a>	822
17.70	<a href="#">csr_pspmv.inl File Reference</a>	822

17.70.1 Macro Definition Documentation . . . . .	823
17.70.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL . . . . .	823
17.71 csr_spm্ম.inl File Reference . . . . .	823
17.71.1 Macro Definition Documentation . . . . .	824
17.71.1.1 __FFLASFFPACK_fflas_sparse_CSR_spm্ম_INL . . . . .	824
17.72 csr_spmv.inl File Reference . . . . .	824
17.72.1 Macro Definition Documentation . . . . .	825
17.72.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmv_INL . . . . .	825
17.73 csr_utils.inl File Reference . . . . .	825
17.74 cuda.C File Reference . . . . .	826
17.74.1 Function Documentation . . . . .	826
17.74.1.1 main() . . . . .	826
17.75 debug.h File Reference . . . . .	826
17.75.1 Detailed Description . . . . .	827
17.75.2 Macro Definition Documentation . . . . .	827
17.75.2.1 FFLASFFPACK_check . . . . .	827
17.75.2.2 FFLASFFPACK_abort . . . . .	827
17.76 det.C File Reference . . . . .	827
17.76.1 Function Documentation . . . . .	828
17.76.1.1 main() . . . . .	828
17.77 ell.h File Reference . . . . .	828
17.78 ell_pspmm.inl File Reference . . . . .	828
17.78.1 Macro Definition Documentation . . . . .	829
17.78.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL . . . . .	829
17.79 ell_pspmv.inl File Reference . . . . .	829
17.79.1 Macro Definition Documentation . . . . .	830
17.79.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL . . . . .	830
17.80 ell_simd.h File Reference . . . . .	830
17.81 ell_simd_pspmv.inl File Reference . . . . .	830
17.81.1 Macro Definition Documentation . . . . .	831
17.81.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL . . . . .	831
17.82 ell_simd_spmv.inl File Reference . . . . .	831
17.82.1 Macro Definition Documentation . . . . .	832
17.82.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL . . . . .	832
17.83 ell_simd_utils.inl File Reference . . . . .	832
17.83.1 Macro Definition Documentation . . . . .	833
17.83.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL . . . . .	833
17.84 ell_spm্ম.inl File Reference . . . . .	833
17.84.1 Macro Definition Documentation . . . . .	834
17.84.1.1 __FFLASFFPACK_fflas_sparse_ELL_spm্ম_INL . . . . .	834
17.85 ell_spmv.inl File Reference . . . . .	834
17.85.1 Macro Definition Documentation . . . . .	834

17.85.1.1 <a href="#">__FFLASFFPACK_fflas_sparse_ELL_spmv_INL</a>	834
17.86 <a href="#">ell_utils.inl</a> File Reference	835
17.86.1 Macro Definition Documentation	835
17.86.1.1 <a href="#">__FFLASFFPACK_fflas_sparse_ELL_utils_INL</a>	835
17.87 <a href="#">fblas.C</a> File Reference	835
17.87.1 Macro Definition Documentation	835
17.87.1.1 <a href="#">__FFLASFFPACK_CONFIGURATION</a>	835
17.87.2 Function Documentation	836
17.87.2.1 <a href="#">dgemm_()</a>	836
17.87.2.2 <a href="#">main()</a>	836
17.88 <a href="#">fflas-101_1.C</a> File Reference	836
17.88.1 Function Documentation	836
17.88.1.1 <a href="#">main()</a>	836
17.89 <a href="#">fflas-101_3.C</a> File Reference	836
17.89.1 Function Documentation	837
17.89.1.1 <a href="#">main()</a>	837
17.90 <a href="#">fflas-ffpack-config.h</a> File Reference	837
17.90.1 Detailed Description	837
17.90.2 Macro Definition Documentation	837
17.90.2.1 <a href="#">GCC_VERSION</a>	837
17.91 <a href="#">fflas-ffpack-default-thresholds.h</a> File Reference	837
17.91.1 Macro Definition Documentation	838
17.91.1.1 <a href="#">__FFLASFFPACK_WINOTHRESHOLD</a>	838
17.91.1.2 <a href="#">__FFLASFFPACK_WINOTHRESHOLD_FLT</a>	838
17.91.1.3 <a href="#">__FFLASFFPACK_WINOTHRESHOLD_BAL</a>	838
17.91.1.4 <a href="#">__FFLASFFPACK_WINOTHRESHOLD_BAL_FLT</a>	838
17.91.1.5 <a href="#">__FFLASFFPACK_PLUQ_THRESHOLD</a>	838
17.91.1.6 <a href="#">__FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD</a>	838
17.91.1.7 <a href="#">__FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD</a>	838
17.91.1.8 <a href="#">__FFLASFFPACK_ARITHPROG_THRESHOLD</a>	838
17.91.1.9 <a href="#">__FFLASFFPACK_FTRTRI_THRESHOLD</a>	838
17.91.1.10 <a href="#">__FFLASFFPACK_FSYTRF_THRESHOLD</a>	838
17.91.1.11 <a href="#">__FFLASFFPACK_FSYRK_THRESHOLD</a>	839
17.92 <a href="#">fflas-ffpack-thresholds.h</a> File Reference	839
17.93 <a href="#">fflas-ffpack.dox</a> File Reference	839
17.94 <a href="#">fflas-ffpack.h</a> File Reference	839
17.94.1 Detailed Description	839
17.95 <a href="#">fflas.dox</a> File Reference	839
17.96 <a href="#">fflas.h</a> File Reference	839
17.96.1 Detailed Description	840
17.96.2 Macro Definition Documentation	840
17.96.2.1 <a href="#">WINOTHRESHOLD</a>	840



17.96.2.2 DOUBLE_TO_FLOAT_CROSSOVER . . . . .	840
17.97 fflas_101.C File Reference . . . . .	840
17.97.1 Function Documentation . . . . .	840
17.97.1.1 main() . . . . .	841
17.98 fflas_101_ivl1.C File Reference . . . . .	841
17.98.1 Function Documentation . . . . .	841
17.98.1.1 main() . . . . .	841
17.99 fflas_bounds.inl File Reference . . . . .	841
17.99.1 Macro Definition Documentation . . . . .	842
17.99.1.1 __FFLASFFPACK_fflas_bounds_INL . . . . .	842
17.99.1.2 FFLAS_INT_TYPE . . . . .	842
17.100 fflas_c.h File Reference . . . . .	842
17.100.1 Macro Definition Documentation . . . . .	844
17.100.1.1 FFLAS_COMPILED . . . . .	844
17.100.2 Enumeration Type Documentation . . . . .	844
17.100.2.1 FFLAS_C_ORDER . . . . .	844
17.100.2.2 FFLAS_C_TRANSPOSE . . . . .	845
17.100.2.3 FFLAS_C_UPLO . . . . .	845
17.100.2.4 FFLAS_C_DIAG . . . . .	845
17.100.2.5 FFLAS_C_SIDE . . . . .	845
17.100.2.6 FFLAS_C_BASE . . . . .	846
17.100.3 Function Documentation . . . . .	846
17.100.3.1 freducein_1_modular_double() . . . . .	846
17.100.3.2 freduce_1_modular_double() . . . . .	846
17.100.3.3 fnegin_1_modular_double() . . . . .	846
17.100.3.4 fneg_1_modular_double() . . . . .	846
17.100.3.5 fzero_1_modular_double() . . . . .	847
17.100.3.6 fiszero_1_modular_double() . . . . .	847
17.100.3.7 fequal_1_modular_double() . . . . .	847
17.100.3.8 fassign_1_modular_double() . . . . .	847
17.100.3.9 fscaln_1_modular_double() . . . . .	847
17.100.3.10 fscal_1_modular_double() . . . . .	848
17.100.3.11 faxpy_1_modular_double() . . . . .	848
17.100.3.12 fdot_1_modular_double() . . . . .	848
17.100.3.13 fswap_1_modular_double() . . . . .	848
17.100.3.14 fadd_1_modular_double() . . . . .	848
17.100.3.15 fsub_1_modular_double() . . . . .	849
17.100.3.16 faddin_1_modular_double() . . . . .	849
17.100.3.17 fsubin_1_modular_double() . . . . .	849
17.100.3.18 fassign_2_modular_double() . . . . .	849
17.100.3.19 fzero_2_modular_double() . . . . .	849
17.100.3.20 fequal_2_modular_double() . . . . .	850

17.100.3.21	<a href="#">fiszero_2_modular_double()</a>	850
17.100.3.22	<a href="#">fidentity_2_modular_double()</a>	850
17.100.3.23	<a href="#">freducein_2_modular_double()</a>	850
17.100.3.24	<a href="#">freduce_2_modular_double()</a>	850
17.100.3.25	<a href="#">fnegin_2_modular_double()</a>	851
17.100.3.26	<a href="#">fneg_2_modular_double()</a>	851
17.100.3.27	<a href="#">fscaln_2_modular_double()</a>	851
17.100.3.28	<a href="#">fscal_2_modular_double()</a>	851
17.100.3.29	<a href="#">faxpy_2_modular_double()</a>	852
17.100.3.30	<a href="#">fmove_2_modular_double()</a>	852
17.100.3.31	<a href="#">fadd_2_modular_double()</a>	852
17.100.3.32	<a href="#">fsub_2_modular_double()</a>	852
17.100.3.33	<a href="#">fsubin_2_modular_double()</a>	852
17.100.3.34	<a href="#">faddin_2_modular_double()</a>	853
17.100.3.35	<a href="#">fgemv_2_modular_double()</a>	853
17.100.3.36	<a href="#">fger_2_modular_double()</a>	853
17.100.3.37	<a href="#">ftrsv_2_modular_double()</a>	854
17.100.3.38	<a href="#">ftrsm_3_modular_double()</a>	854
17.100.3.39	<a href="#">ftrmm_3_modular_double()</a>	854
17.100.3.40	<a href="#">fgemm_3_modular_double()</a>	854
17.100.3.41	<a href="#">fsquare_3_modular_double()</a>	855
17.101	<a href="#">fflas_enum.h</a> File Reference	855
17.102	<a href="#">fflas_fadd.h</a> File Reference	856
17.103	<a href="#">fflas_fadd.inl</a> File Reference	857
17.103.1	Macro Definition Documentation	858
17.103.1.1	<a href="#">__FFLASFFPACK_fadd_INL</a>	858
17.104	<a href="#">fflas_fassign.h</a> File Reference	858
17.105	<a href="#">fflas_fassign.inl</a> File Reference	858
17.105.1	Macro Definition Documentation	859
17.105.1.1	<a href="#">__FFLASFFPACK_fassign_INL</a>	859
17.106	<a href="#">fflas_faxpy.inl</a> File Reference	859
17.106.1	Macro Definition Documentation	860
17.106.1.1	<a href="#">__FFLASFFPACK_faxpy_INL</a>	860
17.107	<a href="#">fflas_fdot.inl</a> File Reference	860
17.107.1	Macro Definition Documentation	860
17.107.1.1	<a href="#">__FFLASFFPACK_fdot_INL</a>	860
17.108	<a href="#">fflas_fgemm.inl</a> File Reference	861
17.108.1	Macro Definition Documentation	863
17.108.1.1	<a href="#">__FFLASFFPACK_fgemm_INL</a>	863
17.109	<a href="#">fflas_fgemv.inl</a> File Reference	863
17.109.1	Macro Definition Documentation	864
17.109.1.1	<a href="#">__FFLASFFPACK_fgemv_INL</a>	864

17.110 fflas_fgmv_mp.inl File Reference . . . . .	864
17.110.1 Macro Definition Documentation . . . . .	865
17.110.1.1 __FFLASFFPACK_fgmv_mp_INL . . . . .	865
17.111 fflas_fger.inl File Reference . . . . .	865
17.111.1 Macro Definition Documentation . . . . .	866
17.111.1.1 __FFLASFFPACK_fger_INL . . . . .	866
17.112 fflas_fger_mp.inl File Reference . . . . .	866
17.112.1 Macro Definition Documentation . . . . .	867
17.112.1.1 __FFPACK_fger_mp_INL . . . . .	867
17.113 fflas_freduce.h File Reference . . . . .	867
17.114 fflas_freduce.inl File Reference . . . . .	868
17.114.1 Macro Definition Documentation . . . . .	869
17.114.1.1 __FFLASFFPACK_fflas_freduce_INL . . . . .	869
17.114.1.2 FFLASFFPACK_COPY_REDUCE . . . . .	869
17.115 fflas_freduce_mp.inl File Reference . . . . .	870
17.115.1 Macro Definition Documentation . . . . .	870
17.115.1.1 __FFLASFFPACK_fflas_freduce_mp_INL . . . . .	870
17.116 fflas_freivalds.inl File Reference . . . . .	870
17.116.1 Macro Definition Documentation . . . . .	870
17.116.1.1 __FFLASFFPACK_freivalds_INL . . . . .	870
17.117 fflas_fscal.h File Reference . . . . .	871
17.118 fflas_fscal.inl File Reference . . . . .	871
17.118.1 Macro Definition Documentation . . . . .	872
17.118.1.1 __FFLASFFPACK_fscal_INL . . . . .	872
17.119 fflas_fscal_mp.inl File Reference . . . . .	872
17.119.1 Macro Definition Documentation . . . . .	873
17.119.1.1 __FFLASFFPACK_fscal_mp_INL . . . . .	873
17.120 fflas_fsyr2k.inl File Reference . . . . .	873
17.120.1 Macro Definition Documentation . . . . .	873
17.120.1.1 __FFLASFFPACK_fflas_fsyr2k_INL . . . . .	873
17.121 fflas_fsyrk.inl File Reference . . . . .	873
17.121.1 Macro Definition Documentation . . . . .	874
17.121.1.1 __FFLASFFPACK_fflas_fsyrk_INL . . . . .	874
17.122 fflas_ftrmm.inl File Reference . . . . .	874
17.122.1 Macro Definition Documentation . . . . .	875
17.122.1.1 __FFLASFFPACK_ftrmm_INL . . . . .	875
17.123 fflas_ftrsm.inl File Reference . . . . .	875
17.123.1 Macro Definition Documentation . . . . .	875
17.123.1.1 __FFLASFFPACK_ftrsm_INL . . . . .	876
17.124 fflas_ftrsm_mp.inl File Reference . . . . .	876
17.124.1 Detailed Description . . . . .	876
17.124.2 Macro Definition Documentation . . . . .	876

17.124.2.1 __FFPACK_frsm_mp_INL . . . . .	876
17.125 fflas_frsv.inl File Reference . . . . .	876
17.125.1 Macro Definition Documentation . . . . .	877
17.125.1.1 __FFLASFFPACK_frsv_INL . . . . .	877
17.126 fflas_helpers.inl File Reference . . . . .	877
17.126.1 Macro Definition Documentation . . . . .	878
17.126.1.1 __FFLASFFPACK_fflas_fflas_mmhelper_INL . . . . .	878
17.127 fflas_intrinsic.h File Reference . . . . .	878
17.128 fflas_io.h File Reference . . . . .	878
17.129 fflas_L1_inst.C File Reference . . . . .	879
17.129.1 Macro Definition Documentation . . . . .	879
17.129.1.1 __FFLAS_L1_INST_C . . . . .	879
17.129.1.2 INST_OR_DECL . . . . .	879
17.129.1.3 FFLAS_FIELD [1/2] . . . . .	879
17.129.1.4 FFLAS_ELT [1/6] . . . . .	880
17.129.1.5 FFLAS_ELT [2/6] . . . . .	880
17.129.1.6 FFLAS_ELT [3/6] . . . . .	880
17.129.1.7 FFLAS_FIELD [2/2] . . . . .	880
17.129.1.8 FFLAS_ELT [4/6] . . . . .	880
17.129.1.9 FFLAS_ELT [5/6] . . . . .	880
17.129.1.10 FFLAS_ELT [6/6] . . . . .	880
17.130 fflas_L1_inst.h File Reference . . . . .	880
17.130.1 Macro Definition Documentation . . . . .	880
17.130.1.1 INST_OR_DECL . . . . .	881
17.130.1.2 FFLAS_FIELD [1/2] . . . . .	881
17.130.1.3 FFLAS_ELT [1/6] . . . . .	881
17.130.1.4 FFLAS_ELT [2/6] . . . . .	881
17.130.1.5 FFLAS_ELT [3/6] . . . . .	881
17.130.1.6 FFLAS_FIELD [2/2] . . . . .	881
17.130.1.7 FFLAS_ELT [4/6] . . . . .	881
17.130.1.8 FFLAS_ELT [5/6] . . . . .	881
17.130.1.9 FFLAS_ELT [6/6] . . . . .	881
17.131 fflas_L1_inst_implem.inl File Reference . . . . .	881
17.132 fflas_L2_inst.C File Reference . . . . .	883
17.132.1 Macro Definition Documentation . . . . .	883
17.132.1.1 __FFLAS_L2_INST_C . . . . .	883
17.132.1.2 INST_OR_DECL . . . . .	883
17.132.1.3 FFLAS_FIELD [1/2] . . . . .	883
17.132.1.4 FFLAS_ELT [1/6] . . . . .	883
17.132.1.5 FFLAS_ELT [2/6] . . . . .	883
17.132.1.6 FFLAS_ELT [3/6] . . . . .	883
17.132.1.7 FFLAS_FIELD [2/2] . . . . .	884

17.132.1.8 FFLAS_ELT [4/6]	884
17.132.1.9 FFLAS_ELT [5/6]	884
17.132.1.10 FFLAS_ELT [6/6]	884
17.133 fflas_L2_inst.h File Reference	884
17.133.1 Macro Definition Documentation	884
17.133.1.1 INST_OR_DECL	884
17.133.1.2 FFLAS_FIELD [1/2]	884
17.133.1.3 FFLAS_ELT [1/6]	884
17.133.1.4 FFLAS_ELT [2/6]	885
17.133.1.5 FFLAS_ELT [3/6]	885
17.133.1.6 FFLAS_FIELD [2/2]	885
17.133.1.7 FFLAS_ELT [4/6]	885
17.133.1.8 FFLAS_ELT [5/6]	885
17.133.1.9 FFLAS_ELT [6/6]	885
17.134 fflas_L2_inst_implem.inl File Reference	885
17.135 fflas_L3_inst.C File Reference	887
17.135.1 Macro Definition Documentation	887
17.135.1.1 __FFLAS_L3_INST_C	887
17.135.1.2 INST_OR_DECL	887
17.135.1.3 FFLAS_FIELD [1/2]	887
17.135.1.4 FFLAS_ELT [1/6]	887
17.135.1.5 FFLAS_ELT [2/6]	887
17.135.1.6 FFLAS_ELT [3/6]	888
17.135.1.7 FFLAS_FIELD [2/2]	888
17.135.1.8 FFLAS_ELT [4/6]	888
17.135.1.9 FFLAS_ELT [5/6]	888
17.135.1.10 FFLAS_ELT [6/6]	888
17.136 fflas_L3_inst.h File Reference	888
17.136.1 Macro Definition Documentation	888
17.136.1.1 INST_OR_DECL	888
17.136.1.2 FFLAS_FIELD [1/2]	888
17.136.1.3 FFLAS_ELT [1/6]	889
17.136.1.4 FFLAS_ELT [2/6]	889
17.136.1.5 FFLAS_ELT [3/6]	889
17.136.1.6 FFLAS_FIELD [2/2]	889
17.136.1.7 FFLAS_ELT [4/6]	889
17.136.1.8 FFLAS_ELT [5/6]	889
17.136.1.9 FFLAS_ELT [6/6]	889
17.137 fflas_L3_inst_implem.inl File Reference	889
17.137.1 Macro Definition Documentation	890
17.137.1.1 __FFLAS__TRSM_READONLY	890
17.138 fflas_level1.inl File Reference	890

17.138.1 Macro Definition Documentation	892
17.138.1.1 __FFLASFFPACK_fflas_fflas_level1_INL	892
17.139 fflas_level2.inl File Reference	892
17.139.1 Macro Definition Documentation	895
17.139.1.1 __FFLASFFPACK_fflas_fflas_level2_INL	895
17.140 fflas_level3.inl File Reference	895
17.140.1 Macro Definition Documentation	897
17.140.1.1 __FFLASFFPACK_fflas_fflas_level3_INL	897
17.140.1.2 __FFLAS__TRSM_READONLY	897
17.141 fflas_lvl1.C File Reference	897
17.141.1 Detailed Description	898
17.141.2 Function Documentation	898
17.141.2.1 freducein_1_modular_double()	899
17.141.2.2 freduce_1_modular_double()	899
17.141.2.3 fnegin_1_modular_double()	899
17.141.2.4 fneg_1_modular_double()	899
17.141.2.5 fzero_1_modular_double()	899
17.141.2.6 fiszero_1_modular_double()	899
17.141.2.7 fequal_1_modular_double()	900
17.141.2.8 fassign_1_modular_double()	900
17.141.2.9 fscaln_1_modular_double()	900
17.141.2.10 fscal_1_modular_double()	900
17.141.2.11 faxpy_1_modular_double()	900
17.141.2.12 fdot_1_modular_double()	901
17.141.2.13 fswap_1_modular_double()	901
17.141.2.14 fadd_1_modular_double()	901
17.141.2.15 fsub_1_modular_double()	901
17.141.2.16 faddn_1_modular_double()	902
17.141.2.17 fsubn_1_modular_double()	902
17.142 fflas_lvl2.C File Reference	902
17.142.1 Detailed Description	903
17.142.2 Function Documentation	903
17.142.2.1 fassign_2_modular_double()	903
17.142.2.2 fzero_2_modular_double()	903
17.142.2.3 fequal_2_modular_double()	904
17.142.2.4 fiszero_2_modular_double()	904
17.142.2.5 fidentity_2_modular_double()	904
17.142.2.6 freducein_2_modular_double()	904
17.142.2.7 freduce_2_modular_double()	904
17.142.2.8 fnegin_2_modular_double()	905
17.142.2.9 fneg_2_modular_double()	905
17.142.2.10 fscaln_2_modular_double()	905

17.142.2.11 fscal_2_modular_double()	905
17.142.2.12 faxpy_2_modular_double()	905
17.142.2.13 fmove_2_modular_double()	906
17.142.2.14 fadd_2_modular_double()	906
17.142.2.15 fsub_2_modular_double()	906
17.142.2.16 fsubin_2_modular_double()	906
17.142.2.17 faddin_2_modular_double()	907
17.142.2.18 fgemv_2_modular_double()	907
17.142.2.19 fger_2_modular_double()	907
17.142.2.20 ftrsv_2_modular_double()	907
17.143 fflas_lvl3.C File Reference	908
17.143.1 Detailed Description	908
17.143.2 Function Documentation	908
17.143.2.1 ftrsm_3_modular_double()	908
17.143.2.2 ftrmm_3_modular_double()	909
17.143.2.3 fgemm_3_modular_double()	909
17.143.2.4 fsquare_3_modular_double()	909
17.144 fflas_memory.h File Reference	910
17.145 fflas_pfgemm.inl File Reference	910
17.145.1 Macro Definition Documentation	911
17.145.1.1 __FFLASFFPACK_fflas_pfgemm_INL	911
17.145.1.2 __FFLASFFPACK_SEQPARTHRESHOLD	911
17.145.1.3 __FFLASFFPACK_DIMKPENALTY	911
17.146 fflas_pftrsm.inl File Reference	911
17.146.1 Macro Definition Documentation	911
17.146.1.1 __FFLASFFPACK_fflas_pftrsm_INL	911
17.146.1.2 PTRSM_HYBRID_THRESHOLD	912
17.147 fflas_plevel1.h File Reference	912
17.148 fflas_randommatrix.h File Reference	912
17.149 fflas_simd.h File Reference	914
17.149.1 Macro Definition Documentation	915
17.149.1.1 SIMD_INT	915
17.149.1.2 INLINE	915
17.149.1.3 CONST	915
17.149.1.4 PURE	915
17.149.1.5 NORML_MOD	915
17.149.1.6 FLOAT_MOD	916
17.149.2 Typedef Documentation	916
17.149.2.1 Simd	916
17.150 fflas_sparse.C File Reference	916
17.150.1 Detailed Description	916
17.151 fflas_sparse.h File Reference	916

17.151.1 Macro Definition Documentation	920
17.151.1.1 index_t	920
17.151.1.2 ROUND_DOWN	920
17.151.1.3 __FFLASFFPACK_CACHE_LINE_SIZE	920
17.151.1.4 assume_aligned	920
17.151.1.5 DENSE_THRESHOLD	921
17.152 fflas_sparse.inl File Reference	921
17.152.1 Macro Definition Documentation	923
17.152.1.1 __FFLASFFPACK_fflas_fflas_sparse_INL	923
17.153 fpack-fgesv.C File Reference	923
17.153.1 Function Documentation	923
17.153.1.1 main()	923
17.154 fpack-solve.C File Reference	923
17.154.1 Function Documentation	923
17.154.1.1 main()	924
17.155 fpack.C File Reference	924
17.155.1 Detailed Description	927
17.155.2 Function Documentation	927
17.155.2.1 LAPACKPerm2MathPerm()	927
17.155.2.2 MathPerm2LAPACKPerm()	927
17.155.2.3 MatrixApplyS_modular_double()	927
17.155.2.4 PermApplyS_double()	928
17.155.2.5 MatrixApplyT_modular_double()	928
17.155.2.6 PermApplyT_double()	928
17.155.2.7 composePermutationsLLM()	928
17.155.2.8 composePermutationsLLL()	929
17.155.2.9 composePermutationsMLM()	929
17.155.2.10 cyclic_shift_mathPerm()	929
17.155.2.11 cyclic_shift_row_modular_double()	929
17.155.2.12 cyclic_shift_col_modular_double()	929
17.155.2.13 applyP_modular_double()	929
17.155.2.14 fgetrsin_modular_double()	930
17.155.2.15 fgetrsv_modular_double()	930
17.155.2.16 fgesvin_modular_double()	930
17.155.2.17 fgesv_modular_double()	931
17.155.2.18 ftrtri_modular_double()	931
17.155.2.19 trinv_left_modular_double()	931
17.155.2.20 ftrtrm_modular_double()	931
17.155.2.21 PLUQ_modular_double()	931
17.155.2.22 LUdivine_modular_double()	932
17.155.2.23 ColumnEchelonForm_modular_double()	932
17.155.2.24 RowEchelonForm_modular_double()	932



17.155.2.25 ReducedColumnEchelonForm_modular_double()	932
17.155.2.26 ReducedRowEchelonForm_modular_double()	933
17.155.2.27 ColumnEchelonForm_modular_float()	933
17.155.2.28 RowEchelonForm_modular_float()	933
17.155.2.29 ReducedColumnEchelonForm_modular_float()	934
17.155.2.30 ReducedRowEchelonForm_modular_float()	934
17.155.2.31 ColumnEchelonForm_modular_int32_t()	934
17.155.2.32 RowEchelonForm_modular_int32_t()	934
17.155.2.33 ReducedColumnEchelonForm_modular_int32_t()	935
17.155.2.34 ReducedRowEchelonForm_modular_int32_t()	935
17.155.2.35 pColumnEchelonForm_modular_double()	935
17.155.2.36 pRowEchelonForm_modular_double()	935
17.155.2.37 pReducedColumnEchelonForm_modular_double()	936
17.155.2.38 pReducedRowEchelonForm_modular_double()	936
17.155.2.39 pColumnEchelonForm_modular_float()	936
17.155.2.40 pRowEchelonForm_modular_float()	936
17.155.2.41 pReducedColumnEchelonForm_modular_float()	937
17.155.2.42 pReducedRowEchelonForm_modular_float()	937
17.155.2.43 pColumnEchelonForm_modular_int32_t()	937
17.155.2.44 pRowEchelonForm_modular_int32_t()	937
17.155.2.45 pReducedColumnEchelonForm_modular_int32_t()	938
17.155.2.46 pReducedRowEchelonForm_modular_int32_t()	938
17.155.2.47 Invertin_modular_double()	938
17.155.2.48 Invert_modular_double()	938
17.155.2.49 Invert2_modular_double()	938
17.155.2.50 KrylovElim_modular_double()	939
17.155.2.51 SpecRankProfile_modular_double()	939
17.155.2.52 Rank_modular_double()	939
17.155.2.53 IsSingular_modular_double()	939
17.155.2.54 Det_modular_double()	940
17.155.2.55 Solve_modular_double()	940
17.155.2.56 solveLB_modular_double()	940
17.155.2.57 solveLB2_modular_double()	940
17.155.2.58 RandomNullSpaceVector_modular_double()	941
17.155.2.59 NullSpaceBasis_modular_double()	941
17.155.2.60 RowRankProfile_modular_double()	941
17.155.2.61 ColumnRankProfile_modular_double()	941
17.155.2.62 RankProfileFromLU()	941
17.155.2.63 LeadingSubmatrixRankProfiles()	942
17.155.2.64 RowRankProfileSubmatrixIndices_modular_double()	942
17.155.2.65 ColRankProfileSubmatrixIndices_modular_double()	942
17.155.2.66 RowRankProfileSubmatrix_modular_double()	942

17.155.2.67 ColRankProfileSubmatrix_modular_double()	943
17.155.2.68 getTriangular_modular_double()	943
17.155.2.69 getTriangularin_modular_double()	943
17.155.2.70 getEchelonForm_modular_double()	943
17.155.2.71 getEchelonFormin_modular_double()	944
17.155.2.72 getEchelonTransform_modular_double()	944
17.155.2.73 getReducedEchelonForm_modular_double()	944
17.155.2.74 getReducedEchelonFormin_modular_double()	945
17.155.2.75 getReducedEchelonTransform_modular_double()	945
17.155.2.76 PLUQtoEchelonPermutation()	945
17.156 ffpack.dox File Reference	945
17.157 ffpack.h File Reference	945
17.157.1 Detailed Description	953
17.157.2 Macro Definition Documentation	953
17.157.2.1 __FFLASFFPACK_FTRSTR_THRESHOLD	953
17.157.2.2 __FFLASFFPACK_FTRSSYR2K_THRESHOLD	953
17.158 ffpack.inl File Reference	954
17.158.1 Macro Definition Documentation	955
17.158.1.1 __FFLASFFPACK_ffpack_INL	955
17.159 ffpack_c.h File Reference	955
17.159.1 Macro Definition Documentation	958
17.159.1.1 FFPACK_COMPILED	958
17.159.2 Enumeration Type Documentation	958
17.159.2.1 FFLAS_C_ORDER	958
17.159.2.2 FFLAS_C_TRANSPOSE	958
17.159.2.3 FFLAS_C_UPLO	959
17.159.2.4 FFLAS_C_DIAG	959
17.159.2.5 FFLAS_C_SIDE	959
17.159.2.6 FFPACK_C_LU_TAG	959
17.159.2.7 FFPACK_C_CHARPOLY_TAG	960
17.159.2.8 FFPACK_C_MINPOLY_TAG	960
17.159.3 Function Documentation	960
17.159.3.1 LAPACKPerm2MathPerm()	960
17.159.3.2 MathPerm2LAPACKPerm()	960
17.159.3.3 MatrixApplyS_modular_double()	960
17.159.3.4 PermApplyS_double()	961
17.159.3.5 MatrixApplyT_modular_double()	961
17.159.3.6 PermApplyT_double()	961
17.159.3.7 composePermutationsLLM()	961
17.159.3.8 composePermutationsLLL()	962
17.159.3.9 composePermutationsMLM()	962
17.159.3.10 cyclic_shift_mathPerm()	962

17.159.3.11 cyclic_shift_row_modular_double()	962
17.159.3.12 cyclic_shift_col_modular_double()	962
17.159.3.13 applyP_modular_double()	962
17.159.3.14 fgetrsin_modular_double()	963
17.159.3.15 fgetrs_modular_double()	963
17.159.3.16 fgesvin_modular_double()	963
17.159.3.17 fgesv_modular_double()	964
17.159.3.18 ftrtri_modular_double()	964
17.159.3.19 trinv_left_modular_double()	964
17.159.3.20 ftrtm_modular_double()	964
17.159.3.21 PLUQ_modular_double()	964
17.159.3.22 LUdivine_modular_double()	965
17.159.3.23 LUdivine_small_modular_double()	965
17.159.3.24 LUdivine_gauss_modular_double()	965
17.159.3.25 ColumnEchelonForm_modular_double()	965
17.159.3.26 RowEchelonForm_modular_double()	966
17.159.3.27 ColumnEchelonForm_modular_float()	966
17.159.3.28 RowEchelonForm_modular_float()	966
17.159.3.29 ColumnEchelonForm_modular_int32_t()	966
17.159.3.30 RowEchelonForm_modular_int32_t()	967
17.159.3.31 ReducedColumnEchelonForm_modular_double()	967
17.159.3.32 ReducedRowEchelonForm_modular_double()	967
17.159.3.33 ReducedColumnEchelonForm_modular_float()	968
17.159.3.34 ReducedRowEchelonForm_modular_float()	968
17.159.3.35 ReducedColumnEchelonForm_modular_int32_t()	968
17.159.3.36 ReducedRowEchelonForm_modular_int32_t()	968
17.159.3.37 ReducedRowEchelonForm2_modular_double()	969
17.159.3.38 REF_modular_double()	969
17.159.3.39 Invertin_modular_double()	969
17.159.3.40 Invert_modular_double()	969
17.159.3.41 Invert2_modular_double()	969
17.159.3.42 KrylovElim_modular_double()	970
17.159.3.43 SpecRankProfile_modular_double()	970
17.159.3.44 Rank_modular_double()	970
17.159.3.45 IsSingular_modular_double()	970
17.159.3.46 Det_modular_double()	971
17.159.3.47 Solve_modular_double()	971
17.159.3.48 solveLB_modular_double()	971
17.159.3.49 solveLB2_modular_double()	971
17.159.3.50 RandomNullSpaceVector_modular_double()	972
17.159.3.51 NullSpaceBasis_modular_double()	972
17.159.3.52 RowRankProfile_modular_double()	972

17.159.3.53 ColumnRankProfile_modular_double()	972
17.159.3.54 RankProfileFromLU()	972
17.159.3.55 LeadingSubmatrixRankProfiles()	973
17.159.3.56 RowRankProfileSubmatrixIndices_modular_double()	973
17.159.3.57 ColRankProfileSubmatrixIndices_modular_double()	973
17.159.3.58 RowRankProfileSubmatrix_modular_double()	973
17.159.3.59 ColRankProfileSubmatrix_modular_double()	974
17.159.3.60 getTriangular_modular_double()	974
17.159.3.61 getTriangularin_modular_double()	974
17.159.3.62 getEchelonForm_modular_double()	974
17.159.3.63 getEchelonFormin_modular_double()	975
17.159.3.64 getEchelonTransform_modular_double()	975
17.159.3.65 getReducedEchelonForm_modular_double()	975
17.159.3.66 getReducedEchelonFormin_modular_double()	976
17.159.3.67 getReducedEchelonTransform_modular_double()	976
17.159.3.68 PLUQtoEchelonPermutation()	976
17.160 ffpack_charpoly.inl File Reference	976
17.160.1 Macro Definition Documentation	977
17.160.1.1 __FFLASFFPACK_charpoly_INL	977
17.161 ffpack_charpoly_danilevski.inl File Reference	977
17.161.1 Macro Definition Documentation	977
17.161.1.1 __FFLASFFPACK_ffpack_charpoly_danilveski_INL	977
17.162 ffpack_charpoly_kgfast.inl File Reference	978
17.162.1 Macro Definition Documentation	978
17.162.1.1 __FFLASFFPACK_ffpack_charpoly_kgfast_INL	978
17.163 ffpack_charpoly_kgfastgeneralized.inl File Reference	978
17.163.1 Macro Definition Documentation	979
17.163.1.1 __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL	979
17.164 ffpack_charpoly_kglu.inl File Reference	979
17.164.1 Macro Definition Documentation	979
17.164.1.1 __FFLASFFPACK_ffpack_charpoly_kglu_INL	979
17.165 ffpack_charpoly_mp.inl File Reference	979
17.165.1 Macro Definition Documentation	980
17.165.1.1 __FFPACK_charpoly_mp_INL	980
17.166 ffpack_det_mp.inl File Reference	980
17.166.1 Macro Definition Documentation	980
17.166.1.1 __FFPACK_det_mp_INL	980
17.167 ffpack_echelonforms.inl File Reference	981
17.167.1 Macro Definition Documentation	982
17.167.1.1 __FFLASFFPACK_ffpack_echelon_forms_INL	982
17.167.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE	982
17.168 ffpack_fgesv.inl File Reference	982

17.168.1 Macro Definition Documentation . . . . .	982
17.168.1.1 __FFLASFFPACK_ffpack_fgesv_INL . . . . .	982
17.169 fpack_fgetrs.inl File Reference . . . . .	983
17.169.1 Macro Definition Documentation . . . . .	983
17.169.1.1 __FFLASFFPACK_ffpack_fgetrs_INL . . . . .	983
17.170 fpack_frobenius.inl File Reference . . . . .	983
17.171 fpack_fsytrf.inl File Reference . . . . .	984
17.171.1 Macro Definition Documentation . . . . .	985
17.171.1.1 __FFLASFFPACK_ffpack_fsytrf_INL . . . . .	985
17.172 fpack_ftrssyr2k.inl File Reference . . . . .	985
17.172.1 Macro Definition Documentation . . . . .	985
17.172.1.1 __FFLASFFPACK_ffpack_ftrssyr2k_INL . . . . .	986
17.173 fpack_ftrstr.inl File Reference . . . . .	986
17.173.1 Macro Definition Documentation . . . . .	986
17.173.1.1 __FFLASFFPACK_ffpack_ftrstr_INL . . . . .	986
17.174 fpack_ftrtr.inl File Reference . . . . .	986
17.174.1 Macro Definition Documentation . . . . .	987
17.174.1.1 ENABLE_ALL_CHECKINGS . . . . .	987
17.174.1.2 __FFLASFFPACK_ffpack_ftrtr_INL . . . . .	987
17.175 fpack_inst.C File Reference . . . . .	987
17.175.1 Macro Definition Documentation . . . . .	987
17.175.1.1 __FFPACK_INST_C . . . . .	987
17.175.1.2 FFLAS_COMPILED . . . . .	987
17.175.1.3 INST_OR_DECL . . . . .	988
17.175.1.4 FFLAS_FIELD [1/2] . . . . .	988
17.175.1.5 FFLAS_ELT [1/6] . . . . .	988
17.175.1.6 FFLAS_ELT [2/6] . . . . .	988
17.175.1.7 FFLAS_ELT [3/6] . . . . .	988
17.175.1.8 FFLAS_FIELD [2/2] . . . . .	988
17.175.1.9 FFLAS_ELT [4/6] . . . . .	988
17.175.1.10 FFLAS_ELT [5/6] . . . . .	988
17.175.1.11 FFLAS_ELT [6/6] . . . . .	988
17.176 fpack_inst.h File Reference . . . . .	988
17.176.1 Macro Definition Documentation . . . . .	989
17.176.1.1 FFLAS_COMPILED . . . . .	989
17.176.1.2 INST_OR_DECL . . . . .	989
17.176.1.3 FFLAS_FIELD [1/2] . . . . .	989
17.176.1.4 FFLAS_ELT [1/6] . . . . .	989
17.176.1.5 FFLAS_ELT [2/6] . . . . .	989
17.176.1.6 FFLAS_ELT [3/6] . . . . .	989
17.176.1.7 FFLAS_FIELD [2/2] . . . . .	989
17.176.1.8 FFLAS_ELT [4/6] . . . . .	989

17.176.1.9 FFLAS_ELT [5/6]	989
17.176.1.10 FFLAS_ELT [6/6]	989
17.177 ffpack_inst_implem.inl File Reference	990
17.178 ffpack_invert.inl File Reference	993
17.178.1 Macro Definition Documentation	993
17.178.1.1 __FFLASFFPACK_ffpack_invert_INL	993
17.179 ffpack_krylovelim.inl File Reference	993
17.179.1 Macro Definition Documentation	993
17.179.1.1 __FFLASFFPACK_ffpack_krylovelim_INL	994
17.180 ffpack_ludivine.inl File Reference	994
17.180.1 Macro Definition Documentation	994
17.180.1.1 __FFLASFFPACK_ffpack_ludivine_INL	994
17.181 ffpack_ludivine_mp.inl File Reference	995
17.181.1 Macro Definition Documentation	995
17.181.1.1 __FFPACK_ludivine_mp_INL	995
17.182 ffpack_minpoly.inl File Reference	995
17.182.1 Macro Definition Documentation	996
17.182.1.1 __FFLASFFPACK_ffpack_minpoly_INL	996
17.183 ffpack_permutation.inl File Reference	996
17.183.1 Macro Definition Documentation	998
17.183.1.1 __FFLASFFPACK_ffpack_permutation_INL	998
17.183.1.2 FFLASFFPACK_PERM_BKSIZE	998
17.184 ffpack_pluq.inl File Reference	998
17.184.1 Macro Definition Documentation	999
17.184.1.1 __FFLASFFPACK_ffpack_pluq_INL	999
17.184.1.2 CROUT	999
17.185 ffpack_pluq_mp.inl File Reference	999
17.185.1 Macro Definition Documentation	1000
17.185.1.1 __FFPACK_pluq_mp_INL	1000
17.186 ffpack_ppluq.inl File Reference	1000
17.186.1 Macro Definition Documentation	1000
17.186.1.1 __FFLASFFPACK_ffpack_ppluq_INL	1000
17.186.1.2 __FFLAS__TRSM_READONLY	1000
17.186.1.3 PBASECASE_K	1000
17.187 ffpack_rankprofiles.inl File Reference	1001
17.187.1 Macro Definition Documentation	1002
17.187.1.1 __FFLASFFPACK_ffpack_rank_profiles_INL	1002
17.188 fgemm_classical.inl File Reference	1002
17.188.1 Macro Definition Documentation	1002
17.188.1.1 __FFLASFFPACK_fflas_fflas_fgemm_classical_INL	1002
17.189 fgemm_classical_mp.inl File Reference	1002
17.189.1 Detailed Description	1004

17.189.2 Macro Definition Documentation	1004
17.189.2.1 __FFPACK_fgemm_classical_INL	1004
17.190 fgemm_winograd.inl File Reference	1004
17.190.1 Macro Definition Documentation	1005
17.190.1.1 __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL	1005
17.190.1.2 NEWWINO	1005
17.191 field-traits.h File Reference	1006
17.191.1 Detailed Description	1008
17.192 field.doxy File Reference	1008
17.193 flimits.h File Reference	1008
17.193.1 Function Documentation	1009
17.193.1.1 in_range() [1/3]	1009
17.193.1.2 in_range() [2/3]	1009
17.193.1.3 in_range() [3/3]	1009
17.194 fsyrk.C File Reference	1009
17.194.1 Macro Definition Documentation	1009
17.194.1.1 CUBE	1009
17.194.1.2 GFOPS	1010
17.194.2 Typedef Documentation	1010
17.194.2.1 TTimer	1010
17.194.3 Function Documentation	1010
17.194.3.1 main()	1010
17.195 fsytrf.C File Reference	1010
17.195.1 Macro Definition Documentation	1010
17.195.1.1 CUBE	1010
17.195.1.2 GFOPS	1011
17.195.2 Typedef Documentation	1011
17.195.2.1 TTimer	1011
17.195.3 Function Documentation	1011
17.195.3.1 main()	1011
17.196 ftrtri.C File Reference	1011
17.196.1 Macro Definition Documentation	1011
17.196.1.1 CUBE	1011
17.196.1.2 GFOPS	1012
17.196.2 Typedef Documentation	1012
17.196.2.1 TTimer	1012
17.196.3 Function Documentation	1012
17.196.3.1 main()	1012
17.197 gmp.C File Reference	1012
17.197.1 Function Documentation	1012
17.197.1.1 main()	1012
17.198 hyb_zo.h File Reference	1012

17.199	hyb_zo_pspmm.inl File Reference	1013
17.199.1	Macro Definition Documentation	1013
17.199.1.1	__FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL	1013
17.200	hyb_zo_pspmv.inl File Reference	1013
17.200.1	Macro Definition Documentation	1013
17.200.1.1	__FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL	1014
17.201	hyb_zo_spmmm.inl File Reference	1014
17.201.1	Macro Definition Documentation	1014
17.201.1.1	__FFLASFFPACK_fflas_sparse_HYB_ZO_spmmm_INL	1014
17.202	hyb_zo_spmv.inl File Reference	1014
17.202.1	Macro Definition Documentation	1015
17.202.1.1	__FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL	1015
17.203	hyb_zo_utils.inl File Reference	1015
17.203.1	Macro Definition Documentation	1015
17.203.1.1	__FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL	1015
17.204	igemm.doxy File Reference	1015
17.205	igemm.h File Reference	1015
17.206	igemm.inl File Reference	1016
17.206.1	Macro Definition Documentation	1016
17.206.1.1	__FFLASFFPACK_fflas_igemm_igemm_INL	1016
17.207	igemm_kernels.h File Reference	1016
17.208	igemm_kernels.inl File Reference	1017
17.208.1	Macro Definition Documentation	1018
17.208.1.1	__FFLASFFPACK_fflas_igemm_igemm_kernels_INL	1018
17.209	igemm_tools.h File Reference	1018
17.210	igemm_tools.inl File Reference	1018
17.210.1	Macro Definition Documentation	1019
17.210.1.1	__FFLASFFPACK_fflas_igemm_igemm_tools_INL	1019
17.211	instrset.h File Reference	1019
17.211.1	Macro Definition Documentation	1020
17.211.1.1	INSTRSET_H	1020
17.211.1.2	INSTRSET	1020
17.211.1.3	const_int	1020
17.211.1.4	const_uint	1020
17.211.2	Typedef Documentation	1020
17.211.2.1	int8_t	1020
17.211.2.2	uint8_t	1020
17.211.2.3	int16_t	1020
17.211.2.4	uint16_t	1020
17.211.2.5	int32_t	1020
17.211.2.6	uint32_t	1020
17.211.2.7	int64_t	1021



17.211.2.8 uint64_t . . . . .	1021
17.211.2.9 intptr_t . . . . .	1021
17.211.3 Function Documentation . . . . .	1021
17.211.3.1 instrset_detect() . . . . .	1021
17.211.3.2 hasFMA3() . . . . .	1021
17.211.3.3 hasFMA4() . . . . .	1021
17.211.3.4 hasXOP() . . . . .	1021
17.211.3.5 hasAVX512ER() . . . . .	1021
17.212 instrset_detect.cpp File Reference . . . . .	1021
17.212.1 Function Documentation . . . . .	1022
17.212.1.1 instrset_detect() . . . . .	1022
17.212.1.2 hasFMA3() . . . . .	1022
17.212.1.3 hasFMA4() . . . . .	1022
17.212.1.4 hasXOP() . . . . .	1022
17.212.1.5 hasF16C() . . . . .	1022
17.212.1.6 hasAVX512ER() . . . . .	1022
17.213 interfaces.doxy File Reference . . . . .	1022
17.214 kaapi_routines.inl File Reference . . . . .	1022
17.214.1 Macro Definition Documentation . . . . .	1022
17.214.1.1 __FFLASFFPACK_KAAPI_ROUTINES_INL . . . . .	1022
17.215 lapack.C File Reference . . . . .	1022
17.215.1 Macro Definition Documentation . . . . .	1023
17.215.1.1 __FFLASFFPACK_CONFIGURATION . . . . .	1023
17.215.1.2 __FFLASFFPACK_HAVE_LAPACK . . . . .	1023
17.215.2 Function Documentation . . . . .	1023
17.215.2.1 main() . . . . .	1023
17.216 mainpage.doxy File Reference . . . . .	1023
17.217 Matio.h File Reference . . . . .	1023
17.217.1 Function Documentation . . . . .	1023
17.217.1.1 read_field() . . . . .	1024
17.217.1.2 write_field() . . . . .	1024
17.218 matmul.C File Reference . . . . .	1024
17.218.1 Function Documentation . . . . .	1024
17.218.1.1 main() . . . . .	1024
17.219 matmul.doxy File Reference . . . . .	1024
17.220 parallel.h File Reference . . . . .	1024
17.220.1 Macro Definition Documentation . . . . .	1025
17.220.1.1 __FFLASFFPACK_SEQUENTIAL . . . . .	1025
17.220.1.2 index_t . . . . .	1025
17.220.1.3 TASK . . . . .	1026
17.220.1.4 WAIT . . . . .	1026
17.220.1.5 CHECK_DEPENDENCIES . . . . .	1026

17.220.1.6 BARRIER . . . . .	1026
17.220.1.7 PAR_BLOCK . . . . .	1026
17.220.1.8 SYNCH_GROUP . . . . .	1026
17.220.1.9 NUM_THREADS . . . . .	1026
17.220.1.10 MAX_THREADS . . . . .	1026
17.220.1.11 READ . . . . .	1026
17.220.1.12 WRITE . . . . .	1026
17.220.1.13 READWRITE . . . . .	1026
17.220.1.14 CONSTREFERENCE . . . . .	1027
17.220.1.15 VALUE . . . . .	1027
17.220.1.16 BEGIN_PARALLEL_MAIN . . . . .	1027
17.220.1.17 END_PARALLEL_MAIN . . . . .	1027
17.220.1.18 FORBLOCK1D . . . . .	1027
17.220.1.19 FOR1D . . . . .	1027
17.220.1.20 PARFORBLOCK1D . . . . .	1027
17.220.1.21 PARFOR1D . . . . .	1028
17.220.1.22 FORBLOCK2D . . . . .	1028
17.220.1.23 FOR2D . . . . .	1028
17.220.1.24 PARFORBLOCK2D . . . . .	1028
17.220.1.25 PARFOR2D . . . . .	1028
17.220.1.26 COMMA . . . . .	1029
17.220.1.27 MODE . . . . .	1029
17.220.1.28 RETURNPARAM . . . . .	1029
17.220.1.29 NUMARGS . . . . .	1029
17.220.1.30 PP_NARG_ . . . . .	1029
17.220.1.31 PP_ARG_N . . . . .	1029
17.220.1.32 PP_RSEQ_N . . . . .	1030
17.220.1.33 NOSPLIT . . . . .	1030
17.220.1.34 splitting_0 . . . . .	1031
17.220.1.35 splitting_1 . . . . .	1031
17.220.1.36 splitting_2 . . . . .	1031
17.220.1.37 splitting_3 . . . . .	1031
17.220.1.38 splitt . . . . .	1031
17.220.1.39 SPLITTER . . . . .	1031
17.221 pfgemm_variants.inl File Reference . . . . .	1031
17.222 pfgemv.inl File Reference . . . . .	1032
17.223 pluq.C File Reference . . . . .	1032
17.223.1 Macro Definition Documentation . . . . .	1033
17.223.1.1 CUBE . . . . .	1033
17.223.1.2 GFOPS . . . . .	1033
17.223.2 Typedef Documentation . . . . .	1033
17.223.2.1 TTimer . . . . .	1033

17.223.3 Function Documentation	1033
17.223.3.1 main()	1033
17.224 pluq.C File Reference	1034
17.224.1 Function Documentation	1034
17.224.1.1 main()	1034
17.225 rank.C File Reference	1034
17.225.1 Function Documentation	1034
17.225.1.1 main()	1034
17.226 read_sparse.h File Reference	1034
17.226.1 Macro Definition Documentation	1035
17.226.1.1 DNS_BIN_VER	1035
17.226.1.2 mask_t	1035
17.227 regression-check.C File Reference	1036
17.227.1 Function Documentation	1036
17.227.1.1 check1()	1036
17.227.1.2 check2()	1036
17.227.1.3 check3()	1036
17.227.1.4 check4()	1036
17.227.1.5 checkZeroDimCharpoly()	1036
17.227.1.6 checkZeroDimMinPoly()	1036
17.227.1.7 gf2ModularBalanced()	1036
17.227.1.8 main()	1037
17.228 rns-double-elt.h File Reference	1037
17.228.1 Detailed Description	1037
17.229 rns-double-recint.inl File Reference	1037
17.229.1 Macro Definition Documentation	1037
17.229.1.1 __FFLASFFPACK_field_rns_double_recint_INL	1037
17.230 rns-double.h File Reference	1038
17.230.1 Detailed Description	1038
17.230.2 Macro Definition Documentation	1038
17.230.2.1 ROUND_DOWN	1038
17.231 rns-double.inl File Reference	1038
17.231.1 Macro Definition Documentation	1039
17.231.1.1 __FFLASFFPACK_field_rns_double_INL	1039
17.232 rns-integer-mod.h File Reference	1039
17.232.1 Detailed Description	1040
17.233 rns-integer.h File Reference	1040
17.233.1 Detailed Description	1040
17.234 rns.h File Reference	1040
17.235 rns.inl File Reference	1041
17.235.1 Macro Definition Documentation	1041
17.235.1.1 __FFLASFFPACK_field_rns_INL	1041

17.236	<a href="#">schedule_bini.inl File Reference</a>	1041
17.236.1	<a href="#">Detailed Description</a>	1041
17.236.2	<a href="#">Macro Definition Documentation</a>	1041
17.236.2.1	<a href="#">__FFLASFFPACK_fgemm_bini_INL</a>	1041
17.237	<a href="#">schedule_winograd.inl File Reference</a>	1042
17.237.1	<a href="#">Macro Definition Documentation</a>	1042
17.237.1.1	<a href="#">__FFLASFFPACK_fgemm_winograd_INL</a>	1042
17.238	<a href="#">schedule_winograd_acc.inl File Reference</a>	1042
17.238.1	<a href="#">Macro Definition Documentation</a>	1043
17.238.1.1	<a href="#">__FFLASFFPACK_fgemm_winograd_acc_INL</a>	1043
17.239	<a href="#">schedule_winograd_acc_ip.inl File Reference</a>	1043
17.239.1	<a href="#">Macro Definition Documentation</a>	1043
17.239.1.1	<a href="#">__FFLASFFPACK_fgemm_winograd_acc_ip_INL</a>	1044
17.240	<a href="#">schedule_winograd_ip.inl File Reference</a>	1044
17.240.1	<a href="#">Macro Definition Documentation</a>	1044
17.240.1.1	<a href="#">__FFLASFFPACK_fgemm_winograd_ip_INL</a>	1044
17.241	<a href="#">sell.h File Reference</a>	1044
17.242	<a href="#">sell_pspmv.inl File Reference</a>	1045
17.242.1	<a href="#">Macro Definition Documentation</a>	1045
17.242.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_sell_pspmv_INL</a>	1045
17.243	<a href="#">sell_spmv.inl File Reference</a>	1045
17.243.1	<a href="#">Macro Definition Documentation</a>	1046
17.243.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_sell_spmv_INL</a>	1046
17.244	<a href="#">sell_utils.inl File Reference</a>	1046
17.244.1	<a href="#">Macro Definition Documentation</a>	1047
17.244.1.1	<a href="#">__FFLASFFPACK_fflas_sparse_sell_utils_INL</a>	1047
17.245	<a href="#">simd.doxy File Reference</a>	1047
17.246	<a href="#">simd128.inl File Reference</a>	1047
17.246.1	<a href="#">Macro Definition Documentation</a>	1047
17.246.1.1	<a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_INL</a>	1048
17.246.2	<a href="#">Typedef Documentation</a>	1048
17.246.2.1	<a href="#">Simd128</a>	1048
17.247	<a href="#">simd128_double.inl File Reference</a>	1048
17.247.1	<a href="#">Macro Definition Documentation</a>	1048
17.247.1.1	<a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL</a>	1048
17.248	<a href="#">simd128_float.inl File Reference</a>	1048
17.248.1	<a href="#">Macro Definition Documentation</a>	1048
17.248.1.1	<a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL</a>	1048
17.249	<a href="#">simd128_int16.inl File Reference</a>	1048
17.249.1	<a href="#">Macro Definition Documentation</a>	1049
17.249.1.1	<a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL</a>	1049
17.250	<a href="#">simd128_int32.inl File Reference</a>	1049

17.250.1 Macro Definition Documentation	1049
17.250.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL</a>	1049
17.251 simd128_int64.inl File Reference	1049
17.251.1 Macro Definition Documentation	1049
17.251.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL</a>	1049
17.251.1.2 vect_t	1050
17.252 simd256.inl File Reference	1050
17.252.1 Macro Definition Documentation	1050
17.252.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_INL</a>	1050
17.252.2 Typedef Documentation	1050
17.252.2.1 Simd256	1050
17.253 simd256_double.inl File Reference	1050
17.253.1 Macro Definition Documentation	1050
17.253.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL</a>	1051
17.254 simd256_float.inl File Reference	1051
17.254.1 Macro Definition Documentation	1051
17.254.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL</a>	1051
17.255 simd256_int16.inl File Reference	1051
17.255.1 Macro Definition Documentation	1051
17.255.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL</a>	1051
17.256 simd256_int32.inl File Reference	1051
17.256.1 Macro Definition Documentation	1052
17.256.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL</a>	1052
17.257 simd256_int64.inl File Reference	1052
17.257.1 Macro Definition Documentation	1052
17.257.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL</a>	1052
17.257.1.2 vect_t	1052
17.258 simd512.inl File Reference	1052
17.258.1 Macro Definition Documentation	1053
17.258.1.1 <a href="#">__FFLASFFPACK_simd512_INL</a>	1053
17.258.2 Typedef Documentation	1053
17.258.2.1 Simd512	1053
17.259 simd512_double.inl File Reference	1053
17.259.1 Macro Definition Documentation	1053
17.259.1.1 <a href="#">__FFLASFFPACK_simd512_double_INL</a>	1053
17.260 simd512_float.inl File Reference	1053
17.260.1 Macro Definition Documentation	1053
17.260.1.1 <a href="#">__FFLASFFPACK_simd512_float_INL</a>	1053
17.261 simd512_int32.inl File Reference	1053
17.261.1 Macro Definition Documentation	1054
17.261.1.1 <a href="#">__FFLASFFPACK_simd512_int32_INL</a>	1054
17.262 simd512_int64.inl File Reference	1054

17.262.1 Macro Definition Documentation	1054
17.262.1.1 _simd512_int64_INL	1054
17.262.1.2 vect_t	1054
17.263 simd_modular.inl File Reference	1054
17.264 solve.C File Reference	1054
17.264.1 Function Documentation	1055
17.264.1.1 main()	1055
17.265 sparse_matrix_traits.h File Reference	1055
17.266 test-charpoly-check.C File Reference	1056
17.266.1 Macro Definition Documentation	1057
17.266.1.1 ENABLE_CHECKER_charpoly	1057
17.266.1.2 TIME_CHECKER_CHARPOLY	1057
17.266.2 Function Documentation	1057
17.266.2.1 printPolynomial()	1057
17.266.2.2 main()	1057
17.267 test-charpoly.C File Reference	1057
17.267.1 Function Documentation	1057
17.267.1.1 launch_test()	1058
17.267.1.2 run_with_field()	1058
17.267.1.3 main()	1058
17.268 test-compressQ.C File Reference	1058
17.268.1 Typedef Documentation	1058
17.268.1.1 Field	1058
17.268.2 Function Documentation	1059
17.268.2.1 printvect()	1059
17.268.2.2 main()	1059
17.269 test-det-check.C File Reference	1059
17.269.1 Macro Definition Documentation	1059
17.269.1.1 ENABLE_CHECKER_Det	1059
17.269.1.2 TIME_CHECKER_Det	1059
17.269.2 Function Documentation	1059
17.269.2.1 main()	1060
17.270 test-det.C File Reference	1060
17.270.1 Function Documentation	1060
17.270.1.1 test_det()	1060
17.270.1.2 main()	1060
17.271 test-echelon.C File Reference	1060
17.271.1 Macro Definition Documentation	1061
17.271.1.1 __FFLASFFPACK_SEQUENTIAL	1061
17.271.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE	1061
17.271.1.3 __FFLASFFPACK_PLUQ_THRESHOLD	1061
17.271.2 Function Documentation	1061

17.271.2.1 test_colechelon()	1061
17.271.2.2 test_rowechelon()	1062
17.271.2.3 test_redcoechelon()	1062
17.271.2.4 test_redrowechelon()	1062
17.271.2.5 run_with_field()	1062
17.271.2.6 main()	1063
17.272 test-fadd.C File Reference	1063
17.272.1 Function Documentation	1063
17.272.1.1 test_fadd()	1063
17.272.1.2 test_faddin()	1063
17.272.1.3 test_fsub()	1064
17.272.1.4 test_fsubin()	1064
17.272.1.5 main()	1064
17.273 test-fdot.C File Reference	1064
17.273.1 Macro Definition Documentation	1065
17.273.1.1 ENABLE_ALL_CHECKINGS	1065
17.273.2 Function Documentation	1065
17.273.2.1 check_fdot()	1065
17.273.2.2 run_with_field()	1065
17.273.2.3 run_with_Integer()	1065
17.273.2.4 main()	1065
17.274 test-fgemm-check.C File Reference	1065
17.274.1 Macro Definition Documentation	1066
17.274.1.1 ENABLE_ALL_CHECKINGS	1066
17.274.2 Function Documentation	1066
17.274.2.1 launch_MM_dispatch()	1066
17.274.2.2 run_with_field()	1066
17.274.2.3 main()	1067
17.275 test-fgemm.C File Reference	1067
17.275.1 Macro Definition Documentation	1067
17.275.1.1 ENABLE_CHECKER_fgemm	1067
17.275.2 Function Documentation	1067
17.275.2.1 check_MM()	1068
17.275.2.2 launch_MM()	1068
17.275.2.3 launch_MM_dispatch()	1068
17.275.2.4 run_with_field()	1069
17.275.2.5 main()	1069
17.276 test-fgemv.C File Reference	1069
17.276.1 Function Documentation	1069
17.276.1.1 check_MV()	1070
17.276.1.2 launch_MV()	1070
17.276.1.3 launch_MV_dispatch()	1070

17.276.1.4 run_with_field()	1070
17.276.1.5 main()	1071
17.277 test-fger.C File Reference	1071
17.277.1 Macro Definition Documentation	1071
17.277.1.1 TIME	1071
17.277.2 Function Documentation	1071
17.277.2.1 check_fger()	1071
17.277.2.2 launch_fger()	1072
17.277.2.3 launch_fger_dispatch()	1072
17.277.2.4 run_with_field()	1072
17.277.2.5 main()	1072
17.278 test-fgesv.C File Reference	1073
17.278.1 Function Documentation	1073
17.278.1.1 test_square_fgesv()	1073
17.278.1.2 test_rect_fgesv()	1073
17.278.1.3 run_with_field()	1073
17.278.1.4 main()	1074
17.279 test-finit.C File Reference	1074
17.279.1 Function Documentation	1074
17.279.1.1 test_freduce()	1074
17.279.1.2 run_with_field()	1075
17.279.1.3 main()	1075
17.280 test-fscal.C File Reference	1075
17.280.1 Function Documentation	1075
17.280.1.1 test_fscal() [1/2]	1075
17.280.1.2 test_fscal() [2/2]	1076
17.280.1.3 test_fscalin() [1/2]	1076
17.280.1.4 test_fscalin() [2/2]	1076
17.280.1.5 main()	1076
17.281 test-fsyr2k.C File Reference	1076
17.281.1 Macro Definition Documentation	1077
17.281.1.1 ENABLE_ALL_CHECKINGS	1077
17.281.2 Function Documentation	1077
17.281.2.1 check_fsyr2k()	1077
17.281.2.2 run_with_field()	1077
17.281.2.3 main()	1077
17.282 test-fsyrk.C File Reference	1078
17.282.1 Macro Definition Documentation	1078
17.282.1.1 ENABLE_ALL_CHECKINGS	1078
17.282.2 Function Documentation	1078
17.282.2.1 check_fsyrk()	1078
17.282.2.2 check_fsyrk_diag()	1079



17.282.2.3 check_fsyrk_bkdiag()	1079
17.282.2.4 run_with_field()	1079
17.282.2.5 main()	1079
17.283 test-fsytrf.C File Reference	1079
17.283.1 Function Documentation	1080
17.283.1.1 operator<<()	1080
17.283.1.2 test_RPM_fsytrf()	1080
17.283.1.3 test_generic_fsytrf()	1080
17.283.1.4 run_with_field()	1080
17.283.1.5 main()	1081
17.284 test-frm.C File Reference	1081
17.284.1 Macro Definition Documentation	1081
17.284.1.1 __FFLASFFPACK_SEQUENTIAL	1081
17.284.2 Function Documentation	1081
17.284.2.1 check_frm()	1081
17.284.2.2 run_with_field()	1082
17.284.2.3 main()	1082
17.285 test-frm.C File Reference	1082
17.285.1 Macro Definition Documentation	1082
17.285.1.1 __FFLASFFPACK_SEQUENTIAL	1082
17.285.1.2 ENABLE_ALL_CHECKINGS	1083
17.285.2 Function Documentation	1083
17.285.2.1 check_frm()	1083
17.285.2.2 run_with_field()	1083
17.285.2.3 main()	1083
17.286 test-frm-check.C File Reference	1083
17.286.1 Macro Definition Documentation	1083
17.286.1.1 ENABLE_ALL_CHECKINGS	1084
17.286.2 Function Documentation	1084
17.286.2.1 main()	1084
17.287 test-frm.C File Reference	1084
17.287.1 Macro Definition Documentation	1084
17.287.1.1 __FFLASFFPACK_SEQUENTIAL	1084
17.287.1.2 ENABLE_ALL_CHECKINGS	1084
17.287.2 Function Documentation	1085
17.287.2.1 check_frm()	1085
17.287.2.2 run_with_field()	1085
17.287.2.3 main()	1085
17.288 test-frm.C File Reference	1085
17.288.1 Macro Definition Documentation	1086
17.288.1.1 ENABLE_ALL_CHECKINGS	1086
17.288.2 Function Documentation	1086

17.288.2.1 check_ftrssyr2k()	1086
17.288.2.2 run_with_field()	1086
17.288.2.3 main()	1086
17.289 test-ftrstr.C File Reference	1086
17.289.1 Macro Definition Documentation	1087
17.289.1.1 ENABLE_ALL_CHECKINGS	1087
17.289.2 Function Documentation	1087
17.289.2.1 check_ftrstr()	1087
17.289.2.2 run_with_field()	1087
17.289.2.3 main()	1087
17.290 test-ftrsv.C File Reference	1088
17.290.1 Macro Definition Documentation	1088
17.290.1.1 __FFLASFFPACK_SEQUENTIAL	1088
17.290.1.2 ENABLE_ALL_CHECKINGS	1088
17.290.2 Function Documentation	1088
17.290.2.1 check_ftrsv()	1088
17.290.2.2 run_with_field()	1089
17.290.2.3 main()	1089
17.291 test-ftrtri.C File Reference	1089
17.291.1 Macro Definition Documentation	1089
17.291.1.1 __FFLASFFPACK_SEQUENTIAL	1089
17.291.1.2 ENABLE_ALL_CHECKINGS	1089
17.291.2 Function Documentation	1090
17.291.2.1 check_ftrtri()	1090
17.291.2.2 run_with_field()	1090
17.291.2.3 main()	1090
17.292 test-interfaces-c.c File Reference	1090
17.292.1 Function Documentation	1090
17.292.1.1 main()	1090
17.293 test-invert-check.C File Reference	1090
17.293.1 Macro Definition Documentation	1091
17.293.1.1 ENABLE_ALL_CHECKINGS	1091
17.293.2 Function Documentation	1091
17.293.2.1 main()	1091
17.294 test-io.C File Reference	1091
17.294.1 Function Documentation	1092
17.294.1.1 run_with_field()	1092
17.294.1.2 main()	1092
17.295 test-lu.C File Reference	1092
17.295.1 Macro Definition Documentation	1093
17.295.1.1 BASECASE_K	1093
17.295.1.2 __FFLASFFPACK_SEQUENTIAL	1093

17.295.1.3 __LUDIVINE_CUTOFF . . . . .	1093
17.295.2 Function Documentation . . . . .	1093
17.295.2.1 test_LUdivine() . . . . .	1093
17.295.2.2 verifPLUQ() . . . . .	1094
17.295.2.3 test_pluq() . . . . .	1094
17.295.2.4 launch_test() . . . . .	1095
17.295.2.5 run_with_field() . . . . .	1095
17.295.2.6 main() . . . . .	1095
17.295.3 Variable Documentation . . . . .	1095
17.295.3.1 tperm . . . . .	1096
17.295.3.2 tgemm . . . . .	1096
17.295.3.3 tBC . . . . .	1096
17.295.3.4 ttrsm . . . . .	1096
17.295.3.5 trest . . . . .	1096
17.295.3.6 timtot . . . . .	1096
17.295.3.7 mvcnt . . . . .	1096
17.296 test-maxdelayeddim.C File Reference . . . . .	1096
17.296.1 Macro Definition Documentation . . . . .	1096
17.296.1.1 MAX_WITH_SIZE_T . . . . .	1097
17.296.2 Function Documentation . . . . .	1097
17.296.2.1 test() . . . . .	1097
17.296.2.2 main() . . . . .	1097
17.297 test-minpoly.C File Reference . . . . .	1097
17.297.1 Function Documentation . . . . .	1097
17.297.1.1 check_minpoly() . . . . .	1097
17.297.1.2 run_with_field() . . . . .	1098
17.297.1.3 main() . . . . .	1098
17.298 test-multifile1.C File Reference . . . . .	1098
17.299 test-multifile2.C File Reference . . . . .	1098
17.299.1 Function Documentation . . . . .	1098
17.299.1.1 main() . . . . .	1098
17.300 test-nullspace.C File Reference . . . . .	1098
17.300.1 Function Documentation . . . . .	1099
17.300.1.1 checkingMessage() . . . . .	1099
17.300.1.2 readOrRandomMatrixWithRankAndRandomRPM() . . . . .	1099
17.300.1.3 test_nullspace() . . . . .	1099
17.300.1.4 run_with_field() . . . . .	1099
17.300.1.5 main() . . . . .	1100
17.301 test-permutations.C File Reference . . . . .	1100
17.301.1 Function Documentation . . . . .	1100
17.301.1.1 checkMonotonicApplyP() . . . . .	1100
17.301.1.2 main() . . . . .	1100

17.301.2 Variable Documentation	1100
17.301.2.1 tperm	1101
17.301.2.2 tgemm	1101
17.301.2.3 tBC	1101
17.301.2.4 ttrsm	1101
17.301.2.5 trest	1101
17.301.2.6 timtot	1101
17.302 test-pluq-check.C File Reference	1101
17.302.1 Macro Definition Documentation	1101
17.302.1.1 ENABLE_ALL_CHECKINGS	1101
17.302.2 Function Documentation	1101
17.302.2.1 main()	1102
17.303 test-rankprofiles.C File Reference	1102
17.303.1 Macro Definition Documentation	1102
17.303.1.1 __FFLASFFPACK_SEQUENTIAL	1102
17.303.2 Function Documentation	1102
17.303.2.1 run_with_field()	1102
17.303.2.2 main()	1103
17.304 test-rpm.C File Reference	1103
17.304.1 Function Documentation	1103
17.304.1.1 checkRPM()	1103
17.304.1.2 checkSymmetricRPM()	1103
17.304.1.3 main()	1103
17.305 test-simd.C File Reference	1103
17.305.1 Macro Definition Documentation	1105
17.305.1.1 REGISTER_TYPE_NAME	1105
17.305.1.2 TEST_ONE_OP	1105
17.305.2 Typedef Documentation	1105
17.305.2.1 integer	1105
17.305.3 Function Documentation	1105
17.305.3.1 TypeName()	1105
17.305.3.2 REGISTER_TYPE_NAME() [1/8]	1105
17.305.3.3 REGISTER_TYPE_NAME() [2/8]	1105
17.305.3.4 REGISTER_TYPE_NAME() [3/8]	1105
17.305.3.5 REGISTER_TYPE_NAME() [4/8]	1106
17.305.3.6 REGISTER_TYPE_NAME() [5/8]	1106
17.305.3.7 REGISTER_TYPE_NAME() [6/8]	1106
17.305.3.8 REGISTER_TYPE_NAME() [7/8]	1106
17.305.3.9 REGISTER_TYPE_NAME() [8/8]	1106
17.305.3.10 generate_random_vector() [1/2]	1106
17.305.3.11 generate_random_vector() [2/2]	1106
17.305.3.12 check_eq() [1/2]	1106

17.305.3.13 <a href="#">check_eq()</a> [2/2]	1106
17.305.3.14 <a href="#">eval_func_on_array()</a> [1/2]	1107
17.305.3.15 <a href="#">eval_func_on_array()</a> [2/2]	1107
17.305.3.16 <a href="#">test_op()</a>	1107
17.305.3.17 <a href="#">test_impl()</a> [1/2]	1107
17.305.3.18 <a href="#">test_impl()</a> [2/2]	1107
17.305.3.19 <a href="#">test()</a> [1/2]	1107
17.305.3.20 <a href="#">test()</a> [2/2]	1107
17.305.3.21 <a href="#">main()</a>	1107
17.306 <a href="#">test-solve.C</a> File Reference	1107
17.306.1 <a href="#">Function Documentation</a>	1108
17.306.1.1 <a href="#">check_solve()</a>	1108
17.306.1.2 <a href="#">run_with_field()</a>	1108
17.306.1.3 <a href="#">main()</a>	1108
17.307 <a href="#">test-utils.h</a> File Reference	1108
17.308 <a href="#">timer.h</a> File Reference	1109
17.309 <a href="#">utils.h</a> File Reference	1109
17.310 <a href="#">winograd.C</a> File Reference	1110
17.310.1 <a href="#">Macro Definition Documentation</a>	1110
17.310.1.1 <a href="#">DOUBLE_TO_FLOAT_CROSSOVER</a>	1110
17.310.1.2 <a href="#">GFOPS</a>	1110
17.310.2 <a href="#">Typedef Documentation</a>	1110
17.310.2.1 <a href="#">TTimer</a>	1110
17.310.3 <a href="#">Function Documentation</a>	1110
17.310.3.1 <a href="#">balanced()</a> [1/2]	1111
17.310.3.2 <a href="#">balanced()</a> [2/2]	1111
17.310.3.3 <a href="#">main()</a>	1111



# Chapter 1

## FFLAS-FFPACK Documentation.

### 1.1 Introduction

FFLAS-FFPACK is a LGPL-2.1+ source code library for basic linear algebra operations over a finite field. It is inspired by BLAS interface (Basic Linear Algebra Subprograms) and the LAPACK library for numerical linear algebra, and shares part of their design. Yet it differs in many aspects due to the specificities of computing over a finite field:

- it is generic with respect to the finite field, so as to accomodate a large variety of field sizes and implementations;
- it is a pure source code library, to be included and compiled in the user's software. Its build system is only used for tests and benchmarks.

### 1.2 Goals

### 1.3 Design

### 1.4 Using FFLAS-FFPACK.

- [Copying and Licence](#).
- [Tutorial](#). This is a brief introduction to FFLAS-FFPACK capabilities.
- [Configuring and Installing FFLAS-FFPACK](#). Explains how to configure/install from sources or from the latest svn version.
- [Architecture of the library](#).. Describes how FFLAS-FFPACK is organized
- [Documentation for Users](#). If everything around is blue, then you are reading the lighter, user-oriented, documentation.
- [Documentation for Developers](#). If everything around is green, then you can get to everything (not necessarily yet) documented.

### 1.5 Contributing to fflas-ffpack, getting assistance.

Version

2.3.0





## Chapter 2

# Configuring and Installing FFLAS-FFPACK

FFLAS-FFPACK is a header-only package.

Howver configuration process can be tweaked a lot. Configure looks for BLAS routines and [Givaro](#) library which are both mandatory dependencies. See the output of `./configure -help` for information about the LAPACK/↵ BLAS discovering strategies.



## Chapter 3

# Copying and Licence

The FFLAS-FFPACK library is licensed under the terms of the GNU LGPL v2.1 or later.

See <https://www.gnu.org/licenses/lgpl-2.1.html>



## Chapter 4

# Tutorial

no doc.



## Chapter 5

# Architecture of the library.

no doc.





## Chapter 6

# Bug List

Global [DOUBLE\\_TO\\_FLOAT\\_CROSSOVER](#)

to be benchmarked.

Global [FFLAS::details::pack\\_lhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)

this is fassign

this is fassign

Global [FFLAS::details::pack\\_rhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)

this is fassign

this is fassign

Global [FFLAS::fconvert](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX, const FFLAS\_ELT \*Y, const size\_t incY)

use cblas\_(d)scal when possible

Global [FFLAS::fconvert](#) (const Field &F, const size\_t n, OtherElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)

use cblas\_(d)scal when possible

Global [FFLAS::finit](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global [FFLAS::finit](#) (const Field &F, const size\_t n, const OtherElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global [FFLAS::fneg](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global [FFLAS::fneg](#) (const Field &F, const size\_t n, typename Field::ConstElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global [FFLAS::fnegin](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global [FFLAS::fnegin](#) (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global [FFLAS::freduce](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::freduce** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fscale** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, const FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)

use cblas\_(d)scal when possible

Global **FFLAS::fscalein** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fsquare** (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)

why double ?

Global **FFLAS::fswap** (const Field &F, const size\_t N, typename Field::Element\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY)

use cblas\_dswap when double

Global **FFLAS::fswap** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t N, FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)

use cblas\_dswap when double

Global **FFLAS::ftrsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb)

$\alpha$  must be non zero.

Global **FFLAS::ftrsm** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const FFLAS\_ELT alpha, const FFLAS\_ELT \*A, const size\_t lda, FFLAS\_ELT \*B, const size\_t ldb)

$\alpha$  must be non zero.

Global **FFPACK::buildMatrix** (const Field &F, typename Field::ConstElement\_ptr E, typename Field::ConstElement\_ptr C, const size\_t lda, const size\_t \*B, const size\_t \*T, const size\_t me, const size\_t mc, const size\_t lambda, const size\_t mu)

is this :

Global **FFPACK::invert2** (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t idx, int &nullity)

not tested.

Global **launch\_fger\_dispatch** (const Field &F, const size\_t nn, const typename Field::Element alpha, const size\_t iters, RandIter &G)

test for incx equal

test for transpo

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size\_t iters, RandIter &G)

test for ldX equal

test for transpo

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size\_t iters, const int nbw, const bool par, RandIter &G)

test for ldX equal

test for transpo

Global **printvect** (std::ostream &o, vector< T > &vect)

does not belong here

## Chapter 7

# Bibliography

- Global **FFLAS::Protected::TRSMBound** (const Givaro::ModularBalanced< Element > &F) .  
Dumas Giorgi Pernet 06, arXiv:cs/0601133
- Global **FFPACK::LeadingSubmatrixRankProfiles** (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP) .  
Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13.
- Global **FFPACK::LUdivine** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const **FFLAS::FFLAS\_TRANSPOSE** trans, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, size\_t \*P, size\_t \*Qt, const FFPACK\_LU\_TAG LuTag=FfpackSlabRecursive, const size\_t cutoff=\_\_FFLASFFPACK\_LUDIVINE\_THRESHOLD) .  
Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013  
• Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002
- Class **ftsmLeftUpperNoTransNonUnit**< Element > .  
Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.
- Global **FFPACK::PLUQ** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, size\_t \*P, size\_t \*Q) .  
Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13, 2013
- Global **FFPACK::Protected::GaussJordan** (const Field &F, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag) .  
Algorithm 2.8 of A. Storjohann Thesis 2000,  
• Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013



## Chapter 8

# Todo List

### File `debug.h`

we should put vector printing elsewhere.

Global `FFLAS::fadd` (const Field &F, const size\_t N, typename `Field::ConstElement_ptr` A, const size\_t incA, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` B, const size\_t incB, typename `Field::Element_ptr` C, const size\_t incC)

optimise here

Global `FFLAS::fassign` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t N, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

variant for triangular matrix

Global `FFLAS::fconvert` (const Field &F, const size\_t m, const size\_t n, OtherElement\_ptr A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb)

check if n == lda

Global `FFLAS::fneg` (const Field &F, const size\_t m, const size\_t n, typename `Field::ConstElement_ptr` B, const size\_t ldb, typename `Field::Element_ptr` A, const size\_t lda)

check if n == lda

Global `FFLAS::fnegin` (const Field &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda)

check if n == lda

Global `FFLAS::fscal` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, const FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::fscaln` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, FFLAS\_ELT \*X, const size\_t incX)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::Protected::igemm` (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size\_t rows, size\_t cols, size\_t depth, const int64\_t alpha, const int64\_t \*A, size\_t lda, const int64\_t \*B, size\_t ldb, const int64\_t beta, int64\_t \*C, size\_t ldc)

use primitive (no `Field()`) and specialise for int64.

Global `FFLAS::Protected::MatF2MatFI_Triangular` (const Field &F, Givaro::FloatDomain::Element\_ptr S, const size\_t lds, typename `Field::ConstElement_ptr` const E, const size\_t lde, const size\_t m, const size\_t n)

do finit(...,FFLAS\_TRANS,FFLAS\_DIAG)

do fconvert(...,FFLAS\_TRANS,FFLAS\_DIAG)

Global **FFPACK::LUdivine** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const **FFLAS::FFLAS\_TRANSPOSE** trans, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, size\_t \*P, size\_t \*Q, const **FFPACK::FFPACK\_LU\_TAG** LuTag, const size\_t cutoff)  
std::swap ?

Global **FFPACK::Protected::RandomKrylovPrecond** (const PolRing &PR, std::list< typename PolRing::↵  
Element > &completedFactors, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const  
size\_t lda, size\_t &Nb, typename PolRing::Domain\_t::Element\_ptr &B, size\_t &ldb, typename PolRing::↵  
Domain\_t::RandIter &g, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)  
don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

swap to save space ??

#### Module **field**

biblio

Global **launch\_fger\_dispatch** (const Field &F, const size\_t nn, const typename **Field::Element** alpha, const  
size\_t iters, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename  
**Field::Element** alpha, const typename **Field::Element** beta, const size\_t iters, const int nbw, const bool  
par, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename  
**Field::Element** alpha, const typename **Field::Element** beta, const size\_t iters, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **test\_redcochelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG**  
LuTag, RandIter &G, bool par)

check lda

Global **test\_redrowechelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG**  
LuTag, RandIter &G, bool par)

check lda

#### Module **MMalgos**

biblio

#### Module **simd**

biblio

Global **test\_cochelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG** LuTag,  
RandIter &G, bool par)

check lda

Global **test\_det** (Field &F, size\_t n, int iter, RandIter &G)

test with stride

Global **test\_rowechelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG** LuTag,  
RandIter &G, bool par)

check lda

## Chapter 9

# Module Index

### 9.1 Modules

Here is a list of all modules:

CHECKER . . . . .	45
Matrix Multiplication Algorithms . . . . .	45
SIMD wrapper . . . . .	46
FFLAS-FFPACK . . . . .	46
FFLAS . . . . .	45
Interfaces . . . . .	47
FFPACK . . . . .	46
FFLAS-FFPACK fields . . . . .	46
RNS . . . . .	47





## Chapter 10

# Namespace Index

### 10.1 Namespace List

Here is a list of all namespaces with brief descriptions:

FFLAS	49
FFLAS::BLAS3	196
FFLAS::csr_hyb_details	202
FFLAS::CuttingStrategy	202
FFLAS::details	203
FFLAS::details_spmv	211
FFLAS::ElementCategories	211
FFLAS::FieldCategories	
Traits and categories will need to be placed in a proper file later	212
FFLAS::MMHelperAlgo	212
FFLAS::ModeCategories	
Specifies the mode of action for an algorithm w.r.t	212
FFLAS::ParSeqHelper	
ParSeqHelper for both fgemm and ftrsm	213
FFLAS::Protected	213
FFLAS::sell_details	226
FFLAS::sparse_details	227
FFLAS::sparse_details_impl	242
FFLAS::StrategyParameter	285
FFLAS::StructureHelper	
StructureHelper for ftrsm	285
FFLAS::vectorised	285
FFLAS::vectorised::unswitch	291
FFPACK	
Finite Field <b>PACK</b> Set of elimination based routines for dense linear algebra	292
FFPACK::Protected	394
Givaro	401
MKL_CONFIG	401
RecInt	401



## Chapter 11

# Hierarchical Index

### 11.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

AlgoChooser< ModeT, ParSeq > . . . . .	403
AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > . . . . .	403
ArbitraryPrecIntTag . . . . .	403
AreEqual< X, Y > . . . . .	404
AreEqual< X, X > . . . . .	404
Argument . . . . .	404
associatedDelayedField< Field > . . . . .	405
associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > . . . . .	405
associatedDelayedField< const Givaro::Modular< T, X > > . . . . .	406
associatedDelayedField< const Givaro::ModularBalanced< T > > . . . . .	406
associatedDelayedField< const Givaro::ZRing< T > > . . . . .	407
Auto . . . . .	407
Bini . . . . .	407
Block . . . . .	407
callLUdivine_small< Element > . . . . .	408
callLUdivine_small< double > . . . . .	408
callLUdivine_small< float > . . . . .	409
CharpolyFailed . . . . .	409
Checker_Empty< Field > . . . . .	409
CheckerImplem_charpoly< Field, Polynomial > . . . . .	410
CheckerImplem_Det< Field > . . . . .	411
CheckerImplem_fgemm< Field > . . . . .	412
CheckerImplem_ftrsm< Field > . . . . .	413
CheckerImplem_invert< Field > . . . . .	414
CheckerImplem_PLUQ< Field > . . . . .	415
Classic . . . . .	416
Column . . . . .	416
CompactElement< Element > . . . . .	416
CompactElement< double > . . . . .	416
CompactElement< float > . . . . .	417
CompactElement< int16_t > . . . . .	417
CompactElement< int32_t > . . . . .	417
CompactElement< int64_t > . . . . .	418
compatible_data_type< Field > . . . . .	418
compatible_data_type< Givaro::ZRing< double > > . . . . .	418

compatible_data_type< Givaro::ZRing< float > > . . . . .	418
Compose< H1, H2 > . . . . .	419
Const_int_t< n > . . . . .	420
Const_uint_t< n > . . . . .	420
Simd128_impl< true, true, false, 2 >::Converter . . . . .	420
Simd128_impl< true, true, false, 4 >::Converter . . . . .	421
Simd128_impl< true, true, false, 8 >::Converter . . . . .	421
Simd128_impl< true, true, true, 2 >::Converter . . . . .	421
Simd128_impl< true, true, true, 4 >::Converter . . . . .	422
Simd128_impl< true, true, true, 8 >::Converter . . . . .	422
Simd256_impl< true, true, false, 2 >::Converter . . . . .	423
Simd256_impl< true, true, false, 4 >::Converter . . . . .	423
Simd256_impl< true, true, false, 8 >::Converter . . . . .	423
Simd256_impl< true, true, true, 2 >::Converter . . . . .	424
Simd256_impl< true, true, true, 4 >::Converter . . . . .	424
Simd256_impl< true, true, true, 8 >::Converter . . . . .	425
Simd512_impl< true, true, false, 8 >::Converter . . . . .	425
Simd512_impl< true, true, true, 8 >::Converter . . . . .	425
ConvertTo< T > . . . . .	426
Coo< ValT, IdxT > . . . . .	426
Coo< Field > . . . . .	428
Coo< ValT, IdxT > . . . . .	429
CooMat< Field > . . . . .	431
CooMat< FFPACK::RNSInteger > . . . . .	431
CsrMat< Field > . . . . .	432
CsrMat< FFPACK::RNSInteger > . . . . .	432
DefaultBoundedTag . . . . .	432
DefaultTag . . . . .	433
DelayedTag . . . . .	433
ElementTraits< Element > . . . . .	433
ElementTraits< double > . . . . .	433
ElementTraits< FFPACK::rns_double_elt > . . . . .	434
ElementTraits< float > . . . . .	434
ElementTraits< Givaro::Integer > . . . . .	434
ElementTraits< int16_t > . . . . .	435
ElementTraits< int32_t > . . . . .	435
ElementTraits< int64_t > . . . . .	435
ElementTraits< int8_t > . . . . .	436
ElementTraits< Reclnt::rint< K > > . . . . .	436
ElementTraits< Reclnt::rmint< K, MG > > . . . . .	436
ElementTraits< Reclnt::ruint< K > > . . . . .	437
ElementTraits< uint16_t > . . . . .	437
ElementTraits< uint32_t > . . . . .	437
ElementTraits< uint64_t > . . . . .	438
ElementTraits< uint8_t > . . . . .	438
EllMat< Field > . . . . .	438
EllMat< FFPACK::RNSInteger > . . . . .	438
Failure . . . . .	439
FailureCharpolyCheck . . . . .	441
FailureDetCheck . . . . .	441
FailureFgemmCheck . . . . .	441
FailureInvertCheck . . . . .	441
FailurePLUQCheck . . . . .	442
FailureTrsmCheck . . . . .	442
false_type . . . . .	
isSparseMatrix< Field, M > . . . . .	477
isSparseMatrixMKLFormat< F, M > . . . . .	481
isSparseMatrixSimdFormat< F, M > . . . . .	481

isZOSparseMatrix< F, M > . . . . .	482
support_fast_mod< T > . . . . .	751
support_simd< T > . . . . .	753
support_simd_add< T > . . . . .	753
support_simd_mod< T > . . . . .	753
FieldSimd< _Field > . . . . .	442
FieldTraits< Field > . . . . .	448
FieldTraits< FFPACK::RNSInteger< T > > . . . . .	449
FieldTraits< FFPACK::RNSIntegerMod< T > > . . . . .	449
FieldTraits< Givaro::Modular< Element > > . . . . .	450
FieldTraits< Givaro::ModularBalanced< Element > > . . . . .	450
FieldTraits< Givaro::ZRing< double > > . . . . .	451
FieldTraits< Givaro::ZRing< float > > . . . . .	451
FieldTraits< Givaro::ZRing< Givaro::Integer > > . . . . .	452
FieldTraits< Givaro::ZRing< int16_t > > . . . . .	453
FieldTraits< Givaro::ZRing< int32_t > > . . . . .	453
FieldTraits< Givaro::ZRing< int64_t > > . . . . .	454
FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > . . . . .	454
FieldTraits< Givaro::ZRing< uint16_t > > . . . . .	455
FieldTraits< Givaro::ZRing< uint32_t > > . . . . .	455
FieldTraits< Givaro::ZRing< uint64_t > > . . . . .	456
Fixed . . . . .	456
FixedPrecIntTag . . . . .	456
ForStrategy1D< blocksize_t, Cut, Param > . . . . .	457
ForStrategy2D< blocksize_t, Cut, Param > . . . . .	459
ftmmLeftLowerNoTransNonUnit< Element > . . . . .	462
ftmmLeftLowerNoTransUnit< Element > . . . . .	462
ftmmLeftLowerTransNonUnit< Element > . . . . .	463
ftmmLeftLowerTransUnit< Element > . . . . .	463
ftmmLeftUpperNoTransNonUnit< Element > . . . . .	463
ftmmLeftUpperNoTransUnit< Element > . . . . .	463
ftmmLeftUpperTransNonUnit< Element > . . . . .	463
ftmmLeftUpperTransUnit< Element > . . . . .	463
ftmmRightLowerNoTransNonUnit< Element > . . . . .	463
ftmmRightLowerNoTransUnit< Element > . . . . .	463
ftmmRightLowerTransNonUnit< Element > . . . . .	464
ftmmRightLowerTransUnit< Element > . . . . .	464
ftmmRightUpperNoTransNonUnit< Element > . . . . .	464
ftmmRightUpperNoTransUnit< Element > . . . . .	464
ftmmRightUpperTransNonUnit< Element > . . . . .	464
ftmmRightUpperTransUnit< Element > . . . . .	464
ftsmLeftLowerNoTransNonUnit< Element > . . . . .	464
ftsmLeftLowerNoTransUnit< Element > . . . . .	464
ftsmLeftLowerTransNonUnit< Element > . . . . .	465
ftsmLeftLowerTransUnit< Element > . . . . .	465
ftsmLeftUpperNoTransNonUnit< Element > . . . . .	465
ftsmLeftUpperNoTransUnit< Element > . . . . .	465
ftsmLeftUpperTransNonUnit< Element > . . . . .	465
ftsmLeftUpperTransUnit< Element > . . . . .	465
ftsmRightLowerNoTransNonUnit< Element > . . . . .	466
ftsmRightLowerNoTransUnit< Element > . . . . .	466
ftsmRightLowerTransNonUnit< Element > . . . . .	466
ftsmRightLowerTransUnit< Element > . . . . .	466
ftsmRightUpperNoTransNonUnit< Element > . . . . .	466
ftsmRightUpperNoTransUnit< Element > . . . . .	466
ftsmRightUpperTransNonUnit< Element > . . . . .	466
ftsmRightUpperTransUnit< Element > . . . . .	466
GenericTag . . . . .	467

GenericTag	467
Grain	467
has_minus_eq_impl< C >	467
has_minus_impl< C >	467
has_mul_eq_impl< C >	468
has_mul_impl< C >	468
has_operation< T >	468
has_plus_eq_impl< C >	469
has_plus_impl< C >	469
HelperFlag	469
HelperMod< Field, ElementTraits >	470
HelperMod< Field, ElementCategories::MachineIntTag >	470
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >	472
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >	472
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >	473
Hybrid	474
Info	474
Info	475
is_simd< T >	477
Iterative	483
LazyTag	484
limits< T >	484
limits< char >	484
limits< double >	485
limits< float >	485
limits< Givaro::Integer >	486
limits< int >	487
limits< long >	487
limits< long long >	488
limits< Reclnt::rint< K > >	489
limits< Reclnt::ruint< K > >	490
limits< short int >	490
limits< signed char >	491
limits< unsigned char >	492
limits< unsigned int >	492
limits< unsigned long >	493
limits< unsigned long long >	494
limits< unsigned short int >	495
MachineFloatTag	495
MachineIntTag	496
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >	496
MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	501
MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	503
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >	505
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >	507
MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	509
ModeTraits< Field >	511
ModeTraits< Givaro::Modular< Element, Compute > >	511
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >	511
ModeTraits< Givaro::Modular< int16_t, Compute > >	512
ModeTraits< Givaro::Modular< int32_t, Compute > >	512
ModeTraits< Givaro::Modular< int8_t, Compute > >	512
ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >	513
ModeTraits< Givaro::Modular< uint16_t, Compute > >	513
ModeTraits< Givaro::Modular< uint32_t, Compute > >	514
ModeTraits< Givaro::Modular< uint8_t, Compute > >	514
ModeTraits< Givaro::ModularBalanced< Element > >	514

ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > . . . . .	515
ModeTraits< Givaro::ModularBalanced< int16_t > > . . . . .	515
ModeTraits< Givaro::ModularBalanced< int32_t > > . . . . .	515
ModeTraits< Givaro::ModularBalanced< int8_t > > . . . . .	516
ModeTraits< Givaro::Montgomery< T > > . . . . .	516
ModeTraits< Givaro::ZRing< double > > . . . . .	516
ModeTraits< Givaro::ZRing< float > > . . . . .	517
ModeTraits< Givaro::ZRing< Givaro::Integer > > . . . . .	517
ModularBalanced< T > . . . . .	517
ModularTag . . . . .	518
Montgomery< T > . . . . .	518
need_field_characteristic< Field > . . . . .	518
need_field_characteristic< Givaro::Modular< Field > > . . . . .	518
need_field_characteristic< Givaro::ModularBalanced< Field > > . . . . .	518
NoSimd< T > . . . . .	519
Parallel< C, P > . . . . .	520
readMyMachineType< Field, T > . . . . .	523
readMyMachineType< Field, mpz_t > . . . . .	524
Recursive . . . . .	525
Recursive . . . . .	525
rint< K > . . . . .	525
rns_double . . . . .	525
rns_double_elt . . . . .	530
rns_double_elt_cstptr . . . . .	532
rns_double_elt_ptr . . . . .	535
rns_double_extended . . . . .	538
RNSElementTag . . . . .	542
RNSInteger< RNS > . . . . .	543
RNSInteger< FFPACK::rns_double > . . . . .	543
RNSIntegerMod< RNS > . . . . .	546
RNSIntegerMod< FFPACK::rns_double > . . . . .	546
rnsRandIter< RNS > . . . . .	553
RNSInteger< RNS >::RandIter . . . . .	521
RNSIntegerMod< RNS >::RandIter . . . . .	522
Row . . . . .	554
ruint< K > . . . . .	554
ScalFunctions< Element, Enable > . . . . .	554
ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type > . . . . .	555
ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type > . . . . .	558
Sequential . . . . .	564
Simd128_impl< ArithType, Int, Signed, Size > . . . . .	565
Simd128fp_base . . . . .	618
Simd128_impl< true, false, true, 4 > . . . . .	565
Simd128_impl< true, false, true, 8 > . . . . .	565
Simd128i_base . . . . .	619
Simd128_impl< true, true, true, 2 > . . . . .	592
Simd128_impl< true, true, false, 2 > . . . . .	566
Simd128_impl< true, true, true, 4 > . . . . .	601
Simd128_impl< true, true, false, 4 > . . . . .	574
Simd128_impl< true, true, true, 8 > . . . . .	609
Simd128_impl< true, true, false, 8 > . . . . .	583
Simd256_impl< ArithType, Int, Signed, Size > . . . . .	620
Simd256fp_base . . . . .	695
Simd256_impl< true, false, true, 4 > . . . . .	620
Simd256_impl< true, false, true, 8 > . . . . .	621
Simd256i_base . . . . .	696

Simd256_impl< true, true, true, 2 > . . . . .	662
Simd256_impl< true, true, false, 2 > . . . . .	628
Simd256_impl< true, true, true, 4 > . . . . .	670
Simd256_impl< true, true, false, 4 > . . . . .	636
Simd256_impl< true, true, false, 4 > . . . . .	636
Simd256_impl< true, true, true, 8 > . . . . .	687
Simd256_impl< true, true, false, 8 > . . . . .	653
Simd512_impl< ArithType, Int, Signed, Size > . . . . .	697
Simd512fp_base . . . . .	723
Simd512_impl< true, false, true, 4 > . . . . .	697
Simd512_impl< true, false, true, 8 > . . . . .	697
Simd512i_base . . . . .	723
Simd256_impl< true, true, true, 4 > . . . . .	670
Simd512_impl< true, true, true, 8 > . . . . .	713
Simd512_impl< true, true, false, 8 > . . . . .	704
SimdChooser< T, bool, bool > . . . . .	725
SimdChooser< T, false, b > . . . . .	725
SimdChooser< T, true, false > . . . . .	725
SimdChooser< T, true, true > . . . . .	725
simdToType< T > . . . . .	726
Single . . . . .	726
Sparse< Field, SparseMatrix_t, IdxT, PtrT > . . . . .	726
Sparse< _Field, SparseMatrix_t::COO > . . . . .	726
Sparse< _Field, SparseMatrix_t::COO_ZO > . . . . .	728
Sparse< _Field, SparseMatrix_t::CSR > . . . . .	729
Sparse< _Field, SparseMatrix_t::CSR_ZO > . . . . .	733
Sparse< _Field, SparseMatrix_t::CSR_HYB > . . . . .	731
Sparse< _Field, SparseMatrix_t::ELL > . . . . .	735
Sparse< _Field, SparseMatrix_t::ELL_ZO > . . . . .	740
Sparse< _Field, SparseMatrix_t::ELL_simd > . . . . .	736
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > . . . . .	738
Sparse< _Field, SparseMatrix_t::HYB_ZO > . . . . .	741
Sparse< _Field, SparseMatrix_t::SELL > . . . . .	743
Sparse< _Field, SparseMatrix_t::SELL_ZO > . . . . .	745
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int16_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int32_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int64_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int16_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int32_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int64_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int16_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int32_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int64_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int16_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int32_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int64_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int16_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int32_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int64_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int16_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int32_t > . . . . .	726
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int64_t > . . . . .	726
SpMat< Field, flag > . . . . .	747
Static_error_check< bool > . . . . .	748
Static_error_check< false > . . . . .	748
StatsMatrix . . . . .	748



tfn_minus . . . . .	753
tfn_minus_eq . . . . .	754
tfn_mul . . . . .	754
tfn_mul_eq . . . . .	755
tfn_plus . . . . .	755
tfn_plus_eq . . . . .	755
Threads . . . . .	756
ThreeD . . . . .	756
ThreeDAdaptive . . . . .	756
ThreeDInPlace . . . . .	756
TRSMHelper< ReclterTrait, ParSeqTrait > . . . . .	756
true_type	
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > . . . . .	478
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . . .	478
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > . . . . .	478
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > . . . . .	478
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . . .	479
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > . . . . .	479
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . . .	480
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > . . . . .	479
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . . .	480
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > . . . . .	480
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > . . . . .	481
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . . .	481
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > . . . . .	482
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > . . . . .	482
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > . . . . .	483
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > . . . . .	483
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > . . . . .	483
support_fast_mod< double > . . . . .	752
support_fast_mod< float > . . . . .	752
support_fast_mod< int64_t > . . . . .	752
TwoD . . . . .	758
TwoDAdaptive . . . . .	758
UnparametricTag . . . . .	758
Winograd . . . . .	758
WinogradPar . . . . .	758



## Chapter 12

# Data Structure Index

### 12.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">AlgoChooser&lt; ModeT, ParSeq &gt;</a>	403
<a href="#">AlgoChooser&lt; ModeCategories::ConvertTo&lt; ElementCategories::RNSElementTag &gt;, ParSeq &gt;</a>	403
<a href="#">ArbitraryPrecIntTag</a>	
Arbitrary precision integers: GMP	403
<a href="#">AreEqual&lt; X, Y &gt;</a>	404
<a href="#">AreEqual&lt; X, X &gt;</a>	404
<a href="#">Argument</a>	404
<a href="#">associatedDelayedField&lt; Field &gt;</a>	405
<a href="#">associatedDelayedField&lt; const FFPACK::RNSIntegerMod&lt; RNS &gt; &gt;</a>	405
<a href="#">associatedDelayedField&lt; const Givaro::Modular&lt; T, X &gt; &gt;</a>	406
<a href="#">associatedDelayedField&lt; const Givaro::ModularBalanced&lt; T &gt; &gt;</a>	406
<a href="#">associatedDelayedField&lt; const Givaro::ZRing&lt; T &gt; &gt;</a>	407
<a href="#">Auto</a>	407
<a href="#">Bini</a>	407
<a href="#">Block</a>	407
<a href="#">callLUdivine_small&lt; Element &gt;</a>	408
<a href="#">callLUdivine_small&lt; double &gt;</a>	408
<a href="#">callLUdivine_small&lt; float &gt;</a>	409
<a href="#">CharpolyFailed</a>	409
<a href="#">Checker_Empty&lt; Field &gt;</a>	409
<a href="#">CheckerImplem_charpoly&lt; Field, Polynomial &gt;</a>	410
<a href="#">CheckerImplem_Det&lt; Field &gt;</a>	411
<a href="#">CheckerImplem_fgemm&lt; Field &gt;</a>	412
<a href="#">CheckerImplem_ftdsm&lt; Field &gt;</a>	413
<a href="#">CheckerImplem_invert&lt; Field &gt;</a>	414
<a href="#">CheckerImplem_PLUQ&lt; Field &gt;</a>	415
<a href="#">Classic</a>	416
<a href="#">Column</a>	416
<a href="#">CompactElement&lt; Element &gt;</a>	416
<a href="#">CompactElement&lt; double &gt;</a>	416
<a href="#">CompactElement&lt; float &gt;</a>	417
<a href="#">CompactElement&lt; int16_t &gt;</a>	417
<a href="#">CompactElement&lt; int32_t &gt;</a>	417
<a href="#">CompactElement&lt; int64_t &gt;</a>	418
<a href="#">compatible_data_type&lt; Field &gt;</a>	418

<a href="#">compatible_data_type&lt; Givaro::ZRing&lt; double &gt; &gt;</a>	418
<a href="#">compatible_data_type&lt; Givaro::ZRing&lt; float &gt; &gt;</a>	418
<a href="#">Compose&lt; H1, H2 &gt;</a>	419
<a href="#">Const_int_t&lt; n &gt;</a>	420
<a href="#">Const_uint_t&lt; n &gt;</a>	420
<a href="#">Simd128_impl&lt; true, true, false, 2 &gt;::Converter</a>	420
<a href="#">Simd128_impl&lt; true, true, false, 4 &gt;::Converter</a>	421
<a href="#">Simd128_impl&lt; true, true, false, 8 &gt;::Converter</a>	421
<a href="#">Simd128_impl&lt; true, true, true, 2 &gt;::Converter</a>	421
<a href="#">Simd128_impl&lt; true, true, true, 4 &gt;::Converter</a>	422
<a href="#">Simd128_impl&lt; true, true, true, 8 &gt;::Converter</a>	422
<a href="#">Simd256_impl&lt; true, true, false, 2 &gt;::Converter</a>	423
<a href="#">Simd256_impl&lt; true, true, false, 4 &gt;::Converter</a>	423
<a href="#">Simd256_impl&lt; true, true, false, 8 &gt;::Converter</a>	423
<a href="#">Simd256_impl&lt; true, true, true, 2 &gt;::Converter</a>	424
<a href="#">Simd256_impl&lt; true, true, true, 4 &gt;::Converter</a>	424
<a href="#">Simd256_impl&lt; true, true, true, 8 &gt;::Converter</a>	425
<a href="#">Simd512_impl&lt; true, true, false, 8 &gt;::Converter</a>	425
<a href="#">Simd512_impl&lt; true, true, true, 8 &gt;::Converter</a>	425
<a href="#">ConvertTo&lt; T &gt;</a>	
Force conversion to appropriate element type of ElementCategory T	426
<a href="#">Coo&lt; ValT, IdxT &gt;</a>	426
<a href="#">Coo&lt; Field &gt;</a>	428
<a href="#">Coo&lt; ValT, IdxT &gt;</a>	429
<a href="#">CooMat&lt; Field &gt;</a>	431
<a href="#">CsrMat&lt; Field &gt;</a>	432
<a href="#">DefaultBoundedTag</a>	
Use standard field operations, but keeps track of bounds on input and output	432
<a href="#">DefaultTag</a>	
No specific mode of action: use standard field operations	433
<a href="#">DelayedTag</a>	
Performs field operations with delayed mod reductions. Ensures result is reduced	433
<a href="#">ElementTraits&lt; Element &gt;</a>	
<a href="#">ElementTraits</a>	433
<a href="#">ElementTraits&lt; double &gt;</a>	433
<a href="#">ElementTraits&lt; FFPACK::rns_double_elt &gt;</a>	434
<a href="#">ElementTraits&lt; float &gt;</a>	434
<a href="#">ElementTraits&lt; Givaro::Integer &gt;</a>	434
<a href="#">ElementTraits&lt; int16_t &gt;</a>	435
<a href="#">ElementTraits&lt; int32_t &gt;</a>	435
<a href="#">ElementTraits&lt; int64_t &gt;</a>	435
<a href="#">ElementTraits&lt; int8_t &gt;</a>	436
<a href="#">ElementTraits&lt; Reclnt::rint&lt; K &gt; &gt;</a>	436
<a href="#">ElementTraits&lt; Reclnt::rmint&lt; K, MG &gt; &gt;</a>	436
<a href="#">ElementTraits&lt; Reclnt::ruint&lt; K &gt; &gt;</a>	437
<a href="#">ElementTraits&lt; uint16_t &gt;</a>	437
<a href="#">ElementTraits&lt; uint32_t &gt;</a>	437
<a href="#">ElementTraits&lt; uint64_t &gt;</a>	438
<a href="#">ElementTraits&lt; uint8_t &gt;</a>	438
<a href="#">EllMat&lt; Field &gt;</a>	438
<a href="#">Failure</a>	
A precondition failed	439
<a href="#">FailureCharpolyCheck</a>	441
<a href="#">FailureDetCheck</a>	441
<a href="#">FailureFgemmCheck</a>	441
<a href="#">FailureInvertCheck</a>	441
<a href="#">FailurePLUQCheck</a>	442
<a href="#">FailureTrsmCheck</a>	442

FieldSimd< _Field > . . . . .	442
FieldTraits< Field >	
FieldTrait . . . . .	448
FieldTraits< FFPACK::RNSInteger< T > > . . . . .	449
FieldTraits< FFPACK::RNSIntegerMod< T > > . . . . .	449
FieldTraits< Givaro::Modular< Element > > . . . . .	450
FieldTraits< Givaro::ModularBalanced< Element > > . . . . .	450
FieldTraits< Givaro::ZRing< double > > . . . . .	451
FieldTraits< Givaro::ZRing< float > > . . . . .	451
FieldTraits< Givaro::ZRing< Givaro::Integer > > . . . . .	452
FieldTraits< Givaro::ZRing< int16_t > > . . . . .	453
FieldTraits< Givaro::ZRing< int32_t > > . . . . .	453
FieldTraits< Givaro::ZRing< int64_t > > . . . . .	454
FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > . . . . .	454
FieldTraits< Givaro::ZRing< uint16_t > > . . . . .	455
FieldTraits< Givaro::ZRing< uint32_t > > . . . . .	455
FieldTraits< Givaro::ZRing< uint64_t > > . . . . .	456
Fixed . . . . .	456
FixedPrecIntTag	
Fixed precision integers above machine precision: Givaro::reclnt . . . . .	456
ForStrategy1D< blocksize_t, Cut, Param > . . . . .	457
ForStrategy2D< blocksize_t, Cut, Param > . . . . .	459
ftmmLeftLowerNoTransNonUnit< Element > . . . . .	462
ftmmLeftLowerNoTransUnit< Element > . . . . .	462
ftmmLeftLowerTransNonUnit< Element > . . . . .	463
ftmmLeftLowerTransUnit< Element > . . . . .	463
ftmmLeftUpperNoTransNonUnit< Element > . . . . .	463
ftmmLeftUpperNoTransUnit< Element > . . . . .	463
ftmmLeftUpperTransNonUnit< Element > . . . . .	463
ftmmLeftUpperTransUnit< Element > . . . . .	463
ftmmRightLowerNoTransNonUnit< Element > . . . . .	463
ftmmRightLowerNoTransUnit< Element > . . . . .	463
ftmmRightLowerTransNonUnit< Element > . . . . .	464
ftmmRightLowerTransUnit< Element > . . . . .	464
ftmmRightUpperNoTransNonUnit< Element > . . . . .	464
ftmmRightUpperNoTransUnit< Element > . . . . .	464
ftmmRightUpperTransNonUnit< Element > . . . . .	464
ftmmRightUpperTransUnit< Element > . . . . .	464
ftsmLeftLowerNoTransNonUnit< Element > . . . . .	464
ftsmLeftLowerNoTransUnit< Element > . . . . .	464
ftsmLeftLowerTransNonUnit< Element > . . . . .	465
ftsmLeftLowerTransUnit< Element > . . . . .	465
ftsmLeftUpperNoTransNonUnit< Element > . . . . .	
Computes the maximal size for delaying the modular reduction in a triangular system resolution	465
ftsmLeftUpperNoTransUnit< Element > . . . . .	465
ftsmLeftUpperTransNonUnit< Element > . . . . .	465
ftsmLeftUpperTransUnit< Element > . . . . .	465
ftsmRightLowerNoTransNonUnit< Element > . . . . .	466
ftsmRightLowerNoTransUnit< Element > . . . . .	466
ftsmRightLowerTransNonUnit< Element > . . . . .	466
ftsmRightLowerTransUnit< Element > . . . . .	466
ftsmRightUpperNoTransNonUnit< Element > . . . . .	466
ftsmRightUpperNoTransUnit< Element > . . . . .	466
ftsmRightUpperTransNonUnit< Element > . . . . .	466
ftsmRightUpperTransUnit< Element > . . . . .	466
GenericTag	
Default is generic . . . . .	467

GenericTag	
Generic ring	467
Grain	467
has_minus_eq_impl< C >	467
has_minus_impl< C >	467
has_mul_eq_impl< C >	468
has_mul_impl< C >	468
has_operation< T >	468
has_plus_eq_impl< C >	469
has_plus_impl< C >	469
HelperFlag	469
HelperMod< Field, ElementTraits >	470
HelperMod< Field, ElementCategories::MachineIntTag >	470
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >	472
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >	472
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >	473
Hybrid	474
Info	474
Info	475
is_simd< T >	477
isSparseMatrix< Field, M >	477
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >	478
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >	478
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >	478
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >	478
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >	479
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >	479
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >	479
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >	480
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >	480
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >	480
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >	481
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >	481
isSparseMatrixMKLFormat< F, M >	481
isSparseMatrixSimdFormat< F, M >	481
isZOSparseMatrix< F, M >	482
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >	482
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >	482
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >	483
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >	483
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >	483
Iterative	483
LazyTag	
Performs field operations with delayed mod only when necessary. Result may not be reduced	484
limits< T >	484
limits< char >	484
limits< double >	485
limits< float >	485
limits< Givaro::Integer >	486
limits< int >	487
limits< long >	487
limits< long long >	488
limits< Reclnt::rint< K > >	489
limits< Reclnt::ruint< K > >	490
limits< short int >	490
limits< signed char >	491
limits< unsigned char >	492
limits< unsigned int >	492

limits< unsigned long > . . . . .	493
limits< unsigned long long > . . . . .	494
limits< unsigned short int > . . . . .	495
MachineFloatTag	
Float or double . . . . .	495
MachineIntTag	
Short, int, long, long long, and unsigned variants . . . . .	496
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > . . . . .	496
MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .	501
MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .	503
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > . . . . .	505
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > . . . . .	507
MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .	
FGEMM Helper for Default and ConvertTo modes of operation . . . . .	509
ModeTraits< Field > . . . . .	
ModeTraits . . . . .	511
ModeTraits< Givaro::Modular< Element, Compute > > . . . . .	511
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > . . . . .	511
ModeTraits< Givaro::Modular< int16_t, Compute > > . . . . .	512
ModeTraits< Givaro::Modular< int32_t, Compute > > . . . . .	512
ModeTraits< Givaro::Modular< int8_t, Compute > > . . . . .	512
ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > > . . . . .	513
ModeTraits< Givaro::Modular< uint16_t, Compute > > . . . . .	513
ModeTraits< Givaro::Modular< uint32_t, Compute > > . . . . .	514
ModeTraits< Givaro::Modular< uint8_t, Compute > > . . . . .	514
ModeTraits< Givaro::ModularBalanced< Element > > . . . . .	514
ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > . . . . .	515
ModeTraits< Givaro::ModularBalanced< int16_t > > . . . . .	515
ModeTraits< Givaro::ModularBalanced< int32_t > > . . . . .	515
ModeTraits< Givaro::ModularBalanced< int8_t > > . . . . .	516
ModeTraits< Givaro::Montgomery< T > > . . . . .	516
ModeTraits< Givaro::ZRing< double > > . . . . .	516
ModeTraits< Givaro::ZRing< float > > . . . . .	517
ModeTraits< Givaro::ZRing< Givaro::Integer > > . . . . .	517
ModularBalanced< T > . . . . .	517
ModularTag	
This is a modular field like e.g. Modular<T> or ModularBalanced<T> . . . . .	518
Montgomery< T > . . . . .	518
need_field_characteristic< Field > . . . . .	518
need_field_characteristic< Givaro::Modular< Field > > . . . . .	518
need_field_characteristic< Givaro::ModularBalanced< Field > > . . . . .	518
NoSimd< T > . . . . .	519
Parallel< C, P > . . . . .	520
RNSInteger< RNS >::RandIter . . . . .	521
RNSIntegerMod< RNS >::RandIter . . . . .	522
readMyMachineType< Field, T > . . . . .	523
readMyMachineType< Field, mpz_t > . . . . .	524
Recursive . . . . .	525
Recursive . . . . .	525
rint< K > . . . . .	525
rns_double . . . . .	525
rns_double_elt . . . . .	530
rns_double_elt_cstptr . . . . .	532
rns_double_elt_ptr . . . . .	535
rns_double_extended . . . . .	538
RNSElementTag	
Representation in a Residue Number System . . . . .	542

RNSInteger< RNS >	543
RNSIntegerMod< RNS >	546
rnsRandIter< RNS >	553
Row	554
ruInt< K >	554
ScalFunctions< Element, Enable >	554
ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type >	555
ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type >	558
Sequential	564
Simd128_impl< ArithType, Int, Signed, Size >	565
Simd128_impl< true, false, true, 4 >	565
Simd128_impl< true, false, true, 8 >	565
Simd128_impl< true, true, false, 2 >	566
Simd128_impl< true, true, false, 4 >	574
Simd128_impl< true, true, false, 8 >	583
Simd128_impl< true, true, true, 2 >	592
Simd128_impl< true, true, true, 4 >	601
Simd128_impl< true, true, true, 8 >	609
Simd128fp_base	618
Simd128i_base	619
Simd256_impl< ArithType, Int, Signed, Size >	620
Simd256_impl< true, false, true, 4 >	620
Simd256_impl< true, false, true, 8 >	621
Simd256_impl< true, true, false, 2 >	628
Simd256_impl< true, true, false, 4 >	636
Simd256_impl< true, true, false, 8 >	653
Simd256_impl< true, true, true, 2 >	662
Simd256_impl< true, true, true, 4 >	670
Simd256_impl< true, true, true, 8 >	687
Simd256fp_base	695
Simd256i_base	696
Simd512_impl< ArithType, Int, Signed, Size >	697
Simd512_impl< true, false, true, 4 >	697
Simd512_impl< true, false, true, 8 >	697
Simd512_impl< true, true, false, 8 >	704
Simd512_impl< true, true, true, 8 >	713
Simd512fp_base	723
Simd512i_base	723
SimdChooser< T, bool, bool >	725
SimdChooser< T, false, b >	725
SimdChooser< T, true, false >	725
SimdChooser< T, true, true >	725
simdToType< T >	726
Single	726
Sparse< Field, SparseMatrix_t, IdxT, PtrT >	726
Sparse< _Field, SparseMatrix_t::COO >	726
Sparse< _Field, SparseMatrix_t::COO_ZO >	728
Sparse< _Field, SparseMatrix_t::CSR >	729
Sparse< _Field, SparseMatrix_t::CSR_HYB >	731
Sparse< _Field, SparseMatrix_t::CSR_ZO >	733
Sparse< _Field, SparseMatrix_t::ELL >	735
Sparse< _Field, SparseMatrix_t::ELL_simd >	736
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >	738
Sparse< _Field, SparseMatrix_t::ELL_ZO >	740
Sparse< _Field, SparseMatrix_t::HYB_ZO >	741
Sparse< _Field, SparseMatrix_t::SELL >	743
Sparse< _Field, SparseMatrix_t::SELL_ZO >	745
SpMat< Field, flag >	747



Static_error_check< bool >	748
Static_error_check< false >	748
StatsMatrix	748
support_fast_mod< T >	751
support_fast_mod< double >	752
support_fast_mod< float >	752
support_fast_mod< int64_t >	752
support_simd< T >	753
support_simd_add< T >	753
support_simd_mod< T >	753
tfn_minus	753
tfn_minus_eq	754
tfn_mul	754
tfn_mul_eq	755
tfn_plus	755
tfn_plus_eq	755
Threads	756
ThreeD	756
ThreeDAdaptive	756
ThreeDInPlace	756
TRSMHelper< ReclterTrait, ParSeqTrait >	
TRSM Helper	756
TwoD	758
TwoDAdaptive	758
UnparametricTag	
If the field uses a representation with infix operators	758
Winograd	758
WinogradPar	758



# Chapter 13

## File Index

### 13.1 File List

Here is a list of all files with brief descriptions:

101-fgemm.C	759
2x2-fgemm.C	759
2x2-ftsrv.C	760
2x2-pluq.C	760
align-allocator.h	761
args-parser.h	761
arithprog.C	763
benchmark-charpoly-mp.C	764
benchmark-charpoly.C	764
benchmark-checkers.C	765
benchmark-dgemm.C	766
benchmark-dgetrf.C	767
benchmark-dgetri.C	768
benchmark-dsytrf.C	769
benchmark-dtrsm.C	770
benchmark-dtrtri.C	770
benchmark-fadd-lvl2.C	771
benchmark-fdot.C	772
benchmark-fgemm-mp.C	773
benchmark-fgemm-rns.C	774
benchmark-fgemm.C	776
benchmark-fgemv-mp.C	776
benchmark-fgemv.C	777
benchmark-fgesv.C	780
benchmark-fsyrk.C	781
benchmark-fsytrf.C	782
benchmark-ftsm-mp.C	783
benchmark-ftsm.C	783
benchmark-ftsrv.C	784
benchmark-fttri.C	785
benchmark-inverse.C	785
benchmark-lqmp-mp.C	786
benchmark-lqmp.C	787
benchmark-pluq.C	787
benchmark-wino.C	789

bit_manipulation.h	790
blockcuts.inl	790
cast.h	792
cblas.C	792
autotune/charpoly.C	793
examples/charpoly.C	794
checker_charpoly.inl	794
checker_det.inl	795
checker_empty.h	795
checker_fgemm.inl	795
checker_ftrsm.inl	796
checker_invert.inl	796
checker_pluq.inl	796
checkers.dox	797
checkers_fflas.h	797
checkers_fflas.inl	797
checkers_ffpack.h	798
checkers_ffpack.inl	798
clapack.C	799
config-blas.h	800
config.h	807
fflas-ffpack/config.h	811
coo.h	815
coo_spmv.inl	815
coo_spmv.inl	816
coo_utils.inl	817
csr.h	818
csr_hyb.h	818
csr_hyb_pspmm.inl	819
csr_hyb_pspmv.inl	819
csr_hyb_spmv.inl	820
csr_hyb_spmv.inl	821
csr_hyb_utils.inl	821
csr_pspmm.inl	822
csr_pspmv.inl	822
csr_spmv.inl	823
csr_spmv.inl	824
csr_utils.inl	825
cuda.C	826
debug.h	
Various utilities for debugging	826
det.C	827
ell.h	828
ell_pspmm.inl	828
ell_pspmv.inl	829
ell_simd.h	830
ell_simd_pspmv.inl	830
ell_simd_spmv.inl	831
ell_simd_utils.inl	832
ell_spmv.inl	833
ell_spmv.inl	834
ell_utils.inl	835
fblas.C	835
fflas-101_1.C	836
fflas-101_3.C	836
fflas-ffpack-config.h	
Defaults for optimised values	837
fflas-ffpack-default-thresholds.h	837

fflas-ffpack-thresholds.h	839
fflas-ffpack.doxy	839
fflas-ffpack.h	
Includes <b>FFLAS</b> and <b>FFPACK</b>	839
fflas.doxy	839
fflas.h	
<b>Finite Field Linear Algebra Subroutines</b>	839
fflas_101.C	840
fflas_101_lv1.C	841
fflas_bounds.inl	841
fflas_c.h	842
fflas_enum.h	855
fflas_fadd.h	856
fflas_fadd.inl	857
fflas_fassign.h	858
fflas_fassign.inl	858
fflas_faxpy.inl	859
fflas_fdot.inl	860
fflas_fgemm.inl	861
fflas_fgemv.inl	863
fflas_fgemv_mp.inl	864
fflas_fger.inl	865
fflas_fger_mp.inl	866
fflas_freduce.h	867
fflas_freduce.inl	868
fflas_freduce_mp.inl	870
fflas_freivalds.inl	870
fflas_fscal.h	871
fflas_fscal.inl	871
fflas_fscal_mp.inl	872
fflas_fsyr2k.inl	873
fflas_fsyrk.inl	873
fflas_ftmm.inl	874
fflas_ftsm.inl	875
fflas_ftsm_mp.inl	
Triangular system with matrix right hand side over multiprecision domain (either over $\mathbb{Z}$ or over $\mathbb{Z}/p\mathbb{Z}$ )	876
fflas_ftsv.inl	876
fflas_helpers.inl	877
fflas_intrinsic.h	878
fflas_io.h	878
fflas_L1_inst.C	879
fflas_L1_inst.h	880
fflas_L1_inst_implem.inl	881
fflas_L2_inst.C	883
fflas_L2_inst.h	884
fflas_L2_inst_implem.inl	885
fflas_L3_inst.C	887
fflas_L3_inst.h	888
fflas_L3_inst_implem.inl	889
fflas_level1.inl	890
fflas_level2.inl	892
fflas_level3.inl	895
fflas_lv1.C	
C functions calls for level 1 <b>FFLAS</b> in fflas-c.h	897
fflas_lv2.C	
C functions calls for level 2 <b>FFLAS</b> in fflas-c.h	902

fflas_lvl3.C	
C functions calls for level 3 <b>FFLAS</b> in fflas-c.h	908
fflas_memory.h	910
fflas_pfgemm.inl	910
fflas_pftsm.inl	911
fflas_plevel1.h	912
fflas_randommatrix.h	912
fflas_simd.h	914
fflas_sparse.C	
C functions calls for level 1.5 and 2.5 <b>FFLAS</b> in fflas-c.h	916
fflas_sparse.h	916
fflas_sparse.inl	921
ffpack-fgesv.C	923
ffpack-solve.C	923
ffpack.C	
C functions calls for <b>FFPACK</b> in ffpack-c.h	924
ffpack.doxy	945
ffpack.h	
Set of elimination based routines for dense linear algebra	945
ffpack.inl	954
ffpack_c.h	955
ffpack_charpoly.inl	976
ffpack_charpoly_danilevski.inl	977
ffpack_charpoly_kgfast.inl	978
ffpack_charpoly_kgfastgeneralized.inl	978
ffpack_charpoly_kglu.inl	979
ffpack_charpoly_mp.inl	979
ffpack_det_mp.inl	980
ffpack_echelonforms.inl	981
ffpack_fgesv.inl	982
ffpack_fgets.inl	983
ffpack_frobenius.inl	983
ffpack_fsytrf.inl	984
ffpack_ftrssyr2k.inl	985
ffpack_ftrstr.inl	986
ffpack_ftrtr.inl	986
ffpack_inst.C	987
ffpack_inst.h	988
ffpack_inst_implem.inl	990
ffpack_invert.inl	993
ffpack_krylovelim.inl	993
ffpack_ludivine.inl	994
ffpack_ludivine_mp.inl	995
ffpack_minpoly.inl	995
ffpack_permutation.inl	996
ffpack_pluq.inl	998
ffpack_pluq_mp.inl	999
ffpack_ppluq.inl	1000
ffpack_rankprofiles.inl	1001
fgemm_classical.inl	1002
fgemm_classical_mp.inl	
Matrix multiplication with multiprecision input (either over $\mathbb{Z}$ or over $\mathbb{Z}/p\mathbb{Z}$ )	1002
fgemm_winograd.inl	1004
field-traits.h	
Field Traits	1006
field.doxy	1008
flimits.h	1008
fsyrk.C	1009

fsytrf.C	1010
fttrtri.C	1011
gmp.C	1012
hyb_zo.h	1012
hyb_zo_pspmm.inl	1013
hyb_zo_pspmv.inl	1013
hyb_zo_spm.inl	1014
hyb_zo_spmv.inl	1014
hyb_zo_utils.inl	1015
igemm.doxy	1015
igemm.h	1015
igemm.inl	1016
igemm_kernels.h	1016
igemm_kernels.inl	1017
igemm_tools.h	1018
igemm_tools.inl	1018
instrset.h	1019
instrset_detect.cpp	1021
interfaces.doxy	1022
kaapi_routines.inl	1022
lapack.C	1022
mainpage.doxy	1023
Matio.h	1023
matmul.C	1024
matmul.doxy	1024
parallel.h	1024
pfgemm_variants.inl	1031
pfgemv.inl	1032
autotune/pluq.C	1032
examples/pluq.C	1034
rank.C	1034
read_sparse.h	1034
regression-check.C	1036
rns-double-elt.h	
Rns elt structure with double support	1037
rns-double-recint.inl	1037
rns-double.h	
Rns structure with double support	1038
rns-double.inl	1038
rns-integer-mod.h	
Representation of $\mathbb{Z}/p\mathbb{Z}$ using RNS representation (note: fixed precision)	1039
rns-integer.h	
Representation of $\mathbb{Z}$ using RNS representation (note: fixed precision)	1040
rns.h	1040
rns.inl	1041
schedule_bini.inl	
Bini implementation	1041
schedule_winograd.inl	1042
schedule_winograd_acc.inl	1042
schedule_winograd_acc_ip.inl	1043
schedule_winograd_ip.inl	1044
sell.h	1044
sell_pspmv.inl	1045
sell_spmv.inl	1045
sell_utils.inl	1046
simd.doxy	1047
simd128.inl	1047
simd128_double.inl	1048

<a href="#">simd128_float.inl</a>	1048
<a href="#">simd128_int16.inl</a>	1048
<a href="#">simd128_int32.inl</a>	1049
<a href="#">simd128_int64.inl</a>	1049
<a href="#">simd256.inl</a>	1050
<a href="#">simd256_double.inl</a>	1050
<a href="#">simd256_float.inl</a>	1051
<a href="#">simd256_int16.inl</a>	1051
<a href="#">simd256_int32.inl</a>	1051
<a href="#">simd256_int64.inl</a>	1052
<a href="#">simd512.inl</a>	1052
<a href="#">simd512_double.inl</a>	1053
<a href="#">simd512_float.inl</a>	1053
<a href="#">simd512_int32.inl</a>	1053
<a href="#">simd512_int64.inl</a>	1054
<a href="#">simd_modular.inl</a>	1054
<a href="#">solve.C</a>	1054
<a href="#">sparse_matrix_traits.h</a>	1055
<a href="#">test-charpoly-check.C</a>	1056
<a href="#">test-charpoly.C</a>	1057
<a href="#">test-compressQ.C</a>	1058
<a href="#">test-det-check.C</a>	1059
<a href="#">test-det.C</a>	1060
<a href="#">test-echelon.C</a>	1060
<a href="#">test-fadd.C</a>	1063
<a href="#">test-fdot.C</a>	1064
<a href="#">test-fgemm-check.C</a>	1065
<a href="#">test-fgemm.C</a>	1067
<a href="#">test-fgemv.C</a>	1069
<a href="#">test-fger.C</a>	1071
<a href="#">test-fgesv.C</a>	1073
<a href="#">test-finit.C</a>	1074
<a href="#">test-fscal.C</a>	1075
<a href="#">test-fsyr2k.C</a>	1076
<a href="#">test-fsyrr.C</a>	1078
<a href="#">test-fsytrf.C</a>	1079
<a href="#">test-ftrmm.C</a>	1081
<a href="#">test-ftrmv.C</a>	1082
<a href="#">test-ftrsm-check.C</a>	1083
<a href="#">test-ftrsm.C</a>	1084
<a href="#">test-ftrssyr2k.C</a>	1085
<a href="#">test-ftrstr.C</a>	1086
<a href="#">test-ftrsv.C</a>	1088
<a href="#">test-ftrtri.C</a>	1089
<a href="#">test-interfaces-c.c</a>	1090
<a href="#">test-invert-check.C</a>	1090
<a href="#">test-io.C</a>	1091
<a href="#">test-lu.C</a>	1092
<a href="#">test-maxdelayeddim.C</a>	1096
<a href="#">test-minpoly.C</a>	1097
<a href="#">test-multifile1.C</a>	1098
<a href="#">test-multifile2.C</a>	1098
<a href="#">test-nullspace.C</a>	1098
<a href="#">test-permutations.C</a>	1100
<a href="#">test-pluq-check.C</a>	1101
<a href="#">test-rankprofiles.C</a>	1102
<a href="#">test-rpm.C</a>	1103
<a href="#">test-simd.C</a>	1103



test-solve.C	1107
test-utils.h	1108
timer.h	1109
utils.h	1109
winograd.C	1110



## Chapter 14

# Module Documentation

### 14.1 CHECKER

Class CHECKER provides functions to verify computations in [FFLAS](#) and [FFPACK](#).

Class CHECKER provides functions to verify computations in [FFLAS](#) and [FFPACK](#).

### 14.2 FFLAS

The C-style wrapper of BLAS for finite field linear algebra.

The C-style wrapper of BLAS for finite field linear algebra.

[FFLAS](#), Finite Field Linear Algebra Subroutines, provide basic linear algebra subroutines based on the BLAS interface. Therefore, the specifications are in C style; only the field given as a template parameter requires C++.

As much as possible, these routines use [ATLAS/BLAS](#) computations and achieve therefore high efficiency.

### 14.3 Matrix Multiplication Algorithms

Matrix Multiplication (level 3) algorithms.

#### Files

- file [schedule\\_bini.inl](#)  
*Bini implementation.*

#### 14.3.1 Detailed Description

Matrix Multiplication (level 3) algorithms.

[Todo](#) biblio

## 14.4 SIMD wrapper

wraps SIMD functions Supportst SSE4.1, AVX, AVX2.

wraps SIMD functions Supportst SSE4.1, AVX, AVX2.

**Todo** biblio

## 14.5 FFLAS-FFPACK

the [FFLAS FFPACK](#) library

### Modules

- [FFLAS](#)  
*The C-style wrapper of BLAS for finite field linear algebra.*
- [Interfaces](#)  
*Intefaces for FFLAS-FFPACK.*

### 14.5.1 Detailed Description

the [FFLAS FFPACK](#) library

C++ header library for fast exact dense linear algebra

See also

[FFLAS](#)  
[FFPACK](#)

## 14.6 FFPACK

Class [FFPACK](#) provides functions using fflas much as Lapack uses BLAS.

Class [FFPACK](#) provides functions using fflas much as Lapack uses BLAS.

## 14.7 FFLAS-FFPACK fields

fields in the FFLAS-FFPACK library

## Files

- file [rns-double-elt.h](#)  
*rns elt structure with double support*
- file [rns-double.h](#)  
*rns structure with double support*
- file [rns-integer-mod.h](#)  
*representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)*
- file [rns-integer.h](#)  
*representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)*
- file [rns.h](#)

### 14.7.1 Detailed Description

fields in the FFLAS-FFPACK library

Unparametric/Random elements

[Todo](#) [biblio](#)

## 14.8 RNS

just include them all

just include them all

## 14.9 Interfaces

Intefaces for FFLAS-FFPACK.

Intefaces for FFLAS-FFPACK.

C interface in folder

See also

[libs](#)



## Chapter 15

# Namespace Documentation

### 15.1 FFLAS Namespace Reference

#### Namespaces

- [BLAS3](#)
- [csr\\_hyb\\_details](#)
- [CuttingStrategy](#)
- [details](#)
- [details\\_spmv](#)
- [ElementCategories](#)
- [FieldCategories](#)

*Traits and categories will need to be placed in a proper file later.*

- [MMHelperAlgo](#)
- [ModeCategories](#)

*Specifies the mode of action for an algorithm w.r.t.*

- [ParSeqHelper](#)

*ParSeqHelper for both fgemm and ftrsm.*

- [Protected](#)
- [sell\\_details](#)
- [sparse\\_details](#)
- [sparse\\_details\\_impl](#)
- [StrategyParameter](#)
- [StructureHelper](#)

*StructureHelper for ftrsm.*

- [vectorised](#)

#### Data Structures

- struct [Checker\\_Empty](#)
- class [CheckerImplem\\_fgemm](#)
- class [CheckerImplem\\_ftrsm](#)
- struct [support\\_simd\\_add](#)
- struct [MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >](#)
- struct [MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >](#)
- struct [MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >](#)

- struct [support\\_simd\\_mod](#)
- struct [support\\_fast\\_mod](#)
- struct [support\\_fast\\_mod< float >](#)
- struct [support\\_fast\\_mod< double >](#)
- struct [support\\_fast\\_mod< int64\\_t >](#)
- struct [AlgoChooser](#)
- struct [AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >](#)
- struct [MMHelper](#)
- struct [MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >](#)

*FGEMM Helper for Default and ConvertTo modes of operation.*

- struct [MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >](#)
- struct [TRSMHelper](#)

*TRSM Helper.*

- struct [support\\_simd](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::COO >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::COO\\_ZO >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::CSR >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_ZO >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_HYB >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_ZO >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::HYB\\_ZO >](#)
- struct [readMyMachineType](#)
- struct [readMyMachineType< Field, mpz\\_t >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::SELL >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::SELL\\_ZO >](#)
- struct [isSparseMatrix](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::COO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::COO\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::SELL > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::SELL\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_simd > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR\\_HYB > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::HYB\\_ZO > >](#)
- struct [isZOSparseMatrix](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR\\_ZO > >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::COO\\_ZO > >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_ZO > >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::SELL\\_ZO > >](#)
- struct [isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO > >](#)
- struct [isSparseMatrixSimdFormat](#)
- struct [isSparseMatrixMKLFormat](#)
- struct [tfn\\_plus](#)
- struct [tfn\\_mul](#)
- struct [tfn\\_mul\\_eq](#)
- struct [tfn\\_minus](#)
- struct [tfn\\_plus\\_eq](#)



- struct [tfn\\_minus\\_eq](#)
- struct [has\\_plus\\_impl](#)
- struct [has\\_mul\\_impl](#)
- struct [has\\_mul\\_eq\\_impl](#)
- struct [has\\_plus\\_eq\\_impl](#)
- struct [has\\_minus\\_eq\\_impl](#)
- struct [has\\_minus\\_impl](#)
- struct [has\\_operation](#)
- struct [StatsMatrix](#)
- struct [Sparse](#)
- struct [HelperFlag](#)
- struct [CsrMat](#)
- struct [CooMat](#)
- struct [EiIMat](#)
- struct [SpMat](#)
- struct [ElementTraits](#)

*ElementTraits.*

- struct [ElementTraits< float >](#)
- struct [ElementTraits< double >](#)
- struct [ElementTraits< int8\\_t >](#)
- struct [ElementTraits< int16\\_t >](#)
- struct [ElementTraits< int32\\_t >](#)
- struct [ElementTraits< int64\\_t >](#)
- struct [ElementTraits< uint8\\_t >](#)
- struct [ElementTraits< uint16\\_t >](#)
- struct [ElementTraits< uint32\\_t >](#)
- struct [ElementTraits< uint64\\_t >](#)
- struct [ElementTraits< Givaro::Integer >](#)
- struct [ElementTraits< Reclnt::rint< K > >](#)
- struct [ElementTraits< Reclnt::ruint< K > >](#)
- struct [ElementTraits< Reclnt::rmint< K, MG > >](#)
- struct [ElementTraits< FFPACK::rns\\_double\\_elt >](#)
- struct [ModeTraits](#)

*ModeTraits.*

- struct [ModeTraits< Givaro::Modular< Element, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< int8\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< int16\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< int32\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< uint8\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< uint16\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< uint32\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< Element > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< int8\\_t > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< int16\\_t > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< int32\\_t > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >](#)
- struct [ModeTraits< Givaro::ZRing< Givaro::Integer > >](#)
- struct [ModeTraits< Givaro::ZRing< float > >](#)
- struct [ModeTraits< Givaro::ZRing< double > >](#)
- struct [ModeTraits< Givaro::Montgomery< T > >](#)
- struct [FieldTraits](#)

*FieldTrait.*

- struct [FieldTraits](#)< [Givaro::ZRing](#)< [Reclnt::ruint](#)< K > > >
- struct [FieldTraits](#)< [Givaro::Modular](#)< [Element](#) > >
- struct [FieldTraits](#)< [Givaro::ModularBalanced](#)< [Element](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [double](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [float](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [int16\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint16\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [int32\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint32\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [int64\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint64\\_t](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > >
- struct [FieldTraits](#)< [FFPACK::RNSInteger](#)< T > >
- struct [FieldTraits](#)< [FFPACK::RNSIntegerMod](#)< T > >
- struct [associatedDelayedField](#)
- struct [associatedDelayedField](#)< const [Givaro::Modular](#)< T, X > >
- struct [associatedDelayedField](#)< const [Givaro::ModularBalanced](#)< T > >
- struct [associatedDelayedField](#)< const [Givaro::ZRing](#)< T > >
- struct [associatedDelayedField](#)< const [FFPACK::RNSIntegerMod](#)< RNS > >
- struct [ForStrategy1D](#)
- struct [ForStrategy2D](#)

## Typedefs

- template<class Field >  
using [Checker\\_fgemm](#) = [FFLAS::Checker\\_Empty](#)< Field >
- template<class Field >  
using [Checker\\_ftrsm](#) = [FFLAS::Checker\\_Empty](#)< Field >
- template<class Field >  
using [ForceCheck\\_fgemm](#) = [CheckerImplem\\_fgemm](#)< Field >
- template<class Field >  
using [ForceCheck\\_ftrsm](#) = [CheckerImplem\\_ftrsm](#)< Field >
- using [ZOSparseMatrix](#) = std::true\_type
- using [NotZOSparseMatrix](#) = std::false\_type
- using [SimdSparseMatrix](#) = std::true\_type
- using [NoSimdSparseMatrix](#) = std::false\_type
- using [MKLSparseMatrixFormat](#) = std::true\_type
- using [NotMKLSparseMatrixFormat](#) = std::false\_type
- template<class T >  
using [has\\_plus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_plus\\_impl](#)< T > >::type
- template<class T >  
using [has\\_minus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_minus\\_impl](#)< T > >::type
- template<class T >  
using [has\\_equal](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, std::is\_copyable\_↵\_assignable< T > >::type
- template<class T >  
using [has\\_plus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_plus\\_eq\\_impl](#)< T > >::type
- template<class T >  
using [has\\_minus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_minus\\_eq\\_impl](#)< T > >::type

- `template<class T >`  
using `has_mul` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_mul_impl< T > >::type`
- `template<class T >`  
using `has_mul_eq` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_mul_eq_impl< T > >::type`
- `typedef Givaro::Timer` `Timer`
- `typedef Givaro::BaseTimer` `BaseTimer`
- `typedef Givaro::UserTimer` `UserTimer`
- `typedef Givaro::SysTimer` `SysTimer`

## Enumerations

- enum `FFLAS_ORDER` { `FflasRowMajor` = 101 , `FflasColMajor` = 102 }  
*Storage by row or col ?*
- enum `FFLAS_TRANSPOSE` { `FflasNoTrans` = 111 , `FflasTrans` = 112 }  
*Is matrix transposed ?*
- enum `FFLAS_UPLO` { `FflasUpper` = 121 , `FflasLower` = 122 }  
*Is triangular matrix's shape upper ?*
- enum `FFLAS_DIAG` { `FflasNonUnit` = 131 , `FflasUnit` = 132 }  
*Is the triangular matrix implicitly unit diagonal ?*
- enum `FFLAS_SIDE` { `FflasLeft` = 141 , `FflasRight` = 142 }  
*On what side ?*
- enum `FFLAS_BASE` { `FflasDouble` = 151 , `FflasFloat` = 152 , `FflasGeneric` = 153 }  
*FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.*
- enum `number_kind` { `zero` = 0 , `one` = 1 , `mone` = -1 , `other` = 2 }
- enum class `SparseMatrix_t` {  
    `CSR` , `CSR_ZO` , `CSC` , `CSC_ZO` ,  
    `COO` , `COO_ZO` , `ELL` , `ELL_ZO` ,  
    `SELL` , `SELL_ZO` , `ELL_simd` , `ELL_simd_ZO` ,  
    `CSR_HYB` , `HYB_ZO` }
- enum `FFLAS_FORMAT` {  
    `FflasAuto` = 0 , `FflasDense` = 1 , `FflasSMS` = 2 , `FflasBinary` = 3 ,  
    `FflasMath` = 4 , `FflasMaple` = 5 , `FflasSageMath` = 6 }

## Functions

- `Givaro::Integer` `InfNorm` (const `size_t` M, const `size_t` N, const `Givaro::Integer` \*A, const `size_t` lda)
- `template<class T >`  
const T & `min3` (const T &m, const T &n, const T &k)
- `template<class T >`  
const T & `max3` (const T &m, const T &n, const T &k)
- `template<class T >`  
const T & `min4` (const T &m, const T &n, const T &k, const T &l)
- `template<class T >`  
const T & `max4` (const T &m, const T &n, const T &k, const T &l)
- `template<class Field >`  
void `fadd` (const `Field` &F, const `size_t` N, `typename Field::ConstElement_ptr` A, const `size_t` inca, `typename Field::ConstElement_ptr` B, const `size_t` incb, `typename Field::Element_ptr` C, const `size_t` incc)
- `template<class Field >`  
void `faddin` (const `Field` &F, const `size_t` N, `typename Field::ConstElement_ptr` B, const `size_t` incb, `typename Field::Element_ptr` C, const `size_t` incc)

- `template<class Field >`  
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition.*
- `template<class Field >`  
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsub : matrix subtraction.*
- `template<class Field >`  
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*faddin*
- `template<class Field >`  
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsubin  $C = C - B$*
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition with scaling.*
- `template<class Field >`  
`void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  
*fassign :  $x \leftarrow y$ .*
- `template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`

- `template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<class Field >`  
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$fassign : A \leftarrow B.$$
- `template<class Field >`  
`void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::ConstElement_ptr a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DelayedTag &MT)`
- `template<> Givaro::DoubleDomain::Element fdot (const Givaro::DoubleDomain &, const size_t N, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<> Givaro::FloatDomain::Element fdot (const Givaro::FloatDomain &, const size_t N, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field, class T >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::ConvertTo< T > &MT)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultBoundedTag &dbt)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, const ParSeqHelper::Sequential seq)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  

$$fdot: \text{dot product } x^T y.$$
- `template<typename RNS, typename ParSeqTrait >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >`

- >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)
- template<typename RNS >  
FFPACK::RNSInteger< RNS >::Element\_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement\_ptr Ad, const size\_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement\_ptr Bd, const size\_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element\_ptr Cd, const size\_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)
  - template<typename RNS, typename ParSeqTrait >  
FFPACK::RNSInteger< RNS >::Element\_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement\_ptr Ad, const size\_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement\_ptr Bd, const size\_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element\_ptr Cd, const size\_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)
  - template<typename RNS, typename Cut, typename Param >  
FFPACK::RNSInteger< RNS >::Element\_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement\_ptr Ad, const size\_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement\_ptr Bd, const size\_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element\_ptr Cd, const size\_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)
  - template<class ParSeq >  
Givaro::Integer \* fgemm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, const Givaro::Integer \*B, const size\_t ldb, Givaro::Integer beta, Givaro::Integer \*C, const size\_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
  - template<typename RNS, class ModeT >  
RNS::Element\_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename RNS::Element alpha, typename RNS::ConstElement\_ptr Ad, const size\_t lda, typename RNS::ConstElement\_ptr Bd, const size\_t ldb, const typename RNS::Element beta, typename RNS::Element\_ptr Cd, const size\_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)
  - template<typename RNS >  
RNS::Element\_ptr fgemm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename RNS::Element alpha, typename RNS::ConstElement\_ptr Ad, const size\_t lda, typename RNS::ConstElement\_ptr Bd, const size\_t ldb, const typename RNS::Element beta, typename RNS::Element\_ptr Cd, const size\_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)
  - Givaro::Integer \* fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, const Givaro::Integer \*B, const size\_t ldb, const Givaro::Integer beta, Givaro::Integer \*C, const size\_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
  - template<class ParSeq >  
Givaro::Integer \* fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, const Givaro::Integer \*B, const size\_t ldb, const Givaro::Integer beta, Givaro::Integer \*C, const size\_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)

- `template<size_t K1, size_t K2, class ParSeq >`  
`Reclnt::ruint< K1 > * fgemm (const Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >`  
`&F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n,`  
`const size_t k, const Reclnt::ruint< K1 > alpha, const Reclnt::ruint< K1 > *A, const size_t lda, const`  
`Reclnt::ruint< K1 > *B, const size_t ldb, Reclnt::ruint< K1 > beta, Reclnt::ruint< K1 > *C, const size_t`  
`ldc, MMHelper< Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >, MMHelperAlgo::Classic,`  
`ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<class Field, class ModeT >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::Winograd, ModeT > &H)`
- `template<class Field, class ModeT, class Cut, class Param >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut, Param > > &H)`
- `template<class Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper<`  
`Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >,`  
`ParSeqHelper::Sequential > &H)`
- `template<typename Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, type-`  
`name Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const`  
`ParSeqHelper::Sequential seq)`
- `template<typename Field, class Cut, class Param >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, type-`  
`name Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const`  
`ParSeqHelper::Parallel< Cut, Param > par)`
- `template<typename Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const`  
`typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  

*fgemm: Field GENERAL Matrix Multiply.*
- `template<typename Field, class ModeT, class ParSeq >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::Auto, ModeT, ParSeq > &H)`
- `template<class Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)`
- `template<class Field >`  
`Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename`



[Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)

*fsquare: Squares a matrix.*

- `template<> double * fsquare (const Givaro::ModularBalanced< double > &F, const FFLAS\_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)`
- `template<> float * fsquare (const Givaro::ModularBalanced< float > &F, const FFLAS\_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)`
- `template<> double * fsquare (const Givaro::Modular< double > &F, const FFLAS\_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)`
- `template<> float * fsquare (const Givaro::Modular< float > &F, const FFLAS\_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)`
- `template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size_t lda, typename Field::ConstElement\_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)`
- `template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size_t lda, typename Field::ConstElement\_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)`
- `template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size_t lda, typename Field::ConstElement\_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size_t lda, typename Field::ConstElement\_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)`
- `template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE TransA, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size_t lda, typename Field::ConstElement\_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size_t incY)`

*finite prime Field GEneral Matrix Vector multiplication.*

- `Givaro::ZRing< int64\_t >::Element\_ptr fgemv (const Givaro::ZRing< int64\_t > &F, const FFLAS\_TRANSPOSE ta, const size_t M, const size_t N, const int64\_t alpha, const int64\_t *A, const size_t lda, const int64\_t *X, const size_t incX, const int64\_t beta, int64\_t *Y, const size_t incY, MMHelper< Givaro::ZRing< int64\_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `Givaro::DoubleDomain::Element\_ptr fgemv (const Givaro::DoubleDomain &F, const FFLAS\_TRANSPOSE ta, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement\_ptr A, const size_t lda, const Givaro::DoubleDomain::ConstElement\_ptr X, const size_t incX, const Givaro::DoubleDomain::Element beta, Givaro::DoubleDomain::Element\_ptr Y, const size_t incY, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement\_ptr A, const size_t lda, const typename Field::ConstElement\_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)`



- [Givaro::FloatDomain::Element\\_ptr fgemv](#) (const [Givaro::FloatDomain](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [Givaro::FloatDomain::Element](#) alpha, const [Givaro::FloatDomain::ConstElement\\_ptr](#) A, const size\_t lda, const [Givaro::FloatDomain::ConstElement\\_ptr](#) X, const size\_t incX, const [Givaro::FloatDomain::Element](#) beta, [Givaro::FloatDomain::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [Givaro::FloatDomain](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- `template<class Field, class Cut, class Param >`  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [ParSeqHelper::Parallel](#)< Cut, Param > &parH)
- `template<class Field >`  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [ParSeqHelper::Sequential](#) &seqH)
- [FFPACK::rns\\_double::Element\\_ptr fgemv](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [FFPACK::rns\\_double::Element\\_ptr fgemv](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [Givaro::Integer \\* fgemv](#) (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [Givaro::Integer](#) alpha, [Givaro::Integer \\*A](#), const size\_t lda, [Givaro::Integer \\*X](#), const size\_t ldx, [Givaro::Integer](#) beta, [Givaro::Integer \\*Y](#), const size\_t ldy, [MMHelper](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) > > &H)
- [Givaro::Integer \\* fgemv](#) (const [Givaro::Modular](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [Givaro::Integer](#) alpha, [Givaro::Integer \\*A](#), const size\_t lda, [Givaro::Integer \\*X](#), const size\_t ldx, [Givaro::Integer](#) beta, [Givaro::Integer \\*Y](#), const size\_t ldy, [MMHelper](#)< [Givaro::Modular](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) > > &H)
- `template<size_t K1, size_t K2, class ParSeq >`  
[RecInt::ruint](#)< K1 > \* [fgemv](#) (const [Givaro::Modular](#)< [RecInt::ruint](#)< K1 >, [RecInt::ruint](#)< K2 > > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [RecInt::ruint](#)< K1 > alpha, const [RecInt::ruint](#)< K1 > \*A, const size\_t lda, const [RecInt::ruint](#)< K1 > \*X, const size\_t incx, [RecInt::ruint](#)< K1 > beta, [RecInt::ruint](#)< K1 > \*Y, const size\_t incy, [MMHelper](#)< [Givaro::Modular](#)< [RecInt::ruint](#)< K1 >, [RecInt::ruint](#)< K2 > >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) >, [ParSeq](#) > &H)
- `template<class Field >`  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fger: rank one update of a general matrix*
- `template<class Field >`  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) > > &H)
- `template<class Field, class AnyTag >`  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), AnyTag > &H)

- void `fger` (const Givaro::DoubleDomain &F, const size\_t M, const size\_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement\_ptr x, const size\_t incx, const Givaro::DoubleDomain::ConstElement\_ptr y, const size\_t incy, Givaro::DoubleDomain::Element\_ptr A, const size\_t lda, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field >  
void `fger` (const Field &F, const size\_t M, const size\_t N, const typename Field::Element alpha, const typename Field::ConstElement\_ptr x, const size\_t incx, const typename Field::ConstElement\_ptr y, const size\_t incy, typename Field::Element\_ptr A, const size\_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)
- void `fger` (const Givaro::FloatDomain &F, const size\_t M, const size\_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement\_ptr x, const size\_t incx, const Givaro::FloatDomain::ConstElement\_ptr y, const size\_t incy, Givaro::FloatDomain::Element\_ptr A, const size\_t lda, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<class Field >  
void `fger` (const Field &F, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr x, const size\_t incx, typename Field::ConstElement\_ptr y, const size\_t incy, typename Field::Element\_ptr A, const size\_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field >  
void `fger` (const Field &F, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr x, const size\_t incx, typename Field::ConstElement\_ptr y, const size\_t incy, typename Field::Element\_ptr A, const size\_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
- void `fger` (const Givaro::Modular< Givaro::Integer > &F, const size\_t M, const size\_t N, const typename Givaro::Integer alpha, typename Givaro::Integer \*x, const size\_t incx, typename Givaro::Integer \*y, const size\_t incy, typename Givaro::Integer \*A, const size\_t lda, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<typename RNS >  
void `fger` (const FFPACK::RNSInteger< RNS > &F, const size\_t M, const size\_t N, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::Element\_ptr x, const size\_t incx, typename FFPACK::RNSInteger< RNS >::Element\_ptr y, const size\_t incy, typename FFPACK::RNSInteger< RNS >::Element\_ptr A, const size\_t lda, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<typename RNS >  
void `fger` (const FFPACK::RNSIntegerMod< RNS > &F, const size\_t M, const size\_t N, const typename FFPACK::RNSIntegerMod< RNS >::Element alpha, typename FFPACK::RNSIntegerMod< RNS >::Element\_ptr x, const size\_t incx, typename FFPACK::RNSIntegerMod< RNS >::Element\_ptr y, const size\_t incy, typename FFPACK::RNSIntegerMod< RNS >::Element\_ptr A, const size\_t lda, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > &H)
- template<class Field >  
void `freduce` (const Field &F, const size\_t n, typename Field::ConstElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)  
$$freduce\ x \leftarrow ymodF.$$
- template<class Field >  
void `freduce` (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)  
$$freduce\ x \leftarrow xmodF.$$
- template<class Field >  
void `freduce_constoverride` (const Field &F, const size\_t m, typename Field::ConstElement\_ptr A, const size\_t incX)
- template<class Field, class ConstOtherElement\_ptr >  
void `finit` (const Field &F, const size\_t n, ConstOtherElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)
- template<class Field >  
void `finit` (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)  
$$finit\ \text{Initializes } X \text{ in } F\$.$$
- template<class Field >  
void `freduce` (const Field &F, const size\_t m, const size\_t n, typename Field::Element\_ptr A, const size\_t lda)

- $$freduce\ A \leftarrow A \bmod F.$$
  - template<class Field >  
void `pfreduce` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda, const size\_t numths)
  - template<class Field >  
void `freduce` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::ConstElement_ptr` B, const size\_t ldb, typename `Field::Element_ptr` A, const size\_t lda)
$$freduce\ A \leftarrow B \bmod F.$$
  - template<class Field >  
void `freduce_constoverride` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::ConstElement_ptr` A, const size\_t lda)
  - template<class Field, class OtherElement\_ptr >  
void `finit` (const `Field` &F, const size\_t m, const size\_t n, const OtherElement\_ptr B, const size\_t ldb, typename `Field::Element_ptr` A, const size\_t lda)
$$finit\ A \leftarrow B \bmod F.$$
  - template<class Field >  
void `finit` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda)
$$finit\ \text{Initializes } A \text{ in } F^S.$$
  - template<> void `freduce` (const `FFPACK::RNSIntegerMod`< `FFPACK::rns_double` > &F, const size\_t n, `FFPACK::RNSIntegerMod`< `FFPACK::rns_double` >::Element\_ptr A, size\_t inc)
  - template<> void `freduce` (const `FFPACK::RNSIntegerMod`< `FFPACK::rns_double` > &F, const size\_t m, const size\_t n, `FFPACK::rns_double::Element_ptr` A, size\_t lda)
  - template<class Field >  
bool `freivalds` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size\_t m, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb, typename `Field::ConstElement_ptr` C, const size\_t ldc)
$$freivalds: \text{Freivalds } \mathbf{GENERAL\ Matrix\ Multiply\ Random\ Check}.$$
  - template<class Field >  
void `fscal` (const `Field` &F, const size\_t n, const typename `Field::Element` alpha, typename `Field::Element_ptr` X, const size\_t incX)
$$fscal\ x \leftarrow \alpha \cdot x.$$
  - template<class Field >  
void `fscal` (const `Field` &F, const size\_t n, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` X, const size\_t incX, typename `Field::Element_ptr` Y, const size\_t incY)
$$fscal\ y \leftarrow \alpha \cdot x.$$
  - template<> void `fscal` (const `Givaro::DoubleDomain` &, const size\_t N, const `Givaro::DoubleDomain::Element` a, `Givaro::DoubleDomain::ConstElement_ptr` x, const size\_t incx, `Givaro::DoubleDomain::Element_ptr` y, const size\_t incy)
  - template<> void `fscal` (const `Givaro::FloatDomain` &, const size\_t N, const `Givaro::FloatDomain::Element` a, `Givaro::FloatDomain::ConstElement_ptr` x, const size\_t incx, `Givaro::FloatDomain::Element_ptr` y, const size\_t incy)
  - template<> void `fscal` (const `Givaro::DoubleDomain` &, const size\_t N, const `Givaro::DoubleDomain::Element` a, `Givaro::DoubleDomain::Element_ptr` y, const size\_t incy)
  - template<> void `fscal` (const `Givaro::FloatDomain` &, const size\_t N, const `Givaro::FloatDomain::Element` a, `Givaro::FloatDomain::Element_ptr` y, const size\_t incy)
  - template<class Field >  
void `fscal` (const `Field` &F, const size\_t m, const size\_t n, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda)
$$fscal\ A \leftarrow a \cdot A.$$
  - template<class Field >  
void `fscal` (const `Field` &F, const size\_t m, const size\_t n, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::Element_ptr` B, const size\_t ldb)
$$fscal\ B \leftarrow a \cdot A.$$
  - template<> void `fscal` (const `FFPACK::RNSInteger`< `FFPACK::rns_double` > &F, const size\_t n, const `FFPACK::rns_double::Element` alpha, `FFPACK::rns_double::Element_ptr` A, const size\_t inc)

- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscaln (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t Ida)`
- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ida, FFPACK::rns_double::Element_ptr B, const size_t ldb)`
- `template<> void fscaln (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha, typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, const size_t inc)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscaln (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t Ida)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ida, FFPACK::rns_double::Element_ptr B, const size_t ldb)`
- `template<class Field >`  
`Field::Element_ptr fsyr2k (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fsyr2k: Symmetric Rank 2K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t Ida, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fsyrk: Symmetric Rank K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t Ida, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)`  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t Ida, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold)`
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t Ida, typename Field::ConstElement_ptr D, const size_t incD, const std::vector< bool > &twoBlock, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)`  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE`

TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb)

*ftmm: TRIangular Matrix Multiply.*

- template<class Field >
   
void **ftmm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)
   
*ftmm: TRIangular Matrix Multiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .*
- template<class Field >
   
void **ftsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb)
- template<class Field >
   
void **ftsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, const ParSeqHelper::Sequential &PSH)
- template<class Field, class Cut, class Param >
   
void **ftsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, const ParSeqHelper::Parallel< Cut, Param > &PSH)
- template<class Field, class ParSeqTrait = ParSeqHelper::Sequential>
   
void **ftsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, TRSMHelper< StructureHelper::Recursive, ParSeqTrait > &H)
- void **ftsm** (const Givaro::Modular< Givaro::Integer > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, Givaro::Integer \*B, const size\_t ldb)
- void **cblas\_imptrsm** (const enum FFLAS\_ORDER Order, const enum FFLAS\_SIDE Side, const enum FFLAS\_UPLO Uplo, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_DIAG Diag, const int M, const int N, const FFPACK::rns\_double\_elt alpha, FFPACK::rns\_double\_elt\_cstptr A, const int lda, FFPACK::rns\_double\_elt\_ptr B, const int ldb)
- template<class Field >
   
void **ftsv** (const Field &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr X, int incX)
   
*ftsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*
- void **igemm** (const enum FFLAS\_ORDER Order, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, const size\_t M, const size\_t N, const size\_t K, const int64\_t alpha, const int64\_t \*A, const size\_t lda, const int64\_t \*B, const size\_t ldb, const int64\_t beta, int64\_t \*C, const size\_t ldc)
- template<class Field, class OtherElement\_ptr >
   
void **finit** (const Field &F, const size\_t n, const OtherElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)
   
*finit  $x \leftarrow y \bmod F$ .*
- template<class Field, class OtherElement\_ptr >
   
void **fconvert** (const Field &F, const size\_t n, OtherElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)
   
*fconvert  $x \leftarrow y \bmod F$ .*
- template<class Field >
   
void **fnegin** (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)
   
*fnegin  $x \leftarrow -x$ .*
- template<class Field >
   
void **fneg** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)



- $f_{neg} x \leftarrow -y.$
- template<class Field >  
 void **fzero** (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)  
 $f_{zero} : A \leftarrow 0.$
- template<class Field , class RandIter >  
 void **frand** (const Field &F, RandIter &G, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)  
 $frand : A \leftarrow random.$
- template<class Field >  
 bool **fiszero** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr X, const size\_t incX)  
 $f_{iszero} : test X = 0.$
- template<class Field >  
 bool **fequal** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)  
 $fequal : test X = Y.$
- template<class Field >  
 void **faxpby** (const Field &F, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY)  
 $faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template<typename Field , class Cut , class Param >  
 Field::Element fdot (const Field &F, const size\_t N, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY, const ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field >  
 void **fswap** (const Field &F, const size\_t N, typename Field::Element\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY)  
 $f_{swap} : X \leftrightarrow Y.$
- template<class Field >  
 void **fzero** (const Field &F, const size\_t m, const size\_t n, typename Field::Element\_ptr A, const size\_t lda)  
 $f_{zero} : A \leftarrow 0.$
- template<class Field , class RandIter >  
 void **frand** (const Field &F, RandIter &G, const size\_t m, const size\_t n, typename Field::Element\_ptr A, const size\_t lda)  
 $frand : A \leftarrow random.$
- template<class Field >  
 bool **fequal** (const Field &F, const size\_t m, const size\_t n, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb)  
 $fequal : test A = B.$
- template<class Field >  
 bool **fiszero** (const Field &F, const size\_t m, const size\_t n, typename Field::ConstElement\_ptr A, const size\_t lda)  
 $f_{iszero} : test A = 0.$
- template<class Field >  
 void **fidentity** (const Field &F, const size\_t m, const size\_t n, typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element &d)  
*creates a diagonal matrix*
- template<class Field >  
 void **fidentity** (const Field &F, const size\_t m, const size\_t n, typename Field::Element\_ptr A, const size\_t lda)  
*creates a diagonal matrix*
- template<class Field , class OtherElement\_ptr >  
 void **finit** (const Field &F, const size\_t m, const size\_t n, typename Field::Element\_ptr A, const size\_t lda)  
 $f_{init}$  Initializes  $A$  in  $F^S$ .
- template<class Field , class OtherElement\_ptr >  
 void **fconvert** (const Field &F, const size\_t m, const size\_t n, OtherElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb)

- $fconvert\ A \leftarrow B \bmod F.$
- template<class Field >  
void **fnegin** (const Field &F, const size\_t m, const size\_t n, typename Field::Element\_ptr A, const size\_t lda)  
 $fnegin\ A \leftarrow -A.$
- template<class Field >  
void **fneg** (const Field &F, const size\_t m, const size\_t n, typename Field::ConstElement\_ptr B, const size\_t ldb, typename Field::Element\_ptr A, const size\_t lda)  
 $fneg\ A \leftarrow -B.$
- template<class Field >  
void **faxpby** (const Field &F, const size\_t m, const size\_t n, const typename Field::Element alpha, typename Field::ConstElement\_ptr X, const size\_t idx, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t ldy)  
 $faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template<class Field >  
void **fmove** (const Field &F, const size\_t m, const size\_t n, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb)  
 $fmove : A \leftarrow B\ and\ B \leftarrow 0.$
- template<class Field >  
size\_t **bitsize** (const Field &F, size\_t M, size\_t N, const typename Field::ConstElement\_ptr A, size\_t lda)  
*bitsize: Computes the largest bitsize of the matrix' coefficients.*
- template<> size\_t **bitsize< Givaro::ZRing< Givaro::Integer > >** (const Givaro::ZRing< Givaro::Integer > &F, size\_t M, size\_t N, const Givaro::Integer \*A, size\_t lda)
- template<class Field >  
void **ftsmv** (const Field &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr X, int incX)  
*ftsm: TRIangular Matrix Vector prodcut Computes  $X \leftarrow op(A)X$*
- template<class Field >  
void **ftsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb)  
*ftsm: TRIangular System solve with Matrix.*
- template<typename Field >  
Field::Element\_ptr **pfgemm** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, size\_t numthreads=0)
- template<class Field >  
Field::Element \* **pfgemm\_1D\_rec** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element \*C, const size\_t ldc, size\_t seuil)
- template<class Field >  
Field::Element \* **pfgemm\_2D\_rec** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element \*C, const size\_t ldc, size\_t seuil)
- template<class Field >  
Field::Element \* **pfgemm\_3D\_rec** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, size\_t seuil, size\_t \*x)
- template<class Field >  
Field::Element\_ptr **pfgemm\_3D\_rec2** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size\_t lda, const typename Field::Element\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, size\_t seuil, size\_t \*x)

- `template<class Field , class ModeTrait , class Strat , class Param >`  
`std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag`  
`>>::value, typename Field::Element\_ptr >::type fgemm (const Field &F, const FFLAS::FFLAS\_TRANSPOSE`  
`ta, const FFLAS::FFLAS\_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const`  
`typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size_t lda, typename`  
`Field::ConstElement\_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element\_ptr`  
`C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,`  
`Param > > &H)`
- `template<class Field , class Cut , class Param >`  
`Field::Element\_ptr ftrsm (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_UPLO`  
`UpLo, const FFLAS::FFLAS\_TRANSPOSE TA, const FFLAS::FFLAS\_DIAG Diag, const size_t m, const`  
`size_t n, const typename Field::Element alpha, typename Field::Element\_ptr A, const size_t lda, typename`  
`Field::Element\_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel<`  
`Cut, Param > > &H)`
- `template<class Field , class Cut , class Param >`  
`Field::Element\_ptr ftrsm (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_UPLO`  
`UpLo, const FFLAS::FFLAS\_TRANSPOSE TA, const FFLAS::FFLAS\_DIAG Diag, const size_t m, const`  
`size_t n, const typename Field::Element alpha, typename Field::Element\_ptr A, const size_t lda, typename`  
`Field::Element\_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut,`  
`Param > > &H)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::COO > &A)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::COO\_ZO > &A)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::COO > &A, const IndexT *row, const IndexT`  
`*col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::COO\_ZO > &A, const IndexT *row, const`  
`IndexT *col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::CSR > &A)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix\_t::CSR\_ZO > &A)`
- `template<class Field >`  
`std::ostream & sparse\_print (std::ostream &os, const Sparse< Field, SparseMatrix\_t::CSR > &A)`
- `template<class IndexT >`  
`void sparse\_init (const Givaro::Modular< Givaro::Integer > &F, Sparse< Givaro::Modular< Givaro::Integer`  
`>, SparseMatrix\_t::CSR > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim,`  
`uint64_t coldim, uint64_t nnz)`
- `template<class IndexT >`  
`void sparse\_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer > ,`  
`SparseMatrix\_t::CSR\_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim,`  
`uint64_t coldim, uint64_t nnz)`
- `template<class IndexT , size_t RECINT_SIZE>`  
`void sparse\_init (const Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >> &F, Sparse< Givaro::ZRing<`  
`Reclnt::rmint< RECINT_SIZE >>, SparseMatrix\_t::CSR\_ZO > &A, const IndexT *row, const IndexT *col,`  
`typename Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>::Element_ptr dat, uint64_t rowdim, uint64_t`  
`coldim, uint64_t nnz)`
- `template<class IndexT , size_t RECINT_SIZE>`  
`void sparse\_init (const Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >> &F, Sparse< Givaro::ZRing<`  
`Reclnt::rmint< RECINT_SIZE >>, SparseMatrix\_t::CSR > &A, const IndexT *row, const IndexT *col, type-`  
`name Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>::Element_ptr dat, uint64_t rowdim, uint64_t coldim,`  
`uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix\_t::CSR > &A, const IndexT *row, const IndexT`  
`*col, typename Field::ConstElement\_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`



- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::HYB_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<typename _Field >`  
`std::ostream & operator<< (std::ostream &os, const Sparse< _Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field , bool sorted = true, bool read_integer = false>`  
`void readSmsFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)`
- `template<class Field >`  
`void readSprFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)`
- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, int > getData Type ()`
- `template<class T >`  
`std::enable_if< std::is_floating_point< T >::value, int > getData Type ()`
- `template<class T >`  
`std::enable_if< std::is_same< T, mpz_t >::value, int > getData Type ()`
- `template<class T >`  
`int getData Type ()`
- `template<class Field >`  
`void readMachineType (const Field &F, typename Field::Element &modulo, typename Field::Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)`
- `template<class Field >`  
`void readDnsFormat (const std::string &path, const Field &F, index_t &rowdim, index_t &coldim, typename Field::Element_ptr &val)`

- `template<class Field >`  
`void writeDnsFormat (const std::string &path, const Field &F, const index_t &rowdim, const index_t &coldim,`  
`typename Field::Element_ptr A, index_t lIdA)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL > &A, const IndexT *row, const In-`  
`dexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz, uint64_t`  
`sigma=0)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL_ZO > &A, const IndexT *row, const`  
`IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class It >`  
`double computeDeviation (It begin, It end)`
- `template<class Field >`  
`StatsMatrix getStat (const Field &F, const index_t *row, const index_t *col, typename Field::ConstElement_ptr`  
`val, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename`  
`Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field, class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,`  
`const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)`
- `template<> void fflas_delete (FFPACK::rns_double_elt_ptr A)`
- `template<> void fflas_delete (FFPACK::rns_double_elt_cstptr A)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double`  
`> &F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double`  
`> &F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`  
`void finit_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const`  
`Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`  
`void finit_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k,`  
`const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`  
`void fconvert_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha,`  
`Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`
- `template<typename RNS >`  
`void fconvert_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n,`  
`Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double >`  
`&F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double >`  
`&F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`  
`void finit_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, size_t k, const`  
`Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::Element_ptr A)`

- template<typename RNS >  
void `fconvert_rns` (const `FFPACK::RNSInteger< RNS >` &F, const size\_t m, const size\_t n, Givaro::Integer alpha, Givaro::Integer \*B, const size\_t ldb, typename `FFPACK::RNSInteger< RNS >::ConstElement_ptr` A)
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{freduce } x \leftarrow x \bmod F.$$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{freduce } x \leftarrow y \bmod F.$$
- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{finit } x \leftarrow y \bmod F.$$
- template `INST_OR_DECL` void `fconvert` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)  
$$\text{fconvert } x \leftarrow y \bmod F.$$
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{fnegin } x \leftarrow -x.$$
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{fneg } x \leftarrow -y.$$
- template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{fzero} : A \leftarrow 0.$$
- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, const `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{fiszero} : \text{test } X = 0.$$
- template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)  
$$\text{fequal} : \text{test } X = Y.$$
- template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t N, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{fassign} : x \leftarrow y.$$
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*X, const size\_t incX)  
$$\text{fscaln } x \leftarrow \alpha \cdot x.$$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  
$$\text{fscal } y \leftarrow \alpha \cdot x.$$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  
$$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$$
- template `INST_OR_DECL` `FFLAS_ELT` `fdot` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t N, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)  
$$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- template `INST_OR_DECL` void `fswap` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t N, `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  
$$\text{fswap} : X \leftrightarrow Y.$$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)

- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $fassign : A \leftarrow B.$
- template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $fzero : A \leftarrow 0.$
- template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb)  
 $fequal : test A = B.$
- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*A, const size\_t lda)  
 $fiszero : test A = 0.$
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` &d)  
*creates a diagonal matrix*
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
*creates a diagonal matrix*
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $freduce A \leftarrow A \bmod F.$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $freduce A \leftarrow B \bmod F.$
- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $finit A \leftarrow B \bmod F.$
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $fnegin A \leftarrow -A.$
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $fneg A \leftarrow -B.$
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*A, const size\_t lda)  
 $fscaln A \leftarrow a \cdot A.$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)  
 $fscal B \leftarrow a \cdot A.$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t ldx, `FFLAS_ELT` \*Y, const size\_t ldy)  
 $faxpy : y \leftarrow \alpha \cdot x + y.$
- template `INST_OR_DECL` void `fmove` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)  
 $faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)

*fadd* : matrix addition.

- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)

*fsub* : matrix subtraction.

- template `INST_OR_DECL` void `fsubin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)

*fsubin*  $C = C - B$

- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)

*fadd* : matrix addition with scaling.

- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)

*faddin*

- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` TransA, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` beta, `FFLAS_ELT` \*Y, const size\_t incY)

*finite prime* `FFLAS_FIELD`<`FFLAS_ELT`> *GE*neral *M*atrix *V*ector multiplication.

- template `INST_OR_DECL` void `fger` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*x, const size\_t incx, const `FFLAS_ELT` \*y, const size\_t incy, `FFLAS_ELT` \*A, const size\_t lda)

*fger*: rank one update of a general matrix

- template `INST_OR_DECL` void `ftsv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, int incX)

*ftsv*: *TRI*angular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

- template `INST_OR_DECL` void `ftsm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_SIDE` Side, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)

*ftsm*: **TRI**angular System solve with **M**atrix.

- template `INST_OR_DECL` void `ftmm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_SIDE` Side, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)

*ftmm*: **TRI**angular **M**atrix **M**ultiply.

- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size\_t m, const size\_t n, const size\_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const size\_t ldc)

*fgemm*: *Field* **GE**neral **M**atrix **M**ultiply.

- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size\_t m, const size\_t n, const size\_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const size\_t ldc, const `ParSeqHelper::Sequential` seq)
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size\_t m, const size\_t n, const size\_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const size\_t ldc, const `ParSeqHelper::Parallel`< `CuttingStrategy::Recursive`, `StrategyParameter::TwoDAdaptive` > par)
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size\_t m, const size\_t n, const size\_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, const

- `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Parallel`< `CuttingStrategy::Block`, `StrategyParameter::Threads` > par)
- template `INST_OR_DECL` `FFLAS_ELT` \* `fsquare` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc)
- fsquare: Squares a matrix.*
- template<class Cut = `CuttingStrategy::Block`, class Strat = `StrategyParameter::Threads`>  
void `BlockCuts` (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` numthreads)
  - template<> void `BlockCuts`< `CuttingStrategy::Single`, `StrategyParameter::Threads` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` numthreads)
  - template<> void `BlockCuts`< `CuttingStrategy::Row`, `StrategyParameter::Fixed` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` numthreads)
  - template<> void `BlockCuts`< `CuttingStrategy::Row`, `StrategyParameter::Grain` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` grainsize)
  - template<> void `BlockCuts`< `CuttingStrategy::Block`, `StrategyParameter::Grain` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` grainsize)
  - template<> void `BlockCuts`< `CuttingStrategy::Column`, `StrategyParameter::Fixed` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` numthreads)
  - template<> void `BlockCuts`< `CuttingStrategy::Column`, `StrategyParameter::Grain` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` grainsize)
  - template<> void `BlockCuts`< `CuttingStrategy::Block`, `StrategyParameter::Fixed` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` numthreads)
  - template<> void `BlockCuts`< `CuttingStrategy::Row`, `StrategyParameter::Threads` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` numthreads)
  - template<> void `BlockCuts`< `CuttingStrategy::Column`, `StrategyParameter::Threads` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` numthreads)
  - template<> void `BlockCuts`< `CuttingStrategy::Block`, `StrategyParameter::Threads` > (`size_t` &`RBLOCKSIZE`, `size_t` &`CBLOCKSIZE`, const `size_t` m, const `size_t` n, const `size_t` numthreads)
  - template<class Cut = `CuttingStrategy::Block`, class Param = `StrategyParameter::Threads`>  
void `BlockCuts` (`size_t` &rowBlockSize, `size_t` &colBlockSize, `size_t` &lastRBS, `size_t` &lastCBS, `size_t` &changeRBS, `size_t` &changeCBS, `size_t` &numRowBlock, `size_t` &numColBlock, `size_t` m, `size_t` n, const `size_t` numthreads)
  - template<class Field >  
void `pfzero` (const `Field` &F, `size_t` m, `size_t` n, typename `Field::Element_ptr` C, `size_t` BS=0)
  - template<class Field , class RandIter >  
void `prand` (const `Field` &F, RandIter &G, `size_t` m, `size_t` n, typename `Field::Element_ptr` C, `size_t` BS=0)
  - template<class Field , class Cut , class Param >  
`Field::Element` & `fdot` (const `Field` &F, const `size_t` N, typename `Field::ConstElement_ptr` x, const `size_t` incx, typename `Field::ConstElement_ptr` y, const `size_t` incy, typename `Field::Element` &d, const `ParSeqHelper::Parallel`< Cut, Param > par)
  - template<class Field , class AlgoT , class FieldTrait >  
`Field::Element` \* `pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, const typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element` \*C, const `size_t` ldc, `MMHelper`< `Field`, AlgoT, FieldTrait, `ParSeqHelper::Parallel`< `CuttingStrategy::Block`, `StrategyParameter::Threads` > > &H)
  - template<class Field , class AlgoT , class FieldTrait >  
`Field::Element` \* `pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, const typename `Field::ConstElement_ptr` AA, const `size_t` lda, const typename `Field::ConstElement_ptr` BB, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element` \*C, const `size_t` ldc, `MMHelper`< `Field`, AlgoT, FieldTrait, `ParSeqHelper::Parallel`< `CuttingStrategy::Recursive`, `StrategyParameter::ThreeDAdaptive` > > &H)
  - template<class Field , class AlgoT , class FieldTrait >  
`Field::Element` \* `pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, const typename



- `Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoD > > &H)`
  - `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)`
  - `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace > > &H)`
  - `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)`
  - `template<class Field, class AlgoT, class FieldTrait, class Cut >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row, Cut > > &H)`
  - `void parseArguments (int argc, char **argv, Argument *args, bool printDefaults=true)`
  - `std::ostream & writeCommandString (std::ostream &os, Argument *args, const char *programName=nullptr)`  
*writes the values of all arguments, preceded by the programName*
  - `template<class Field >`  
`std::ostream & WriteMatrix (std::ostream &c, const Field &F, size_t m, size_t n, typename Field::ConstElement_ptr A, size_t lda, FFLAS_FORMAT format, bool column_major)`  
*WriteMatrix: write a matrix to an output stream.*
  - `void preamble (std::ifstream &if, FFLAS_FORMAT &format)`
  - `template<class Field >`  
`Field::Element_ptr ReadMatrix (std::ifstream &if, Field &F, size_t &m, size_t &n, typename Field::Element_ptr &A, FFLAS_FORMAT format=FflasAuto)`  
*ReadMatrix: read a matrix from an input stream.*
  - `template<class Field >`  
`Field::Element_ptr ReadMatrix (const std::string &matrix_file, Field &F, size_t &m, size_t &n, typename Field::Element_ptr &A, FFLAS_FORMAT format=FflasAuto)`  
*ReadMatrix: read a matrix from a file.*
  - `template<class Field >`  
`void WriteMatrix (std::string &matrix_file, const Field &F, int m, int n, typename Field::ConstElement_ptr A, size_t lda, FFLAS_FORMAT format=FflasDense, bool column_major=false)`  
*WriteMatrix: write a matrix to a file.*
  - `std::ostream & WritePermutation (std::ostream &c, const size_t *P, size_t N)`

*WritePermutation: write a permutation matrix to an output stream.*

- `template<class Element >`  
`bool alignable ()`
- `template<> bool alignable< Givaro::Integer * > ()`
- `template<class Field >`  
`Field::Element_ptr fflas_new (const Field &F, const size_t m, const Alignment align=Alignment::DEFAULT)`
- `template<class Field >`  
`Field::Element_ptr fflas_new (const Field &F, const size_t m, const size_t n, const Alignment align=Alignment::DEFAULT)`
- `template<class Element >`  
`Element * fflas_new (const size_t m, const Alignment align=Alignment::DEFAULT)`
- `template<class Ptr, class ... Args>`  
`void fflas_delete (Ptr p, Args ... args)`
- `void prefetch (const int64_t *)`
- `void getTLBSize (int &tlb)`
- `void queryCacheSizes (int &l1, int &l2, int &l3)`
- `int queryL1CacheSize ()`
- `int queryTopLevelCacheSize ()`
- `uint64_t getSeed ()`

## 15.1.1 Typedef Documentation

### 15.1.1.1 Checker\_fgemm

```
using Checker_fgemm = FFLAS::Checker_Empty<Field>
```

### 15.1.1.2 Checker\_ftrsm

```
using Checker_ftrsm = FFLAS::Checker_Empty<Field>
```

### 15.1.1.3 ForceCheck\_fgemm

```
using ForceCheck_fgemm = CheckerImplem_fgemm<Field>
```

### 15.1.1.4 ForceCheck\_ftrsm

```
using ForceCheck_ftrsm = CheckerImplem_ftrsm<Field>
```



#### 15.1.1.5 ZOSparseMatrix

```
using ZOSparseMatrix = std::true_type
```

#### 15.1.1.6 NotZOSparseMatrix

```
using NotZOSparseMatrix = std::false_type
```

#### 15.1.1.7 SimdSparseMatrix

```
using SimdSparseMatrix = std::true_type
```

#### 15.1.1.8 NoSimdSparseMatrix

```
using NoSimdSparseMatrix = std::false_type
```

#### 15.1.1.9 MKLSparseMatrixFormat

```
using MKLSparseMatrixFormat = std::true_type
```

#### 15.1.1.10 NotMKLSparseMatrixFormat

```
using NotMKLSparseMatrixFormat = std::false_type
```

#### 15.1.1.11 has\_plus

```
using has_plus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_plus_impl<T> >::type
```

#### 15.1.1.12 has\_minus

```
using has_minus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_minus_impl<T> >::type
```

#### 15.1.1.13 has\_equal

```
using has_equal = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
std::is_copy_assignable<T> >::type
```

#### 15.1.1.14 has\_plus\_eq

```
using has_plus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_plus_eq_impl<T> >::type
```

#### 15.1.1.15 has\_minus\_eq

```
using has_minus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_minus_eq_impl<T> >::type
```

#### 15.1.1.16 has\_mul

```
using has_mul = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_impl<T>  
>::type
```

#### 15.1.1.17 has\_mul\_eq

```
using has_mul_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_mul_eq_impl<T> >::type
```

#### 15.1.1.18 Timer

```
typedef Givaro::Timer Timer
```

#### 15.1.1.19 BaseTimer

```
typedef Givaro::BaseTimer BaseTimer
```

#### 15.1.1.20 UserTimer

```
typedef Givaro::UserTimer UserTimer
```

#### 15.1.1.21 SysTimer

```
typedef Givaro::SysTimer SysTimer
```

### 15.1.2 Enumeration Type Documentation

#### 15.1.2.1 FFLAS\_ORDER

```
enum FFLAS_ORDER
```

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major

#### 15.1.2.2 FFLAS\_TRANSPOSE

```
enum FFLAS_TRANSPOSE
```

Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.

### 15.1.2.3 FFLAS\_UPLO

enum [FFLAS\\_UPLO](#)

Is triangular matrix's shape upper ?

Enumerator

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ )
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ )

### 15.1.2.4 FFLAS\_DIAG

enum [FFLAS\\_DIAG](#)

Is the triangular matrix implicitly unit diagonal ?

Enumerator

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ )

### 15.1.2.5 FFLAS\_SIDE

enum [FFLAS\\_SIDE](#)

On what side ?

Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the righth.

### 15.1.2.6 FFLAS\_BASE

enum [FFLAS\\_BASE](#)

FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.

(deprecated, should not be used)

## Enumerator

FflasDouble	to use the double precision BLAS
FflasFloat	to use the single precison BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

## 15.1.2.7 number\_kind

```
enum number_kind
```

## Enumerator

zero	
one	
mone	
other	

## 15.1.2.8 SparseMatrix\_t

```
enum SparseMatrix_t [strong]
```

## Enumerator

CSR	
CSR_ZO	
CSC	
CSC_ZO	
COO	
COO_ZO	
ELL	
ELL_ZO	
SELL	
SELL_ZO	
ELL_simd	
ELL_simd_ZO	
CSR_HYB	
HYB_ZO	

## 15.1.2.9 FFLAS\_FORMAT

```
enum FFLAS_FORMAT
```

## Enumerator

FflasAuto	
FflasDense	
FflasSMS	
FflasBinary	
FflasMath	
FflasMaple	
FflasSageMath	

### 15.1.3 Function Documentation

#### 15.1.3.1 InfNorm()

```
Givaro::Integer FFLAS::InfNorm (
    const size_t M,
    const size_t N,
    const Givaro::Integer * A,
    const size_t lda ) [inline]
```

#### 15.1.3.2 min3()

```
const T& FFLAS::min3 (
    const T & m,
    const T & n,
    const T & k )
```

#### 15.1.3.3 max3()

```
const T& FFLAS::max3 (
    const T & m,
    const T & n,
    const T & k )
```

#### 15.1.3.4 min4()

```
const T& FFLAS::min4 (
    const T & m,
    const T & n,
    const T & k,
    const T & l )
```

### 15.1.3.5 max4()

```
const T& FFLAS::max4 (
    const T & m,
    const T & n,
    const T & k,
    const T & l )
```

### 15.1.3.6 fadd() [1/8]

```
void FFLAS::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

### 15.1.3.7 faddin() [1/4]

```
void FFLAS::faddin (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

### 15.1.3.8 fsub() [1/4]

```
void FFLAS::fsub (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**15.1.3.9 fsubin()** [1/3]

```
void FFLAS::fsubin (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**15.1.3.10 fadd()** [2/8]

```
void FFLAS::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**Todo** optimise here

**15.1.3.11 pfadd()**

```
void FFLAS::pfadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t numths )
```



### 15.1.3.12 pfsub()

```
void FFLAS::pfsub (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t numths )
```

### 15.1.3.13 pfaddin()

```
void FFLAS::pfaddin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numths )
```

### 15.1.3.14 pfsubin()

```
void FFLAS::pfsubin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numths )
```

### 15.1.3.15 fadd() [3/8]

```
void FFLAS::fadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
```

```
const size_t ldb,  
typename Field::Element_ptr C,  
const size_t ldc )
```

fadd : matrix addition.

Computes  $C = A + B$ .

## Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size $M \times N$
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size $M \times N$
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size $M \times N$
<i>ldc</i>	leading dimension of C

**15.1.3.16 fsub()** [2/4]

```
void FFLAS::fsub (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )
```

fsub : matrix subtraction.

Computes  $C = A - B$ .

## Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size $M \times N$
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size $M \times N$
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size $M \times N$
<i>ldc</i>	leading dimension of C

**15.1.3.17 faddin()** [2/4]

```
void FFLAS::faddin (
    const Field & F,
```

```

    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

faddin

### 15.1.3.18 fsubin() [2/3]

```

void FFLAS::fsubin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fsubin  $C = C - B$

### 15.1.3.19 fadd() [4/8]

```

void FFLAS::fadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fadd : matrix addition with scaling.

Computes  $C = A + \text{alpha } B$ .

#### Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>alpha</i>	some scalar
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

**15.1.3.20 fassign()** [1/10]

```
void FFLAS::fassign (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX ) [inline]
```

$\text{fassign} : x \leftarrow y.$

X is preallocated

**Todo** variant for triangular matrix

**Parameters**

	$F$	field
	$N$	size of the vectors
out	$X$	vector in F
	$incX$	stride of X
in	$Y$	vector in F
	$incY$	stride of Y

**15.1.3.21 fassign()** [2/10]

```
void FFLAS::fassign (
    const Givaro::Modular< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

**15.1.3.22 fassign()** [3/10]

```
void FFLAS::fassign (
    const Givaro::ModularBalanced< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

**15.1.3.23 fassign()** [4/10]

```
void FFLAS::fassign (
    const Givaro::ZRing< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

**15.1.3.24 fassign()** [5/10]

```
void FFLAS::fassign (
    const Givaro::Modular< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.25 fassign()** [6/10]

```
void FFLAS::fassign (
    const Givaro::ModularBalanced< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.26 fassign()** [7/10]

```
void FFLAS::fassign (
    const Givaro::ZRing< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.27 fassign()** [8/10]

```
void FFLAS::fassign (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

fassign :  $A \leftarrow B$ .

**Parameters**

$F$	field
$m$	number of rows to copy
$n$	number of cols to copy
$A$	matrix in $F$
$lda$	stride of $A$
$B$	vector in $F$
$ldb$	stride of $B$

**15.1.3.28 faxpy()** [1/6]

```
void FFLAS::faxpy (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

faxpy :  $y \leftarrow \alpha \cdot x + y$ .

**Parameters**

	$F$	field
	$N$	size of the vectors
	$alpha$	scalar
in	$X$	vector in $F$
	$incX$	stride of $X$
in, out	$Y$	vector in $F$
	$incY$	stride of $Y$

**15.1.3.29 faxpy()** [2/6]

```
void FFLAS::faxpy (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.30 faxpy()** [3/6]

```
void FFLAS::faxpy (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.31 faxpy()** [4/6]

```
void FFLAS::faxpy (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t ldx,
    typename Field::Element_ptr Y,
    const size_t ldy ) [inline]
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

**Parameters**

	$F$	field
	$m$	row dimension
	$n$	column dimension
	$\alpha$	scalar
in	$X$	vector in $F$
	$ldx$	leading dimension of $X$
in, out	$Y$	vector in $F$
	$ldy$	leading dimension of $Y$



**15.1.3.32 fdot()** [1/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

**15.1.3.33 fdot()** [2/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DelayedTag & MT ) [inline]
```

**15.1.3.34 fdot()** [3/11]

```
Givaro::DoubleDomain::Element FFLAS::fdot (
    const Givaro::DoubleDomain & ,
    const size_t N,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

**15.1.3.35 fdot()** [4/11]

```
Givaro::FloatDomain::Element FFLAS::fdot (
    const Givaro::FloatDomain & ,
    const size_t N,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

**15.1.3.36 fdot()** [5/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::ConvertTo< T > & MT ) [inline]
```

**15.1.3.37 fdot()** [6/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultBoundedTag & dbt ) [inline]
```

**15.1.3.38 fdot()** [7/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    const ParSeqHelper::Sequential seq ) [inline]
```

**15.1.3.39 fdot()** [8/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY ) [inline]
```

fdot: dot product  $x^T y$ .

## Parameters

$F$	field
$N$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

## 15.1.3.40 fgemm() [1/23]

```
FFPACK::RNSInteger<RNS>::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > & H ) [inline]
```

## 15.1.3.41 fgemm() [2/23]

```
FFPACK::RNSInteger<RNS>::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Sequential > & H ) [inline]
```

**15.1.3.42 fgemm()** [3/23]

```
FFPACK::RNSInteger<RNS>::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads
    >, ParSeqTrait > > & H ) [inline]
```

**15.1.3.43 fgemm()** [4/23]

```
FFPACK::RNSInteger<RNS>::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Parallel< Cut, Param > > & H ) [inline]
```

**15.1.3.44 fgemm()** [5/23]

```
Givaro::Integer* FFLAS::fgemm (
    const Givaro::ZRing< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
```

```

    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]

```

#### 15.1.3.45 fgemm() [6/23]

```

RNS::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename RNS::Element alpha,
    typename RNS::ConstElement_ptr Ad,
    const size_t lda,
    typename RNS::ConstElement_ptr Bd,
    const size_t ldb,
    const typename RNS::Element beta,
    typename RNS::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential
> & H ) [inline]

```

#### 15.1.3.46 fgemm() [7/23]

```

RNS::Element_ptr FFLAS::fgemm (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename RNS::Element alpha,
    typename RNS::ConstElement_ptr Ad,
    const size_t lda,
    typename RNS::ConstElement_ptr Bd,
    const size_t ldb,
    const typename RNS::Element beta,
    typename RNS::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > & H ) [inline]

```

**15.1.3.47 fgemm()** [8/23]

```
Givaro::Integer* FFLAS::fgemm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    const Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]
```

**15.1.3.48 fgemm()** [9/23]

```
Givaro::Integer* FFLAS::fgemm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    const Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.49 fgemm()** [10/23]

```
RecInt::ruint<K1>* FFLAS::fgemm (
    const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const RecInt::ruint< K1 > alpha,
```

```

    const RecInt::ruint< K1 > * A,
    const size_t lda,
    const RecInt::ruint< K1 > * B,
    const size_t ldb,
    RecInt::ruint< K1 > beta,
    RecInt::ruint< K1 > * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
    ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]

```

#### 15.1.3.50 fgemm() [11/23]

```

Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeT > & H ) [inline]

```

#### 15.1.3.51 fgemm() [12/23]

```

Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut,
    Param > > & H ) [inline]

```

**15.1.3.52 fgemm()** [13/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFl
>, ParSeqHelper::Sequential > & H ) [inline]
```

**15.1.3.53 fgemm()** [14/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Sequential seq ) [inline]
```

**15.1.3.54 fgemm()** [15/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
```



```

const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

### 15.1.3.55 fgemm() [16/23]

```

Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]

```

fgemm: Field **G**eneral **M**atrix **M**ultiply.

Computes  $C = \alpha \text{op}(A) \times \text{op}(B) + \beta C$  Automatically set Winograd recursion level

#### Parameters

<i>F</i>	field.
<i>ta</i>	if $ta == \text{FflasTrans}$ then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ ,
<i>tb</i>	same for matrix B
<i>m</i>	see A
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	$\text{op}(A)$ is $m \times k$
<i>B</i>	$\text{op}(B)$ is $k \times n$
<i>C</i>	$C$ is $m \times n$
<i>lda</i>	leading dimension of A
<i>ldb</i>	leading dimension of B
<i>ldc</i>	leading dimension of C
<i>w</i>	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of $w$ .

## Warning

$\alpha$  must be invertible

## 15.1.3.56 fgemm() [17/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > & H ) [inline]
```

## 15.1.3.57 fgemm() [18/23]

```
Field::Element_ptr FFLAS::fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential
> & H ) [inline]
```

**15.1.3.58 fsquare()** [1/6]

```
Field::Element_ptr FFLAS::fsquare (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsquare: Squares a matrix.

compute  $C \leftarrow \alpha \text{op}(A) \text{op}(A) + \beta C$  over a Field  $F$  Avoid the conversion of  $B$

**Parameters**

<i>ta</i>	if $ta == \text{FflasTrans}$ , $\text{op}(A) = A^T$ .
<i>F</i>	field
<i>n</i>	size of $A$
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	dense matrix of size $n \times n$
<i>lda</i>	leading dimension of $A$
<i>C</i>	dense matrix of size $n \times n$
<i>ldc</i>	leading dimension of $C$

**Bug** why double ?

**15.1.3.59 fsquare()** [2/6]

```
double* FFLAS::fsquare (
    const Givaro::ModularBalanced< double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double beta,
    double * C,
    const size_t ldc ) [inline]
```

**15.1.3.60 fsquare()** [3/6]

```
float* FFLAS::fsquare (
    const Givaro::ModularBalanced< float > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const float alpha,
    const float * A,
    const size_t lda,
    const float beta,
    float * C,
    const size_t ldc ) [inline]
```

**15.1.3.61 fsquare()** [4/6]

```
double* FFLAS::fsquare (
    const Givaro::Modular< double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double beta,
    double * C,
    const size_t ldc ) [inline]
```

**15.1.3.62 fsquare()** [5/6]

```
float* FFLAS::fsquare (
    const Givaro::Modular< float > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const float alpha,
    const float * A,
    const size_t lda,
    const float beta,
    float * C,
    const size_t ldc ) [inline]
```

**15.1.3.63 fgemv()** [1/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
```

```

    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H ) [inline]

```

#### 15.1.3.64 fgemv() [2/19]

```

Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]

```

#### 15.1.3.65 fgemv() [3/19]

```

Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]

```

**15.1.3.66 fgemv()** [4/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]
```

**15.1.3.67 fgemv()** [5/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE TransA,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

finite prime Field GEneral Matrix Vector multiplication.

Computes  $Y \leftarrow \alpha \text{op}(A)X + \beta Y$ .

**Parameters**

	$F$	field
	$TransA$	if $TransA == FflasTrans$ then $\text{op}(A) = A^t$ .
	$M$	rows
	$N$	cols
	$alpha$	scalar
	$A$	dense matrix of size $M \times N$
	$lda$	leading dimension of $A$
	$X$	dense vector of size $N$
	$incX$	stride of $X$
	$beta$	scalar
out	$Y$	dense vector of size $M$
	$incY$	stride of $Y$

**15.1.3.68 fgemv()** [6/19]

```
Givaro::ZRing<int64_t>::Element_ptr FFLAS::fgemv (
    const Givaro::ZRing< int64_t > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const int64_t alpha,
    const int64_t * A,
    const size_t lda,
    const int64_t * X,
    const size_t incX,
    const int64_t beta,
    int64_t * Y,
    const size_t incY,
    MMHelper< Givaro::ZRing< int64_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

**15.1.3.69 fgemv()** [7/19]

```
Givaro::DoubleDomain::Element_ptr FFLAS::fgemv (
    const Givaro::DoubleDomain & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const Givaro::DoubleDomain::Element alpha,
    const Givaro::DoubleDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::DoubleDomain::ConstElement_ptr X,
    const size_t incX,
    const Givaro::DoubleDomain::Element beta,
    Givaro::DoubleDomain::Element_ptr Y,
    const size_t incY,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

**15.1.3.70 fgemv()** [8/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
```

```

    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]

```

### 15.1.3.71 fgemv() [9/19]

```

Givaro::FloatDomain::Element_ptr FFLAS::fgemv (
    const Givaro::FloatDomain & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const Givaro::FloatDomain::Element alpha,
    const Givaro::FloatDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::FloatDomain::ConstElement_ptr X,
    const size_t incX,
    const Givaro::FloatDomain::Element beta,
    Givaro::FloatDomain::Element_ptr Y,
    const size_t incY,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

### 15.1.3.72 fgemv() [10/19]

```

Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    ParSeqHelper::Parallel< Cut, Param > & parH )

```



**15.1.3.73 fgemv()** [11/19]

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    ParSeqHelper::Sequential & seqH )
```

**15.1.3.74 fgemv()** [12/19]

```
FFPACK::rns_double::Element_ptr FFLAS::fgemv (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::ConstElement_ptr X,
    const size_t incX,
    const FFPACK::rns_double::Element beta,
    FFPACK::rns_double::Element_ptr Y,
    const size_t incY,
    MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]
```

**15.1.3.75 fgemv()** [13/19]

```
FFPACK::rns_double::Element_ptr FFLAS::fgemv (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::ConstElement_ptr X,
    const size_t incX,
    const FFPACK::rns_double::Element beta,
    FFPACK::rns_double::Element_ptr Y,
    const size_t incY,
    MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]
```

**15.1.3.76 fgemv()** [14/19]

```
Givaro::Integer* FFLAS::fgemv (
    const Givaro::ZRing< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const Givaro::Integer alpha,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * X,
    const size_t ldx,
    Givaro::Integer beta,
    Givaro::Integer * Y,
    const size_t ldy,
    MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]
```

**15.1.3.77 fgemv()** [15/19]

```
Givaro::Integer* FFLAS::fgemv (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const Givaro::Integer alpha,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * X,
    const size_t ldx,
    Givaro::Integer beta,
    Givaro::Integer * Y,
    const size_t ldy,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]
```

**15.1.3.78 fgemv()** [16/19]

```
RecInt::ruint<K1>* FFLAS::fgemv (
    const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const RecInt::ruint< K1 > alpha,
    const RecInt::ruint< K1 > * A,
    const size_t lda,
    const RecInt::ruint< K1 > * X,
    const size_t incx,
    RecInt::ruint< K1 > beta,
    RecInt::ruint< K1 > * Y,
    const size_t incy,
    MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.79 fger()** [1/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]

```

fger: rank one update of a general matrix

Computes  $A \leftarrow \alpha x y^T + A$

**Parameters**

	$F$	field
	$M$	rows
	$N$	cols
	$\alpha$	scalar
in, out	$A$	dense matrix of size M×N and leading dimension lda
	$lda$	leading dimension of A
	$x$	dense vector of size M
	$incx$	stride of X
	$y$	dense vector of size N
	$incy$	stride of Y

**15.1.3.80 fger()** [2/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H ) [inline]

```

**15.1.3.81 fger()** [3/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, AnyTag > & H ) [inline]

```

**15.1.3.82 fger()** [4/12]

```

void FFLAS::fger (
    const Givaro::DoubleDomain & F,
    const size_t M,
    const size_t N,
    const Givaro::DoubleDomain::Element alpha,
    const Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    const Givaro::DoubleDomain::ConstElement_ptr y,
    const size_t incy,
    Givaro::DoubleDomain::Element_ptr A,
    const size_t lda,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.83 fger()** [5/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr x,
    const size_t incx,
    const typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]

```

**15.1.3.84 fger()** [6/12]

```

void FFLAS::fger (
    const Givaro::FloatDomain & F,
    const size_t M,
    const size_t N,
    const Givaro::FloatDomain::Element alpha,
    const Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    const Givaro::FloatDomain::ConstElement_ptr y,
    const size_t incy,
    Givaro::FloatDomain::Element_ptr A,
    const size_t lda,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.85 fger()** [7/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]

```

**15.1.3.86 fger()** [8/12]

```

void FFLAS::fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]

```

**15.1.3.87 fger()** [9/12]

```

void FFLAS::fger (
    const Givaro::Modular< Givaro::Integer > & F,
    const size_t M,
    const size_t N,
    const typename Givaro::Integer alpha,
    typename Givaro::Integer * x,
    const size_t incx,
    typename Givaro::Integer * y,
    const size_t incy,
    typename Givaro::Integer * A,
    const size_t lda,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]

```

**15.1.3.88 fger()** [10/12]

```

void FFLAS::fger (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t M,
    const size_t N,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::Element_ptr x,
    const size_t incx,
    typename FFPACK::RNSInteger< RNS >::Element_ptr y,
    const size_t incy,
    typename FFPACK::RNSInteger< RNS >::Element_ptr A,
    const size_t lda,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.89 fger()** [11/12]

```

void FFLAS::fger (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t M,
    const size_t N,
    const typename FFPACK::RNSIntegerMod< RNS >::Element alpha,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr x,
    const size_t incx,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y,
    const size_t incy,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A,
    const size_t lda,
    MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > & H ) [inline]

```

**15.1.3.90 freduce()** [1/10]

```
void FFLAS::freduce (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )
```

freduce  $x \leftarrow y \bmod F$ .

**Parameters**

$F$	field
$n$	size of the vectors
$Y$	vector of Element
$incY$	stride of Y
$X$	vector in F
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

**15.1.3.91 freduce()** [2/10]

```
void FFLAS::freduce (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

freduce  $x \leftarrow x \bmod F$ .

**Parameters**

$F$	field
$n$	size of the vectors
$X$	vector in F
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

**15.1.3.92 freduce\_constoverride()** [1/2]

```
void FFLAS::freduce_constoverride (
    const Field & F,
```

```

const size_t m,
typename Field::ConstElement_ptr A,
const size_t incX )

```

### 15.1.3.93 finit() [1/8]

```

void FFLAS::finit (
    const Field & F,
    const size_t n,
    ConstOtherElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )

```

### 15.1.3.94 finit() [2/8]

```

void FFLAS::finit (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )

```

finit Initializes X in F\$.

#### Parameters

$F$	field
$n$	size of the vectors
$X$	vector in F
$incX$	stride of X

### 15.1.3.95 freduce() [3/10]

```

void FFLAS::freduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

freduce  $A \leftarrow A \bmod F$ .

#### Parameters

$F$	field
-----	-------



## Parameters

<i>m</i>	number of rows
<i>n</i>	number of cols
<i>A</i>	matrix in F
<i>lda</i>	stride of A

## 15.1.3.96 pfreduce()

```
void FFLAS::pfreduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t numths )
```

## 15.1.3.97 freduce() [4/10]

```
void FFLAS::freduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

freduce  $A \leftarrow B \bmod F$ .

## Parameters

<i>F</i>	field
<i>m</i>	number of rows
<i>n</i>	number of cols
<i>A</i>	matrix in F
<i>lda</i>	stride of A
<i>B</i>	matrix in Element
<i>ldb</i>	stride of B

## 15.1.3.98 freduce\_constoverride() [2/2]

```
void FFLAS::freduce_constoverride (
    const Field & F,
```

```

    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda )

```

### 15.1.3.99 finit() [3/8]

```

void FFLAS::finit (
    const Field & F,
    const size_t m,
    const size_t n,
    const OtherElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{finit } A \leftarrow B \bmod F$ .

#### Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$
$B$	matrix in $\text{OtherElement}$
$ldb$	stride of $B$

### 15.1.3.100 finit() [4/8]

```

void FFLAS::finit (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{finit}$  Initializes  $A$  in  $F$ .

#### Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.101 freduce()** [5/10]

```
void FFLAS::freduce (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
    size_t inc ) [inline]
```

**15.1.3.102 freduce()** [6/10]

```
void FFLAS::freduce (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    FFPACK::rns_double::Element_ptr A,
    size_t lda ) [inline]
```

**15.1.3.103 freivalds()**

```
bool FFLAS::freivalds (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::ConstElement_ptr C,
    const size_t ldc ) [inline]
```

freivalds: **F**reivalds **G**eneral **M**atrix **M**ultiply **R**andom **C**heck.

Randomly Checks  $C = \alpha \text{op}(A) \times \text{op}(B)$

**Parameters**

<i>F</i>	field.
<i>ta</i>	if ta==FflasTrans then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ ,
<i>tb</i>	same for matrix B
<i>m</i>	see A
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar

## Parameters

$A$	$\text{op}(A)$ is $m \times k$
$B$	$\text{op}(B)$ is $k \times n$
$C$	$C$ is $m \times n$
$lda$	leading dimension of $A$
$ldb$	leading dimension of $B$
$ldc$	leading dimension of $C$

**15.1.3.104 fscaln()** [1/10]

```
void FFLAS::fscaln (
    const Field & F,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr X,
    const size_t incX ) [inline]
```

$\text{fscaln } x \leftarrow \alpha \cdot x.$

## Parameters

$F$	field
$n$	size of the vectors
$alpha$	scalar
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use `cblas_(d)scal` when possible

**Todo** check if comparison with  $\pm 1, 0$  is necessary.

**15.1.3.105 fscal()** [1/10]

```
void FFLAS::fscal (
    const Field & F,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

$\text{fscal } y \leftarrow \alpha \cdot x.$

## Parameters

	$F$	field
	$n$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in $F$
	$incX$	stride of $X$
out	$Y$	vector in $F$
	$incY$	stride of $Y$

**Bug** use `cblas_(d)scal` when possible

**Todo** check if comparison with  $\pm 1, 0$  is necessary.

**15.1.3.106 fscal() [2/10]**

```
void FFLAS::fscal (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.107 fscal() [3/10]**

```
void FFLAS::fscal (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.108 fscaln() [2/10]**

```
void FFLAS::fscaln (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.109 fscaln()** [3/10]

```
void FFLAS::fscaln (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.110 fscaln()** [4/10]

```
void FFLAS::fscaln (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

$\text{fscaln } A \leftarrow a \cdot A.$

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$\alpha$	homotecie scalar
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.111 fscal()** [4/10]

```
void FFLAS::fscal (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]
```

$\text{fscal } B \leftarrow a \cdot A.$

**Parameters**

	$F$	field
	$m$	number of rows

## Parameters

	$n$	number of cols
	$\alpha$	homotecie scalar
in	$A$	matrix in F
	$lda$	stride of A
out	$B$	matrix in F
	$ldb$	stride of B

**15.1.3.112 fscaln()** [5/10]

```
void FFLAS::fscaln (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t inc ) [inline]
```

**15.1.3.113 fscal()** [5/10]

```
void FFLAS::fscal (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t Ainc,
    FFPACK::rns_double::Element_ptr B,
    const size_t Binc ) [inline]
```

**15.1.3.114 fscaln()** [6/10]

```
void FFLAS::fscaln (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t lda ) [inline]
```

**15.1.3.115 fscal()** [6/10]

```

void FFLAS::fscal (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::Element_ptr B,
    const size_t ldb ) [inline]

```

**15.1.3.116 fscaln()** [7/10]

```

void FFLAS::fscaln (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha,
    typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
    const size_t inc ) [inline]

```

**15.1.3.117 fscal()** [7/10]

```

void FFLAS::fscal (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t Ainc,
    FFPACK::rns_double::Element_ptr B,
    const size_t Binc ) [inline]

```

**15.1.3.118 fscaln()** [8/10]

```

void FFLAS::fscaln (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t lda ) [inline]

```



**15.1.3.119 fscal()** [8/10]

```

void FFLAS::fscal (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::Element_ptr B,
    const size_t ldb ) [inline]

```

**15.1.3.120 fsyr2k()**

```

Field::Element_ptr FFLAS::fsyr2k (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]

```

fsyr2k: Symmetric Rank 2K update

Computes the Lower or Upper triangular part of  $C = \alpha(A \times B^T + B \times A^T) + \beta C$  or  $C = \alpha(A^T \times B + B^T \times A) + \beta C$

**Parameters**

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if <code>ta==FflasNoTrans</code> then compute $C = \alpha(A \times B^T + B \times A^T) + \beta C$ , else $C = \alpha(A^T \times B + B^T \times A) + \beta C$
<i>n</i>	order of matrix C
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	A is $n \times k$ (FflasNoTrans) or A is $k \times n$ (FflasTrans)
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	C is $n \times n$
<i>ldc</i>	leading dimension of C

**Warning**

$\alpha$  must be invertible

**15.1.3.121 fsyrk()** [1/5]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsyrk: Symmetric Rank K update

Computes the Lower or Upper triangular part of  $C = \alpha A \times A^T + \beta C$  or  $C = \alpha A^T \times A + \beta C$

**Parameters**

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if <code>ta==FflasNoTrans</code> then compute $C = \alpha A \times A^T + \beta C$ , else $C = \alpha A^T \times A + \beta C$
<i>n</i>	order of matrix C
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	$A$ is $n \times k$ or $A$ is $k \times n$
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	$C$ is $n \times n$
<i>ldc</i>	leading dimension of C

**Warning**

$\alpha$  must be invertible

**15.1.3.122 fsyrk()** [2/5]

```
Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
```

```

const FFLAS_TRANSPOSE trans,
const size_t n,
const size_t k,
const typename Field::Element alpha,
typename Field::Element_ptr A,
const size_t lda,
typename Field::ConstElement_ptr D,
const size_t incD,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD ) [inline]

```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of  $C = \alpha A \times D \times A^T + \beta C$  or  $C = \alpha A^T \times D \times A + \beta C$  where  $D$  is a diagonal matrix. Matrix  $A$  is updated into  $D \times A$  (if  $\text{trans} = \text{FflasTrans}$ ) or  $A \times D$  (if  $\text{trans} = \text{FflasNoTrans}$ ).

#### Parameters

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix $C$
<i>trans</i>	if $\text{ta} == \text{FflasNoTrans}$ then compute $C = \alpha A \times A^T + \beta C$ , else $C = \alpha A^T \times A + \beta C$
<i>n</i>	order of matrix $C$
<i>k</i>	see $A$
<i>alpha</i>	scalar
<i>A</i>	$A$ is $n \times k$ or $A$ is $k \times n$
<i>lda</i>	leading dimension of $A$
<i>D</i>	$D$ is $k \times k$ diagonal matrix, stored as a vector of $k$ coefficients
<i>lda</i>	leading dimension of $A$
<i>beta</i>	scalar
<i>C</i>	$C$ is $n \times n$
<i>ldc</i>	leading dimension of $C$

#### Warning

$\alpha$  must be invertible

#### 15.1.3.123 fsyrk() [3/5]

```

Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,

```

```

const size_t incD,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const ParSeqHelper::Sequential seq,
const size_t threshold ) [inline]

```

#### 15.1.3.124 fsyrk() [4/5]

```

Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Parallel< Cut, Param > par,
    const size_t threshold ) [inline]

```

#### 15.1.3.125 fsyrk() [5/5]

```

Field::Element_ptr FFLAS::fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const std::vector< bool > & twoBlock,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD ) [inline]

```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of  $C = \alpha A \times \text{Delta} D \times A^T + \beta C$  or  $C = \alpha A^T \times \text{Delta} D \times A + \beta C$  where D is a diagonal matrix and Delta is a block diagonal with either 1 on the diagonal or 2x2 swap blocks Matrix A is updated into  $D \times A$  (if trans = FflasTrans) or  $A \times D$  (if trans = FflasNoTrans).

## Parameters

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if <code>ta==FflasNoTrans</code> then compute $C = \alpha A \Delta D \times A^T + \beta C$ , else $C = \alpha A^T \Delta D \times A + \beta C$
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	<i>A</i> is $n \times k$ or <i>A</i> is $k \times n$
<i>lda</i>	leading dimension of A
<i>D</i>	<i>D</i> is $k \times k$ diagonal matrix, stored as a vector of k coefficients
<i>twoBlocks</i>	a vector boolean indicating the beginning of each 2x2 blocs in Delta
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	<i>C</i> is $n \times n$
<i>ldc</i>	leading dimension of C

## Warning

$\alpha$  must be invertible

## 15.1.3.126 ftrmm() [1/3]

```
void FFLAS::ftrmm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]
```

ftrmm: **TR**iangular **M**atrix **M**ultiply.

Computes  $B \leftarrow \alpha \text{op}(A)B$  or  $B \leftarrow \alpha B \text{op}(A)$ .

## Parameters

<i>F</i>	field
<i>Side</i>	if <code>Side==FflasLeft</code> then $B \leftarrow \alpha \text{op}(A)B$ is computed.
<i>Uplo</i>	if <code>Uplo==FflasUpper</code> then A is upper triangular
<i>TransA</i>	if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .
<i>Diag</i>	if <code>Diag==FflasUnit</code> then A is implicitly unit.

## Parameters

$M$	rows of B
$N$	cols of B
$\alpha$	scalar
$A$	triangular matrix. If Side==FflasLeft then A is $N \times N$ , otherwise A is $M \times M$
$lda$	leading dim of A
$B$	matrix of size MxN
$ldb$	leading dim of B

## 15.1.3.127 ftrmm() [2/3]

```
void FFLAS::ftrmm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

ftrmm: **TR**iangular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \beta C$  or  $C \leftarrow \alpha B \text{op}(A) + \beta C$ .

## Parameters

$F$	field
$Side$	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A)B$ is computed.
$Uplo$	if Uplo==FflasUpper then A is upper triangular
$TransA$	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
$Diag$	if Diag==FflasUnit then A is implicitly unit.
$M$	rows of B
$N$	cols of B
$\alpha$	scalar
$A$	triangular matrix. If Side==FflasLeft then A is $N \times N$ , otherwise A is $M \times M$
$lda$	leading dim of A
$B$	matrix of size MxN
$ldb$	leading dim of B
$\beta$	scalar
$C$	matrix of size MxN
$ldc$	leading dim of C

**15.1.3.128 ftrsm()** [1/9]

```

void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]

```

**15.1.3.129 ftrsm()** [2/9]

```

void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const ParSeqHelper::Sequential & PSH ) [inline]

```

**15.1.3.130 ftrsm()** [3/9]

```

void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const ParSeqHelper::Parallel< Cut, Param > & PSH ) [inline]

```

**15.1.3.131 ftrsm()** [4/9]

```

void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Recursive, ParSeqTrait > & H ) [inline]

```

**15.1.3.132 ftrsm()** [5/9]

```

void FFLAS::ftrsm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * B,
    const size_t ldb ) [inline]

```

**15.1.3.133 cblas\_impstrsm()**

```

void FFLAS::cblas_impstrsm (
    const enum FFLAS_ORDER Order,
    const enum FFLAS_SIDE Side,
    const enum FFLAS_UPLO Uplo,
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_DIAG Diag,
    const int M,
    const int N,
    const FFPACK::rns_double_elt alpha,
    FFPACK::rns_double_elt_cstptr A,
    const int lda,
    FFPACK::rns_double_elt_ptr B,
    const int ldb ) [inline]

```



**15.1.3.134 ftrsv()** [1/2]

```

void FFLAS::ftrsv (
    const Field & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    int incX ) [inline]

```

ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

**Parameters**

<i>F</i>	field
<i>X</i>	vector of size N on a field F
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size N
<i>lda</i>	leading dimension of A
<i>N</i>	number of rows or columns of A according to TransA
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular

**15.1.3.135 igemm\_()**

```

void igemm_ (
    const enum FFLAS_ORDER Order,
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_TRANSPOSE TransB,
    const size_t M,
    const size_t N,
    const size_t K,
    const int64_t alpha,
    const int64_t * A,
    const size_t lda,
    const int64_t * B,
    const size_t ldb,
    const int64_t beta,
    int64_t * C,
    const size_t ldc ) [inline]

```

**15.1.3.136 finit()** [5/8]

```
void FFLAS::finit (
    const Field & F,
    const size_t n,
    const OtherElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )
```

$\text{finit } x \leftarrow y \bmod F.$

**Parameters**

$F$	field
$n$	size of the vectors
$Y$	vector of OtherElement
$incY$	stride of Y
$X$	vector in F
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

**15.1.3.137 fconvert()** [1/3]

```
void FFLAS::fconvert (
    const Field & F,
    const size_t n,
    OtherElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY )
```

$\text{fconvert } x \leftarrow y \bmod F.$

**Parameters**

$F$	field
$n$	size of the vectors
$Y$	vector of F
$incY$	stride of Y
$X$	vector in OtherElement
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

**15.1.3.138 fnegin()** [1/4]

```
void FFLAS::fnegin (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

$\text{fnegin } x \leftarrow -x.$

**Parameters**

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use `cblas_(d)scal` when possible

**15.1.3.139 fneg()** [1/4]

```
void FFLAS::fneg (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )
```

$\text{fneg } x \leftarrow -y.$

**Parameters**

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

**Bug** use `cblas_(d)scal` when possible

**15.1.3.140 fzero()** [1/4]

```
void FFLAS::fzero (
    const Field & F,
```

```

    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )

```

fzero :  $A \leftarrow 0$ .

#### Parameters

$F$	field
$n$	number of elements to zero
$X$	vector in $F$
$incX$	stride of $X$

#### 15.1.3.141 frand() [1/2]

```

void FFLAS::frand (
    const Field & F,
    RandIter & G,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )

```

frand :  $A \leftarrow \text{random}$ .

#### Parameters

$F$	field
$G$	randomiterator
$n$	number of elements to randomize
$X$	vector in $F$
$incX$	stride of $X$

#### 15.1.3.142 fiszero() [1/4]

```

bool FFLAS::fiszero (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr X,
    const size_t incX )

```

fiszero : test  $X = 0$ .

#### Parameters

$F$	field
$n$	vector dimension
$X$	vector in $F$
$incX$	increment of $X$

**15.1.3.143 fequal()** [1/4]

```
bool FFLAS::fequal (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY )
```

fequal : test  $X = Y$ .

**Parameters**

$F$	field
$n$	vector dimension
$X$	vector in $F$
$incX$	increment of $X$
$Y$	vector in $F$
$incY$	increment of $Y$

**15.1.3.144 faxpby()** [1/2]

```
void FFLAS::faxpby (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY )
```

faxpby :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .

**Parameters**

	$F$	field
	$N$	size of the vectors
	$alpha$	scalar
in	$X$	vector in $F$
	$incX$	stride of $X$
	$beta$	scalar
in, out	$Y$	vector in $F$
	$incY$	stride of $Y$

**Note**

this is a catlas function

**15.1.3.145 fdot()** [9/11]

```
Field::Element FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    const ParSeqHelper::Parallel< Cut, Param > par )
```

**15.1.3.146 fswap()** [1/2]

```
void FFLAS::fswap (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY )
```

fswap:  $X \leftrightarrow Y$ .

**Bug** use cblas\_dswap when double

**Parameters**

$F$	field
$N$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

**15.1.3.147 fzero()** [2/4]

```
void FFLAS::fzero (
    const Field & F,
    const size_t m,
```

```

    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{fzero} : A \leftarrow 0.$

#### Parameters

$F$	field
$m$	number of rows to zero
$n$	number of cols to zero
$A$	matrix in $F$
$lda$	stride of $A$

#### Warning

may be buggy if Element is larger than int

### 15.1.3.148 frand() [2/2]

```

void FFLAS::frand (
    const Field & F,
    RandIter & G,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{frand} : A \leftarrow \text{random}.$

#### Parameters

$F$	field
$G$	randomiterator
$m$	number of rows to randomize
$n$	number of cols to randomize
$A$	matrix in $F$
$lda$	stride of $A$

### 15.1.3.149 fequal() [2/4]

```

bool FFLAS::fequal (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,

```

```

const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb )

```

fequal : test  $A = B$ .

#### Parameters

$F$	field
$m$	row dimension
$n$	column dimension
$A$	m x n matrix in $F$
$lda$	leading dimension of A
$B$	m x n matrix in $F$
$ldb$	leading dimension of B

#### 15.1.3.150 fiszero() [2/4]

```

bool FFLAS::fiszero (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda )

```

fiszero : test  $A = 0$ .

#### Parameters

$F$	field
$m$	row dimension
$n$	column dimension
$A$	m x n matrix in $F$
$lda$	leading dimension of A

#### 15.1.3.151 fidentity() [1/4]

```

void FFLAS::fidentity (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element & d )

```

creates a diagonal matrix



**15.1.3.152 fidentity() [2/4]**

```
void FFLAS::fidentity (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

creates a diagonal matrix

**15.1.3.153 finit() [6/8]**

```
void FFLAS::finit (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

finit Initializes A in  $F^{\$}$ .

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of A

**15.1.3.154 fconvert() [2/3]**

```
void FFLAS::fconvert (
    const Field & F,
    const size_t m,
    const size_t n,
    OtherElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb )
```

fconvert  $A \leftarrow B \bmod F$ .

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols

## Parameters

$A$	matrix in <code>OtherElement</code>
$lda$	stride of $A$
$B$	matrix in $F$
$ldb$	stride of $B$

**Todo** check if  $n == lda$

15.1.3.155 `fnegin()` [2/4]

```
void FFLAS::fnegin (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

`fnegin`  $A \leftarrow -A$ .

## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**Todo** check if  $n == lda$

15.1.3.156 `fneg()` [2/4]

```
void FFLAS::fneg (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

`fneg`  $A \leftarrow -B$ .

## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**Todo** check if  $n == lda$

## 15.1.3.157 faxpby() [2/2]

```
void FFLAS::faxpby (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t ldx,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t ldy )
```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

## Parameters

	$F$	field
	$m$	row dimension
	$n$	column dimension
	$\alpha$	scalar
in	$X$	vector in $F$
	$ldx$	leading dimension of $X$
	$\beta$	scalar
in, out	$Y$	vector in $F$
	$ldy$	leading dimension of $Y$

## Note

this is a catlas function

## 15.1.3.158 fmove() [1/2]

```
void FFLAS::fmove (
    const Field & F,
```

```

    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb )

```

fmove :  $A \leftarrow B$  and  $B \leftarrow 0$ .

#### Parameters

<i>F</i>	field
<i>m</i>	number of rows to copy
<i>n</i>	number of cols to copy
<i>A</i>	matrix in F
<i>lda</i>	stride of A
<i>B</i>	matrix in F
<i>ldb</i>	stride of B

#### 15.1.3.159 bitsize()

```

size_t FFLAS::bitsize (
    const Field & F,
    size_t M,
    size_t N,
    const typename Field::ConstElement_ptr A,
    size_t lda ) [inline]

```

bitsize: Computes the largest bitsize of the matrix' coefficients.

If the matrix is over a modular prime field, it returns the bitsize of the largest element (in a bsolute value)

#### Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size MxN
<i>lda</i>	leading dimension of A

#### 15.1.3.160 bitsize< Givaro::ZRing< Givaro::Integer > >()

```

size_t FFLAS::bitsize< Givaro::ZRing< Givaro::Integer > > (
    const Givaro::ZRing< Givaro::Integer > & F,
    size_t M,

```

```

size_t N,
const Givaro::Integer * A,
size_t lda ) [inline]

```

### 15.1.3.161 ftrmv()

```

void FFLAS::ftrmv (
    const Field & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    int incX )

```

ftrsm: TRIangular Matrix Vector prodcut Computes  $X \leftarrow \text{op}(A)X$

#### Parameters

<i>F</i>	field
<i>X</i>	vector of size N on a field F
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size N
<i>lda</i>	leading dimension of A
<i>N</i>	number of rows and columns of A
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^T$ .
<i>Diag</i>	if Diag==FflasUnit then A is unit diagonal.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular

### 15.1.3.162 ftrsm() [6/9]

```

void FFLAS::ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb )

```

ftrsm: TRIangular System solve with Matrix.

Computes  $B \leftarrow \alpha \text{op}(A^{-1})B$  or  $B \leftarrow \alpha B \text{op}(A^{-1})$ .

## Parameters

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A^{-1})B$ is computed.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular invertible matrix. If Side==FflasLeft then A is $N \times N$ , otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B

**Bug**  $\alpha$  must be non zero.

## 15.1.3.163 pfgemm() [1/7]

```
Field::Element_ptr FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numthreads = 0 )
```

## 15.1.3.164 pfgemm\_1D\_rec()

```
Field::Element* FFLAS::pfgemm_1D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
```

```

const typename Field::Element_ptr A,
const size_t lda,
const typename Field::Element_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element * C,
const size_t ldc,
size_t seuil )

```

### 15.1.3.165 pfgemm\_2D\_rec()

```

Field::Element* FFLAS::pfgemm_2D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    size_t seuil )

```

### 15.1.3.166 pfgemm\_3D\_rec()

```

Field::Element* FFLAS::pfgemm_3D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t seuil,
    size_t * x )

```

**15.1.3.167 pfgemm\_3D\_rec2()**

```
Field::Element_ptr FFLAS::pfgemm_3D_rec2 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t seuil,
    size_t * x )
```

**15.1.3.168 fgemm() [19/23]**

```
std::enable_if<!std::is_same<ModeTrait, ModeCategories::ConvertTo<ElementCategories::RNSElementTag>
>::value, typename Field::Element_ptr>::type FFLAS::fgemm (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE ta,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,
Param > > & H ) [inline]
```

**15.1.3.169 ftrsm() [7/9]**

```
Field::Element_ptr FFLAS::ftrsm (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_TRANSPOSE TA,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t m,
```



```

    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &
H ) [inline]

```

#### 15.1.3.170 ftrsm() [8/9]

```

Field::Element_ptr FFLAS::ftrsm (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_TRANSPOSE TA,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > & H
) [inline]

```

#### 15.1.3.171 sparse\_delete() [1/12]

```

void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::COO > & A ) [inline]

```

#### 15.1.3.172 sparse\_delete() [2/12]

```

void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A ) [inline]

```

#### 15.1.3.173 sparse\_init() [1/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::COO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.174 sparse\_init()** [2/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.175 sparse\_delete()** [3/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR > & A ) [inline]
```

**15.1.3.176 sparse\_delete()** [4/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A ) [inline]
```

**15.1.3.177 sparse\_print()** [1/3]

```
std::ostream& FFLAS::sparse_print (
    std::ostream & os,
    const Sparse< Field, SparseMatrix_t::CSR > & A ) [inline]
```

**15.1.3.178 sparse\_init()** [3/16]

```
void FFLAS::sparse_init (
    const Givaro::Modular< Givaro::Integer > & F,
    Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > & A,
    const IndexT * row,
    const IndexT * col,
    Givaro::Integer * dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.179 sparse\_init()** [4/16]

```

void FFLAS::sparse_init (
    const Givaro::ZRing< Givaro::Integer > & F,
    Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    Givaro::Integer * dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.180 sparse\_init()** [5/16]

```

void FFLAS::sparse_init (
    const Givaro::ZRing< RecInt::rmint< RECINT_SIZE >> & F,
    Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE >>, SparseMatrix_t::CSR_ZO >
& A,
    const IndexT * row,
    const IndexT * col,
    typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE >>::Element_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.181 sparse\_init()** [6/16]

```

void FFLAS::sparse_init (
    const Givaro::ZRing< RecInt::rmint< RECINT_SIZE >> & F,
    Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE >>, SparseMatrix_t::CSR > &
A,
    const IndexT * row,
    const IndexT * col,
    typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE >>::Element_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.182 sparse\_init()** [7/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.183 sparse\_init()** [8/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.184 sparse\_delete()** [5/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A ) [inline]
```

**15.1.3.185 sparse\_init()** [9/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.186 sparse\_delete()** [6/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL > & A ) [inline]
```

**15.1.3.187 sparse\_delete()** [7/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A ) [inline]
```

**15.1.3.188 sparse\_init()** [10/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.189 sparse\_init()** [11/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.190 sparse\_delete()** [8/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A ) [inline]
```

**15.1.3.191 sparse\_delete()** [9/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A ) [inline]
```

**15.1.3.192 sparse\_print()** [2/3]

```
void FFLAS::sparse_print (
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A ) [inline]
```

**15.1.3.193 sparse\_init()** [12/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.194 sparse\_init()** [13/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.195 sparse\_delete()** [10/12]

```

void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A ) [inline]

```

**15.1.3.196 sparse\_init()** [14/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.197 operator<<()**

```
std::ostream& FFLAS::operator<< (
    std::ostream & os,
    const Sparse<_Field, SparseMatrix_t::HYB_ZO > & A )
```

**15.1.3.198 readSmsFormat()**

```
void FFLAS::readSmsFormat (
    const std::string & path,
    const Field & f,
    index_t *& row,
    index_t *& col,
    typename Field::Element_ptr & val,
    index_t & rowdim,
    index_t & coldim,
    uint64_t & nnz )
```

**15.1.3.199 readSprFormat()**

```
void FFLAS::readSprFormat (
    const std::string & path,
    const Field & f,
    index_t *& row,
    index_t *& col,
    typename Field::Element_ptr & val,
    index_t & rowdim,
    index_t & coldim,
    uint64_t & nnz )
```

**15.1.3.200 getDataType() [1/4]**

```
std::enable_if<std::is_integral<T>::value,int> FFLAS::getDataType ( )
```

**15.1.3.201 getDataType() [2/4]**

```
std::enable_if<std::is_floating_point<T>::value,int> FFLAS::getDataType ( )
```

**15.1.3.202** `getDataType()` [3/4]

```
std::enable_if<std::is_same<T,mpz_t>::value,int> FFLAS::getDataType ( )
```

**15.1.3.203** `getDataType()` [4/4]

```
int FFLAS::getDataType ( )
```

**15.1.3.204** `readMachineType()`

```
void FFLAS::readMachineType (
    const Field & F,
    typename Field::Element & modulo,
    typename Field::Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

**15.1.3.205** `readDnsFormat()`

```
void FFLAS::readDnsFormat (
    const std::string & path,
    const Field & F,
    index_t & rowdim,
    index_t & coldim,
    typename Field::Element_ptr & val )
```

**15.1.3.206** `writeDnsFormat()`

```
void FFLAS::writeDnsFormat (
    const std::string & path,
    const Field & F,
    const index_t & rowdim,
    const index_t & coldim,
    typename Field::Element_ptr A,
    index_t ldA )
```



**15.1.3.207 fspmv()** [1/2]

```
void FFLAS::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ) [inline]
```

**15.1.3.208 sparse\_delete()** [11/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::SELL > & A ) [inline]
```

**15.1.3.209 sparse\_delete()** [12/12]

```
void FFLAS::sparse_delete (
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A ) [inline]
```

**15.1.3.210 sparse\_print()** [3/3]

```
void FFLAS::sparse_print (
    const Sparse< Field, SparseMatrix_t::SELL > & A ) [inline]
```

**15.1.3.211 sparse\_init()** [15/16]

```
void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::SELL > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz,
    uint64_t sigma = 0 ) [inline]
```

**15.1.3.212 sparse\_init()** [16/16]

```

void FFLAS::sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.213 computeDeviation()**

```

double FFLAS::computeDeviation (
    It begin,
    It end )

```

**15.1.3.214 getStat()**

```

StatsMatrix FFLAS::getStat (
    const Field & F,
    const index_t * row,
    const index_t * col,
    typename Field::ConstElement_ptr val,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz )

```

**15.1.3.215 fspmv()** [2/2]

```

void FFLAS::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    const typename Field::Element & beta,
    typename Field::Element_ptr y ) [inline]

```

**15.1.3.216 fspmm()**

```

void FFLAS::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    const typename Field::Element & beta,
    typename Field::Element_ptr y,
    int ldy ) [inline]

```

**15.1.3.217 fflas\_delete() [1/3]**

```

void FFLAS::fflas_delete (
    FFPACK::rns_double_elt_ptr A ) [inline]

```

**15.1.3.218 fflas\_delete() [2/3]**

```

void FFLAS::fflas_delete (
    FFPACK::rns_double_elt_cstptr A ) [inline]

```

**15.1.3.219 fflas\_new() [1/7]**

```

FFPACK::rns_double_elt_ptr FFLAS::fflas_new (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const Alignment align ) [inline]

```

**15.1.3.220 fflas\_new() [2/7]**

```

FFPACK::rns_double_elt_ptr FFLAS::fflas_new (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const Alignment align ) [inline]

```

**15.1.3.221 finit\_rns()** [1/2]

```

void FFLAS::finit_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename RNS::Element_ptr A )

```

**15.1.3.222 finit\_trans\_rns()**

```

void FFLAS::finit_trans_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename RNS::Element_ptr A )

```

**15.1.3.223 fconvert\_rns()** [1/2]

```

void FFLAS::fconvert_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename RNS::ConstElement_ptr A )

```

**15.1.3.224 fconvert\_trans\_rns()**

```

void FFLAS::fconvert_trans_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename RNS::ConstElement_ptr A )

```

**15.1.3.225 fflas\_new()** [3/7]

```
FFPACK::rns_double_elt_ptr FFLAS::fflas_new (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const Alignment align ) [inline]
```

**15.1.3.226 fflas\_new()** [4/7]

```
FFPACK::rns_double_elt_ptr FFLAS::fflas_new (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const Alignment align ) [inline]
```

**15.1.3.227 finit\_rns()** [2/2]

```
void FFLAS::finit_rns (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename FFPACK::RNSInteger< RNS >::Element_ptr A )
```

**15.1.3.228 fconvert\_rns()** [2/2]

```
void FFLAS::fconvert_rns (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A )
```

**15.1.3.229 freduce()** [7/10]

```
template INST_OR_DECL void FFLAS::freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

freduce  $x \leftarrow x \bmod F$ .

## Parameters

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use `cblas_(d)scal` when possible

### 15.1.3.230 `freduce()` [8/10]

```
template INST_OR_DECL void FFLAS::freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

`freduce`  $x \leftarrow y \bmod F$ .

## Parameters

$F$	field
$n$	size of the vectors
$Y$	vector of Element
$incY$	stride of $Y$
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use `cblas_(d)scal` when possible

### 15.1.3.231 `finit()` [7/8]

```
template INST_OR_DECL void FFLAS::finit (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

`finit`  $x \leftarrow y \bmod F$ .

## Parameters

$F$	field
$n$	size of the vectors
$Y$	vector of OtherElement
$incY$	stride of $Y$
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use `cblas_(d)scal` when possible

15.1.3.232 `fconvert()` [3/3]

```
template INST_OR_DECL void FFLAS::fconvert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

`fconvert`  $x \leftarrow y \bmod F$ .

## Parameters

$F$	field
$n$	size of the vectors
$Y$	vector of $F$
$incY$	stride of $Y$
$X$	vector in OtherElement
$incX$	stride of $X$

**Bug** use `cblas_(d)scal` when possible

15.1.3.233 `fnegin()` [3/4]

```
template INST_OR_DECL void FFLAS::fnegin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

`fnegin`  $x \leftarrow -x$ .

## Parameters

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$

**Bug** use `cblas_(d)scal` when possible

15.1.3.234 `fneg()` [3/4]

```
template INST_OR_DECL void FFLAS::fneg (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

`fneg`  $x \leftarrow -y$ .

## Parameters

$F$	field
$n$	size of the vectors
$X$	vector in $F$
$incX$	stride of $X$
$Y$	vector in $F$
$incY$	stride of $Y$

**Bug** use `cblas_(d)scal` when possible

15.1.3.235 `fzero()` [3/4]

```
template INST_OR_DECL void FFLAS::fzero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

`fzero` :  $A \leftarrow 0$ .



## Parameters

$F$	field
$n$	number of elements to zero
$X$	vector in $F$
$incX$	stride of $X$

**15.1.3.236 fiszero()** [3/4]

```
template INST_OR_DECL bool FFLAS::fiszero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * X,
    const size_t incX )
```

**fiszero** : test  $X = 0$ .

## Parameters

$F$	field
$n$	vector dimension
$X$	vector in $F$
$incX$	increment of $X$

**15.1.3.237 fequal()** [3/4]

```
template INST_OR_DECL bool FFLAS::fequal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

**fequal** : test  $X = Y$ .

## Parameters

$F$	field
$n$	vector dimension
$X$	vector in $F$
$incX$	increment of $X$
$Y$	vector in $F$
$incY$	increment of $Y$

**15.1.3.238 fassign()** [9/10]

```
template INST_OR_DECL void FFLAS::fassign (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

fassign :  $x \leftarrow y$ .

X is preallocated

**Todo** variant for triangular matrix

**Parameters**

	$F$	field
	$N$	size of the vectors
out	$X$	vector in F
	$incX$	stride of X
in	$Y$	vector in F
	$incY$	stride of Y

**15.1.3.239 fscaln()** [9/10]

```
template INST_OR_DECL void FFLAS::fscaln (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT alpha,
    FFLAS_ELT * X,
    const size_t incX )
```

fscaln  $x \leftarrow \alpha \cdot x$ .

**Parameters**

$F$	field
$n$	size of the vectors
$alpha$	scalar
$X$	vector in F
$incX$	stride of X

**Bug** use cblas\_(d)scal when possible

**Todo** check if comparison with +/-1,0 is necessary.

**15.1.3.240 fscal()** [9/10]

```
template INST_OR_DECL void FFLAS::fscal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

$\text{fscal } y \leftarrow \alpha \cdot x.$

**Parameters**

	$F$	field
	$n$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in F
	$\text{incX}$	stride of X
out	$Y$	vector in F
	$\text{incY}$	stride of Y

**Bug** use cblas\_(d)scal when possible

**Todo** check if comparison with +/-1,0 is necessary.

**15.1.3.241 faxpy()** [5/6]

```
template INST_OR_DECL void FFLAS::faxpy (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

**Parameters**

	$F$	field
	$N$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in F
	$\text{incX}$	stride of X
in, out	$Y$	vector in F
	$\text{incY}$	stride of Y

**15.1.3.242 fdot()** [10/11]

```
template INST_OR_DECL FFLAS_ELT FFLAS::fdot (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

**Parameters**

	$F$	field
	$N$	size of the vectors
	$\alpha$	scalar
in	$X$	vector in F
	$\text{incX}$	stride of X
	$\beta$	scalar
in, out	$Y$	vector in F
	$\text{incY}$	stride of Y

**Note**

this is a catlas function

**fdot**: dot product  $x^T y$ .

**Parameters**

$F$	field
$N$	size of the vectors
$X$	vector in F
$\text{incX}$	stride of X
$Y$	vector in F
$\text{incY}$	stride of Y

**15.1.3.243 fswap()** [2/2]

```
template INST_OR_DECL void FFLAS::fswap (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    FFLAS_ELT * X,
    const size_t incX,
```

```

FFLAS_ELT * Y,
const size_t incY )

```

fswap:  $X \leftrightarrow Y$ .

**Bug** use `cblas_dswap` when double

#### Parameters

<i>F</i>	field
<i>N</i>	size of the vectors
<i>X</i>	vector in $\mathbb{F}$
<i>incX</i>	stride of $X$
<i>Y</i>	vector in $\mathbb{F}$
<i>incY</i>	stride of $Y$

#### 15.1.3.244 fadd() [5/8]

```

template INST_OR_DECL void FFLAS::fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )

```

#### 15.1.3.245 fsub() [3/4]

```

template INST_OR_DECL void FFLAS::fsub (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )

```

**15.1.3.246 faddin()** [3/4]

```
template INST_OR_DECL void FFLAS::faddin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

**15.1.3.247 fadd()** [6/8]

```
template INST_OR_DECL void FFLAS::fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

**15.1.3.248 fassign()** [10/10]

```
template INST_OR_DECL void FFLAS::fassign (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

fassign :  $A \leftarrow B$ .

**Parameters**

$F$	field
$m$	number of rows to copy
$n$	number of cols to copy
$A$	matrix in F
$lda$	stride of A
$B$	vector in F
$ldb$	stride of B

**15.1.3.249 fzero()** [4/4]

```
template INST_OR_DECL void FFLAS::fzero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

fzero :  $A \leftarrow 0$ .

**Parameters**

$F$	field
$m$	number of rows to zero
$n$	number of cols to zero
$A$	matrix in F
$lda$	stride of A

**Warning**

may be buggy if Element is larger than int

**15.1.3.250 fequal()** [4/4]

```
template INST_OR_DECL bool FFLAS::fequal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb )
```

fequal : test  $A = B$ .

**Parameters**

$F$	field
$m$	row dimension
$n$	column dimension
$A$	m x n matrix in F
$lda$	leading dimension of A
$B$	m x n matrix in F
$ldb$	leading dimension of B

**15.1.3.251 fiszero()** [4/4]

```
template INST_OR_DECL bool FFLAS::fiszero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * A,
    const size_t lda )
```

**fiszero** : test  $A = 0$ .

**Parameters**

$F$	field
$m$	row dimension
$n$	column dimension
$A$	m x n matrix in F
$lda$	leading dimension of A

**15.1.3.252 fidentity()** [3/4]

```
template INST_OR_DECL void FFLAS::fidentity (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT & d )
```

creates a diagonal matrix

**15.1.3.253 fidentity()** [4/4]

```
template INST_OR_DECL void FFLAS::fidentity (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

creates a diagonal matrix

**15.1.3.254 freduce()** [9/10]

```
template INST_OR_DECL void FFLAS::freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

**freduce**  $A \leftarrow A \bmod F$ .



## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.255** **freduce()** [10/10]

```
template INST_OR_DECL void FFLAS::freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

freduce  $A \leftarrow B \bmod F$ .

## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$
$B$	matrix in Element
$ldb$	stride of $B$

**15.1.3.256** **finit()** [8/8]

```
template INST_OR_DECL void FFLAS::finit (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

finit  $A \leftarrow B \bmod F$ .

## Parameters

$F$	field
-----	-------

## Parameters

$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$
$B$	matrix in $F$
$ldb$	stride of $B$

## 15.1.3.257 fnegin() [4/4]

```
template INST_OR_DECL void FFLAS::fnegin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

fnegin  $A \leftarrow -A$ .

## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

## 15.1.3.258 fneg() [4/4]

```
template INST_OR_DECL void FFLAS::fneg (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

fneg  $A \leftarrow -B$ .

## Parameters

$F$	field
$m$	number of rows
$n$	number of cols
$A$	matrix in $F$
$lda$	stride of $A$

**15.1.3.259 fscaln()** [10/10]

```
template INST_OR_DECL void FFLAS::fscaln (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fscaln } A \leftarrow a \cdot A.$

**Parameters**

$F$	field
$m$	number of rows
$n$	number of cols
$\alpha$	homotecie scalar
$A$	matrix in F
$lda$	stride of A

**15.1.3.260 fscl()** [10/10]

```
template INST_OR_DECL void FFLAS::fscl (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

$\text{fscl } B \leftarrow a \cdot A.$

**Parameters**

	$F$	field
	$m$	number of rows
	$n$	number of cols
	$\alpha$	homotecie scalar
in	$A$	matrix in F
	$lda$	stride of A
out	$B$	matrix in F
	$ldb$	stride of B

**15.1.3.261 faxpy()** [6/6]

```
template INST_OR_DECL void FFLAS::faxpy (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t ldx,
    FFLAS_ELT * Y,
    const size_t ldy )
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

**Parameters**

	$F$	field
	$m$	row dimension
	$n$	column dimension
	$\alpha$	scalar
in	$X$	vector in $F$
	$ldx$	leading dimension of $X$
in, out	$Y$	vector in $F$
	$ldy$	leading dimension of $Y$

**15.1.3.262 fmove()** [2/2]

```
template INST_OR_DECL void FFLAS::fmove (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

**Parameters**

	$F$	field
	$m$	row dimension
	$n$	column dimension
	$\alpha$	scalar
in	$X$	vector in $F$
	$ldx$	leading dimension of $X$
	$\beta$	scalar
in, out	$Y$	vector in $F$
	$ldy$	leading dimension of $Y$

**Note**

this is a catlas function

fmove :  $A \leftarrow B$  and  $B \leftarrow 0$ .

**Parameters**

$F$	field
$m$	number of rows to copy
$n$	number of cols to copy
$A$	matrix in $F$
$lda$	stride of $A$
$B$	vector in $F$
$ldb$	stride of $B$

**15.1.3.263 fadd() [7/8]**

```
template INST_OR_DECL void FFLAS::fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

fadd : matrix addition.

Computes  $C = A + B$ .

**Parameters**

$F$	field
$M$	rows
$N$	cols
$A$	dense matrix of size $M \times N$
$lda$	leading dimension of $A$
$B$	dense matrix of size $M \times N$
$ldb$	leading dimension of $B$
$C$	dense matrix of size $M \times N$
$ldc$	leading dimension of $C$

**15.1.3.264 fsub() [4/4]**

```
template INST_OR_DECL void FFLAS::fsub (
```

```

    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

**fsub** : matrix subtraction.

Computes  $C = A - B$ .

#### Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

#### 15.1.3.265 fsubin() [3/3]

```

template INST_OR_DECL void FFLAS::fsubin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

**fsubin**  $C = C - B$

#### 15.1.3.266 fadd() [8/8]

```

template INST_OR_DECL void FFLAS::fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT alpha,

```

```

    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

fadd : matrix addition with scaling.

Computes  $C = A + \alpha B$ .

#### Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>alpha</i>	some scalar
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

#### 15.1.3.267 faddin() [4/4]

```

template INST_OR_DECL void FFLAS::faddin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

faddin

#### 15.1.3.268 fgemv() [17/19]

```

template INST_OR_DECL FFLAS_ELT* FFLAS::fgemv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE TransA,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT beta,

```

```
FFLAS_ELT * Y,  
const size_t incY )
```

finite prime FFLAS\_FIELD<FFLAS\_ELT> GEneral Matrix Vector multiplication.

Computes  $Y \leftarrow \alpha \text{op}(A)X + \beta Y$ .



## Parameters

	<i>F</i>	field
	<i>TransA</i>	if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .
	<i>M</i>	rows
	<i>N</i>	cols
	<i>alpha</i>	scalar
	<i>A</i>	dense matrix of size MxN
	<i>lda</i>	leading dimension of A
	<i>X</i>	dense vector of size N
	<i>incX</i>	stride of X
	<i>beta</i>	scalar
out	<i>Y</i>	dense vector of size M
	<i>incY</i>	stride of Y

## 15.1.3.269 fger() [12/12]

```
template INST_OR_DECL void FFLAS::fger (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * x,
    const size_t incx,
    const FFLAS_ELT * y,
    const size_t incy,
    FFLAS_ELT * A,
    const size_t lda )
```

fger: rank one update of a general matrix

Computes  $A \leftarrow \alpha x y^T + A$

## Parameters

	<i>F</i>	field
	<i>M</i>	rows
	<i>N</i>	cols
	<i>alpha</i>	scalar
in, out	<i>A</i>	dense matrix of size MxN and leading dimension lda
	<i>lda</i>	leading dimension of A
	<i>x</i>	dense vector of size M
	<i>incx</i>	stride of X
	<i>y</i>	dense vector of size N
	<i>incy</i>	stride of Y

**15.1.3.270 ftrsv()** [2/2]

```
template INST_OR_DECL void FFLAS::ftrsv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    int incX )
```

ftrsv: **TR**iangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

**Parameters**

<i>F</i>	field
<i>X</i>	vector of size N on a field F
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size N
<i>lda</i>	leading dimension of A
<i>N</i>	number of rows or columns of A according to TransA
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$ .
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular

**15.1.3.271 ftrsm()** [9/9]

```
template INST_OR_DECL void FFLAS::ftrsm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

ftrsm: **TR**iangular System solve with **M**atrix.

Computes  $B \leftarrow \alpha \text{op}(A^{-1})B$  or  $B \leftarrow \alpha B \text{op}(A^{-1})$ .

**Parameters**

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A^{-1})B$ is computed.

## Parameters

<i>Uplo</i>	if <code>Uplo==FflasUpper</code> then A is upper triangular
<i>TransA</i>	if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .
<i>Diag</i>	if <code>Diag==FflasUnit</code> then A is unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular invertible matrix. If <code>Side==FflasLeft</code> then A is $N \times N$ , otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B

**Bug**  $\alpha$  must be non zero.

## 15.1.3.272 ftrmm() [3/3]

```
template INST_OR_DECL void FFLAS::ftrmm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

ftrmm: **TR**iangular **M**atrix **M**ultiply.

Computes  $B \leftarrow \alpha \text{op}(A)B$  or  $B \leftarrow \alpha B \text{op}(A)$ .

## Parameters

<i>F</i>	field
<i>Side</i>	if <code>Side==FflasLeft</code> then $B \leftarrow \alpha \text{op}(A)B$ is computed.
<i>Uplo</i>	if <code>Uplo==FflasUpper</code> then A is upper triangular
<i>TransA</i>	if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .
<i>Diag</i>	if <code>Diag==FflasUnit</code> then A is implicitly unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular matrix. If <code>Side==FflasLeft</code> then A is $N \times N$ , otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B

### 15.1.3.273 fgemm() [20/23]

```
template INST_OR_DECL FFLAS_ELT* FFLAS::fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc )
```

fgemm: **F**ield **G**eneral **M**atrix **M**ultiply.

Computes  $C = \alpha \text{op}(A) \times \text{op}(B) + \beta C$  Automatically set Winograd recursion level

#### Parameters

<i>F</i>	field.
<i>ta</i>	if <code>ta==FflasTrans</code> then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ ,
<i>tb</i>	same for matrix B
<i>m</i>	see A
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	$\text{op}(A)$ is $m \times k$
<i>B</i>	$\text{op}(B)$ is $k \times n$
<i>C</i>	$C$ is $m \times n$
<i>lda</i>	leading dimension of A
<i>ldb</i>	leading dimension of B
<i>ldc</i>	leading dimension of C
<i>w</i>	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of $w$ .

#### Warning

$\alpha$  must be invertible

### 15.1.3.274 fgemm() [21/23]

```
template INST_OR_DECL FFLAS_ELT* FFLAS::fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```

```

const FFLAS_TRANSPOSE ta,
const FFLAS_TRANSPOSE tb,
const size_t m,
const size_t n,
const size_t k,
const FFLAS_ELT alpha,
const FFLAS_ELT * A,
const size_t lda,
const FFLAS_ELT * B,
const size_t ldb,
const FFLAS_ELT beta,
FFLAS_ELT * C,
const size_t ldc,
const ParSeqHelper::Sequential seq )

```

### 15.1.3.275 fgemm() [22/23]

```

template INST_OR_DECL FFLAS_ELT* FFLAS::fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc,
    const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive
> par )

```

### 15.1.3.276 fgemm() [23/23]

```

template INST_OR_DECL FFLAS_ELT* FFLAS::fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,

```

```

        const size_t ldc,
        const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads
> par )

```

### 15.1.3.277 fsquare() [6/6]

```

template INST_OR_DECL FFLAS_ELT* FFLAS::fsquare (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc )

```

fsquare: Squares a matrix.

compute  $C \leftarrow \alpha \text{op}(A) \text{op}(A) + \beta C$  over a FFLAS\_FIELD <FFLAS\_ELT> F Avoid the conversion of B

#### Parameters

<i>ta</i>	if ta==FflasTrans, $\text{op}(A) = A^T$ .
<i>F</i>	field
<i>n</i>	size of A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	dense matrix of size n×n
<i>lda</i>	leading dimension of A
<i>C</i>	dense matrix of size n×n
<i>ldc</i>	leading dimension of C

### 15.1.3.278 BlockCuts() [1/2]

```

void FFLAS::BlockCuts (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]

```

**15.1.3.279 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.280 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.281 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.282 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.283 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.284 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.285 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.286 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.287 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.288 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()**

```
void FFLAS::BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```



**15.1.3.289 BlockCuts()** [2/2]

```
void FFLAS::BlockCuts (
    size_t & rowBlockSize,
    size_t & colBlockSize,
    size_t & lastRBS,
    size_t & lastCBS,
    size_t & changeRBS,
    size_t & changeCBS,
    size_t & numRowsBlock,
    size_t & numColBlock,
    size_t m,
    size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.290 pfzero()**

```
void FFLAS::pfzero (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr C,
    size_t BS = 0 )
```

**15.1.3.291 pfrand()**

```
void FFLAS::pfrand (
    const Field & F,
    RandIter & G,
    size_t m,
    size_t n,
    typename Field::Element_ptr C,
    size_t BS = 0 )
```

**15.1.3.292 fdot()** [11/11]

```
Field::Element& FFLAS::fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element & d,
    const ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

**15.1.3.293 pfgemm()** [2/7]

```
Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block,
StrategyParameter::Threads > > & H )
```

**15.1.3.294 pfgemm()** [3/7]

```
Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDAdaptive > > & H )
```

**15.1.3.295 pfgemm()** [4/7]

```
Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
```

```

    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoDAdaptive > > & H )

```

### 15.1.3.296 pfgemm() [5/7]

```

Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoD > > & H )

```

### 15.1.3.297 pfgemm() [6/7]

```

Field::Element_ptr FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeD > > & H )

```

**15.1.3.298 pfgemm() [7/7]**

```
Field::Element* FFLAS::pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
    StrategyParameter::ThreeDInPlace > > & H )
```

**15.1.3.299 fgemv() [18/19]**

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
    StrategyParameter::Threads > > & H )
```

**15.1.3.300 fgemv() [19/19]**

```
Field::Element_ptr FFLAS::fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
```

```

    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row,
Cut > > & H )

```

### 15.1.3.301 parseArguments()

```

void parseArguments (
    int argc,
    char ** argv,
    Argument * args,
    bool printDefaults = true )

```

### 15.1.3.302 writeCommandString()

```

std::ostream& FFLAS::writeCommandString (
    std::ostream & os,
    Argument * args,
    const char * programName = nullptr )

```

writes the values of all arguments, preceded by the programName

### 15.1.3.303 WriteMatrix() [1/2]

```

std::ostream& FFLAS::WriteMatrix (
    std::ostream & c,
    const Field & F,
    size_t m,
    size_t n,
    typename Field::ConstElement_ptr A,
    size_t lda,
    FFLAS_FORMAT format,
    bool column_major )

```

WriteMatrix: write a matrix to an output stream.

#### Parameters

<i>c</i>	output stream
<i>F</i>	base field
<i>m</i>	row dimension
<i>n</i>	column dimension
<i>A</i>	matrix
<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
<i>column_major</i>	whether the matrix is stored in column or row major (row by default)

**15.1.3.304 preamble()**

```
void FFLAS::preamble (
    std::ifstream & ifs,
    FFLAS_FORMAT & format ) [inline]
```

**15.1.3.305 ReadMatrix() [1/2]**

```
Field::Element_ptr FFLAS::ReadMatrix (
    std::ifstream & ifs,
    Field & F,
    size_t & m,
    size_t & n,
    typename Field::Element_ptr & A,
    FFLAS_FORMAT format = FflasAuto )
```

ReadMatrix: read a matrix from an input stream.

**Parameters**

	<i>ifs</i>	input stream
	<i>F</i>	base field
out	<i>m</i>	row dimension
out	<i>n</i>	column dimension
out	<i>A</i>	output matrix
	<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

**15.1.3.306 ReadMatrix() [2/2]**

```
Field::Element_ptr FFLAS::ReadMatrix (
    const std::string & matrix_file,
    Field & F,
    size_t & m,
    size_t & n,
    typename Field::Element_ptr & A,
    FFLAS_FORMAT format = FflasAuto ) [inline]
```

ReadMatrix: read a matrix from a file.

**Parameters**

	<i>matrix_file</i>	filename
	<i>F</i>	base field
out	<i>m</i>	row dimension
out	<i>n</i>	column dimension
out	<i>A</i>	output matrix
	<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

**15.1.3.307 WriteMatrix()** [2/2]

```
void FFLAS::WriteMatrix (
    std::string & matrix_file,
    const Field & F,
    int m,
    int n,
    typename Field::ConstElement_ptr A,
    size_t lda,
    FFLAS_FORMAT format = FflasDense,
    bool column_major = false )
```

WriteMatrix: write a matrix to a file.

**Parameters**

<i>matrix_file</i>	file name
<i>F</i>	base field
<i>m</i>	row dimension
<i>n</i>	column dimension
<i>A</i>	matrix
<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
<i>column_major</i>	whether the matrix is stored in column or row major (row by default)

**15.1.3.308 WritePermutation()**

```
std::ostream& FFLAS::WritePermutation (
    std::ostream & c,
    const size_t * P,
    size_t N ) [inline]
```

WritePermutation: write a permutation matrix to an output stream.

**Parameters**

<i>c</i>	output stream
<i>P</i>	permutation
<i>N</i>	size of the permutation

**15.1.3.309 alignable()**

```
bool FFLAS::alignable ( ) [inline]
```

**15.1.3.310 alignable< Givaro::Integer \* >()**

```
bool FFLAS::alignable< Givaro::Integer * > ( ) [inline]
```

**15.1.3.311 fflas\_new() [5/7]**

```
Field::Element_ptr FFLAS::fflas_new (
    const Field & F,
    const size_t m,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.312 fflas\_new() [6/7]**

```
Field::Element_ptr FFLAS::fflas_new (
    const Field & F,
    const size_t m,
    const size_t n,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.313 fflas\_new() [7/7]**

```
Element* FFLAS::fflas_new (
    const size_t m,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.314 fflas\_delete() [3/3]**

```
void FFLAS::fflas_delete (
    Ptr p,
    Args ... args ) [inline]
```

**15.1.3.315 prefetch()**

```
void FFLAS::prefetch (
    const int64_t * ) [inline]
```



**15.1.3.316 getTLBSize()**

```
void FFLAS::getTLBSize (
    int & tlb ) [inline]
```

**15.1.3.317 queryCacheSizes()**

```
void FFLAS::queryCacheSizes (
    int & l1,
    int & l2,
    int & l3 ) [inline]
```

Queries and returns the cache sizes in Bytes of the L1, L2, and L3 data caches respectively

**15.1.3.318 queryL1CacheSize()**

```
int FFLAS::queryL1CacheSize ( ) [inline]
```

**Returns**

the size in Bytes of the L1 data cache

**15.1.3.319 queryTopLevelCacheSize()**

```
int FFLAS::queryTopLevelCacheSize ( ) [inline]
```

**Returns**

the size in Bytes of the L2 or L3 cache if this later is present

**15.1.3.320 getSeed()**

```
uint64_t FFLAS::getSeed ( )
```

## 15.2 FFLAS::BLAS3 Namespace Reference

### Functions

- `template<class Field >`  
`void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE base, const size_t rec_level)`
- `template<class Field , class FieldTrait , class Strat , class Param >`  
`Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_3_23 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_3_21 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_2_24 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_2_27 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_LR (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

- `template<class Field , class FieldTrait >`  
`void WinogradAcc_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd_LR_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

## 15.2.1 Function Documentation

### 15.2.1.1 Bini()

```
void FFLAS::BLAS3::Bini (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t kmax,
    const size_t w,
    const FFLAS_BASE base,
    const size_t rec_level ) [inline]
```

### 15.2.1.2 WinoPar()

```
Field::Element_ptr FFLAS::BLAS3::WinoPar (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel<
Strat, Param > > & WH ) [inline]
```

### 15.2.1.3 Winograd()

```
void FFLAS::BLAS3::Winograd (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

### 15.2.1.4 WinogradAcc\_3\_23()

```
void FFLAS::BLAS3::WinogradAcc_3_23 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
```

```

const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.2.1.5 WinogradAcc\_3\_21()

```

void FFLAS::BLAS3::WinogradAcc_3_21 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.2.1.6 WinogradAcc\_2\_24()

```

void FFLAS::BLAS3::WinogradAcc_2_24 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

### 15.2.1.7 WinogradAcc\_2\_27()

```
void FFLAS::BLAS3::WinogradAcc_2_27 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

### 15.2.1.8 WinogradAcc\_LR()

```
void FFLAS::BLAS3::WinogradAcc_LR (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

### 15.2.1.9 WinogradAcc\_R\_S()

```
void FFLAS::BLAS3::WinogradAcc_R_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
```

```

typename Field::Element_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.2.1.10 WinogradAcc\_L\_S()

```

void FFLAS::BLAS3::WinogradAcc_L_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.2.1.11 Winograd\_LR\_S()

```

void FFLAS::BLAS3::Winograd_LR_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

### 15.2.1.12 Winograd\_L\_S()

```
void FFLAS::BLAS3::Winograd_L_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

### 15.2.1.13 Winograd\_R\_S()

```
void FFLAS::BLAS3::Winograd_R_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

## 15.3 FFLAS::csr\_hyb\_details Namespace Reference

### Data Structures

- struct [Info](#)
- struct [Coo](#)

## 15.4 FFLAS::CuttingStrategy Namespace Reference

### Data Structures

- struct [Single](#)
- struct [Row](#)
- struct [Column](#)
- struct [Block](#)
- struct [Recursive](#)



## Typedefs

- typedef [Row](#) [RNSModulus](#)

### 15.4.1 Typedef Documentation

#### 15.4.1.1 RNSModulus

```
typedef Row RNSModulus
```

## 15.5 FFLAS::details Namespace Reference

### Functions

- template<class Field , bool ADD>  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::ModularTag](#))
- template<class Field , bool ADD>  
std::enable\_if<![FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::ModularTag](#))
- template<class Field , bool ADD>  
void [fadd](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, type-  
name [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::GenericTag](#))
- template<class Field , bool ADD>  
std::enable\_if<![FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::UnparametricTag](#))
- template<class Field , bool ADD>  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::UnparametricTag](#))
- template<class Field >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [freduce](#) (const [Field](#) &F, const size\_t m, typename [Field::Element\\_ptr](#) A, const size\_t incX, [FieldCategories::ModularTag](#))
- template<class Field >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [freduce](#)  
(const [Field](#) &F, const size\_t m, typename [Field::ConstElement\\_ptr](#) B, const size\_t incY, typename [Field::Element\\_ptr](#) A, const size\_t incX, [FieldCategories::ModularTag](#))
- template<class Field , class FC >  
void [freduce](#) (const [Field](#) &F, const size\_t m, typename [Field::Element\\_ptr](#) A, const size\_t incX, FC)
- template<class Field , class FC >  
void [freduce](#) (const [Field](#) &F, const size\_t m, typename [Field::ConstElement\\_ptr](#) B, const size\_t incY, type-  
name [Field::Element\\_ptr](#) A, const size\_t incX, FC)

- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type fscalin` (const `Field` &F, const `size_t` N, const typename `Field::Element` a, typename `Field::Element_ptr` X, const `size_t` incX, `FieldCategories::ModularTag`)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type fscal` (const `Field` &F, const `size_t` N, const typename `Field::Element` a, typename `Field::ConstElement_ptr` X, const `size_t` incX, typename `Field::Element_ptr` Y, const `size_t` incY, `FieldCategories::ModularTag`)
- `template<class Field, class FC >`  
`void fscalin` (const `Field` &F, const `size_t` n, const typename `Field::Element` a, typename `Field::Element_ptr` X, const `size_t` incX, FC)
- `template<class Field, class FC >`  
`void fscal` (const `Field` &F, const `size_t` N, const typename `Field::Element` a, typename `Field::ConstElement_ptr` X, const `size_t` incX, typename `Field::Element_ptr` Y, const `size_t` incY, FC)
- `template<enum number_kind K>`  
`void igebb44` (`size_t` i, `size_t` j, `size_t` depth, `size_t` pdeth, const `int64_t` alpha, const `int64_t` \*blA, const `int64_t` \*blB, `int64_t` \*C, `size_t` ldc)
- `template<enum number_kind K>`  
`void igebb24` (`size_t` i, `size_t` j, `size_t` depth, `size_t` pdeth, const `int64_t` alpha, const `int64_t` \*blA, const `int64_t` \*blB, `int64_t` \*C, `size_t` ldc)
- `template<enum number_kind K>`  
`void igebb14` (`size_t` i, `size_t` j, `size_t` depth, `size_t` pdeth, const `int64_t` alpha, const `int64_t` \*blA, const `int64_t` \*blB, `int64_t` \*C, `size_t` ldc)
- `template<enum number_kind K>`  
`void igebb41` (`size_t` i, `size_t` j, `size_t` depth, `size_t` pdeth, const `int64_t` alpha, const `int64_t` \*blA, const `int64_t` \*blB, `int64_t` \*C, `size_t` ldc)
- `template<enum number_kind K>`  
`void igebb21` (`size_t` i, `size_t` j, `size_t` depth, `size_t` pdeth, const `int64_t` alpha, const `int64_t` \*blA, const `int64_t` \*blB, `int64_t` \*C, `size_t` ldc)
- `template<enum number_kind K>`  
`void igebb11` (`size_t` i, `size_t` j, `size_t` depth, `size_t` pdeth, const `int64_t` alpha, const `int64_t` \*blA, const `int64_t` \*blB, `int64_t` \*C, `size_t` ldc)
- `template<enum number_kind K>`  
`void igebp` (`size_t` rows, `size_t` cols, `size_t` depth, const `int64_t` alpha, const `int64_t` \*blockA, `size_t` lda, const `int64_t` \*blockB, `size_t` ldb, `int64_t` \*C, `size_t` ldc)
- `template<size_t k, bool transpose>`  
`void pack_lhs` (`int64_t` \*XX, const `int64_t` \*X, `size_t` ldx, `size_t` rows, `size_t` cols)
- `template<size_t k, bool transpose>`  
`void pack_rhs` (`int64_t` \*XX, const `int64_t` \*X, `size_t` ldx, `size_t` rows, `size_t` cols)
- `void gebp` (`size_t` rows, `size_t` cols, `size_t` depth, `int64_t` \*C, `size_t` ldc, const `int64_t` \*blockA, `size_t` lda, const `int64_t` \*BlockB, `size_t` ldb, `int64_t` \*BlockW)
- `void BlockingFactor` (`size_t` &m, `size_t` &n, `size_t` &k)

## 15.5.1 Function Documentation

### 15.5.1.1 fadd() [1/5]

```
std::enable_if<FFLAS::support_simd_add<typename Field::Element>::value, void>::type FFLAS<
::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
```

```

    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::ModularTag )

```

### 15.5.1.2 fadd() [2/5]

```

std::enable_if<!FFLAS::support_simd_add<typename Field::Element>::value, void>::type FFLAS↵
::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::ModularTag )

```

### 15.5.1.3 fadd() [3/5]

```

void FFLAS::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::GenericTag )

```

### 15.5.1.4 fadd() [4/5]

```

std::enable_if<!FFLAS::support_simd_add<typename Field::Element>::value, void>::type FFLAS↵
::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::UnparametricTag ) [inline]

```

**15.5.1.5 fadd()** [5/5]

```
std::enable_if<FFLAS::support_simd_add<typename Field::Element>::value, void>::type FFLAS↵
::details::fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::UnparametricTag ) [inline]
```

**15.5.1.6 freduce()** [1/4]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value , void>::type FFLAS↵
::details::freduce (
    const Field & F,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

**15.5.1.7 freduce()** [2/4]

```
std::enable_if< FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::details::freduce (
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr B,
    const size_t incY,
    typename Field::Element_ptr A,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

**15.5.1.8 freduce()** [3/4]

```
void FFLAS::details::freduce (
    const Field & F,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t incX,
    FC ) [inline]
```

**15.5.1.9 freduce()** [4/4]

```
void FFLAS::details::freduce (
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr B,
    const size_t incY,
    typename Field::Element_ptr A,
    const size_t incX,
    FC ) [inline]
```

**15.5.1.10 fscaln()** [1/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::details::fscaln (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::Element_ptr X,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

**15.5.1.11 fscl()** [1/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::details::fscl (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FieldCategories::ModularTag ) [inline]
```

**15.5.1.12 fscaln()** [2/2]

```
void FFLAS::details::fscaln (
    const Field & F,
    const size_t n,
    const typename Field::Element a,
    typename Field::Element_ptr X,
    const size_t incX,
    FC ) [inline]
```

#### 15.5.1.13 fscal() [2/2]

```
void FFLAS::details::fscal (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FC ) [inline]
```

#### 15.5.1.14 igebb44()

```
void igebb44 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

#### 15.5.1.15 igebb24()

```
void igebb24 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

#### 15.5.1.16 igebb14()

```
void igebb14 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

#### 15.5.1.17 igebb41()

```
void igebb41 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

bug ,B\_0 dans VEC\_MADD\_32 ?

bug ,B\_0 dans VEC\_MADD\_32 ?

#### 15.5.1.18 igebb21()

```
void igebb21 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

#### 15.5.1.19 igebb11()

```
void igebb11 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

### 15.5.1.20 igebp()

```
void igebp (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * blockA,
    size_t lda,
    const int64_t * blockB,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

### 15.5.1.21 pack\_lhs()

```
void pack_lhs (
    int64_t * XX,
    const int64_t * X,
    size_t ldx,
    size_t rows,
    size_t cols )
```

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

### 15.5.1.22 pack\_rhs()

```
void pack_rhs (
    int64_t * XX,
    const int64_t * X,
    size_t ldx,
    size_t rows,
    size_t cols )
```

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign



### 15.5.1.23 gebp()

```
void FFLAS::details::gebp (
    size_t rows,
    size_t cols,
    size_t depth,
    int64_t * C,
    size_t ldc,
    const int64_t * blockA,
    size_t lda,
    const int64_t * BlockB,
    size_t ldb,
    int64_t * BlockW )
```

### 15.5.1.24 BlockingFactor()

```
void BlockingFactor (
    size_t & m,
    size_t & n,
    size_t & k ) [inline]
```

## 15.6 FFLAS::details\_spmv Namespace Reference

### Data Structures

- struct [Coo](#)

## 15.7 FFLAS::ElementCategories Namespace Reference

### Data Structures

- struct [GenericTag](#)  
*default is generic*
- struct [MachineFloatTag](#)  
*float or double*
- struct [MachineIntTag](#)  
*short, int, long, long long, and unsigned variants*
- struct [FixedPrecIntTag](#)  
*Fixed precision integers above machine precision: Givaro::reclnt.*
- struct [ArbitraryPrecIntTag](#)  
*Arbitrary precision integers: GMP.*
- struct [RNSElementTag](#)  
*Representation in a Residue Number System.*

## 15.8 FFLAS::FieldCategories Namespace Reference

Traits and categories will need to be placed in a proper file later.

### Data Structures

- struct [GenericTag](#)  
*generic ring.*
- struct [ModularTag](#)  
*This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`*
- struct [UnparametricTag](#)  
*If the field uses a representation with infix operators.*

### 15.8.1 Detailed Description

Traits and categories will need to be placed in a proper file later.

## 15.9 FFLAS::MMHelperAlgo Namespace Reference

### Data Structures

- struct [Auto](#)
- struct [Classic](#)
- struct [Winograd](#)
- struct [WinogradPar](#)
- struct [Bini](#)

## 15.10 FFLAS::ModeCategories Namespace Reference

Specifies the mode of action for an algorithm w.r.t.

### Data Structures

- struct [DefaultTag](#)  
*No specific mode of action: use standard field operations.*
- struct [DefaultBoundedTag](#)  
*Use standard field operations, but keeps track of bounds on input and output.*
- struct [ConvertTo](#)  
*Force conversion to appropriate element type of `ElementCategory T`.*
- struct [DelayedTag](#)  
*Performs field operations with delayed mod reductions. Ensures result is reduced.*
- struct [LazyTag](#)  
*Performs field operations with delayed mod only when necessary. Result may not be reduced.*

### 15.10.1 Detailed Description

Specifies the mode of action for an algorithm w.r.t.

its field

## 15.11 FFLAS::ParSeqHelper Namespace Reference

[ParSeqHelper](#) for both fgemm and ftrsm.

### Data Structures

- struct [Parallel](#)
- struct [Sequential](#)
- struct [Compose](#)

### 15.11.1 Detailed Description

[ParSeqHelper](#) for both fgemm and ftrsm.

[ParSeqHelper](#) for both fgemm and ftrsm

## 15.12 FFLAS::Protected Namespace Reference

### Data Structures

- class [AreEqual](#)
  - class [AreEqual< X, X >](#)
  - class [ftrsmLeftUpperNoTransNonUnit](#)
- Computes the maximal size for delaying the modular reduction in a triangular system resolution.*
- class [ftrsmLeftUpperNoTransUnit](#)
  - class [ftrsmLeftUpperTransNonUnit](#)
  - class [ftrsmLeftUpperTransUnit](#)
  - class [ftrsmLeftLowerNoTransNonUnit](#)
  - class [ftrsmLeftLowerNoTransUnit](#)
  - class [ftrsmLeftLowerTransNonUnit](#)
  - class [ftrsmLeftLowerTransUnit](#)
  - class [ftrsmRightUpperNoTransNonUnit](#)
  - class [ftrsmRightUpperNoTransUnit](#)
  - class [ftrsmRightUpperTransNonUnit](#)
  - class [ftrsmRightUpperTransUnit](#)
  - class [ftrsmRightLowerNoTransNonUnit](#)
  - class [ftrsmRightLowerNoTransUnit](#)
  - class [ftrsmRightLowerTransNonUnit](#)
  - class [ftrsmRightLowerTransUnit](#)
  - class [ftrmmLeftUpperNoTransNonUnit](#)
  - class [ftrmmLeftUpperNoTransUnit](#)

- class [ftrmmLeftUpperTransNonUnit](#)
- class [ftrmmLeftUpperTransUnit](#)
- class [ftrmmLeftLowerNoTransNonUnit](#)
- class [ftrmmLeftLowerNoTransUnit](#)
- class [ftrmmLeftLowerTransNonUnit](#)
- class [ftrmmLeftLowerTransUnit](#)
- class [ftrmmRightUpperNoTransNonUnit](#)
- class [ftrmmRightUpperNoTransUnit](#)
- class [ftrmmRightUpperTransNonUnit](#)
- class [ftrmmRightUpperTransUnit](#)
- class [ftrmmRightLowerNoTransNonUnit](#)
- class [ftrmmRightLowerNoTransUnit](#)
- class [ftrmmRightLowerTransNonUnit](#)
- class [ftrmmRightLowerTransUnit](#)

## Functions

- template<class Field >  
double [computeFactorClassic](#) (const [Field](#) &F)
- template<> double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< double > &F)
- template<> double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< float > &F)
- template<class Field >  
size\_t [DotProdBoundClassic](#) (const [Field](#) &F, const typename [Field::Element](#) &beta)
- template<class Field >  
size\_t [TRSMBound](#) (const [Field](#) &)  
*TRSMBound.*
- template<class Element >  
size\_t [TRSMBound](#) (const [Givaro::Modular](#)< Element > &F)  
*Specialization for positive modular representation over float.*
- template<class Element >  
size\_t [TRSMBound](#) (const [Givaro::ModularBalanced](#)< Element > &F)  
*Specialization for balanced modular representation over double.*
- template<class Field >  
int [WinogradThreshold](#) (const [Field](#) &F)  
*Computes the number of recursive levels to perform.*
- template<> int [WinogradThreshold](#) (const [Givaro::Modular](#)< float > &F)
- template<> int [WinogradThreshold](#) (const [Givaro::ModularBalanced](#)< double > &F)
- template<> int [WinogradThreshold](#) (const [Givaro::ModularBalanced](#)< float > &F)
- template<class Field >  
int [WinogradSteps](#) (const [Field](#) &F, const size\_t &m)  
*Computes the number of recursive levels to perform.*
- template<class Field , class FieldMode >  
void [DynamicPeeling](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), FieldMode > &H, const typename [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), FieldMode >::DelayedField::Element Cmin, const typename [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), FieldMode >::DelayedField::Element Cmax)

- `template<class Field , class FieldMode >`  
`void DynamicPeeling2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`  
`void WinogradCalc (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`
- `template<class NewField , class Field , class FieldMode >`  
`Field::Element_ptr fgemm_convert (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`
- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ParSeqTrait >`  
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >`  
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class Field , class AlgoT , class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)`
- `template<class Field , class AlgoT , class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)`
- `template<class Field >`  
`Field::Element_ptr fsquareCommon (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`
- `template<typename FloatElement , class Field >`  
`Field::Element_ptr fgemv_convert (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda,`

- typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY)
- `template<class FloatElement, class Field >`  
`void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement\_ptr x, const size_t incx, typename Field::ConstElement\_ptr y, const size_t`  
`incy, typename Field::Element\_ptr A, const size_t lda)`
  - `template<class DFE >`  
`size_t min_types (const DFE &k)`
  - `template<> size_t min_types (const Reclnt::rint< 6 > &k)`
  - `template<> size_t min_types (const Reclnt::rint< 7 > &k)`
  - `template<> size_t min_types (const Reclnt::rint< 8 > &k)`
  - `template<> size_t min_types (const Reclnt::rint< 9 > &k)`
  - `template<> size_t min_types (const Reclnt::rint< 10 > &k)`
  - `template<> size_t min_types (const Givaro::Integer &k)`
  - `template<class T >`  
`bool unfit (T x)`
  - `template<> bool unfit (int64\_t x)`
  - `template<size_t K>`  
`bool unfit (Reclnt::rint< K > x)`
  - `template<> bool unfit (Reclnt::rint< 6 > x)`
  - `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>`  
`void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64\_t alpha, const int64\_t *A, size_t lda,`  
`const int64\_t *B, size_t ldb, int64\_t *C, size_t ldc)`
  - `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>`  
`void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64\_t alpha, const int64\_t *A, size_t lda,`  
`const int64\_t *B, size_t ldb, int64\_t *C, size_t ldc)`
  - `void igemm (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size_t`  
`rows, size_t cols, size_t depth, const int64\_t alpha, const int64\_t *A, size_t lda, const int64\_t *B, size_t ldb,`  
`const int64\_t beta, int64\_t *C, size_t ldc)`
  - `template<class Field >`  
`void MatF2MatD_Triangular (const Field &F, Givaro::DoubleDomain::Element\_ptr S, const size_t lds, type-`  
`name Field::ConstElement\_ptr const E, const size_t lde, const size_t m, const size_t n)`
  - `template<class Field >`  
`void MatF2MatFI_Triangular (const Field &F, Givaro::FloatDomain::Element\_ptr S, const size_t lds, typename`  
`Field::ConstElement\_ptr const E, const size_t lde, const size_t m, const size_t n)`

## 15.12.1 Function Documentation

### 15.12.1.1 computeFactorClassic() [1/3]

```
double FFLAS::Protected::computeFactorClassic (
    const Field & F ) [inline]
```

### 15.12.1.2 computeFactorClassic() [2/3]

```
double FFLAS::Protected::computeFactorClassic (
    const Givaro::ModularBalanced< double > & F ) [inline]
```

**15.12.1.3 computeFactorClassic() [3/3]**

```
double FFLAS::Protected::computeFactorClassic (
    const Givaro::ModularBalanced< float > & F ) [inline]
```

**15.12.1.4 DotProdBoundClassic()**

```
size_t FFLAS::Protected::DotProdBoundClassic (
    const Field & F,
    const typename Field::Element & beta ) [inline]
```

**15.12.1.5 TRSMBound() [1/3]**

```
size_t FFLAS::Protected::TRSMBound (
    const Field & ) [inline]
```

TRSMBound.

computes the maximal size for delaying the modular reduction in a triangular system resolution

This is the default version over an arbitrary field. It is currently never used (the recursive algorithm is run until  $n=1$  in this case)

**Parameters**

$F$	Finite Field/Ring of the computation
-----	--------------------------------------

**15.12.1.6 TRSMBound() [2/3]**

```
size_t FFLAS::Protected::TRSMBound (
    const Givaro::Modular< Element > & F ) [inline]
```

Specialization for positive modular representation over float.

Computes  $n_{\max}$  s.t.  $(p-1)/2 * (p^{\{n_{\max}-1\}} + (p-2)^{\{n_{\max}-1\}}) < 2^{24}$  @pbi See [Dumas Giorgi Pernet 06, arXiv:cs/0601133]

**15.12.1.7 TRSMBound() [3/3]**

```
size_t FFLAS::Protected::TRSMBound (
    const Givaro::ModularBalanced< Element > & F ) [inline]
```

Specialization for balanced modular representation over double.

Computes  $n_{\max}$  s.t.  $(p-1)/2 * (((p+1)/2)^{\{n_{\max}-1\}}) < 2^{53}$

**Bibliography** • Dumas Giorgi Pernet 06, arXiv:cs/0601133

**15.12.1.8 WinogradThreshold()** [1/4]

```
int FFLAS::Protected::WinogradThreshold (
    const Field & F ) [inline]
```

Computes the number of recursive levels to perform.

**Parameters**

<i>m</i>	the common dimension in the product AxB
----------	---

**15.12.1.9 WinogradThreshold()** [2/4]

```
int FFLAS::Protected::WinogradThreshold (
    const Givaro::Modular< float > & F ) [inline]
```

**15.12.1.10 WinogradThreshold()** [3/4]

```
int FFLAS::Protected::WinogradThreshold (
    const Givaro::ModularBalanced< double > & F ) [inline]
```

**15.12.1.11 WinogradThreshold()** [4/4]

```
int FFLAS::Protected::WinogradThreshold (
    const Givaro::ModularBalanced< float > & F ) [inline]
```

**15.12.1.12 WinogradSteps()**

```
int FFLAS::Protected::WinogradSteps (
    const Field & F,
    const size_t & m ) [inline]
```

Computes the number of recursive levels to perform.

**Parameters**

<i>m</i>	the common dimension in the product AxB
----------	---



## 15.12.1.13 DynamicPeeling()

```

void FFLAS::Protected::DynamicPeeling (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↔
Field::Element Cmin,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↔
Field::Element Cmax ) [inline]

```

## 15.12.1.14 DynamicPeeling2()

```

void FFLAS::Protected::DynamicPeeling2 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↔
Field::Element Cmin,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↔
Field::Element Cmax ) [inline]

```

### 15.12.1.15 WinogradCalc()

```
void FFLAS::Protected::WinogradCalc (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H ) [inline]
```

### 15.12.1.16 fgemv\_convert()

```
Field::Element_ptr FFLAS::Protected::fgemv_convert (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H ) [inline]
```

### 15.12.1.17 NeedPreAddReduction() [1/2]

```
bool FFLAS::Protected::NeedPreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

**15.12.1.18 NeedPreAddReduction()** [2/2]

```
bool FFLAS::Protected::NeedPreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

**15.12.1.19 NeedPreSubReduction()** [1/2]

```
bool FFLAS::Protected::NeedPreSubReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

**15.12.1.20 NeedPreSubReduction()** [2/2]

```
bool FFLAS::Protected::NeedPreSubReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

**15.12.1.21 NeedDoublePreAddReduction()** [1/2]

```
bool FFLAS::Protected::NeedDoublePreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    Element beta,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

**15.12.1.22 NeedDoublePreAddReduction() [2/2]**

```
bool FFLAS::Protected::NeedDoublePreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    Element beta,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

**15.12.1.23 ScalAndReduce() [1/2]**

```
void FFLAS::Protected::ScalAndReduce (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr X,
    const size_t incX,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]
```

**15.12.1.24 ScalAndReduce() [2/2]**

```
void FFLAS::Protected::ScalAndReduce (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]
```

**15.12.1.25 fsquareCommon()**

```
Field::Element_ptr FFLAS::Protected::fsquareCommon (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

**15.12.1.26 fgemv\_convert()**

```
Field::Element_ptr FFLAS::Protected::fgemv_convert (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

**15.12.1.27 fger\_convert()**

```
void FFLAS::Protected::fger_convert (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

**15.12.1.28 min\_types() [1/7]**

```
size_t FFLAS::Protected::min_types (
    const DFE & k ) [inline]
```

**15.12.1.29 min\_types() [2/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 6 > & k ) [inline]
```

**15.12.1.30 min\_types() [3/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 7 > & k ) [inline]
```

**15.12.1.31 min\_types() [4/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 8 > & k ) [inline]
```

**15.12.1.32 min\_types() [5/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 9 > & k ) [inline]
```

**15.12.1.33 min\_types() [6/7]**

```
size_t FFLAS::Protected::min_types (
    const RecInt::rint< 10 > & k ) [inline]
```

**15.12.1.34 min\_types() [7/7]**

```
size_t FFLAS::Protected::min_types (
    const Givaro::Integer & k ) [inline]
```

**15.12.1.35 unfit() [1/4]**

```
bool FFLAS::Protected::unfit (
    T x ) [inline]
```

**15.12.1.36 unfit() [2/4]**

```
bool FFLAS::Protected::unfit (
    int64_t x ) [inline]
```

**15.12.1.37 unfit() [3/4]**

```
bool FFLAS::Protected::unfit (
    RecInt::rint< K > x ) [inline]
```

**15.12.1.38 unfit()** [4/4]

```
bool FFLAS::Protected::unfit (
    RecInt::rint< 6 > x ) [inline]
```

**15.12.1.39 igemm\_colmajor()** [1/2]

```
void igemm_colmajor (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

**15.12.1.40 igemm\_colmajor()** [2/2]

```
void igemm_colmajor (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

**15.12.1.41 igemm()**

```
void igemm (
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_TRANSPOSE TransB,
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    const int64_t beta,
    int64_t * C,
    size_t ldc ) [inline]
```

**Todo** use primitive (no [Field\(\)](#)) and specialise for int64.

**Todo** use primitive (no [Field\(\)](#)) and specialise for int64.

#### 15.12.1.42 MatF2MatD\_Triangular()

```
void FFLAS::Protected::MatF2MatD_Triangular (
    const Field & F,
    Givaro::DoubleDomain::Element\_ptr S,
    const size_t lds,
    typename Field::ConstElement\_ptr const E,
    const size_t lde,
    const size_t m,
    const size_t n )
```

#### 15.12.1.43 MatF2MatFI\_Triangular()

```
void FFLAS::Protected::MatF2MatFI_Triangular (
    const Field & F,
    Givaro::FloatDomain::Element\_ptr S,
    const size_t lds,
    typename Field::ConstElement\_ptr const E,
    const size_t lde,
    const size_t m,
    const size_t n )
```

**Todo** do finit(...,FFLAS\_TRANS,FFLAS\_DIAG)  
do fconvert(...,FFLAS\_TRANS,FFLAS\_DIAG)

## 15.13 FFLAS::sell\_details Namespace Reference

### Data Structures

- struct [Info](#)
- struct [Coo](#)



## 15.14 FFLAS::sparse\_details Namespace Reference

### Functions

- `template<class Field >`  
`void init_y (const Field &F, const size_t m, const typename Field::Element b, typename Field::Element_ptr y)`
- `template<class Field >`  
`void init_y (const Field &F, const size_t m, const size_t n, const typename Field::Element b, typename Field::Element_ptr y, const int ldy)`
- `template<class Field , class SM , class FC , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FC fc, MZO mzo)`
- `template<class Field , class SM , class FC , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FC fc, MZO mzo)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`

- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloa`  
`>::value||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag`  
`>::value >::type fspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr`  
`x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx,`  
`typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy,`  
`FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy,`  
`FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy,`  
`FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy,`  
`FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx,`  
`typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy,`  
`FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy,`  
`FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx,`  
`typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloa`  
`>::value)||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag`  
`>::value)>::type pfspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename`  
`Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloa`  
`>::value||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag`  
`>::value >::type pfspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename`  
`Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx,`  
`typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const`  
`SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy,`  
`FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const`

- SM &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::GenericTag, std::false_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, std::false_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, std::false_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::GenericTag, std::true_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, std::true_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, std::true_type)`
  - `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support\_simd< typename Field::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support\_simd< typename Field::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support\_simd< typename Field::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`

### 15.14.1 Function Documentation

**15.14.1.1 init\_y() [1/2]**

```
void FFLAS::sparse_details::init_y (
    const Field & F,
    const size_t m,
    const typename Field::Element b,
    typename Field::Element_ptr y ) [inline]
```

**15.14.1.2 init\_y() [2/2]**

```
void FFLAS::sparse_details::init_y (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element b,
    typename Field::Element_ptr y,
    const int ldy ) [inline]
```

**15.14.1.3 fspmv\_dispatch() [1/2]**

```
std::enable_if< !(std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1>::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1>::value)>::type FFLAS::sparse_details::fspmv_dispatch (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FC fC,
    MZO mzo ) [inline]
```

**15.14.1.4 fspmv\_dispatch() [2/2]**

```
std::enable_if< std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1>::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1>::value>::type FFLAS::sparse_details::fspmv_dispatch (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FC fC,
    MZO mzo ) [inline]
```

**15.14.1.5 fspmv()** [1/12]

```
void FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.6 fspmv()** [2/12]

```
std::enable_if<!isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse_details←
::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.7 fspmv()** [3/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse_details::fspmv
(
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.8 fspmv()** [4/12]

```
std::enable_if<!isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse_details←
::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.9 fspmv()** [5/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse_details::fspmv
(
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.10 fspmv()** [6/12]

```
void FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.11 fspmv()** [7/12]

```
std::enable_if<!isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse_details↵
::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.12 fspmv()** [8/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value>::type FFLAS::sparse_details::fspmv
(
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.13 fspmv()** [9/12]

```
void FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::true_type ) [inline]
```

**15.14.1.14 fspmm\_dispatch()** [1/2]

```
std::enable_if< !(std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineIntT>::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineIntT>::value)>::type FFLAS::sparse_details::fspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

**15.14.1.15 fspmm\_dispatch()** [2/2]

```
std::enable_if< std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineIntT>::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineIntT>::value>::type FFLAS::sparse_details::fspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

**15.14.1.16 fspmm() [1/9]**

```

void FFLAS::sparse_details::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.17 fspmm() [2/9]**

```

std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.18 fspmm() [3/9]**

```

std::enable_if<!support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```



**15.14.1.19 fspmm()** [4/9]

```
std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.20 fspmm()** [5/9]

```
std::enable_if<!support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.21 fspmm()** [6/9]

```
void FFLAS::sparse_details::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.22 fspmm() [7/9]**

```
std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.23 fspmm() [8/9]**

```
std::enable_if<!support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.24 fspmm() [9/9]**

```
void FFLAS::sparse_details::fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.25 pfspmm\_dispatch() [1/2]**

```
std::enable_if< !(std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value)>::type FFLAS::sparse_details::pfspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

**15.14.1.26 pfspmm\_dispatch() [2/2]**

```
std::enable_if< std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value || std::is_same<typename ElementTraits<typename Field::Element>::value, ElementCategories::MachineInt1
::value)>::type FFLAS::sparse_details::pfspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

**15.14.1.27 pfspmm() [1/9]**

```
void FFLAS::sparse_details::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.28 pfspmm()** [2/9]

```
std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.29 pfspmm()** [3/9]

```
std::enable_if<!support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.30 pfspmm()** [4/9]

```
std::enable_if<support_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.31 pfspmm()** [5/9]

```
std::enable_if<!support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.32 pfspmm()** [6/9]

```
void FFLAS::sparse_details::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.33 pfspmm()** [7/9]

```
std::enable_if<support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details←
::pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.34 pfsppmm()** [8/9]

```
std::enable_if<!support\_simd<typename Field::Element>::value>::type FFLAS::sparse_details↵
::pfsppmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.35 pfsppmm()** [9/9]

```
void FFLAS::sparse_details::pfsppmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement\_ptr x,
    int ldx,
    typename Field::Element\_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.36 pfsppmv()** [1/6]

```
void FFLAS::sparse_details::pfsppmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement\_ptr x,
    typename Field::Element\_ptr y,
    FieldCategories::GenericTag ,
    std::false_type ) [inline]
```

**15.14.1.37 pfsppmv()** [2/6]

```
void FFLAS::sparse_details::pfsppmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement\_ptr x,
    typename Field::Element\_ptr y,
    FieldCategories::UnparametricTag ,
    std::false_type ) [inline]
```

**15.14.1.38 pfspmv()** [3/6]

```
void FFLAS::sparse_details::pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::false_type ) [inline]
```

**15.14.1.39 pfspmv()** [4/6]

```
void FFLAS::sparse_details::pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    std::true_type ) [inline]
```

**15.14.1.40 pfspmv()** [5/6]

```
void FFLAS::sparse_details::pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    std::true_type ) [inline]
```

**15.14.1.41 pfspmv()** [6/6]

```
void FFLAS::sparse_details::pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::true_type ) [inline]
```

**15.14.1.42 fspmv()** [10/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value && support_simd<typename Field::Element>>↵
::value >::type FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.43 fspmv()** [11/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value && support_simd<typename Field::Element>>↵
::value >::type FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.44 fspmv()** [12/12]

```
std::enable_if<isSparseMatrixSimdFormat<Field, SM>::value && support_simd<typename Field::Element>>↵
::value >::type FFLAS::sparse_details::fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.15 FFLAS::sparse\_details\_impl Namespace Reference****Functions**

- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::GenericTag)
- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::UnparametricTag)
- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, const int64\_t kmax)



- `template<class Field >`  
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`

- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t blockSize, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`



- `template<class Field , class Func >`  
`void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x, typename Field::Element_ptr y, Func &&func)`
- `template<class Field , class Func >`  
`void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`



- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`

- [illegible]





- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 15.15.1 Function Documentation

### 15.15.1.1 fspmm() [1/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

### 15.15.1.2 fspmm() [2/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

### 15.15.1.3 fspmm() [3/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.15.1.4 fspmm\_simd\_aligned()** [1/2]

```
void FFLAS::sparse_details_impl::fspmm_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldY,
    const int64_t kmax ) [inline]
```

**15.15.1.5 fspmm\_simd\_unaligned()** [1/2]

```
void FFLAS::sparse_details_impl::fspmm_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldY,
    const int64_t kmax ) [inline]
```

**15.15.1.6 fspmm\_one()** [1/4]

```
void FFLAS::sparse_details_impl::fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldY,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.7 fspmm\_mone()** [1/4]

```
void FFLAS::sparse_details_impl::fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldY,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.8 fspmm\_one\_simd\_aligned()** [1/3]

```
void FFLAS::sparse_details_impl::fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.9 fspmm\_one\_simd\_unaligned()** [1/3]

```
void FFLAS::sparse_details_impl::fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.10 fspmm\_mone\_simd\_aligned()** [1/3]

```
void FFLAS::sparse_details_impl::fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.11 fspmm\_mone\_simd\_unaligned()** [1/3]

```
void FFLAS::sparse_details_impl::fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.12 fspmv()** [1/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.13 fspmv()** [2/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.14 fspmv()** [3/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.15 fspmv\_one()** [1/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.16 fspmv\_mone()** [1/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.17 fspmv\_one()** [2/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.18 fspmv\_mone()** [2/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.19 pfspmm()** [1/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.20 pfspmm()** [2/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.21 pfsppmm()** [3/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.15.1.22 pfsppmm\_one()** [1/2]

```
void FFLAS::sparse_details_impl::pfsppmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.23 pfsppmm\_mone()** [1/2]

```
void FFLAS::sparse_details_impl::pfsppmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.24 pfsppmm\_one()** [2/2]

```
void FFLAS::sparse_details_impl::pfsppmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.25 pfspmm\_mone()** [2/2]

```

void FFLAS::sparse_details_impl::pfspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.26 pfspmv()** [1/18]

```

void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.27 pfspmv\_task()**

```

void FFLAS::sparse_details_impl::pfspmv_task (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const index_t iStart,
    const index_t iStop,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.28 pfspmv()** [2/18]

```

void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.29 pfspmv()** [3/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.15.1.30 pfspmv\_one()** [1/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.31 pfspmv\_mone()** [1/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.32 pfspmv\_one()** [2/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.33 pfspmv\_mone()** [2/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```



**15.15.134 fspmm()** [4/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.135 fspmm()** [5/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    index_t blockSize,
    typename Field::ConstElement_ptr x_,
    index_t ldx,
    typename Field::Element_ptr y_,
    index_t ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.136 fspmm\_simd\_aligned()** [2/2]

```
void FFLAS::sparse_details_impl::fspmm_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.137 fspmm\_simd\_unaligned()** [2/2]

```
void FFLAS::sparse_details_impl::fspmm_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.38 fspmm()** [6/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.15.1.39 fspmm\_one()** [2/4]

```
void FFLAS::sparse_details_impl::fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.40 fspmm\_mone()** [2/4]

```
void FFLAS::sparse_details_impl::fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.41 fspmm\_one\_simd\_aligned()** [2/3]

```
void FFLAS::sparse_details_impl::fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.142 fspmm\_one\_simd\_unaligned()** [2/3]

```
void FFLAS::sparse_details_impl::fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.143 fspmm\_mone\_simd\_aligned()** [2/3]

```
void FFLAS::sparse_details_impl::fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.144 fspmm\_mone\_simd\_unaligned()** [2/3]

```
void FFLAS::sparse_details_impl::fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.145 fspmv()** [4/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.46 fspmv()** [5/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.47 fspmv()** [6/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.15.1.48 fspmv\_one()** [3/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.49 fspmv\_mone()** [3/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.50 fspmv\_one()** [4/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.151 fspmv\_mone()** [4/10]

```

void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.152 pfspmm()** [4/18]

```

void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]

```

**15.15.153 pfspmm()** [5/18]

```

void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.154 pfspmm()** [6/18]

```

void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.55 pfsppmm()** [7/18]

```

void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.56 pfsppmm()** [8/18]

```

void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    const int64_t kmax ) [inline]

```

**15.15.1.57 pfsppmm()** [9/18]

```

void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.58 pfsppmv()** [4/18]

```

void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.59 pfspmv()** [5/18]

```

void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.60 pfspmv()** [6/18]

```

void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]

```

**15.15.1.61 fspmm()** [7/15]

```

void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.62 fspmm()** [8/15]

```

void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.63 fspmm()** [9/15]

```

void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.64 fspmv()** [7/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.65 fspmv()** [8/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.66 fspmv()** [9/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]

```



**15.15.1.67 pfspmm()** [10/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.68 pfspmm()** [11/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.69 pfspmm()** [12/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.70 pfspmm()** [13/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.71 pfsppmm()** [14/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    const int64_t kmax ) [inline]
```

**15.15.1.72 pfsppmm()** [15/18]

```
void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.15.1.73 pfsppmm\_zo()** [1/2]

```
void FFLAS::sparse_details_impl::pfsppmm_zo (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    Func && func ) [inline]
```

**15.15.1.74 pfsppmm\_zo()** [2/2]

```
void FFLAS::sparse_details_impl::pfsppmm_zo (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    Func && func ) [inline]
```

**15.15.1.75 pfspmv()** [7/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.76 pfspmv()** [8/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.77 pfspmv()** [9/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.15.1.78 pfspmv\_one()** [3/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.79 pfspmv\_mone()** [3/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.80 pfspmv\_one() [4/8]**

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.81 pfspmv\_mone() [4/8]**

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.82 fspmm() [10/15]**

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.83 fspmm() [11/15]**

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.84 fspmm()** [12/15]

```
void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.15.1.85 fspmm\_mone()** [3/4]

```
void FFLAS::sparse_details_impl::fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.86 fspmm\_one()** [3/4]

```
void FFLAS::sparse_details_impl::fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.87 fspmm\_mone()** [4/4]

```
void FFLAS::sparse_details_impl::fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.88 fspmm\_one() [4/4]**

```

void FFLAS::sparse_details_impl::fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.89 fspmm\_one\_simd\_aligned() [3/3]**

```

void FFLAS::sparse_details_impl::fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.90 fspmm\_one\_simd\_unaligned() [3/3]**

```

void FFLAS::sparse_details_impl::fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.91 fspmm\_mone\_simd\_aligned() [3/3]**

```

void FFLAS::sparse_details_impl::fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.92 fspmm\_mone\_simd\_unaligned()** [3/3]

```

void FFLAS::sparse_details_impl::fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.93 fspmv()** [10/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.94 fspmv()** [11/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.95 fspmv()** [12/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]

```

**15.15.1.96 fspmv\_one()** [5/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.97 fspmv\_mone()** [5/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.98 fspmv\_one()** [6/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.99 fspmv\_mone()** [6/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.100 pfspmv()** [10/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```



**15.15.1.101 pfspmv()** [11/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.102 pfspmv()** [12/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.103 pfspmv\_one()** [5/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.104 pfspmv\_mone()** [5/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.105 pfspmv\_one()** [6/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.106 pfspmv\_mone()** [6/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.107 fspmv()** [13/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.108 fspmv\_simd()** [1/4]

```
void FFLAS::sparse_details_impl::fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.109 fspmv()** [14/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.110 fspmv\_simd()** [2/4]

```
void FFLAS::sparse_details_impl::fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.111 fspmv()** [15/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.112 fspmv\_one()** [7/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.113 fspmv\_mone()** [7/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.114 fspmv\_one()** [8/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.115 fspmv\_mone()** [8/10]

```
void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.116 fspmv\_one\_simd()** [1/2]

```
void FFLAS::sparse_details_impl::fspmv_one_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.117 fspmv\_mone\_simd()** [1/2]

```
void FFLAS::sparse_details_impl::fspmv_mone_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.118 pfspmm()** [16/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.119 pfspmm()** [17/18]

```
void FFLAS::sparse_details_impl::pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.120 pfsppmm()** [18/18]

```

void FFLAS::sparse_details_impl::pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    uint64_t kmax ) [inline]

```

**15.15.1.121 pfsppmv()** [13/18]

```

void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.122 pfsppmv()** [14/18]

```

void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.123 pfsppmv()** [15/18]

```

void FFLAS::sparse_details_impl::pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    uint64_t kmax ) [inline]

```

**15.15.1.124 fspmm()** [13/15]

```

void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.125 fspmm()** [14/15]

```

void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.126 fspmm()** [15/15]

```

void FFLAS::sparse_details_impl::fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    uint64_t kmax ) [inline]

```

**15.15.1.127 fspmv()** [16/21]

```

void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.128 fspmv()** [17/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.129 fspmv()** [18/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    uint64_t kmax ) [inline]
```

**15.15.1.130 pfspmv()** [16/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.131 pfspmv()** [17/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.132 pfspmv()** [18/18]

```
void FFLAS::sparse_details_impl::pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.15.1.133 pfspmv\_one()** [7/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.134 pfspmv\_mone()** [7/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.135 pfspmv\_one()** [8/8]

```
void FFLAS::sparse_details_impl::pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.136 pfspmv\_mone()** [8/8]

```
void FFLAS::sparse_details_impl::pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.137 fspmv()** [19/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```



**15.15.1.138 fspmv\_simd()** [3/4]

```
void FFLAS::sparse_details_impl::fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.139 fspmv()** [20/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.140 fspmv\_simd()** [4/4]

```
void FFLAS::sparse_details_impl::fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.141 fspmv()** [21/21]

```
void FFLAS::sparse_details_impl::fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.142 fspmv\_one()** [9/10]

```
void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.143 fspmv\_mone()** [9/10]

```

void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.144 fspmv\_one\_simd()** [2/2]

```

void FFLAS::sparse_details_impl::fspmv_one_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.145 fspmv\_mone\_simd()** [2/2]

```

void FFLAS::sparse_details_impl::fspmv_mone_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.146 fspmv\_one()** [10/10]

```

void FFLAS::sparse_details_impl::fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.147 fspmv\_mone()** [10/10]

```

void FFLAS::sparse_details_impl::fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

## 15.16 FFLAS::StrategyParameter Namespace Reference

### Data Structures

- struct [Fixed](#)
- struct [Threads](#)
- struct [Grain](#)
- struct [TwoD](#)
- struct [TwoDAdaptive](#)
- struct [ThreeD](#)
- struct [ThreeDInPlace](#)
- struct [ThreeDAdaptive](#)

## 15.17 FFLAS::StructureHelper Namespace Reference

[StructureHelper](#) for ftrsm.

### Data Structures

- struct [Recursive](#)
- struct [Iterative](#)
- struct [Hybrid](#)

### 15.17.1 Detailed Description

[StructureHelper](#) for ftrsm.

## 15.18 FFLAS::vectorised Namespace Reference

### Namespaces

- [unswitch](#)

### Data Structures

- struct [HelperMod](#)
- struct [HelperMod](#)< Field, ElementCategories::MachineIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::MachineFloatTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::FixedPrecIntTag >

## Functions

- `template<class SimdT , class Element , bool positive>`  
`std::enable_if< is\_simd< SimdT >::value, void >::type VEC\_ADD (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)`
- `template<bool positive, class Element , class T1 , class T2 >`  
`std::enable_if< FFLAS::support\_simd\_add< Element >::value, void >::type addp (Element *T, const Element *TA, const Element *TB, size_t n, Element p, T1 min_, T2 max_)`
- `template<class SimdT , class Element , bool positive>`  
`std::enable_if< is\_simd< SimdT >::value, void >::type VEC\_SUB (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)`
- `template<bool positive, class Element , class T1 , class T2 >`  
`std::enable_if< FFLAS::support\_simd\_add< Element >::value, void >::type subp (Element *T, const Element *TA, const Element *TB, const size_t n, const Element p, const T1 min_, const T2 max_)`
- `template<class Element >`  
`std::enable_if< FFLAS::support\_simd\_add< Element >::value, void >::type add (Element *T, const Element *TA, const Element *TB, size_t n)`
- `template<class Element >`  
`std::enable_if< FFLAS::support\_simd\_add< Element >::value, void >::type sub (Element *T, const Element *TA, const Element *TB, size_t n)`
- `template<class T >`  
`std::enable_if< ! std::is_integral< T >::value, T >::type reduce (T A, T B)`
- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, T >::type reduce (T A, T B)`
- `template<> Givaro::Integer reduce (Givaro::Integer A, Givaro::Integer B)`
- `float reduce (float A, float B, float invB, float min, float max)`
- `double reduce (double A, double B, double invB, double min, double max)`
- `int64\_t reduce (int64_t A, int64_t p, double invp, double min, double max, int64\_t pow50rem)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineIntTag > &H)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineFloatTag > &H)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size_t &n, typename Field::Element\_ptr T)`
- `template<class Field >`  
`std::enable_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size_t &n, const size_t &incX, typename Field::Element\_ptr T)`
- `template<class Field >`  
`std::enable_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type scalp (const Field &F, typename Field::Element\_ptr T, const typename Field::Element alpha, typename Field::ConstElement\_ptr U, const size_t n)`
- `template<class Field >`  
`std::enable_if< FFLAS::support\_fast\_mod< typename Field::Element >::value, void >::type scalp (const Field &F, typename Field::Element\_ptr T, const typename Field::Element alpha, typename Field::ConstElement\_ptr U, const size_t n, const size_t &incX)`

### 15.18.1 Function Documentation

### 15.18.1.1 VEC\_ADD()

```
std::enable_if<is_simd<SimdT>::value, void>::type FFLAS::vectorised::VEC_ADD (
    SimdT & C,
    SimdT & A,
    SimdT & B,
    SimdT & Q,
    SimdT & T,
    SimdT & P,
    SimdT & NEGP,
    SimdT & MIN,
    SimdT & MAX ) [inline]
```

### 15.18.1.2 addp()

```
std::enable_if<FFLAS::support_simd_add<Element>::value, void>::type FFLAS::vectorised::addp (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n,
    Element p,
    T1 min_,
    T2 max_ ) [inline]
```

### 15.18.1.3 VEC\_SUB()

```
std::enable_if<is_simd<SimdT>::value, void>::type FFLAS::vectorised::VEC_SUB (
    SimdT & C,
    SimdT & A,
    SimdT & B,
    SimdT & Q,
    SimdT & T,
    SimdT & P,
    SimdT & NEGP,
    SimdT & MIN,
    SimdT & MAX ) [inline]
```

### 15.18.1.4 subp()

```
std::enable_if<FFLAS::support_simd_add<Element>::value, void>::type FFLAS::vectorised::subp (
    Element * T,
    const Element * TA,
    const Element * TB,
    const size_t n,
    const Element p,
    const T1 min_,
    const T2 max_ ) [inline]
```

**15.18.1.5 add()**

```
std::enable_if<FFLAS::support_simd_add<Element>::value, void>::type FFLAS::vectorised::add (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n ) [inline]
```

**15.18.1.6 sub()**

```
std::enable_if<FFLAS::support_simd_add<Element>::value, void>::type FFLAS::vectorised::sub (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n ) [inline]
```

**15.18.1.7 reduce() [1/9]**

```
std::enable_if< ! std::is_integral<T>::value, T>::type FFLAS::vectorised::reduce (
    T A,
    T B ) [inline]
```

**15.18.1.8 reduce() [2/9]**

```
std::enable_if< std::is_integral<T>::value, T>::type FFLAS::vectorised::reduce (
    T A,
    T B ) [inline]
```

**15.18.1.9 reduce() [3/9]**

```
Givaro::Integer FFLAS::vectorised::reduce (
    Givaro::Integer A,
    Givaro::Integer B ) [inline]
```

**15.18.1.10 reduce() [4/9]**

```
float FFLAS::vectorised::reduce (
    float A,
    float B,
    float invB,
    float min,
    float max ) [inline]
```

**15.18.1.11 reduce() [5/9]**

```
double FFLAS::vectorised::reduce (
    double A,
    double B,
    double invB,
    double min,
    double max ) [inline]
```

**15.18.1.12 reduce() [6/9]**

```
int64_t FFLAS::vectorised::reduce (
    int64_t A,
    int64_t p,
    double invp,
    double min,
    double max,
    int64_t pow50rem ) [inline]
```

**15.18.1.13 reduce() [7/9]**

```
Field::Element FFLAS::vectorised::reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::MachineIntTag > & H ) [inline]
```

**15.18.1.14 reduce() [8/9]**

```
Field::Element FFLAS::vectorised::reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::MachineFloatTag > & H ) [inline]
```

**15.18.1.15 reduce() [9/9]**

```
Field::Element FFLAS::vectorised::reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > & H ) [inline]
```

**15.18.1.16 modp() [1/2]**

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    typename Field::Element_ptr T ) [inline]
```

**15.18.1.17 modp() [2/2]**

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    const size_t & incX,
    typename Field::Element_ptr T ) [inline]
```

**15.18.1.18 scalp() [1/2]**

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n ) [inline]
```

**15.18.1.19 scalp() [2/2]**

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔
::vectorised::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX ) [inline]
```



## 15.19 FFLAS::vectorised::unswitch Namespace Reference

### Functions

- `template<class Field >`  
`std::enable_if<!FFLAS::support_simd_mod< typename Field::Element >::value &&FFLAS::support_fast_mod<`  
`typename Field::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U,`  
`const size_t &n, typename Field::Element_ptr T, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type modp`  
`(const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename`  
`Field::Element_ptr T, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if<!FFLAS::support_simd_mod< typename Field::Element >::value &&FFLAS::support_fast_mod<`  
`typename Field::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const`  
`typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, HelperMod< Field >`  
`&H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX, HelperMod< Field > &H)`

### 15.19.1 Function Documentation

#### 15.19.1.1 modp() [1/2]

```
std::enable_if<!FFLAS::support_simd_mod<typename Field::Element>::value && FFLAS::support_fast_mod<typename
Field::Element>::value, void>::type FFLAS::vectorised::unswitch::modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    typename Field::Element_ptr T,
    HelperMod< Field > & H ) [inline]
```

#### 15.19.1.2 modp() [2/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↵
::vectorised::unswitch::modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    const size_t & incX,
    typename Field::Element_ptr T,
    HelperMod< Field > & H ) [inline]
```

### 15.19.1.3 `scalp()` [1/2]

```
std::enable_if<!FFLAS::support_simd_mod<typename Field::Element>::value && FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS::vectorised::unswitch::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    HelperMod< Field > & H ) [inline]
```

### 15.19.1.4 `scalp()` [2/2]

```
std::enable_if<FFLAS::support_fast_mod<typename Field::Element>::value, void>::type FFLAS↔::vectorised::unswitch::scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX,
    HelperMod< Field > & H ) [inline]
```

## 15.20 FFPACK Namespace Reference

Finite Field **PACK** Set of elimination based routines for dense linear algebra.

### Namespaces

- [Protected](#)

### Data Structures

- class [CheckerImplem\\_charpoly](#)
- class [CheckerImplem\\_Det](#)
- class [CheckerImplem\\_invert](#)
- class [CheckerImplem\\_PLUQ](#)
- class [CharpolyFailed](#)
- class [callLUdivine\\_small](#)
- class [callLUdivine\\_small< double >](#)
- class [callLUdivine\\_small< float >](#)
- class [RNSInteger](#)
- class [RNSIntegerMod](#)
- struct [rns\\_double\\_elt](#)
- struct [rns\\_double\\_elt\\_ptr](#)
- struct [rns\\_double\\_elt\\_cstptr](#)
- struct [rns\\_double](#)
- struct [rns\\_double\\_extended](#)
- class [rnsRandIter](#)
- class [Failure](#)

*A precondition failed.*

## Typedefs

- template<class Field >  
using [Checker\\_PLUQ](#) = FFLAS::Checker\_Empty< Field >
- template<class Field >  
using [Checker\\_Det](#) = FFLAS::Checker\_Empty< Field >
- template<class Field >  
using [Checker\\_invert](#) = FFLAS::Checker\_Empty< Field >
- template<class Field , class Polynomial >  
using [Checker\\_charpoly](#) = FFLAS::Checker\_Empty< Field >
- template<class Field >  
using [ForceCheck\\_PLUQ](#) = CheckerImplem\_PLUQ< Field >
- template<class Field >  
using [ForceCheck\\_Det](#) = CheckerImplem\_Det< Field >
- template<class Field >  
using [ForceCheck\\_invert](#) = CheckerImplem\_invert< Field >
- template<class Field , class Polynomial >  
using [ForceCheck\\_charpoly](#) = CheckerImplem\_charpoly< Field, Polynomial >

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)  
*Conversion of a permutation from LAPACK format to Math format.*
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)  
*Conversion of a permutation from Maths format to LAPACK format.*
- template<class Field >  
void [applyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- template<class Field >  
void [applyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const FFLAS::ParSeqHelper::Sequential seq)
- template<class Field , class Cut , class Param >  
void [applyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field >  
void [MonotonicApplyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t R)  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- template<class Field >  
void [fgetrs](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*
- template<class Field >  
[Field::Element\\_ptr fgetrs](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr X, const size\_t ldx, typename Field::ConstElement\_ptr B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*

- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Square system solver.*
- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Rectangular system solver.*
- `template<class Field >`  
`void ftrtri (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FTRTRI_THRESHOLD)`  
*Compute the inverse of a triangular matrix.*
- `template<class Field >`  
`void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldl, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field >`  
`void ftrtrm (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda)`  
*Compute the product of two triangular matrices of opposite shape.*
- `template<class Field >`  
`void ftrstr (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const FFLAS::FFLAS_DIAG diagB, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSTR_THRESHOLD)`  
*Solve a triangular system with a triangular right hand side of the same shape.*
- `template<class Field >`  
`void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldx, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)`  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`  
*Compute a PLUQ factorization of the given matrix.*
- `template<class Field >`  
`size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`

- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, type-`  
`name Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential`  
`&PSHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut,`  
`Param > &PSHelper)`
- `template<class Field >`  
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans,`  
`const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt,`  
`const FFPACK_LU_TAG LuTag=FfpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE↵`  
`_THRESHOLD)`  

*Compute the CUP or PLE factorization of the given matrix.*
- `template<class Field >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`  
`A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG Lu↵`  
`Tag=FfpackSlabRecursive)`  

*Compute the Column Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG`  
`LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper`  
`&psH)`
- `template<class Field >`  
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`  
`A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG Lu↵`  
`Tag=FfpackSlabRecursive)`  

*Compute the Row Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_↵`  
`LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper`  
`&psH)`
- `template<class Field >`  
`size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_↵`  
`LU_TAG LuTag=FfpackSlabRecursive)`  

*Compute the Reduced Column Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0,`  
`const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU↵`  
`_TAG LuTag, const PSHelper &psH)`
- `template<class Field >`  
`size_t ReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`  
`A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG Lu↵`  
`Tag=FfpackSlabRecursive)`

*Compute the Reduced Row Echelon form of the input matrix in-place.*

- template<class Field >  
size\_t **pReducedRowEchelonForm** (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=FpackTileRecursive)
- template<class Field , class PSHelper >  
size\_t **ReducedRowEchelonForm** (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class Field >  
**Field::Element\_ptr Invert** (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, int &>nullity)  
*Invert the given matrix in place or computes its nullity if it is singular.*
- template<class Field >  
**Field::Element\_ptr Invert** (const Field &F, const size\_t M, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, int &>nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class Field >  
**Field::Element\_ptr Invert2** (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, int &>nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class PolRing >  
std::list< typename PolRing::Element > & **CharPoly** (const PolRing &R, std::list< typename PolRing::Element > &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, typename PolRing::Domain\_t::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag=FpackAuto, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)  
*Compute the characteristic polynomial of the matrix A.*
- template<class PolRing >  
PolRing::Element & **CharPoly** (const PolRing &R, typename PolRing::Element &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, typename PolRing::Domain\_t::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag=FpackAuto, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)  
*Compute the characteristic polynomial of the matrix A.*
- template<class PolRing >  
PolRing::Element & **CharPoly** (const PolRing &R, typename PolRing::Element &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, const FFPACK\_CHARPOLY\_TAG CharpTag=FpackAuto, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)  
*Compute the characteristic polynomial of the matrix A.*
- template<class Field , class Polynomial >  
Polynomial & **MinPoly** (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda)  
*Compute the minimal polynomial of the matrix A.*
- template<class Field , class Polynomial , class RandIter >  
Polynomial & **MinPoly** (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, RandIter &G)  
*Compute the minimal polynomial of the matrix A.*
- template<class Field , class Polynomial >  
Polynomial & **MatVecMinPoly** (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr v, const size\_t incv)  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- template<class Field >  
size\_t **Rank** (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)  
*Computes the rank of the given matrix using a PLUQ factorization.*
- template<class Field >  
size\_t **pRank** (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t numthreads=0)

- `template<class Field, class PSHelper>`  
`size_t Rank` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, const `PSHelper` &psH)
- `template<class Field>`  
`bool IsSingular` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda)  
*Returns true if the given matrix is singular.*
- `template<class Field>`  
`Field::Element & Det` (const `Field` &F, typename `Field::Element` &det, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P=NULL, `size_t` \*Q=NULL)  
*Returns the determinant of the given square matrix.*
- `template<class Field>`  
`Field::Element & pDet` (const `Field` &F, typename `Field::Element` &det, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` numthreads=0, `size_t` \*P=NULL, `size_t` \*Q=NULL)
- `template<class Field, class PSHelper>`  
`Field::Element & Det` (const `Field` &F, typename `Field::Element` &det, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, const `PSHelper` &psH, `size_t` \*P=NULL, `size_t` \*Q=NULL)
- `template<class Field>`  
`Field::Element_ptr Solve` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::Element_ptr` x, const `int` incx, typename `Field::ConstElement_ptr` b, const `int` incb)  
*Solves a linear system  $AX = b$  using PLUQ factorization.*
- `template<class Field, class PSHelper>`  
`Field::Element_ptr Solve` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::Element_ptr` x, const `int` incx, typename `Field::ConstElement_ptr` b, const `int` incb, `PSHelper` &psH)
- `template<class Field>`  
`Field::Element_ptr pSolve` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::Element_ptr` x, const `int` incx, typename `Field::ConstElement_ptr` b, const `int` incb, `size_t` numthreads=0)
- `template<class Field>`  
`*void RandomNullSpaceVector` (const `Field` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::Element_ptr` X, const `size_t` incX)  
*Solve  $LX = B$  or  $XL = B$  in place.*
- `template<class Field>`  
`size_t NullSpaceBasis` (const `Field` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::Element_ptr` &NS, `size_t` &ldn, `size_t` &NSdim)  
*Computes a basis of the Left/Right nullspace of the matrix A.*
- `template<class Field>`  
`size_t RowRankProfile` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag=`FfpackSlabRecursive`)  
*Computes the row rank profile of A.*
- `template<class Field>`  
`size_t pRowRankProfile` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*&rkprofile, `size_t` numthreads=0, const `FFPACK_LU_TAG` LuTag=`FfpackTileRecursive`)
- `template<class Field, class PSHelper>`  
`size_t RowRankProfile` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag, `PSHelper` &psH)
- `template<class Field>`  
`size_t ColumnRankProfile` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag=`FfpackSlabRecursive`)  
*Computes the column rank profile of A.*
- `template<class Field>`  
`size_t pColumnRankProfile` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*&rkprofile, `size_t` numthreads=0, const `FFPACK_LU_TAG` LuTag=`FfpackTileRecursive`)



- `template<class Field, class PSHelper>`  
`size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)`
- `void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const FFPACK_LU_TAG LuTag)`  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- `size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)`  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*
- `template<class Field>`  
`size_t RowRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rowindices, size_t *colindices, size_t &R)`  
*RowRankProfileSubmatrixIndices.*
- `template<class Field>`  
`size_t ColRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rowindices, size_t *colindices, size_t &R)`  
*Computes the indices of the submatrix  $r \times r$   $X$  of  $A$  whose columns correspond to the column rank profile of  $A$ .*
- `template<class Field>`  
`size_t RowRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)`  
*Computes the  $r \times r$  submatrix  $X$  of  $A$ , by picking the row rank profile rows of  $A$ .*
- `template<class Field>`  
`size_t ColRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)`  
*Compute the  $r \times r$  submatrix  $X$  of  $A$ , by picking the row rank profile rows of  $A$ .*
- `template<class Field>`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false)`  
*Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank  $R$ .*
- `template<class Field>`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda)`  
*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank  $R$ .*
- `template<class Field>`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `template<class Field>`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .*
- `template<class Field>`  
`void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*



- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.*
- `template<class Field >`  
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `RowEchelonForm` or `ColumnEchelonForm`.*
- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- `template<class Field >`  
`Field::Element_ptr LQUptInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X, const size_t ldx)`  
*LQUptInverseOfFullRankMinor.*
- `template<class Field >`  
`void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t incX)`  
*Solve  $LX = B$  or  $XL = B$  in place.*
- `template<class Field >`  
`void solveLB (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field >`  
`void solveLB2 (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field, class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`  
`Field::Element_ptr buildMatrix (const Field &F, typename Field::ConstElement_ptr E, typename Field::ConstElement_ptr C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda, const size_t mu)`
- `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)`
- `template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R, Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp, const size_t N, Givaro::Integer *A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)`
- `template<class PSHelper >`  
`FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (const FFPACK::RNSInteger<`

- `FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr &det, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, const PSHelper &psH)`
- `template<class PSHelper >`  
`Givaro::Integer & Det (const Givaro::ZRing< Givaro::Integer > &F, Givaro::Integer &det, const size_t N, Givaro::Integer *A, const size_t lda, const PSHelper &psH, size_t *P, size_t *Q)`
  - `template<class Field >`  
`bool fsytrf_BC_Crout (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)`
  - `template<class Field >`  
`size_t fsytrf_BC_RL (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)`
  - `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_RL (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
  - `template<class Field >`  
`size_t fsytrf_LOW_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
  - `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
  - `template<class Field >`  
`size_t fsytrf_UP_RPM (const Field &Fi, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P, size_t BCThreshold)`
  - `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Sequential seq, size_t threshold)`
  - `template<class Field, class Cut, class Param >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Parallel< Cut, Param > par, size_t threshold)`
  - `template<class Field >`  
`size_t fsytrf_RPM (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t threshold)`
  - `template<class Field >`  
`void getTridiagonal (const Field &F, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, size_t *P, typename Field::Element_ptr T, const size_t ldt)`
  - `template<class Field >`  
`size_t LUdivine_gauss (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`
  - `template<class Field >`  
`size_t LUdivine_small (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`
  - `template<class Field >`  
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)`
  - `template<> size_t LUdivine (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)`
  - `template<class Field >`  
`void MonotonicCompress (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`

- `template<class Field >`  
`void MonotonicCompressMorePivots (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, type-`  
`name Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const`  
`size_t rowstomove, const size_t lenP)`
- `template<class Field >`  
`void MonotonicCompressCycles (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename`  
`Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t lenP)`
- `template<class Field >`  
`void MonotonicExpand (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename`  
`Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t`  
`maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`
- `template<class Field >`  
`void applyP_block (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE`  
`Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda,`  
`const size_t *P)`
- `template<class Field >`  
`void doApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr`  
`tmp, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const`  
`size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t`  
`width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const`  
`FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t`  
`width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const`  
`FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t`  
`R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr`  
`tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const`  
`size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t`  
`width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const`  
`FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t`  
`width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const`  
`FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t`  
`R2, const size_t R3, const size_t R4)`
- `void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times Diag(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- `void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const`  
`size_t N)`  
*Computes  $P1 \times Diag(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $MathP$  as a MathPermutation format.*
- `void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)`

*Computes MathP1 x Diag (I\_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.*

- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- template<class Field >  
void [cyclic\\_shift\\_row\\_col](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
void [cyclic\\_shift\\_row](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<typename T >  
void [cyclic\\_shift\\_row](#) (const RNSIntegerMod< T > &F, typename T::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
void [cyclic\\_shift\\_col](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<typename T >  
void [cyclic\\_shift\\_col](#) (const RNSIntegerMod< T > &F, typename T::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
size\_t [PLUQ\\_basecaseV3](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::Element \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template<class Field >  
size\_t [PLUQ\\_basecaseV2](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::Element \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template<class Field >  
size\_t [PLUQ\\_basecaseCrout](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template<class Field >  
size\_t [\\_PLUQ](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Q, size\_t BCThreshold)
- template<class Cut, class Param >  
size\_t [PLUQ](#) (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Givaro::Integer \*A, const size\_t lda, size\_t \*P, size\_t \*Q, size\_t BCThreshold, FFLAS::ParSeqHelper::Parallel< Cut, Param > &PSHelper)
- template<class Field >  
void [threads\\_fgemm](#) (const size\_t m, const size\_t n, const size\_t r, int nbthreads, size\_t \*W1, size\_t \*W2, size\_t \*W3, size\_t gamma)
- template<class Field >  
void [threads\\_ftrsm](#) (const size\_t m, const size\_t n, int nbthreads, size\_t \*t1, size\_t \*t2)
- template<class Field >  
size\_t [PLUQ](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Q, const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &PSHelper)
- template<> [rns\\_double\\_elt\\_ptr fflas\\_const\\_cast](#) (rns\_double\_elt\_cstptr x)
- template<> [rns\\_double\\_elt\\_cstptr fflas\\_const\\_cast](#) (rns\_double\_elt\_ptr x)
- template<typename Base\_t >  
void [cyclic\\_shift\\_row\\_col](#) (Base\_t \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [cyclic\\_shift\\_row](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, FFLAS\_ELT \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [cyclic\\_shift\\_col](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, FFLAS\_ELT \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [applyP](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P)
- template [INST\\_OR\\_DECL](#) void [fgetrs](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) FFLAS\_ELT \* [fgetrs](#) (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, FFLAS\_ELT \*X, const size\_t ldx, const FFLAS\_ELT \*B, const size\_t ldb, int \*info)

- template [INST\\_OR\\_DECL](#) size\_t [fgesv](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) size\_t [fgesv](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*X, const size\_t ldx, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) void [ftrtri](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t threshold)
- template [INST\\_OR\\_DECL](#) void [trinv\\_left](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t N, const [FFLAS\\_ELT](#) \*L, const size\_t ldl, [FFLAS\\_ELT](#) \*X, const size\_t ldx)
- template [INST\\_OR\\_DECL](#) void [ftrtrm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda)
- template [INST\\_OR\\_DECL](#) size\_t [PLUQ](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template [INST\\_OR\\_DECL](#) size\_t [LUdivine](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const [FFPACK\\_LU\\_TAG](#) LuTag, const size\_t cutoff)
- template [INST\\_OR\\_DECL](#) size\_t [LUdivine\\_small](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) size\_t [LUdivine\\_gauss](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) size\_t [RowEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) size\_t [ReducedRowEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) size\_t [ColumnEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) size\_t [ReducedColumnEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, [FFLAS\\_ELT](#) \*A, const size\_t lda, int &nullity)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*X, const size\_t ldx, int &nullity)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert2](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t M, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*X, const size\_t ldx, int &nullity)
- template [INST\\_OR\\_DECL](#) std::list< [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element > & [CharPoly](#) (const [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, std::list< [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element > &charp, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const size\_t degree)
- template [INST\\_OR\\_DECL](#) [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element & [CharPoly](#) (const [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element &charp, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const size\_t degree)
- template [INST\\_OR\\_DECL](#) [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element & [CharPoly](#) (const [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, [Givaro::Poly1Dom](#)< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element &charp, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const size\_t degree)
- template [INST\\_OR\\_DECL](#) std::vector< [FFLAS\\_ELT](#) > & [MinPoly](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, std::vector< [FFLAS\\_ELT](#) > &minP, const size\_t N, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G)



- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MinPoly` (const `FFLAS_FIELD< FFLAS_ELT > &F`, `std::vector< FFLAS_ELT > &minP`, const `size_t N`, const `FFLAS_ELT *A`, const `size_t lda`)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MatVecMinPoly` (const `FFLAS_FIELD< FFLAS_ELT > &F`, `std::vector< FFLAS_ELT > &minP`, const `size_t N`, const `FFLAS_ELT *A`, const `size_t lda`, const `FFLAS_ELT *V`, const `size_t incv`)
- template `INST_OR_DECL` `size_t KrylovElim` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `size_t *P`, `size_t *Q`, const `size_t deg`, `size_t *iterates`, `size_t *inviterates`, const `size_t maxit`, `size_t virt`)
- template `INST_OR_DECL` `size_t SpecRankProfile` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, const `size_t deg`, `size_t *rankProfile`)
- template `INST_OR_DECL` `size_t Rank` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`)
- template `INST_OR_DECL` `bool IsSingular` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`)
- template `INST_OR_DECL` `FFLAS_ELT & Det` (const `FFLAS_FIELD< FFLAS_ELT > &F`, `FFLAS_ELT &det`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `size_t *P`, `size_t *Q`)
- template `INST_OR_DECL` `FFLAS_ELT & Det` (const `FFLAS_FIELD< FFLAS_ELT > &F`, `FFLAS_ELT &det`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, const `FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &parH`, `size_t *P`, `size_t *Q`)
- template `INST_OR_DECL` `FFLAS_ELT * Solve` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, `FFLAS_ELT *A`, const `size_t lda`, `FFLAS_ELT *x`, const `int incx`, const `FFLAS_ELT *b`, const `int incb`)
- template `INST_OR_DECL` `void solveLB` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `FFLAS::FFLAS_SIDE Side`, const `size_t M`, const `size_t N`, const `size_t R`, `FFLAS_ELT *L`, const `size_t ldl`, const `size_t *Q`, `FFLAS_ELT *B`, const `size_t ldb`)
- template `INST_OR_DECL` `void solveLB2` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `FFLAS::FFLAS_SIDE Side`, const `size_t M`, const `size_t N`, const `size_t R`, `FFLAS_ELT *L`, const `size_t ldl`, const `size_t *Q`, `FFLAS_ELT *B`, const `size_t ldb`)
- template `INST_OR_DECL` `void RandomNullSpaceVector` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `FFLAS::FFLAS_SIDE Side`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `FFLAS_ELT *X`, const `size_t incX`)
- template `INST_OR_DECL` `size_t NullSpaceBasis` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `FFLAS::FFLAS_SIDE Side`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `FFLAS_ELT *NS`, `size_t &ldn`, `size_t &NSdim`)
- template `INST_OR_DECL` `size_t RowRankProfile` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `size_t *rkprofile`, const `FFPACK_LU_TAG LuTag`)
- template `INST_OR_DECL` `size_t ColumnRankProfile` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `size_t *rkprofile`, const `FFPACK_LU_TAG LuTag`)
- template `INST_OR_DECL` `size_t RowRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `size_t *rowindices`, `size_t *colindices`, `size_t &R`)
- template `INST_OR_DECL` `size_t ColRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `size_t *rowindices`, `size_t *colindices`, `size_t &R`)
- template `INST_OR_DECL` `size_t RowRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `FFLAS_ELT *X`, `size_t &R`)
- template `INST_OR_DECL` `size_t ColRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `size_t M`, const `size_t N`, `FFLAS_ELT *A`, const `size_t lda`, `FFLAS_ELT *X`, `size_t &R`)
- template `INST_OR_DECL` `void getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `FFLAS::FFLAS_UPLO Uplo`, const `FFLAS::FFLAS_DIAG diag`, const `size_t M`, const `size_t N`, const `size_t R`, const `FFLAS_ELT *A`, const `size_t lda`, `FFLAS_ELT *T`, const `size_t ldt`, const `bool OnlyNonZeroVectors`)
- template `INST_OR_DECL` `void getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `FFLAS::FFLAS_UPLO Uplo`, const `FFLAS::FFLAS_DIAG diag`, const `size_t M`, const `size_t N`, const `size_t R`, `FFLAS_ELT *A`, const `size_t lda`)
- template `INST_OR_DECL` `void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT > &F`, const `FFLAS::FFLAS_UPLO Uplo`, const `FFLAS::FFLAS_DIAG diag`, const `size_t M`, const `size_t N`, const `size_t R`, const `size_t *P`, const `FFLAS_ELT *A`, const `size_t lda`, `FFLAS_ELT *T`, const `size_t ldt`, const `bool OnlyNonZeroVectors`, const `FFPACK_LU_TAG LuTag`)

- template `INST_OR_DECL` void `getEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, `FFLAS_ELT` \*A, const size\_t Ida, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonTransform`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const `FFLAS_ELT` \*A, const size\_t Ida, `FFLAS_ELT` \*T, const size\_t Idt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const `FFLAS_ELT` \*A, const size\_t Ida, `FFLAS_ELT` \*T, const size\_t Idt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, `FFLAS_ELT` \*A, const size\_t Ida, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonTransform`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const `FFLAS_ELT` \*A, const size\_t Ida, `FFLAS_ELT` \*T, const size\_t Idt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `FFLAS_ELT` \* `LQUPtoInverseOfFullRankMinor` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t rank, `FFLAS_ELT` \*A\_factors, const size\_t Ida, const size\_t \*QtPointer, `FFLAS_ELT` \*X, const size\_t Idx)
- template<class T, class CT = const T>  
T `fflas_const_cast` (CT x)
- `Failure` & `failure` ()
- template<class T >  
bool `isOdd` (const T &a)
- bool `isOdd` (const float &a)
- bool `isOdd` (const double &a)
- template<class Field, class RandIter >  
`Field::Element_ptr` `NonZeroRandomMatrix` (const `Field` &F, size\_t m, size\_t n, typename `Field::Element_ptr` A, size\_t Ida, RandIter &G)  
*Random non-zero Matrix.*
- template<class Field, class RandIter >  
`Field::Element_ptr` `NonZeroRandomMatrix` (const `Field` &F, size\_t m, size\_t n, typename `Field::Element_ptr` A, size\_t Ida)  
*Random non-zero Matrix.*
- template<class Field, class RandIter >  
`Field::Element_ptr` `RandomMatrix` (const `Field` &F, size\_t m, size\_t n, typename `Field::Element_ptr` A, size\_t Ida, RandIter &G)  
*Random Matrix.*
- template<class Field >  
`Field::Element_ptr` `RandomMatrix` (const `Field` &F, size\_t m, size\_t n, typename `Field::Element_ptr` A, size\_t Ida)  
*Random Matrix.*
- template<class Field, class RandIter >  
`Field::Element_ptr` `RandomTriangularMatrix` (const `Field` &F, size\_t m, size\_t n, const `FFLAS::FFLAS_UPLO` UpLo, const `FFLAS::FFLAS_DIAG` Diag, bool nonsingular, typename `Field::Element_ptr` A, size\_t Ida, RandIter &G)  
*Random Triangular Matrix.*
- template<class Field >  
`Field::Element_ptr` `RandomTriangularMatrix` (const `Field` &F, size\_t m, size\_t n, const `FFLAS::FFLAS_UPLO` UpLo, const `FFLAS::FFLAS_DIAG` Diag, bool nonsingular, typename `Field::Element_ptr` A, size\_t Ida)  
*Random Triangular Matrix.*
- size\_t `RandInt` (size\_t a, size\_t b)

- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomSymmetricMatrix` (const `Field` &F, size\_t n, bool nonsingular, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)  
*Random Symmetric Matrix.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomMatrixWithRank` (const `Field` &F, size\_t m, size\_t n, size\_t r, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)  
*Random Matrix with prescribed rank.*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRank` (const `Field` &F, size\_t m, size\_t n, size\_t r, typename `Field::Element_ptr` A, size\_t lda)  
*Random Matrix with prescribed rank.*
- `size_t * RandomIndexSubset` (size\_t N, size\_t R, size\_t \*P)  
*Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.*
- `size_t * RandomPermutation` (size\_t N, size\_t \*P)  
*Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.*
- `void RandomRankProfileMatrix` (size\_t M, size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.*
- `void swapval` (size\_t k, size\_t N, size\_t \*P, size\_t val)
- `void RandomSymmetricRankProfileMatrix` (size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random a symmetric  $R$ -subpermutation of dimension  $N \times N$  : a symmetric matrix with only  $R$  non-zeros, all equal to one, in a random rook placement.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, Randlter &G)  
*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, Randlter &G)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*
- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)  
*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)  
*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)



*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*

- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (const Field &F, size_t N, size_t R,`  
`typename Field::Element_ptr A, size_t lda)`  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithDet (const Field &F, size_t n, const typename Field::Element d, type-`  
`name Field::Element_ptr A, size_t lda)`  
*Random Matrix with prescribed det.*
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithDet (const Field &F, size_t n, const typename Field::Element d, type-`  
`name Field::Element_ptr A, size_t lda, RandIter &G)`  
*Random Matrix with prescribed det.*
- `template<typename Field >`  
`Givaro::Integer maxFieldElt ()`
- `template<> Givaro::Integer maxFieldElt< Givaro::ZRing< Givaro::Integer > > ()`
- `template<typename Field >`  
`Field * chooseField (Givaro::Integer q, uint64_t b, uint64_t seed)`
- `template<> Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (Givaro::Integer q,`  
`uint64_t b, uint64_t seed)`
- `template<> Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (Givaro::Integer q,`  
`uint64_t b, uint64_t seed)`
- `template<> Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (Givaro::Integer q, uint64_t`  
`b, uint64_t seed)`
- `template<> Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (Givaro::Integer q,`  
`uint64_t b, uint64_t seed)`

## 15.20.1 Detailed Description

**Finite Field PACK** Set of elimination based routines for dense linear algebra.

This namespace enlarges the set of BLAS routines of the class **FFLAS**, with higher level routines based on elimination.

## 15.20.2 Typedef Documentation

### 15.20.2.1 Checker\_PLUQ

```
using Checker_PLUQ = FFLAS::Checker_Empty<Field>
```

### 15.20.2.2 Checker\_Det

```
using Checker_Det = FFLAS::Checker_Empty<Field>
```

**15.20.2.3 Checker\_invert**

```
using Checker_invert = FFLAS::Checker_Empty<Field>
```

**15.20.2.4 Checker\_charpoly**

```
using Checker_charpoly = FFLAS::Checker_Empty<Field>
```

**15.20.2.5 ForceCheck\_PLUQ**

```
using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
```

**15.20.2.6 ForceCheck\_Det**

```
using ForceCheck_Det = CheckerImplem_Det<Field>
```

**15.20.2.7 ForceCheck\_invert**

```
using ForceCheck_invert = CheckerImplem_invert<Field>
```

**15.20.2.8 ForceCheck\_charpoly**

```
using ForceCheck_charpoly = CheckerImplem_charpoly<Field, Polynomial>
```

**15.20.3 Function Documentation****15.20.3.1 LAPACKPerm2MathPerm()**

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N ) [inline]
```

Conversion of a permutation from LAPACK format to Math format.

### 15.20.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N ) [inline]
```

Conversion of a permutation from Maths format to LAPACK format.

### 15.20.3.3 applyP() [1/4]

```
void FFPACK::applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P )
```

Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.

#### Parameters

in, out	$P1$	a LAPACK permutation of size N
	$P2$	a LAPACK permutation of size N-R

Applies a permutation  $P$  to the matrix  $A$ . Apply a permutation  $P$ , stored in the LAPACK format (a sequence of transpositions) between indices  $ibeg$  and  $iend$  of  $P$  to  $(iend-ibeg)$  vectors of size  $M$  stored in  $A$  (as column for NoTrans and rows for Trans).  $Side == \text{FFLAS::FflasLeft}$  for row permutation  $Side == \text{FFLAS::FflasRight}$  for a column permutation  $Trans == \text{FFLAS::FflasTrans}$  for the inverse permutation of  $P$

#### Parameters

$F$	base field
$Side$	decides if rows (FflasLeft) or columns (FflasRight) are permuted
$Trans$	decides if the matrix is seen as columns (FflasTrans) or rows (FflasNoTrans)
$M$	size of the elements to permute
$ibeg$	first index to consider in $P$
$iend$	last index to consider in $P$
$A$	input matrix
$lda$	leading dimension of $A$
$P$	permutation in LAPACK format
$psh$	(optional): a sequential or parallel helper, to choose between sequential or parallel execution

**Warning**

not sure the submatrix is still a permutation and the one we expect in all cases... examples for iend=2, ibeg=1 and P=[2,2,2]

**15.20.3.4 applyP() [2/4]**

```
void FFPACK::applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t m,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const FFLAS::ParSeqHelper::Sequential seq )
```

**15.20.3.5 applyP() [3/4]**

```
void FFPACK::applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t m,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par )
```

**15.20.3.6 MonotonicApplyP()**

```
void FFPACK::MonotonicApplyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t R )
```

Apply a R-monotonically increasing permutation P, to the matrix A.

MonotonicApplyP Apply a permutation defined by the first R entries of the vector P (the pivots).

The permutation represented by P is defined as follows:

- the first  $R$  values of  $P$  is a LAPACK representation (a sequence of transpositions)
- the remaining  $iend-ibeg-R$  values of the permutation are in a monotonically increasing progression  $Side==FflasLeft$  for row permutation  $Side==FflasRight$  for a column permutation  $Trans==FflasTrans$  for the inverse permutation of  $P$

## Parameters

<i>F</i>	base field
<i>Side</i>	selects if it is a row (FflasLeft) or column (FflasRight) permutation
<i>Trans</i>	inverse permutation (FflasTrans/NoTrans)
<i>M</i>	
<i>ibeg</i>	
<i>iend</i>	
<i>A</i>	input matrix
<i>lda</i>	leading dimension of <i>A</i>
<i>P</i>	LAPACK permuation
<i>R</i>	first values of $P$

The non pivot elements, are located in montonically increasing order.

## 15.20.3.7 fgetrs() [1/4]

```
void FFPACK::fgetrs (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb,
    int * info )
```

Solve the system  $AX = B$  or  $XA = B$ .

Solving using the PLUQ decomposition of  $A$  already computed inplace with PLUQ ([FFLAS::FflasNonUnit](#)). Version for  $A$  square. If  $A$  is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

## Parameters

<i>F</i>	base field
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
<i>M</i>	row dimension of $B$
<i>N</i>	col dimension of $B$
<i>R</i>	rank of $A$
<i>A</i>	input matrix
<i>lda</i>	leading dimension of $A$

## Parameters

<i>P</i>	row permutation of the PLUQ decomposition of A
<i>Q</i>	column permutation of the PLUQ decomposition of A
<i>B</i>	Right/Left hand side matrix. Initially stores B, finally stores the solution X.
<i>ldb</i>	leading dimension of B
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

## 15.20.3.8 fgetrs() [2/4]

```
Field::Element_ptr FFPACK::fgetrs (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    int * info )
```

Solve the system  $A X = B$  or  $X A = B$ .

Solving using the PLUQ decomposition of A already computed inplace with PLUQ(FFLAS::FflasNonUnit). Version for A rectangular. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

## Parameters

<i>F</i>	base field
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
<i>M</i>	row dimension of A
<i>N</i>	col dimension of A
<i>NRHS</i>	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
<i>R</i>	rank of A
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	row permutation of the PLUQ decomposition of A
<i>Q</i>	column permutation of the PLUQ decomposition of A
<i>X</i>	solution matrix
<i>ldx</i>	leading dimension of X
<i>B</i>	Right/Left hand side matrix.
<i>ldb</i>	leading dimension of B
<i>info</i>	Succes of the computation: 0 if successfull, >0 if system is inconsistent

**15.20.3.9 fgesv() [1/4]**

```
size_t FFPACK::fgesv (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    int * info )
```

Square system solver.

**Parameters**

<i>F</i>	The computation domain
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
<i>M</i>	row dimension of B
<i>N</i>	col dimension of B
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>B</i>	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
<i>ldb</i>	leading dimension of B
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

**Returns**

the rank of the system

Solve the system  $A X = B$  or  $X A = B$ . Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

**15.20.3.10 fgesv() [2/4]**

```
size_t FFPACK::fgesv (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    int * info )
```

Rectangular system solver.

## Parameters

<i>F</i>	The computation domain
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
<i>M</i>	row dimension of A
<i>N</i>	col dimension of A
<i>NRHS</i>	number of columns (if Side = <a href="#">FFLAS::FflasLeft</a> ) or row (if Side = <a href="#">FFLAS::FflasRight</a> ) of the matrices X and B
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>B</i>	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
<i>ldb</i>	leading dimension of B
<i>X</i>	
<i>ldx</i>	
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

## Returns

the rank of the system

Solve the system  $A X = B$  or  $X A = B$ . Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

**15.20.3.11 ftrtri()** [1/2]

```
void FFPACK::ftrtri (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t threshold = __FFLASFFPACK_FTRTRI_THRESHOLD )
```

Compute the inverse of a triangular matrix.

## Parameters

<i>F</i>	base field
<i>Uplo</i>	whether the matrix is upper or lower triangular
<i>Diag</i>	whether the matrix is unit diagonal (FflasUnit/NoUnit)
<i>N</i>	input matrix order
<i>A</i>	the input matrix
<i>lda</i>	leading dimension of A

**15.20.3.12 trinv\_left()** [1/2]

```
void FFPACK::trinv_left (
```



```

const Field & F,
const size_t N,
typename Field::ConstElement_ptr L,
const size_t ldl,
typename Field::Element_ptr X,
const size_t ldx )

```

### 15.20.3.13 ftrtrm() [1/2]

```

void FFPACK::ftrtrm (
    const Field & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )

```

Compute the product of two triangular matrices of opposite shape.

Product UL or LU of the upper, resp lower triangular matrices U and L stored one above the other in the square matrix A.

#### Parameters

<i>F</i>	base field
<i>Side</i>	set to FflasLeft to compute the product UL, FflasRight to compute LU
<i>diag</i>	whether the matrix U is unit diagonal (FflasUnit/NoUnit)
<i>N</i>	input matrix order
<i>A</i>	the input matrix
<i>lda</i>	leading dimension of A

### 15.20.3.14 ftrstr()

```

void FFPACK::ftrstr (
    const Field & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diagA,
    const FFLAS::FFLAS_DIAG diagB,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const size_t threshold = __FFLASFFPACK_FTRSTR_THRESHOLD )

```

Solve a triangular system with a triangular right hand side of the same shape.

## Parameters

<i>F</i>	base field
<i>Side</i>	set to FflasLeft to compute $U1^{-1} * U2$ or $L1^{-1} * L2$ , FflasRight to compute $U1 * U2^{-1}$ or $L1 * L2^{-1}$
<i>Uplo</i>	whether the matrix A is upper or lower triangular
<i>diag1</i>	whether the matrix U1 or L2 is unit diagonal (FflasUnit/NoUnit)
<i>diag2</i>	whether the matrix U2 or L2 is unit diagonal (FflasUnit/NoUnit)
<i>N</i>	order of the input matrices
<i>A</i>	the input matrix to be inverted (U1 or L1)
<i>lda</i>	leading dimension of A
<i>B</i>	the input right hand side (U2 or L2)
<i>ldb</i>	leading dimension of B

## 15.20.3.15 ftrssyr2k()

```
void FFPACK::ftrssyr2k (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diagA,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const size_t threshold = __FFLASFFPACK_FTRSSYR2K_THRESHOLD )
```

Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.

C is overwritten by B.

## Parameters

	<i>F</i>	base field
	<i>Side</i>	set to FflasLeft to compute $U1^{-1} * U2$ or $L1^{-1} * L2$ , FflasRight to compute $U1 * U2^{-1}$ or $L1 * L2^{-1}$
	<i>Uplo</i>	whether the matrix A is upper or lower triangular
	<i>diagA</i>	whether the matrix A is unit diagonal (FflasUnit/NoUnit)
	<i>N</i>	order of the input matrices
	<i>A</i>	the input matrix
	<i>lda</i>	leading dimension of A
<i>in, out</i>	<i>B</i>	the input right hand side where the output is written
	<i>ldb</i>	leading dimension of B

**15.20.3.16 fsytrf()** [1/3]

```
bool FFPACK::fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

Triangular factorization of symmetric matrices.

**Parameters**

	$F$	The computation domain
	$UpLo$	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor
	$N$	order of the matrix A
$in, out$	$A$	input matrix
	$lda$	leading dimension of A

**Returns**

false if the A does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix A:  $A = L \times D \times L^T$  if UpLo = FflasLower or  $A = U^T \times D \times U$  otherwise. D is a diagonal matrix. The matrices L and U are unit diagonal lower (resp. upper) triangular and overwrite the input matrix A. The matrix D is stored on the diagonal of A, as the diagonal of L or U is known to be all ones. If A does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

**15.20.3.17 fsytrf()** [2/3]

```
bool FFPACK::fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Sequential seq,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

**15.20.3.18 fsytrf()** [3/3]

```
bool FFPACK::fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

**15.20.3.19 fsytrf\_nonunit()** [1/3]

```
bool FFPACK::fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr D,
    const size_t incD,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

Triangular factorization of symmetric matrices.

**Parameters**

	$F$	The computation domain
	$UpLo$	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor
	$N$	order of the matrix A
in, out	$A$	input matrix
in, out	$D$	
	$lda$	leading dimension of A

**Returns**

false if the A does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix A:  $A = L \times D_{inv} \times L^T$  if UpLo = FflasLower or  $A = U^T \times D \times U$  otherwise. D is a diagonal matrix. The matrices L and U are lower (resp. upper) triangular and overwrite the input matrix A. The matrix D need to be stored separately, as the diagonal of L or U are not unit. If A does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

**15.20.3.20 PLUQ()** [1/6]

```
size_t FFPACK::PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

Compute a PLUQ factorization of the given matrix.

Return its rank. The permutations P and Q are represented using LAPACK's convention.

**Parameters**

$F$	base field
$Diag$	whether U should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
$M$	matrix row dimension

## Parameters

$N$	matrix column dimension
$A$	input matrix
$lda$	leading dimension of $A$
$P$	the row permutation
$Q$	the column permutation

## Returns

the rank of  $A$

- Bibliography** • Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13, 2013

## 15.20.3.21 pPLUQ()

```
size_t FFPACK::pPLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

## 15.20.3.22 PLUQ() [2/6]

```
size_t FFPACK::PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Sequential & PSHelper,
    size_t BCThreshold = __FFLASFFPACK_PLUQ_THRESHOLD )
```

**15.20.3.23 PLUQ()** [3/6]

```

size_t FFPACK::PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper )

```

**15.20.3.24 LUdivine()** [1/4]

```

size_t FFPACK::LUdivine (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive,
    const size_t cutoff = __FFLASFFPACK_LUDIVINE_THRESHOLD )

```

Compute the CUP or PLE factorization of the given matrix.

Using a block algorithm and return its rank. The permutations P and Q are represented using LAPACK's convention.

**Parameters**

<i>F</i>	base field
<i>Diag</i>	whether the transformation matrix (U of the CUP, L of the PLE) should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
<i>trans</i>	whether to compute the CUP decomposition (FflasNoTrans) or the PLE decomposition (FflasTrans)
<i>M</i>	matrix row dimension
<i>N</i>	matrix column dimension
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	the factor of CUP or PLE
<i>Q</i>	a permutation indicating the pivot position in the echelon form C or E in its first r positions
<i>LuTag</i>	flag for setting the earling termination if the matrix is singular
<i>cutoff</i>	threshold to basecase

**Returns**

the rank of A

**Bibliography**

- Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013
- Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002

**15.20.3.25 ColumnEchelonForm()** [1/3]

```
size_t ColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Compute the Column Echelon form of the input matrix in-place.

If LuTag == FfpackTileRecursive, then after the computation  $A = [M \setminus V]$  such that  $AU = C$  is a column echelon decomposition of A, with  $U = P^T [V]$  and  $C = M + Q [I_r] [0 \text{ } I_{n-r}] [0]$  If LuTag == FfpackTileRecursive then  $A = [N \setminus V]$  such that the same holds with  $M = Q N$

$Q_t = Q^T$  If transform=false, the matrix V is not computed. See also test-colechelon for an example of use

**Parameters**

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix
	$lda$	leading dimension of A
	$P$	the column permutation
	$Q_t$	the row position of the pivots in the echelon form
	$transform$	decides whether V is computed
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

**15.20.3.26 pColumnEchelonForm()**

```
size_t pColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
```

```

size_t * Qt,
bool transform = false,
size_t numthreads = 0,
const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

### 15.20.3.27 ColumnEchelonForm() [2/3]

```

size_t ColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]

```

### 15.20.3.28 RowEchelonForm() [1/3]

```

size_t RowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Row Echelon form of the input matrix in-place.

If  $\text{LuTag} == \text{FfpackTileRecursive}$ , then after the computation  $A = [L \setminus M]$  such that  $XA = R$  is a row echelon decomposition of  $A$ , with  $X = [L \ 0] P$  and  $R = M + [I \ 0] Q^T [In-r]$  If  $\text{LuTag} == \text{FfpackTileRecursive}$  then  $A = [L \setminus N]$  such that the same holds with  $M = N Q^T Q^T = Q^T$  If  $\text{transform} = \text{false}$ , the matrix  $L$  is not computed. See also `test-rowechelon` for an example of use

#### Parameters

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	the input matrix
	$lda$	leading dimension of $A$
	$P$	the row permutation
	$Qt$	the column position of the pivots in the echelon form
	$transform$	decides whether $L$ is computed
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ



**15.20.3.29 pRowEchelonForm()**

```

size_t pRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

**15.20.3.30 RowEchelonForm() [2/3]**

```

size_t RowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]

```

**15.20.3.31 ReducedColumnEchelonForm() [1/3]**

```

size_t ReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Reduced Column Echelon form of the input matrix in-place.

After the computation  $A = [V]$  such that  $AX = R$  is a reduced col echelon  $[M \ 0]$  decomposition of  $A$ , where  $X = P^T [V]$  and  $R = Q [I_r] [0 \ I_{n-r}] [M \ 0] Qt = Q^T$  If transform=false, the matrix  $X$  is not computed and the matrix  $A = R$

## Parameters

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix
	$lda$	leading dimension of A
	$P$	the column permutation
	$Qt$	the row position of the pivots in the echelon form
	$transform$	decides whether X is computed
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

## 15.20.3.32 pReducedColumnEchelonForm()

```

size_t pReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

## 15.20.3.33 ReducedColumnEchelonForm() [2/3]

```

size_t ReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]

```

**15.20.3.34 ReducedRowEchelonForm()** [1/3]

```

size_t ReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Reduced Row Echelon form of the input matrix in-place.

After the computation  $A = [V1 \ M]$  such that  $X A = R$  is a reduced row echelon  $[V2 \ 0]$  decomposition of  $A$ , where  $X = [V1 \ 0] P$  and  $R = [I_r \ M] Q^T [V2 \ In-r] [0] Qt = Q^T$  If transform=false, the matrix  $X$  is not computed and the matrix  $A = R$

**Parameters**

	<i>F</i>	base field
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
	<i>P</i>	the row permutation
	<i>Qt</i>	the column position of the pivots in the echelon form
	<i>transform</i>	decides whether X is computed
	<i>LuTag</i>	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

**15.20.3.35 pReducedRowEchelonForm()**

```

size_t pReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

### 15.20.3.36 ReducedRowEchelonForm() [2/3]

```
size_t ReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]
```

### 15.20.3.37 Invert() [1/4]

```
Field::Element_ptr FFPACK::Invert (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    int & nullity )
```

Invert the given matrix in place or computes its nullity if it is singular.

An inplace  $2n^3$  algorithm is used.

#### Parameters

	$F$	The computation domain
	$M$	order of the matrix
in, out	$A$	input matrix ( $M \times M$ )
	$lda$	leading dimension of A
	$nullity$	dimension of the kernel of A

#### Returns

pointer to  $A$  and  $A \leftarrow A^{-1}$

### 15.20.3.38 Invert() [2/4]

```
Field::Element_ptr FFPACK::Invert (
    const Field & F,
    const size_t M,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    int & nullity )
```

Invert the given matrix or computes its nullity if it is singular.

**Precondition**

$X$  is preallocated and should be large enough to store the  $m \times m$  matrix  $A$ .

**Parameters**

	$F$	The computation domain
	$M$	order of the matrix
in	$A$	input matrix ( $M \times M$ )
	$lda$	leading dimension of $A$
out	$X$	this is the inverse of $A$ if $A$ is invertible (non NULL and <code>nullity = 0</code> ). It is untouched otherwise.
	$ldx$	leading dimension of $X$
	$nullity$	dimension of the kernel of $A$

**Returns**

pointer to  $X = A^{-1}$

**15.20.3.39 Invert2() [1/2]**

```
Field::Element_ptr FFPACK::Invert2 (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    int & nullity )
```

Invert the given matrix or computes its nullity if it is singular.

An  $2n^3f$  algorithm is used. This routine can be % faster than [FFPACK::Invert](#) but is not totally inplace.

**Precondition**

$X$  is preallocated and should be large enough to store the  $m \times m$  matrix  $A$ .

**Warning**

$A$  is overwritten here !

**Bug** not tested.

**Parameters**

	$F$	the computation domain
	$M$	order of the matrix
in, out	$A$	input matrix ( $M \times M$ ). On output, $A$ is modified and represents a "psychological" factorisation LU.
Generated by Doxygen	$lda$	leading dimension of $A$
out	$X$	this is the inverse of $A$ if $A$ is invertible (non NULL and <code>nullity = 0</code> ). It is untouched otherwise.
	$ldx$	leading dimension of $X$

## Returns

pointer to  $X = A^{-1}$

**Todo** this init is not all necessary (done after ftrtri)

### 15.20.3.40 CharPoly() [1/8]

```
std::list< typename PolRing::Element > & CharPoly (
    const PolRing & R,
    std::list< typename PolRing::Element > & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    typename PolRing::Domain_t::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]
```

Compute the characteristic polynomial of the matrix A.

## Parameters

	<i>R</i>	the polynomial ring of charp (contains the base field)
out	<i>charp</i>	the characteristic polynomial of as a list of factors
	<i>N</i>	order of the matrix A
in	<i>A</i>	the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants)
	<i>lda</i>	leading dimension of A
	<i>CharpTag</i>	the algorithmic variant
	<i>G</i>	a random iterator (required for the randomized variants LUKrylov and ArithProg)

### 15.20.3.41 CharPoly() [2/8]

```
PolRing::Element & CharPoly (
    const PolRing & R,
    typename PolRing::Element & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    typename PolRing::Domain_t::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]
```

Compute the characteristic polynomial of the matrix A.

## Parameters

	<i>R</i>	the polynomial ring of charp (contains the base field)
--	----------	--

## Parameters

out	<i>charp</i>	the characteristic polynomial of <i>as</i> a single polynomial
	<i>N</i>	order of the matrix <i>A</i>
in	<i>A</i>	the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants)
	<i>lda</i>	leading dimension of <i>A</i>
	<i>CharpTag</i>	the algorithmic variant
	<i>G</i>	a random iterator (required for the randomized variants LUKrylov and ArithProg)

## 15.20.3.42 CharPoly() [3/8]

```
PolRing::Element& FFPACK::CharPoly (
    const PolRing & R,
    typename PolRing::Element & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]
```

Compute the characteristic polynomial of the matrix *A*.

## Parameters

	<i>R</i>	the polynomial ring of <i>charp</i> (contains the base field)
out	<i>charp</i>	the characteristic polynomial of <i>as</i> a single polynomial
	<i>N</i>	order of the matrix <i>A</i>
in	<i>A</i>	the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants)
	<i>lda</i>	leading dimension of <i>A</i>
	<i>CharpTag</i>	the algorithmic variant

## 15.20.3.43 MinPoly() [1/4]

```
Polynomial& FFPACK::MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

Compute the minimal polynomial of the matrix *A*.

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector: (*v*, *Av*, ..., *A*<sup>*k*</sup>*v*)

## Parameters

	$F$	the base field
out	$minP$	the minimal polynomial of $A$
	$N$	order of the matrix $A$
in	$A$	the input matrix ( $N \times N$ )
	$lda$	leading dimension of $A$

**15.20.3.44 MinPoly()** [2/4]

```
Polynomial& FFPACK::MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    RandIter & G )
```

Compute the minimal polynomial of the matrix  $A$ .

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector:  $(v, Av, \dots, A^k v)$

## Parameters

	$F$	the base field
out	$minP$	the minimal polynomial of $A$
	$N$	order of the matrix $A$
in	$A$	the input matrix ( $N \times N$ )
	$lda$	leading dimension of $A$
	$G$	a random iterator

**15.20.3.45 MatVecMinPoly()** [1/2]

```
Polynomial& FFPACK::MatVecMinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr v,
    const size_t incv )
```

Compute the minimal polynomial of the matrix  $A$  and a vector  $v$ , namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .



## Parameters

	$F$	the base field
out	$minP$	the minimal polynomial of $A$ and $v$
	$N$	order of the matrix $A$
in	$A$	the input matrix ( $N \times N$ )
	$lda$	leading dimension of $A$
	$K$	an $N \times (N + 1)$ matrix containing the vector $v$ on its first row
	$ldk$	leading dimension of $K$
	$P$	[out] (optional) the permutation used in the elimination of the Krylov matrix $K$

## 15.20.3.46 Rank() [1/3]

```
size_t FFPACK::Rank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

Computes the rank of the given matrix using a PLUQ factorization.

The input matrix is modified.

## Parameters

	$F$	base field
	$M$	row dimension of the matrix
	$N$	column dimension of the matrix
in	$A$	input matrix
	$lda$	leading dimension of $A$
	$psH$	(optional) a ParSeqHelper to choose between sequential and parallel execution

## 15.20.3.47 pRank()

```
size_t FFPACK::pRank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t numthreads = 0 )
```

**15.20.3.48 Rank()** [2/3]

```
size_t FFPACK::Rank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const PSHelper & pSH )
```

**15.20.3.49 IsSingular()** [1/2]

```
bool FFPACK::IsSingular (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

Returns true if the given matrix is singular.

The method is a block elimination with early termination

using LQUP factorization with early termination. If  $M \neq N$ , then the matrix is virtually padded with zeros to make it square and it's determinant is zero.

**Warning**

The input matrix is modified.

**Parameters**

	$F$	base field
	$M$	row dimension of the matrix
	$N$	column dimension of the matrix.
in, out	$A$	input matrix
	$lda$	leading dimension of A

**15.20.3.50 Det()** [1/6]

```
Field::Element& FFPACK::Det (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
```

```
size_t * P = NULL,
size_t * Q = NULL )
```

Returns the determinant of the given square matrix.

The method is a block elimination using PLUQ factorization. The input matrix A is overwritten.

#### Warning

The input matrix is modified.

#### Parameters

	<i>F</i>	base field
out	<i>det</i>	the determinant of A
	<i>N</i>	the order of the square matrix A.
in, out	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
	<i>psH</i>	(optional) a ParSeqHelper to choose between sequential and parallel execution
	<i>P,Q</i>	(optional) row and column permutations to be used by the PLUQ factorization. randomized checkers (see cherckes/checker_det.inl) need them for certification

#### 15.20.3.51 pDet()

```
Field::Element& FFPACK::pDet (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t numthreads = 0,
    size_t * P = NULL,
    size_t * Q = NULL )
```

#### 15.20.3.52 Det() [2/6]

```
Field::Element& FFPACK::Det (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const PSHelper & psH,
    size_t * P = NULL,
    size_t * Q = NULL )
```

**15.20.3.53 Solve()** [1/3]

```
Field::Element_ptr FFPACK::Solve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb )
```

Solves a linear system  $AX = b$  using PLUQ factorization.

@oaram F base field @oaram M matrix order

**Parameters**

in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
out	<i>x</i>	output solution vector
	<i>incx</i>	increment of x
	<i>b</i>	input right hand side of the system
	<i>incb</i>	increment of b

**15.20.3.54 Solve()** [2/3]

```
Field::Element_ptr FFPACK::Solve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb,
    PSHelper & psH )
```

**15.20.3.55 pSolve()**

```
Field::Element_ptr FFPACK::pSolve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb,
    size_t numthreads = 0 )
```

**15.20.3.56 RandomNullSpaceVector()** [1/3]

```
* void FFPACK::RandomNullSpaceVector (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t incX )
```

Solve  $LX = B$  or  $XL = B$  in place.

L is  $M \times M$  if Side == FFLAS::FflasLeft and  $N \times N$  if Side == FFLAS::FflasRight, B is  $M \times N$ . Only the R non trivial column of L are stored in the  $M \times R$  matrix L Requirement : so that L could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix A.

**Parameters**

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in, out	<i>A</i>	input matrix of dimension $M \times N$ , A is modified to its LU version
	<i>lda</i>	leading dimension of A
out	<i>X</i>	output vector
	<i>incX</i>	increment of X

**15.20.3.57 NullSpaceBasis()** [1/2]

```
size_t FFPACK::NullSpaceBasis (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & NS,
    size_t & ldn,
    size_t & NSdim )
```

Computes a basis of the Left/Right nullspace of the matrix A.

return the dimension of the nullspace.

**Parameters**

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows

## Parameters

	<i>N</i>	number of columns
in, out	<i>A</i>	input matrix of dimension M x N, A is modified
	<i>lda</i>	leading dimension of A
out	<i>NS</i>	output matrix of dimension N x NSdim (allocated here)
out	<i>ldn</i>	leading dimension of NS
out	<i>NSdim</i>	the dimension of the Nullspace (N-rank(A))

## 15.20.3.58 RowRankProfile() [1/3]

```
size_t FFPACK::RowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )
```

Computes the row rank profile of A.

## Parameters

	<i>F</i>	base field
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in	<i>A</i>	input matrix of dimension M x N
	<i>lda</i>	leading dimension of A
out	<i>rkprofile</i>	return the rank profile as an array of row indexes, of dimension r=rank(A)
	<i>LuTag</i>	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

## Returns

R

## 15.20.3.59 pRowRankProfile()

```
size_t FFPACK::pRowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
```

```

const size_t lda,
size_t *& rkprofile,
size_t numthreads = 0,
const FFPACK_LU_TAG LuTag = FfpackTileRecursive )

```

### 15.20.3.60 RowRankProfile() [2/3]

```

size_t FFPACK::RowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag,
    PSHelper & psH )

```

### 15.20.3.61 ColumnRankProfile() [1/3]

```

size_t FFPACK::ColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Computes the column rank profile of A.

#### Parameters

	<i>F</i>	base field
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in	<i>A</i>	input matrix of dimension
	<i>lda</i>	leading dimension of A
out	<i>rkprofile</i>	return the rank profile as an array of row indexes, of dimension r=rank(A)
	<i>LuTag</i>	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

#### Returns

R

**15.20.3.62 pColumnRankProfile()**

```

size_t FFPACK::pColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive )

```

**15.20.3.63 ColumnRankProfile() [2/3]**

```

size_t FFPACK::ColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    const FFPACK_LU_TAG LuTag,
    PSHelper & psH )

```

**15.20.3.64 RankProfileFromLU()**

```

void RankProfileFromLU (
    const size_t * P,
    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const FFPACK_LU_TAG LuTag ) [inline]

```

Recovers the column/row rank profile from the permutation of an LU decomposition.

Works with both the CUP/PLE decompositions (obtained by LUdivine) or the PLUQ decomposition. Assumes that the output vector containing the rank profile is already allocated.

**Parameters**

	<i>P</i>	the permutation carrying the rank profile information
	<i>N</i>	the row/col dimension for a row/column rank profile
	<i>R</i>	the rank of the matrix
out	<i>rkprofile</i>	return the rank profile as an array of indices
	<i>LuTag</i>	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ



**15.20.3.65 LeadingSubmatrixRankProfiles()**

```
size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP ) [inline]
```

Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.

Only works with the PLUQ decomposition Assumes that the output vectors containing the rank profiles are already allocated.

**Parameters**

<i>P</i>	the permutation carrying the rank profile information
<i>M</i>	the row dimension of the initial matrix
<i>N</i>	the column dimension of the initial matrix
<i>R</i>	the rank of the initial matrix
<i>LSm</i>	the row dimension of the leading submatrix considered
<i>LSn</i>	the column dimension of the leading submatrix considered
<i>P</i>	the row permutation of the PLUQ decomposition
<i>Q</i>	the column permutation of the PLUQ decomposition
<i>RRP</i>	return the row rank profile of the leading submatrix

**Returns**

the rank of the LSm x LSn leading submatrix

A is modified

**Bibliography** • Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13.

**15.20.3.66 RowRankProfileSubmatrixIndices()** [1/2]

```
size_t FFPACK::RowRankProfileSubmatrixIndices (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

RowRankProfileSubmatrixIndices.

Computes the indices of the submatrix  $r \times r$  X of A whose rows correspond to the row rank profile of A.

## Parameters

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix of dimension
	<i>rowindices</i>	array of the row indices of X in A
	<i>colindices</i>	array of the col indices of X in A
	<i>lda</i>	leading dimension of A
out	$R$	list of indices

rowindices and colindices are allocated during the computation. A is modified

## Returns

$R$

**15.20.3.67 ColRankProfileSubmatrixIndices()** [1/2]

```
size_t FFPACK::ColRankProfileSubmatrixIndices (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.

## Parameters

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix of dimension
	<i>rowindices</i>	array of the row indices of X in A
	<i>colindices</i>	array of the col indices of X in A
	<i>lda</i>	leading dimension of A
out	$R$	list of indices

rowindices and colindices are allocated during the computation.

## Warning

A is modified

## Returns

R

**15.20.3.68 RowRankProfileSubmatrix()** [1/2]

```
size_t FFPACK::RowRankProfileSubmatrix (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & X,
    size_t & R )
```

Computes the  $r \times r$  submatrix  $X$  of  $A$ , by picking the row rank profile rows of  $A$ .

## Parameters

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix of dimension $M \times N$
	$lda$	leading dimension of $A$
out	$X$	the output matrix
out	$R$	list of indices

$A$  is not modified  $X$  is allocated during the computation.

## Returns

R

**15.20.3.69 ColRankProfileSubmatrix()** [1/2]

```
size_t FFPACK::ColRankProfileSubmatrix (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & X,
    size_t & R )
```

Compute the  $r \times r$  submatrix  $X$  of  $A$ , by picking the row rank profile rows of  $A$ .

## Parameters

	$F$	base field
	$M$	number of rows
	$N$	number of columns
in	$A$	input matrix of dimension $M \times N$
	$lda$	leading dimension of $A$
out	$X$	the output matrix
out	$R$	list of indices

$A$  is not modified  $X$  is allocated during the computation.

## Returns

$R$

## 15.20.3.70 getTriangular() [1/2]

```
void FFPACK::getTriangular (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false )
```

Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank  $R$ .

if `OnlyNonZeroVectors` is false, then  $T$  and  $A$  have the same dimensions Otherwise,  $T$  is  $R \times N$  if `UpLo` = `FflasUpper`, else  $T$  is  $M \times R$

## Parameters

	$F$	base field
	$UpLo$	selects if the upper ( <code>FflasUpper</code> ) or lower ( <code>FflasLower</code> ) triangular matrix is returned
	$diag$	selects if the triangular matrix unit-diagonal ( <code>FflasUnit/NoUnit</code> )
	$M$	row dimension of $T$
	$N$	column dimension of $T$
	$R$	rank of the triangular matrix (how many rows/columns need to be copied)
in	$A$	input matrix
	$lda$	leading dimension of $A$
out	$T$	output matrix
	$ldt$	leading dimension of $T$
	<code>OnlyNonZeroVectors</code>	decides whether the last zero rows/columns should be ignored

**Todo** just one triangular fzero+fassign ?

### 15.20.3.71 getTriangular() [2/2]

```
void FFPACK::getTriangular (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda )
```

Cleans up a compact storage  $A=LU$  to reveal a triangular matrix of rank  $R$ .

#### Parameters

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is revealed
	<i>diag</i>	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
	<i>R</i>	rank of the triangular matrix
in, out	<i>A</i>	input/output matrix
	<i>lda</i>	leading dimension of A

**Todo** just one triangular fzero+fassign ?

### 15.20.3.72 getEchelonForm() [1/2]

```
void FFPACK::getEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )
```

Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.

Either  $L$  or  $U$  is in Echelon form (depending on Uplo) The echelon structure is defined by the first  $R$  values of the array  $P$ . row and column dimension of  $T$  are greater or equal to that of  $A$

#### Parameters

	$F$	base field
	$UpLo$	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	$diag$	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	$M$	row dimension of $T$
	$N$	column dimension of $T$
	$R$	rank of the triangular matrix (how many rows/columns need to be copied)
	$P$	positions of the $R$ pivots
in	$A$	input matrix
	$lda$	leading dimension of $A$
out	$T$	output matrix
	$ldt$	leading dimension of $T$
	<i>OnlyNonZeroVectors</i>	decides whether the last zero rows/columns should be ignored
	$LuTag$	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

#### 15.20.3.73 getEchelonForm() [2/2]

```
void FFPACK::getEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )
```

Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .

Either  $L$  or  $U$  is in Echelon form (depending on Uplo) The echelon structure is defined by the first  $R$  values of the array  $P$ .

#### Parameters

	$F$	base field
	$UpLo$	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	$diag$	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	$M$	row dimension of $A$
	$N$	column dimension of $A$

## Parameters

	$R$	rank of the triangular matrix (how many rows/columns need to be copied)
	$P$	positions of the R pivots
in, out	$A$	input/output matrix
	$lda$	leading dimension of A
	$LuTag$	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

## 15.20.3.74 getEchelonTransform()

```

void FFPACK::getEchelonTransform (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Extracts a transformation matrix to echelon form from a compact storage  $A=LU$  of rank  $R$  obtained by Row↔EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower:  $T$  is  $N \times N$  (already allocated) such that  $A T = C$  is a transformation of  $A$  in Column echelon form Else  $T$  is  $M \times M$  (already allocated) such that  $T A = E$  is a transformation of  $A$  in Row Echelon form

## Parameters

	$F$	base field
	$UpLo$	Lower (FflasLower) means Transformation to Column Echelon Form, Upper (FflasUpper), to Row Echelon Form
	$diag$	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	$M$	row dimension of A
	$N$	column dimension of A
	$R$	rank of the triangular matrix
	$P$	permutation matrix
in	$A$	input matrix
	$lda$	leading dimension of A
out	$T$	output matrix
	$ldt$	leading dimension of T
	$LuTag$	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

**15.20.3.75 getReducedEchelonForm() [1/2]**

```

void FFPACK::getReducedEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```

Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first  $R$  values of the array  $P$ . row and column dimension of  $T$  are greater or equal to that of  $A$

**Parameters**

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	<i>diag</i>	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	<i>M</i>	row dimension of T
	<i>N</i>	column dimension of T
	<i>R</i>	rank of the triangular matrix (how many rows/columns need to be copied)
	<i>P</i>	positions of the R pivots
in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
	<i>ldt</i>	leading dimension of T
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)
	<i>OnlyNonZeroVectors</i>	decides whether the last zero rows/columns should be ignored

**15.20.3.76 getReducedEchelonForm() [2/2]**

```

void FFPACK::getReducedEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )

```



Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.

Either  $L$  or  $U$  is in Echelon form (depending on `Uplo`) The echelon structure is defined by the first  $R$  values of the array  $P$ .

#### Parameters

	$F$	base field
	$UpLo$	selects if the upper ( <code>FflasUpper</code> ) or lower ( <code>FflasLower</code> ) triangular matrix is returned
	$diag$	selects if the echelon matrix has unit pivots ( <code>FflasUnit/NoUnit</code> )
	$M$	row dimension of $A$
	$N$	column dimension of $A$
	$R$	rank of the triangular matrix (how many rows/columns need to be copied)
	$P$	positions of the $R$ pivots
$in, out$	$A$	input/output matrix
	$lda$	leading dimension of $A$
	$LuTag$	which factorized form ( <code>CUP/PLE</code> if <code>FfpackSlabRecursive</code> , <code>PLUQ</code> if <code>FfpackTileRecursive</code> )

#### 15.20.3.77 getReducedEchelonTransform()

```
void FFPACK::getReducedEchelonTransform (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive )
```

Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `RowEchelonForm` or `ColumnEchelonForm`.

If `Uplo == FflasLower`:  $T$  is  $N \times N$  (already allocated) such that  $A T = C$  is a transformation of  $A$  in Column echelon form Else  $T$  is  $M \times M$  (already allocated) such that  $T A = E$  is a transformation of  $A$  in Row Echelon form

#### Parameters

	$F$	base field
	$UpLo$	selects Col ( <code>FflasLower</code> ) or Row ( <code>FflasUpper</code> ) Echelon Form
	$diag$	selects if the echelon matrix has unit pivots ( <code>FflasUnit/NoUnit</code> )
	$M$	row dimension of $A$
	$N$	column dimension of $A$
	$R$	rank of the triangular matrix
	$P$	permutation matrix

## Parameters

in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
out	<i>T</i>	output matrix
	<i>ldt</i>	leading dimension of T
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

**15.20.3.78 PLUQtoEchelonPermutation()**

```
void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm ) [inline]
```

Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.

**15.20.3.79 LQUPtoInverseOfFullRankMinor()** [1/2]

```
Field::Element_ptr FFPACK::LQUPtoInverseOfFullRankMinor (
    const Field & F,
    const size_t rank,
    typename Field::Element_ptr A_factors,
    const size_t lda,
    const size_t * QtPointer,
    typename Field::Element_ptr X,
    const size_t ldx )
```

LQUPtoInverseOfFullRankMinor.

Suppose A has been factorized as L.Q.U.P, with rank r. Then Qt.A.Pt has an invertible leading principal r x r submatrix This procedure efficiently computes the inverse of this minor and puts it into X.

**Note**

It changes the lower entries of A\_factors in the process (NB: unless A was nonsingular and square)

## Parameters

<i>F</i>	base field
<i>rank</i>	rank of the matrix.
<i>A_factors</i>	matrix containing the L and U entries of the factorization
<i>lda</i>	leading dimension of A
<i>QtPointer</i>	theLQUP->getQ()->getPointer() (note: getQ returns Qt!)
<i>X</i>	desired location for output
<i>ldx</i>	leading dimension of X

**15.20.3.80 RandomNullSpaceVector()** [2/3]

```
void FFPACK::RandomNullSpaceVector (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t incX )
```

Solve  $LX = B$  or  $XL = B$  in place.

L is  $M \times M$  if `Side == FFLAS::FflasLeft` and  $N \times N$  if `Side == FFLAS::FflasRight`, B is  $M \times N$ . Only the R non trivial column of L are stored in the  $M \times R$  matrix L Requirement : so that L could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix A.

**Parameters**

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in, out	<i>A</i>	input matrix of dimension $M \times N$ , A is modified to its LU version
	<i>lda</i>	leading dimension of A
out	<i>X</i>	output vector
	<i>incX</i>	increment of X

**15.20.3.81 solveLB()** [1/2]

```
void FFPACK::solveLB (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr L,
    const size_t ldl,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb )
```

**15.20.3.82 solveLB2()** [1/2]

```
void FFPACK::solveLB2 (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr L,
    const size_t ldl,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb )
```

**15.20.3.83 Danilevski()**

```
std::list<Polynomial>& FFPACK::Danilevski (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

**15.20.3.84 buildMatrix()**

```
Field::Element_ptr FFPACK::buildMatrix (
    const Field & F,
    typename Field::ConstElement_ptr E,
    typename Field::ConstElement_ptr C,
    const size_t lda,
    const size_t * B,
    const size_t * T,
    const size_t me,
    const size_t mc,
    const size_t lambda,
    const size_t mu )
```

**Bug** is this :

**15.20.3.85 CharPoly()** [4/8]

```
FFPACK::RNSInteger<FFPACK::rns_double>::Element_ptr FFPACK::CharPoly (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp,
    const size_t N,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
    const size_t lda,
    Givaro::ZRing< Givaro::Integer >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    size_t degree ) [inline]
```

**15.20.3.86 CharPoly()** [5/8]

```
Givaro::Poly1Dom<Givaro::ZRing<Givaro::Integer> >::Element& FFPACK::CharPoly (
    const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > & R,
    Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & charp,
    const size_t N,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::ZRing< Givaro::Integer >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    size_t degree ) [inline]
```

**15.20.3.87 Det()** [3/6]

```
FFPACK::RNSInteger<FFPACK::rns_double>::Element_ptr& FFPACK::Det (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & det,
    const size_t N,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
    const size_t lda,
    const PSHelper & psH ) [inline]
```

**15.20.3.88 Det()** [4/6]

```
Givaro::Integer& FFPACK::Det (
    const Givaro::ZRing< Givaro::Integer > & F,
    Givaro::Integer & det,
    const size_t N,
    Givaro::Integer * A,
    const size_t lda,
    const PSHelper & psH,
    size_t * P,
    size_t * Q ) [inline]
```

**15.20.3.89 fsytrf\_BC\_Crout()**

```
bool FFPACK::fsytrf_BC_Crout (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv ) [inline]
```

**15.20.3.90 fsytrf\_BC\_RL()**

```

size_t FFPACK::fsytrf_BC_RL (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv ) [inline]

```

**15.20.3.91 fsytrf\_UP\_RPM\_BC\_RL()**

```

size_t FFPACK::fsytrf_UP_RPM_BC_RL (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]

```

**15.20.3.92 fsytrf\_LOW\_RPM\_BC\_Crout()**

```

size_t FFPACK::fsytrf_LOW_RPM_BC_Crout (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]

```

**15.20.3.93 fsytrf\_UP\_RPM\_BC\_Crout()**

```

size_t FFPACK::fsytrf_UP_RPM_BC_Crout (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]

```

**15.20.3.94 fsytrf\_UP\_RPM()**

```
size_t FFPACK::fsytrf_UP_RPM (
    const Field & Fi,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P,
    size_t BCThreshold ) [inline]
```

$$\text{MathP} \leftarrow \begin{bmatrix} I \\ P_1 \end{bmatrix} \begin{bmatrix} L(N+R_2) \\ P_2^T \end{bmatrix} \begin{bmatrix} P_3^T \end{bmatrix} \begin{bmatrix} Q_2^T \end{bmatrix}$$

Changing  $\begin{bmatrix} U_1 & V_1 & E_1 & E_{21} & E_{22} \end{bmatrix}$  into  $\begin{bmatrix} U_1 & E_{11} & E_{12} & V_1 & E^* & E^* \end{bmatrix} \begin{bmatrix} 0 & L_2 \setminus U_2 & V_{21} & V_{22} \end{bmatrix} \begin{bmatrix} U_4 & V_{41} & 0 & V_{42} & V_{43} \end{bmatrix} \begin{bmatrix} 0 & M_2 & 0 & 0 \end{bmatrix} \begin{bmatrix} U_3 & 0 & 0 & V_3 \end{bmatrix}$  where  $U_4$  is the  $2R_2 \times 2R_2$  matrix formed by interleaving  $U_2$ ,  $L_2^T$  and  $H_1$

**15.20.3.95 fsytrf\_nonunit() [2/3]**

```
bool FFPACK::fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    FFLAS::ParSeqHelper::Sequential seq,
    size_t threshold ) [inline]
```

**15.20.3.96 fsytrf\_nonunit() [3/3]**

```
bool FFPACK::fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
    size_t threshold ) [inline]
```

**15.20.3.97 fsytrf\_RPM()**

```
size_t FFPACK::fsytrf_RPM (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t threshold ) [inline]
```

**15.20.3.98 getTridiagonal()**

```
void FFPACK::getTridiagonal (
    const Field & F,
    const size_t N,
    const size_t R,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    size_t * P,
    typename Field::Element_ptr T,
    const size_t ldt ) [inline]
```

**15.20.3.99 LUdivine\_gauss() [1/2]**

```
size_t FFPACK::LUdivine_gauss (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

**15.20.3.100 LUdivine\_small() [1/2]**

```
size_t FFPACK::LUdivine_small (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```



**15.20.3.101 LUdivine()** [2/4]

```

size_t FFPACK::LUdivine (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag,
    const size_t cutoff ) [inline]

```

**Todo** std::swap ?

**15.20.3.102 LUdivine()** [3/4]

```

size_t FFPACK::LUdivine (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Givaro::Integer * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag,
    const size_t cutoff ) [inline]

```

**15.20.3.103 MonotonicCompress()**

```

void FFPACK::MonotonicCompress (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t maxpiv,
    const size_t rowstomove,
    const std::vector< bool > & ispiv ) [inline]

```

**15.20.3.104 MonotonicCompressMorePivots()**

```

void FFPACK::MonotonicCompressMorePivots (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t rowstomove,
    const size_t lenP ) [inline]

```

**15.20.3.105 MonotonicCompressCycles()**

```

void FFPACK::MonotonicCompressCycles (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t lenP ) [inline]

```

**15.20.3.106 MonotonicExpand()**

```

void FFPACK::MonotonicExpand (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t maxpiv,
    const size_t rowstomove,
    const std::vector< bool > & ispiv )

```

### 15.20.3.107 applyP\_block()

```
void FFPACK::applyP_block (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P ) [inline]
```

### 15.20.3.108 doApplyS()

```
void FFPACK::doApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

### 15.20.3.109 MatrixApplyS() [1/3]

```
void FFPACK::MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.20.3.110 MatrixApplyS()** [2/3]

```

void FFPACK::MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Sequential seq ) [inline]

```

**15.20.3.111 MatrixApplyS()** [3/3]

```

void FFPACK::MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

**15.20.3.112 PermApplyS()**

```

void FFPACK::PermApplyS (
    T * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

**15.20.3.113 doApplyT()**

```
void FFPACK::doApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.20.3.114 MatrixApplyT() [1/3]**

```
void FFPACK::MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.20.3.115 MatrixApplyT() [2/3]**

```
void FFPACK::MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Sequential seq ) [inline]
```

**15.20.3.116 MatrixApplyT()** [3/3]

```
void FFPACK::MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

**15.20.3.117 PermApplyT()**

```
void FFPACK::PermApplyT (
    T * A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.20.3.118 composePermutationsLLL()**

```
void composePermutationsLLL (
    size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]
```

Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.

**Parameters**

in, out	$P1$	a LAPACK permutation of size N
	$P2$	a LAPACK permutation of size N-R

**15.20.3.119 composePermutationsLLM()**

```
void composePermutationsLLM (
    size_t * MathP,
```

```

const size_t * P1,
const size_t * P2,
const size_t R,
const size_t N ) [inline]

```

Computes  $P1 \times \text{Diag}(I\_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $\text{MathP}$  as a  $\text{MathPermutation}$  format.

#### Parameters

out		
-----	--	--

### 15.20.3.120 composePermutationsMLM()

```

void composePermutationsMLM (
    size_t * MathP1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]

```

Computes  $\text{MathP1} \times \text{Diag}(I\_R, P2)$  where  $\text{MathP1}$  is a  $\text{MathPermutation}$  and  $P2$  a LAPACK permutation and store the result in  $\text{MathP1}$  as a  $\text{MathPermutation}$  format.

#### Parameters

in, out	<i>MathP1</i>	a $\text{MathPermutation}$ of size $N$
	<i>P2</i>	a LAPACK permutation of size $N-R$

### 15.20.3.121 cyclic\_shift\_mathPerm()

```

void cyclic_shift_mathPerm (
    size_t * P,
    const size_t s ) [inline]

```

### 15.20.3.122 cyclic\_shift\_row\_col() [1/2]

```

void FFPACK::cyclic_shift_row_col (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]

```

**15.20.3.123 cyclic\_shift\_row() [1/3]**

```
void FFPACK::cyclic_shift_row (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.124 cyclic\_shift\_row() [2/3]**

```
void FFPACK::cyclic_shift_row (
    const RNSIntegerMod< T > & F,
    typename T::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.125 cyclic\_shift\_col() [1/3]**

```
void FFPACK::cyclic_shift_col (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.126 cyclic\_shift\_col() [2/3]**

```
void FFPACK::cyclic_shift_col (
    const RNSIntegerMod< T > & F,
    typename T::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.127 PLUQ\_basecaseV3()**

```
size_t FFPACK::PLUQ_basecaseV3 (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element * A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```



**15.20.3.128 PLUQ\_basecaseV2()**

```
size_t FFPACK::PLUQ_basecaseV2 (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element * A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

**15.20.3.129 PLUQ\_basecaseCrout()**

```
size_t FFPACK::PLUQ_basecaseCrout (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

**15.20.3.130 \_PLUQ()**

```
size_t FFPACK::_PLUQ (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    size_t BCThreshold ) [inline]
```

**15.20.3.131 PLUQ() [4/6]**

```
size_t FFPACK::PLUQ (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Givaro::Integer * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    size_t BCThreshold,
    FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper ) [inline]
```

**15.20.3.132 threads\_fgemm()**

```
void FFPACK::threads_fgemm (
    const size_t m,
    const size_t n,
    const size_t r,
    int nbthreads,
    size_t * W1,
    size_t * W2,
    size_t * W3,
    size_t gamma )
```

**15.20.3.133 threads\_ftrsm()**

```
void FFPACK::threads_ftrsm (
    const size_t m,
    const size_t n,
    int nbthreads,
    size_t * t1,
    size_t * t2 )
```

**15.20.3.134 PLUQ() [5/6]**

```
size_t FFPACK::PLUQ (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & PSHelper ) [inline]
```

**15.20.3.135 fflas\_const\_cast() [1/3]**

```
rns_double_elt_ptr FFPACK::fflas_const_cast (
    rns_double_elt_cstp_ptr x ) [inline]
```

**15.20.3.136 fflas\_const\_cast() [2/3]**

```
rns_double_elt_cstp_ptr FFPACK::fflas_const_cast (
    rns_double_elt_ptr x ) [inline]
```

**15.20.3.137 cyclic\_shift\_row\_col() [2/2]**

```
void FFPACK::cyclic_shift_row_col (
    Base_t * A,
    size_t m,
    size_t n,
    size_t lda )
```

**15.20.3.138 cyclic\_shift\_row() [3/3]**

```
template INST_OR_DECL void FFPACK::cyclic_shift_row (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT * A,
    size_t m,
    size_t n,
    size_t lda )
```

**15.20.3.139 cyclic\_shift\_col() [3/3]**

```
template INST_OR_DECL void FFPACK::cyclic_shift_col (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT * A,
    size_t m,
    size_t n,
    size_t lda )
```

**15.20.3.140 applyP() [4/4]**

```
template INST_OR_DECL void FFPACK::applyP (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P )
```

**15.20.3.141 fgetrs() [3/4]**

```
template INST_OR_DECL void FFPACK::fgetrs (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.20.3.142 fgetrs() [4/4]**

```
template INST_OR_DECL FFLAS_ELT* FFPACK::fgetrs (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    FFLAS_ELT * X,
    const size_t ldX,
    const FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.20.3.143 fgesv() [3/4]**

```
template INST_OR_DECL size_t FFPACK::fgesv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.20.3.144 fgesv()** [4/4]

```
template INST_OR_DECL size_t FFPACK::fgesv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    const FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.20.3.145 ftrtri()** [2/2]

```
template INST_OR_DECL void FFPACK::ftrtri (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t threshold )
```

**15.20.3.146 trinv\_left()** [2/2]

```
template INST_OR_DECL void FFPACK::trinv_left (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * L,
    const size_t ldl,
    FFLAS_ELT * X,
    const size_t ldx )
```

**15.20.3.147 ftrtrm()** [2/2]

```
template INST_OR_DECL void FFPACK::ftrtrm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.148 PLUQ()** [6/6]

```
template INST_OR_DECL size_t FFPACK::PLUQ (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

**15.20.3.149 LUdivine()** [4/4]

```
template INST_OR_DECL size_t FFPACK::LUdivine (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const FFPACK_LU_TAG LuTag,
    const size_t cutoff )
```

**15.20.3.150 LUdivine\_small()** [2/2]

```
template INST_OR_DECL size_t FFPACK::LUdivine_small (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.151 LUdivine\_gauss()** [2/2]

```
template INST_OR_DECL size_t FFPACK::LUdivine_gauss (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.152 RowEchelonForm()** [3/3]

```
template INST_OR_DECL size_t FFPACK::RowEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.153 ReducedRowEchelonForm()** [3/3]

```
template INST_OR_DECL size_t FFPACK::ReducedRowEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.154 ColumnEchelonForm()** [3/3]

```
template INST_OR_DECL size_t FFPACK::ColumnEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.155 ReducedColumnEchelonForm()** [3/3]

```
template INST_OR_DECL size_t FFPACK::ReducedColumnEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.156 Invert()** [3/4]

```
template INST_OR_DECL FFLAS_ELT* FFPACK::Invert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    int & nullity )
```

**15.20.3.157 Invert()** [4/4]

```
template INST_OR_DECL FFLAS_ELT* FFPACK::Invert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    int & nullity )
```

**15.20.3.158 Invert2()** [2/2]

```
template INST_OR_DECL FFLAS_ELT* FFPACK::Invert2 (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    int & nullity )
```



**15.20.3.159 CharPoly() [6/8]**

```
template INST_OR_DECL std::list<Givaro::Poly1Dom<FFLAS_FIELD<FFLAS_ELT> >::Element>& FFPACK←
::CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.20.3.160 CharPoly() [7/8]**

```
template INST_OR_DECL Givaro::Poly1Dom<FFLAS_FIELD<FFLAS_ELT> >::Element& FFPACK::CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.20.3.161 CharPoly() [8/8]**

```
template INST_OR_DECL Givaro::Poly1Dom<FFLAS_FIELD<FFLAS_ELT> >::Element& FFPACK::CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.20.3.162 MinPoly() [3/4]**

```
template INST_OR_DECL std::vector<FFLAS_ELT>& FFPACK::MinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G )
```

**15.20.3.163 MinPoly()** [4/4]

```
template INST_OR_DECL std::vector<FFLAS_ELT>& FFPACK::MinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.164 MatVecMinPoly()** [2/2]

```
template INST_OR_DECL std::vector<FFLAS_ELT>& FFPACK::MatVecMinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * V,
    const size_t incv )
```

**15.20.3.165 KrylovElim()**

```
template INST_OR_DECL size_t FFPACK::KrylovElim (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const size_t deg,
    size_t * iterates,
    size_t * inviterates,
    const size_t maxit,
    size_t virt )
```

**15.20.3.166 SpecRankProfile()**

```
template INST_OR_DECL size_t FFPACK::SpecRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t deg,
    size_t * rankProfile )
```

**15.20.3.167 Rank()** [3/3]

```
template INST_OR_DECL size_t FFPACK::Rank (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.168 IsSingular()** [2/2]

```
template INST_OR_DECL bool FFPACK::IsSingular (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.169 Det()** [5/6]

```
template INST_OR_DECL FFLAS_ELT& FFPACK::Det (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT & det,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

**15.20.3.170 Det()** [6/6]

```
template INST_OR_DECL FFLAS_ELT& FFPACK::Det (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT & det,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & parH,
    size_t * P,
    size_t * Q )
```

**15.20.3.171 Solve()** [3/3]

```
template INST_OR_DECL FFLAS_ELT* FFPACK::Solve (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * x,
    const int incx,
    const FFLAS_ELT * b,
    const int incb )
```

**15.20.3.172 solveLB()** [2/2]

```
template INST_OR_DECL void FFPACK::solveLB (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * L,
    const size_t ldl,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb )
```

**15.20.3.173 solveLB2()** [2/2]

```
template INST_OR_DECL void FFPACK::solveLB2 (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * L,
    const size_t ldl,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb )
```

**15.20.3.174 RandomNullSpaceVector()** [3/3]

```
template INST_OR_DECL void FFPACK::RandomNullSpaceVector (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t incX )
```

**15.20.3.175 NullSpaceBasis()** [2/2]

```
template INST_OR_DECL size_t FFPACK::NullSpaceBasis (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& NS,
    size_t & ldn,
    size_t & NSdim )
```

**15.20.3.176 RowRankProfile()** [3/3]

```
template INST_OR_DECL size_t FFPACK::RowRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.177 ColumnRankProfile()** [3/3]

```
template INST_OR_DECL size_t FFPACK::ColumnRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.178 RowRankProfileSubmatrixIndices()** [2/2]

```
template INST_OR_DECL size_t FFPACK::RowRankProfileSubmatrixIndices (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

**15.20.3.179 ColRankProfileSubmatrixIndices()** [2/2]

```
template INST_OR_DECL size_t FFPACK::ColRankProfileSubmatrixIndices (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

**15.20.3.180 RowRankProfileSubmatrix()** [2/2]

```
template INST_OR_DECL size_t FFPACK::RowRankProfileSubmatrix (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& X,
    size_t & R )
```

**15.20.3.181 ColRankProfileSubmatrix()** [2/2]

```
template INST_OR_DECL size_t FFPACK::ColRankProfileSubmatrix (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& X,
    size_t & R )
```

**15.20.3.182 getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >()** [1/2]

```
template INST_OR_DECL void FFPACK::getTriangular< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors )
```

**15.20.3.183** `getTriangular< FFLAS_FIELD< FFLAS_ELT > >()` [2/2]

```
template INST_OR_DECL void FFPACK::getTriangular< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.184** `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >()` [1/2]

```
template INST_OR_DECL void FFPACK::getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.185** `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >()` [2/2]

```
template INST_OR_DECL void FFPACK::getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.186 getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >()**

```
template INST_OR_DECL void FFPACK::getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.187 getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >() [1/2]**

```
template INST_OR_DECL void FFPACK::getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.188 getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >() [2/2]**

```
template INST_OR_DECL void FFPACK::getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag )
```



**15.20.3.189 getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >()**

```
template INST_OR_DECL void FFPACK::getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.190 LQUPtoInverseOfFullRankMinor() [2/2]**

```
template INST_OR_DECL FFLAS_ELT* FFPACK::LQUPtoInverseOfFullRankMinor (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t rank,
    FFLAS_ELT * A_factors,
    const size_t lda,
    const size_t * QtPointer,
    FFLAS_ELT * X,
    const size_t ldx )
```

**15.20.3.191 fflas\_const\_cast() [3/3]**

```
T FFPACK::fflas_const_cast (
    CT x )
```

**15.20.3.192 failure()**

```
Failure& FFPACK::failure ( ) [inline]
```

**15.20.3.193 isOdd() [1/3]**

```
bool FFPACK::isOdd (
    const T & a ) [inline]
```

**15.20.3.194 isOdd()** [2/3]

```
bool FFPACK::isOdd (
    const float & a ) [inline]
```

**15.20.3.195 isOdd()** [3/3]

```
bool FFPACK::isOdd (
    const double & a ) [inline]
```

**15.20.3.196 NonZeroRandomMatrix()** [1/2]

```
Field::Element_ptr FFPACK::NonZeroRandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random non-zero Matrix.

Creates a  $m \times n$  matrix with random entries, and at least one of them is non zero.

**Parameters**

	$F$	field
	$m$	number of rows in A
	$n$	number of cols in A
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of A
	$G$	a random iterator

**Returns**

A.

**15.20.3.197 NonZeroRandomMatrix()** [2/2]

```
Field::Element_ptr FFPACK::NonZeroRandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
```

```

typename Field::Element_ptr A,
size_t lda ) [inline]

```

Random non-zero Matrix.

Creates a  $m \times n$  matrix with random entries, and at least one of them is non zero.

#### Parameters

	<i>F</i>	field
	<i>m</i>	number of rows in A
	<i>n</i>	number of cols in A
out	<i>A</i>	the matrix (preallocated to at least $m \times lda$ field elements)
	<i>lda</i>	leading dimension of A

#### Returns

A.

### 15.20.3.198 RandomMatrix() [1/2]

```

Field::Element_ptr FFPACK::RandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]

```

Random Matrix.

Creates a  $m \times n$  matrix with random entries.

#### Parameters

	<i>F</i>	field
	<i>m</i>	number of rows in A
	<i>n</i>	number of cols in A
out	<i>A</i>	the matrix (preallocated to at least $m \times lda$ field elements)
	<i>lda</i>	leading dimension of A
	<i>G</i>	a random iterator

#### Returns

A.

**15.20.3.199 RandomMatrix()** [2/2]

```
Field::Element_ptr FFPACK::RandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix.

Creates a  $m \times n$  matrix with random entries.

**Parameters**

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$

**Returns**

$A$ .

**15.20.3.200 RandomTriangularMatrix()** [1/2]

```
Field::Element_ptr FFPACK::RandomTriangularMatrix (
    const Field & F,
    size_t m,
    size_t n,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG Diag,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Triangular Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The `UpLo` parameter defines whether it is upper or lower triangular.

**Parameters**

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
	$UpLo$	whether $A$ is upper or lower triangular
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$
	$G$	a random iterator

## Returns

$A$ .

## 15.20.3.201 RandomTriangularMatrix() [2/2]

```
Field::Element_ptr FFPACK::RandomTriangularMatrix (
    const Field & F,
    size_t m,
    size_t n,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG Diag,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Triangular Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The `UpLo` parameter defines whether it is upper or lower triangular.

## Parameters

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
	$UpLo$	whether $A$ is upper or lower triangular
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$

## Returns

$A$ .

## 15.20.3.202 RandInt()

```
size_t FFPACK::RandInt (
    size_t a,
    size_t b ) [inline]
```

### 15.20.3.203 RandomSymmetricMatrix()

```
Field::Element_ptr FFPACK::RandomSymmetricMatrix (
    const Field & F,
    size_t n,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Symmetric Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The `UpLo` parameter defines whether it is upper or lower triangular.

#### Parameters

	$F$	field
	$n$	order of $A$
out	$A$	the matrix (preallocated to at least $n \times lda$ field elements)
	$lda$	leading dimension of $A$
	$G$	a random iterator

#### Returns

$A$ .

### 15.20.3.204 RandomMatrixWithRank() [1/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithRank (
    const Field & F,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank.

Creates an  $m \times n$  matrix with random entries and rank  $r$ .

#### Parameters

$F$	field
$m$	number of rows in $A$
$n$	number of cols in $A$
$r$	rank of the matrix to build
$A$	the matrix (preallocated to at least $m \times lda$ field elements)
$lda$	leading dimension of $A$
$G$	a random iterator

## Returns

$A$ .

## 15.20.3.205 RandomMatrixWithRank() [2/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithRank (
    const Field & F,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed rank.

Creates an  $m \times n$  matrix with random entries and rank  $r$ .

## Parameters

	$F$	field
	$m$	number of rows in $A$
	$n$	number of cols in $A$
	$r$	rank of the matrix to build
out	$A$	the matrix (preallocated to at least $m \times lda$ field elements)
	$lda$	leading dimension of $A$

## Returns

$A$ .

## 15.20.3.206 RandomIndexSubset()

```
size_t* FFPACK::RandomIndexSubset (
    size_t N,
    size_t R,
    size_t * P ) [inline]
```

Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.

## Parameters

	$N$	the cardinality of the sampling set
	$R$	the number of elements to sample
out	$P$	the output sequence (pre-allocated to at least $R$ indices)

**15.20.3.207 RandomPermutation()**

```
size_t* FFPACK::RandomPermutation (
    size_t N,
    size_t * P ) [inline]
```

Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.

**Parameters**

	$N$	the length of the permutation
out	$P$	the output permutation (pre-allocated to at least $N$ indices)

**15.20.3.208 RandomRankProfileMatrix()**

```
void FFPACK::RandomRankProfileMatrix (
    size_t M,
    size_t N,
    size_t R,
    size_t * rows,
    size_t * cols ) [inline]
```

Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.

**Parameters**

	$M$	row dimension
	$N$	column dimension
out	$rows$	the row position of each non zero element (pre-allocated)
out	$cols$	the column position of each non zero element (pre-allocated)

**15.20.3.209 swapval()**

```
void FFPACK::swapval (
    size_t k,
    size_t N,
    size_t * P,
    size_t val ) [inline]
```



**15.20.3.210 RandomSymmetricRankProfileMatrix()**

```
void FFPACK::RandomSymmetricRankProfileMatrix (
    size_t N,
    size_t R,
    size_t * rows,
    size_t * cols ) [inline]
```

Pick uniformly at random a symmetric R-subpermutation of dimension  $N \times N$  : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.

**Parameters**

	<i>N</i>	matrix order
out	<i>rows</i>	the row position of each non zero element (pre-allocated)
out	<i>cols</i>	the column position of each non zero element (pre-allocated)

**15.20.3.211 RandomMatrixWithRankandRPM() [1/2]**

```
Field::Element_ptr FFPACK::RandomMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

<i>F</i>	field
<i>m</i>	number of rows in A
<i>n</i>	number of cols in A
<i>r</i>	rank of the matrix to build
<i>A</i>	the matrix (preallocated to at least $m \times lda$ field elements)
<i>lda</i>	leading dimension of A
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots
<i>G</i>	a random iterator

**Returns**

A.

**15.20.3.212 RandomMatrixWithRankandRPM()** [2/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP ) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

$F$	field
$m$	number of rows in A
$n$	number of cols in A
$r$	rank of the matrix to build
$A$	the matrix (preallocated to at least $m \times lda$ field elements)
$lda$	leading dimension of A
$RRP$	the R dimensional array with row positions of the rank profile matrix' pivots
$CRP$	the R dimensional array with column positions of the rank profile matrix' pivots

**Returns**

A.

**15.20.3.213 RandomSymmetricMatrixWithRankandRPM()** [1/2]

```
Field::Element_ptr FFPACK::RandomSymmetricMatrixWithRankandRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP,
    RandIter & G ) [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .

**Parameters**

$F$	field
$n$	order of A
$r$	rank of A
$A$	the matrix (preallocated to at least $n \times lda$ field elements)

## Parameters

<i>lda</i>	leading dimension of $A$
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots
<i>G</i>	a random iterator

## Returns

$A$ .

## 15.20.3.214 RandomSymmetricMatrixWithRankandRPM() [2/2]

```
Field::Element_ptr FFPACK::RandomSymmetricMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP ) [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .

## Parameters

<i>F</i>	field
<i>n</i>	order of $A$
<i>r</i>	rank of $A$
<i>A</i>	the matrix (preallocated to at least $n \times lda$ field elements)
<i>lda</i>	leading dimension of $A$
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots

## Returns

$A$ .

## 15.20.3.215 RandomMatrixWithRankandRandomRPM() [1/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithRankandRandomRPM (
    const Field & F,
    size_t M,
```

```

    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]

```

Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

#### Parameters

$F$	field
$m$	number of rows in $A$
$n$	number of cols in $A$
$r$	rank of the matrix to build
$A$	the matrix (preallocated to at least $m \times lda$ field elements)
$lda$	leading dimension of $A$
$G$	a random iterator

#### Returns

$A$ .

### 15.20.3.216 RandomMatrixWithRankandRandomRPM() [2/2]

```

Field::Element_ptr FFPACK::RandomMatrixWithRankandRandomRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda ) [inline]

```

Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

#### Parameters

$F$	field
$m$	number of rows in $A$
$n$	number of cols in $A$
$r$	rank of the matrix to build
$A$	the matrix (preallocated to at least $m \times lda$ field elements)
$lda$	leading dimension of $A$

#### Returns

$A$ .

**15.20.3.217 RandomSymmetricMatrixWithRankandRandomRPM()** [1/2]

```
Field::Element_ptr FFPACK::RandomSymmetricMatrixWithRankandRandomRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

**Parameters**

$F$	field
$n$	order of $A$
$r$	rank of $A$
$A$	the matrix (preallocated to at least $n \times lda$ field elements)
$lda$	leading dimension of $A$
$G$	a random iterator

**Returns**

$A$ .

**15.20.3.218 RandomSymmetricMatrixWithRankandRandomRPM()** [2/2]

```
Field::Element_ptr FFPACK::RandomSymmetricMatrixWithRankandRandomRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

**Parameters**

$F$	field
$n$	order of $A$
$r$	rank of $A$
$A$	the matrix (preallocated to at least $n \times lda$ field elements)
$lda$	leading dimension of $A$

**Returns**

$A$ .

**15.20.3.219 RandomMatrixWithDet()** [1/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithDet (
    const Field & F,
    size_t n,
    const typename Field::Element d,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed det.

Creates a  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

$F$	field
$d$	the prescribed value for the determinant of A
$n$	number of cols in A
$A$	the matrix to be generated (preallocated to at least $n \times lda$ field elements)
$lda$	leading dimension of A

**Returns**

A.

**15.20.3.220 RandomMatrixWithDet()** [2/2]

```
Field::Element_ptr FFPACK::RandomMatrixWithDet (
    const Field & F,
    size_t n,
    const typename Field::Element d,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed det.

Creates a  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

$F$	field
$d$	the prescribed value for the determinant of A
$n$	number of cols in A
$A$	the matrix to be generated (preallocated to at least $n \times lda$ field elements)
$lda$	leading dimension of A

## Returns

A.

**15.20.3.221 maxFieldElt()**

```
Givaro::Integer FFPACK::maxFieldElt ( )
```

**15.20.3.222 maxFieldElt< Givaro::ZRing< Givaro::Integer > >()**

```
Givaro::Integer FFPACK::maxFieldElt< Givaro::ZRing< Givaro::Integer > > ( )
```

**15.20.3.223 chooseField()**

```
Field* FFPACK::chooseField (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.20.3.224 chooseField< Givaro::ZRing< int32\_t > >()**

```
Givaro::ZRing<int32_t>* FFPACK::chooseField< Givaro::ZRing< int32_t > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.20.3.225 chooseField< Givaro::ZRing< int64\_t > >()**

```
Givaro::ZRing<int64_t>* FFPACK::chooseField< Givaro::ZRing< int64_t > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

### 15.20.3.226 chooseField< Givaro::ZRing< float > >()

```
Givaro::ZRing<float>* FFPACK::chooseField< Givaro::ZRing< float > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

### 15.20.3.227 chooseField< Givaro::ZRing< double > >()

```
Givaro::ZRing<double>* FFPACK::chooseField< Givaro::ZRing< double > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

## 15.21 FFPACK::Protected Namespace Reference

### Functions

- template<class Field >  
size\_t LUdivine\_construct (const Field &F, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, typename Field::Element\_ptr u, const size\_t incu, size\_t \*P, bool computeX, const FFPACK\_MINPOLY\_TAG MinTag=FFpackDense, const size\_t kg\_mc=0, const size\_t kg\_mb=0, const size\_t kg\_j=0)
- template<class Field >  
size\_t GaussJordan (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag)  
*Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.*
- template<class Field , class Polynomial >  
std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda)
- template<class Field , class Polynomial >  
int KGFast (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*kg\_mc, size\_t \*kg\_mb, size\_t \*kg\_j)
- template<class Field , class Polynomial >  
std::list< Polynomial > & KGFast\_generalized (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)
- template<class Field >  
void fgemv\_kgf (const Field &F, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY, const size\_t kg\_mc, const size\_t kg\_mb, const size\_t kg\_j)
- template<class Field , class Polynomial , class RandIter >  
std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr U, const size\_t ldu, RandIter &G)
- template<class Field , class Polynomial >  
std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)
- template<class PolRing >  
void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, size\_t &Nb, typename PolRing::Domain\_t::Element\_ptr &B, size\_t &ldb, typename PolRing::Domain\_t::RandIter &g, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)



- `template<class PolRing >`  
`std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const size_t degree)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr v, const size_t incv, typename Field::Element_ptr K, const size_t ldk, size_t *P)`
- `template<class Field , class Polynomial >`  
`Polynomial & Hybrid_KGF_LUK_MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, size_t *P, const FFPACK_MINPOLY_TAG MinTag=FFPACK::FfpackDense, const size_t kg_mc=0, const size_t kg_mb=0, const size_t kg_j=0)`
- `template<class Field >`  
`size_t updatedD (const Field &F, size_t *d, size_t k, std::vector< std::vector< typename Field::Element > > &minpt)`
- `template<class Field >`  
`size_t newD (const Field &F, size_t *d, bool &KeepOn, const size_t l, const size_t N, typename Field::Element_ptr X, const size_t *Q, std::vector< std::vector< typename Field::Element > > &minpt)`
- `template<class Field >`  
`void CompressRows (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQK (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRowsQK (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQA (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRowsQA (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK::FFPACK_MINPOLY_TAG MinTag, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)`

### 15.21.1 Function Documentation

### 15.21.1.1 LUdivine\_construct() [1/2]

```
size_t FFPACK::Protected::LUdivine_construct (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    typename Field::Element_ptr u,
    const size_t incu,
    size_t * P,
    bool computeX,
    const FFPACK_MINPOLY_TAG MinTag = FfpackDense,
    const size_t kg_mc = 0,
    const size_t kg_mb = 0,
    const size_t kg_j = 0 )
```

### 15.21.1.2 GaussJordan()

```
size_t GaussJordan (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t colbeg,
    const size_t rowbeg,
    const size_t colsize,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.

#### Bibliography

- Algorithm 2.8 of A. Storjohann Thesis 2000,
- Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013

#### Parameters

	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
in, out	<i>A</i>	an m x n matrix
	<i>lda</i>	leading dimension of A
	<i>P</i>	row permutation
	<i>Q</i>	column permutation
	<i>LuTag</i>	set the base case to a Tile (FfpackGaussJordanTile) or Slab (FfpackGaussJordanSlab) recursive RedEchelon

where the transformation matrix is stored at the pivot column position

### 15.21.1.3 KellerGehrig()

```
std::list<Polynomial>& FFPACK::Protected::KellerGehrig (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

### 15.21.1.4 KGFast()

```
int FFPACK::Protected::KGFast (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * kg_mc,
    size_t * kg_mb,
    size_t * kg_j )
```

### 15.21.1.5 KGFast\_generalized()

```
std::list<Polynomial>& FFPACK::Protected::KGFast_generalized (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

### 15.21.1.6 fgemv\_kgf()

```
void FFPACK::Protected::fgemv_kgf (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    const size_t kg_mc,
    const size_t kg_mb,
    const size_t kg_j )
```

### 15.21.1.7 LUKrylov()

```
std::list<Polynomial>& FFPACK::Protected::LUKrylov (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
```

```

typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr U,
const size_t ldu,
RandIter & G )

```

#### 15.21.1.8 Danilevski()

```

std::list<Polynomial>& FFPACK::Protected::Danilevski (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )

```

#### 15.21.1.9 RandomKrylovPrecond()

```

void RandomKrylovPrecond (
    const PolRing & PR,
    std::list< typename PolRing::Element > & completedFactors,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    size_t & Nb,
    typename PolRing::Domain_t::Element_ptr & B,
    size_t & ldb,
    typename PolRing::Domain_t::RandIter & g,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

**Todo** swap to save space ??

**Todo** don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

**Todo** swap to save space ??

**Todo** don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

#### 15.21.1.10 ArithProg()

```

std::list< typename PolRing::Element > & ArithProg (
    const PolRing & PR,
    std::list< typename PolRing::Element > & frobeniusForm,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    const size_t degree ) [inline]

```

#### 15.21.1.11 LUKrylov\_KGFast()

```

std::list<Polynomial>& FFPACK::Protected::LUKrylov_KGFast (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,

```

```

typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr X,
const size_t ldX )

```

#### 15.21.1.12 MatVecMinPoly()

```

Polynomial& FFPACK::Protected::MatVecMinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr v,
    const size_t incv,
    typename Field::Element_ptr K,
    const size_t ldK,
    size_t * P )

```

#### 15.21.1.13 Hybrid\_KGF\_LUK\_MinPoly()

```

Polynomial& FFPACK::Protected::Hybrid_KGF_LUK_MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    size_t * P,
    const FFPACK_MINPOLY_TAG MinTag = FFPACK::FfpackDense,
    const size_t kg_mc = 0,
    const size_t kg_mb = 0,
    const size_t kg_j = 0 )

```

#### 15.21.1.14 updateD()

```

size_t FFPACK::Protected::updateD (
    const Field & F,
    size_t * d,
    size_t k,
    std::vector< std::vector< typename Field::Element > > & minpt )

```

#### 15.21.1.15 newD()

```

size_t FFPACK::Protected::newD (
    const Field & F,
    size_t * d,
    bool & KeepOn,
    const size_t l,
    const size_t N,
    typename Field::Element_ptr X,
    const size_t * Q,
    std::vector< std::vector< typename Field::Element > > & minpt )

```

**15.21.1.16 CompressRows()**

```
void FFPACK::Protected::CompressRows (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]
```

**15.21.1.17 CompressRowsQK()**

```
void FFPACK::Protected::CompressRowsQK (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t deg,
    const size_t nb_blocs ) [inline]
```

**15.21.1.18 DeCompressRows()**

```
void FFPACK::Protected::DeCompressRows (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]
```

**15.21.1.19 DeCompressRowsQK()**

```
void FFPACK::Protected::DeCompressRowsQK (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t deg,
    const size_t nb_blocs ) [inline]
```

**15.21.1.20 CompressRowsQA()**

```
void FFPACK::Protected::CompressRowsQA (
    Field & F,
```

```

const size_t M,
typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr tmp,
const size_t ldtmp,
const size_t * d,
const size_t nb_blocs ) [inline]

```

#### 15.21.1.21 DeCompressRowsQA()

```

void FFPACK::Protected::DeCompressRowsQA (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]

```

#### 15.21.1.22 LUdivine\_construct() [2/2]

```

size_t FFPACK::Protected::LUdivine_construct (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    typename Field::Element_ptr u,
    const size_t incu,
    size_t * P,
    bool computeX,
    const FFPACK::FFPACK_MINPOLY_TAG MinTag,
    const size_t kg_mc,
    const size_t kg_mb,
    const size_t kg_j )

```

## 15.22 Givaro Namespace Reference

### Data Structures

- class [ModularBalanced](#)
- class [Montgomery](#)

## 15.23 MKL\_CONFIG Namespace Reference

## 15.24 Reclnt Namespace Reference

### Data Structures

- class [rint](#)

- class [ruint](#)



## Chapter 16

# Data Structure Documentation

### 16.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference

#### Public Types

- typedef [MMHelperAlgo::Winograd value](#)

#### 16.1.1 Member Typedef Documentation

##### 16.1.1.1 value

typedef [MMHelperAlgo::Winograd value](#)

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

### 16.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference

#### Public Types

- typedef [MMHelperAlgo::Classic value](#)

#### 16.2.1 Member Typedef Documentation

##### 16.2.1.1 value

typedef [MMHelperAlgo::Classic value](#)

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

### 16.3 ArbitraryPrecIntTag Struct Reference

Arbitrary precision integers: GMP.

```
#include <field-traits.h>
```

### 16.3.1 Detailed Description

Arbitrary precision integers: GMP.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.4 `AreEqual< X, Y >` Class Template Reference

```
#include <fflas_enum.h>
```

### Static Public Attributes

- static const bool [value](#) = false

### 16.4.1 Field Documentation

#### 16.4.1.1 `value`

```
const bool value = false [static]
```

The documentation for this class was generated from the following file:

- [fflas\\_enum.h](#)

## 16.5 `AreEqual< X, X >` Class Template Reference

```
#include <fflas_enum.h>
```

### Static Public Attributes

- static const bool [value](#) = true

### 16.5.1 Field Documentation

#### 16.5.1.1 `value`

```
const bool value = true [static]
```

The documentation for this class was generated from the following file:

- [fflas\\_enum.h](#)

## 16.6 Argument Struct Reference

```
#include <args-parser.h>
```

### Data Fields

- char [c](#)
- const char \* [example](#)
- const char \* [helpString](#)
- [ArgumentType](#) type
- void \* [data](#)

## 16.6.1 Field Documentation

### 16.6.1.1 c

char c

### 16.6.1.2 example

const char\* example

### 16.6.1.3 helpString

const char\* helpString

### 16.6.1.4 type

[ArgumentType](#) type

### 16.6.1.5 data

void\* data

The documentation for this struct was generated from the following file:

- [args-parser.h](#)

## 16.7 associatedDelayedField< Field > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [Field](#) field
- typedef [Field](#) & [type](#)

### 16.7.1 Member Typedef Documentation

#### 16.7.1.1 field

typedef [Field](#) field

#### 16.7.1.2 type

typedef [Field](#)& [type](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.8 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FFPACK::RNSInteger< RNS >](#) [field](#)
- typedef [FFPACK::RNSInteger< RNS >](#) [type](#)

### 16.8.1 Member Typedef Documentation

#### 16.8.1.1 field

```
typedef FFPACK::RNSInteger<RNS> field
```

#### 16.8.1.2 type

```
typedef FFPACK::RNSInteger<RNS> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.9 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

### 16.9.1 Member Typedef Documentation

#### 16.9.1.1 field

```
typedef Givaro::ZRing<T> field
```

#### 16.9.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.10 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

## 16.10.1 Member Typedef Documentation

### 16.10.1.1 field

```
typedef Givaro::ZRing<T> field
```

### 16.10.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.11 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

## 16.11.1 Member Typedef Documentation

### 16.11.1.1 field

```
typedef Givaro::ZRing<T> field
```

### 16.11.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.12 Auto Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.13 Bini Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.14 Block Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.15 callLUdivine\_small< Element > Class Template Reference

### Public Member Functions

- `template<class Field >`  
`size_t operator()` (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const `size_t` M, const `size_t` N, typename [Field::Element\\_ptr](#) A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)

### 16.15.1 Member Function Documentation

#### 16.15.1.1 operator>()()

```
size_t operator() (
    const Field & F,
    const FFLAS::FFLAS\_DIAG Diag,
    const FFLAS::FFLAS\_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element\_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK\_LU\_TAG LuTag ) [inline]
```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

## 16.16 callLUdivine\_small< double > Class Reference

### Public Member Functions

- `template<class Field >`  
`size_t operator()` (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const `size_t` M, const `size_t` N, typename [Field::Element\\_ptr](#) A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)

### 16.16.1 Member Function Documentation

#### 16.16.1.1 operator>()()

```
size_t operator() (
    const Field & F,
    const FFLAS::FFLAS\_DIAG Diag,
    const FFLAS::FFLAS\_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element\_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK\_LU\_TAG LuTag ) [inline]
```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

## 16.17 callLUdivine\_small< float > Class Reference

### Public Member Functions

- `template<class Field >`  
`size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`

### 16.17.1 Member Function Documentation

#### 16.17.1.1 operator>()()

```
size_t operator() (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

## 16.18 CharpolyFailed Class Reference

```
#include <ffpack.h>
```

The documentation for this class was generated from the following file:

- [ffpack.h](#)

## 16.19 Checker\_Empty< Field > Struct Template Reference

```
#include <checker_empty.h>
```

### Public Member Functions

- `template<typename... Params>`  
`Checker_Empty (Params... parameters)`
- `template<typename... Params>`  
`bool check (Params... parameters)`

### 16.19.1 Constructor & Destructor Documentation

#### 16.19.1.1 Checker\_Empty()

```
Checker_Empty (
    Params... parameters ) [inline]
```

## 16.19.2 Member Function Documentation

### 16.19.2.1 check()

```
bool check (
    Params... parameters ) [inline]
```

The documentation for this struct was generated from the following file:

- [checker\\_empty.h](#)

## 16.20 CheckerImplem\_charpoly< Field, Polynomial > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_charpoly](#) (const [Field](#) &F\_, const size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda\_)
- [CheckerImplem\\_charpoly](#) (typename [Field::RandIter](#) &G, const size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda\_)
- [~CheckerImplem\\_charpoly](#) ()
- bool [check](#) (Polynomial &g)

### 16.20.1 Constructor & Destructor Documentation

#### 16.20.1.1 CheckerImplem\_charpoly() [1/2]

```
CheckerImplem_charpoly (
    const Field & F_,
    const size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda_ ) [inline]
```

#### 16.20.1.2 CheckerImplem\_charpoly() [2/2]

```
CheckerImplem_charpoly (
    typename Field::RandIter & G,
    const size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda_ ) [inline]
```

#### 16.20.1.3 ~CheckerImplem\_charpoly()

```
~CheckerImplem_charpoly ( ) [inline]
```

## 16.20.2 Member Function Documentation

### 16.20.2.1 check()

```
bool check (
    Polynomial & g ) [inline]
```

The documentation for this class was generated from the following file:



- [checker\\_charpoly.inl](#)

## 16.21 CheckerImplem\_Det< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_Det](#) (const [Field](#) &F\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [CheckerImplem\\_Det](#) (typename [Field::RandIter](#) &G, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [~CheckerImplem\\_Det](#) ()
- bool [check](#) (const typename [Field::Element](#) &det, typename [Field::ConstElement\\_ptr](#) LU, size\_t lda, size\_t \*P, size\_t \*Q) const

*check if the Det factorization is correct.*

### 16.21.1 Constructor & Destructor Documentation

#### 16.21.1.1 CheckerImplem\_Det() [1/2]

```
CheckerImplem_Det (
    const Field & F_,
    size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda ) [inline]
```

#### 16.21.1.2 CheckerImplem\_Det() [2/2]

```
CheckerImplem_Det (
    typename Field::RandIter & G,
    size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda ) [inline]
```

#### 16.21.1.3 ~CheckerImplem\_Det()

```
~CheckerImplem_Det ( ) [inline]
```

### 16.21.2 Member Function Documentation

#### 16.21.2.1 check()

```
bool check (
    const typename Field::Element & det,
    typename Field::ConstElement\_ptr LU,
    size_t lda,
    size_t * P,
    size_t * Q ) const [inline]
```

check if the Det factorization is correct.

Needs matrix in LU form

#### Parameters

<i>LU,storage</i>	for L and U
-------------------	-------------

## Parameters

<i>det</i>	
<i>P</i>	
<i>Q</i>	

The documentation for this class was generated from the following file:

- [checker\\_det.inl](#)

## 16.22 CheckerImplem\_fgemm< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_fgemm](#) (const [Field](#) &F\_, const size\_t m\_, const size\_t n\_, const size\_t k\_, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc\_)
- [CheckerImplem\\_fgemm](#) (typename [Field::RandIter](#) &G, const size\_t m\_, const size\_t n\_, const size\_t k\_, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc\_)
- [~CheckerImplem\\_fgemm](#) ()
- bool [check](#) (const [FFLAS::FFLAS\\_TRANSPOSE](#) ta, const [FFLAS::FFLAS\\_TRANSPOSE](#) tb, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::ConstElement\\_ptr](#) C)

### 16.22.1 Constructor & Destructor Documentation

#### 16.22.1.1 CheckerImplem\_fgemm() [1/2]

```
CheckerImplem_fgemm (
    const Field & F_,
    const size_t m_,
    const size_t n_,
    const size_t k_,
    const typename Field::Element beta,
    typename Field::Element\_ptr C,
    const size_t ldc_ ) [inline]
```

#### 16.22.1.2 CheckerImplem\_fgemm() [2/2]

```
CheckerImplem_fgemm (
    typename Field::RandIter & G,
    const size_t m_,
    const size_t n_,
    const size_t k_,
    const typename Field::Element beta,
    typename Field::Element\_ptr C,
    const size_t ldc_ ) [inline]
```

#### 16.22.1.3 ~CheckerImplem\_fgemm()

```
~CheckerImplem_fgemm ( ) [inline]
```

### 16.22.2 Member Function Documentation

### 16.22.2.1 check()

```
bool check (
    const FFLAS::FFLAS_TRANSPOSE ta,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::ConstElement_ptr C ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_fgemm.inl](#)

## 16.23 CheckerImplem\_ftrsm< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_ftrsm](#) (const [Field](#) &F\_, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)
- [CheckerImplem\\_ftrsm](#) (typename [Field::RandIter](#) &G, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)
- [~CheckerImplem\\_ftrsm](#) ()
- bool [check](#) (const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_UPLO](#) uplo, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, size\_t ldx)

### 16.23.1 Constructor & Destructor Documentation

#### 16.23.1.1 CheckerImplem\_ftrsm() [1/2]

```
CheckerImplem_ftrsm (
    const Field & F_,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

#### 16.23.1.2 CheckerImplem\_ftrsm() [2/2]

```
CheckerImplem_ftrsm (
    typename Field::RandIter & G,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

#### 16.23.1.3 ~CheckerImplem\_ftrsm()

```
~CheckerImplem_ftrsm ( ) [inline]
```

## 16.23.2 Member Function Documentation

### 16.23.2.1 check()

```
bool check (
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_UPLO uplo,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const FFLAS::FFLAS_DIAG diag,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    const typename Field::ConstElement_ptr X,
    size_t ldx ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_ftsm.inl](#)

## 16.24 CheckerImplem\_invert< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_invert](#) (const [Field](#) &F\_, const size\_t m\_, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda\_)
- [CheckerImplem\\_invert](#) (typename [Field::RandIter](#) &G, const size\_t m\_, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda\_)
- [~CheckerImplem\\_invert](#) ()
- bool [check](#) (typename [Field::ConstElement\\_ptr](#) A, int nullity)

### 16.24.1 Constructor & Destructor Documentation

#### 16.24.1.1 CheckerImplem\_invert() [1/2]

```
CheckerImplem_invert (
    const Field & F_,
    const size_t m_,
    typename Field::ConstElement_ptr A,
    const size_t lda_ ) [inline]
```

#### 16.24.1.2 CheckerImplem\_invert() [2/2]

```
CheckerImplem_invert (
    typename Field::RandIter & G,
    const size_t m_,
    typename Field::ConstElement_ptr A,
    const size_t lda_ ) [inline]
```

#### 16.24.1.3 ~CheckerImplem\_invert()

```
~CheckerImplem_invert ( ) [inline]
```

## 16.24.2 Member Function Documentation

### 16.24.2.1 check()

```
bool check (
    typename Field::ConstElement_ptr A,
    int nullity ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_invert.inl](#)

## 16.25 CheckerImplem\_PLUQ< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_PLUQ](#) (const [Field](#) &F\_, size\_t m\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [CheckerImplem\\_PLUQ](#) (typename [Field::RandIter](#) &G, size\_t m\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [~CheckerImplem\\_PLUQ](#) ()
- bool [check](#) (typename [Field::ConstElement\\_ptr](#) A, size\_t lda, const [FFLAS::FFLAS\\_DIAG](#) Diag, size\_t r, size\_t \*P, size\_t \*Q) const  
*check if the PLUQ factorization is correct.*

### 16.25.1 Constructor & Destructor Documentation

#### 16.25.1.1 CheckerImplem\_PLUQ() [1/2]

```
CheckerImplem_PLUQ (
    const Field & F_,
    size_t m_,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

#### 16.25.1.2 CheckerImplem\_PLUQ() [2/2]

```
CheckerImplem_PLUQ (
    typename Field::RandIter & G,
    size_t m_,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

#### 16.25.1.3 ~CheckerImplem\_PLUQ()

```
~CheckerImplem_PLUQ ( ) [inline]
```

## 16.25.2 Member Function Documentation

### 16.25.2.1 check()

```
bool check (
    typename Field::ConstElement_ptr A,
    size_t lda,
    const FFLAS::FFLAS_DIAG Diag,
    size_t r,
    size_t * P,
    size_t * Q ) const [inline]
```

check if the PLUQ factorization is correct.

Returns true if  $w - P(L(U(Q.v))) == 0$

#### Parameters

<i>A</i>	
<i>r</i>	
<i>P</i>	
<i>Q</i>	

The documentation for this class was generated from the following file:

- [checker\\_pluq.inl](#)

## 16.26 Classic Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.27 Column Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.28 CompactElement< Element > Struct Template Reference

### Public Types

- typedef Element [type](#)

### 16.28.1 Member Typedef Documentation

#### 16.28.1.1 type

```
typedef Element type
```

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.29 CompactElement< double > Struct Reference

### Public Types

- typedef [int32\\_t](#) [type](#)

## 16.29.1 Member Typedef Documentation

### 16.29.1.1 type

typedef [int32\\_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.30 CompactElement< float > Struct Reference

### Public Types

- typedef [int16\\_t](#) type

## 16.30.1 Member Typedef Documentation

### 16.30.1.1 type

typedef [int16\\_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.31 CompactElement< int16\_t > Struct Reference

### Public Types

- typedef [int8\\_t](#) type

## 16.31.1 Member Typedef Documentation

### 16.31.1.1 type

typedef [int8\\_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.32 CompactElement< int32\_t > Struct Reference

### Public Types

- typedef [int16\\_t](#) type

## 16.32.1 Member Typedef Documentation

### 16.32.1.1 type

typedef [int16\\_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.33 CompactElement< int64\_t > Struct Reference

### Public Types

- typedef [int32\\_t](#) [type](#)

### 16.33.1 Member Typedef Documentation

#### 16.33.1.1 type

typedef [int32\\_t](#) [type](#)

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.34 compatible\_data\_type< Field > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = true

### 16.34.1 Field Documentation

#### 16.34.1.1 value

constexpr bool value = true [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.35 compatible\_data\_type< Givaro::ZRing< double > > Struct Reference

### Static Public Attributes

- static constexpr bool [value](#) = false

### 16.35.1 Field Documentation

#### 16.35.1.1 value

constexpr bool value = false [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.36 compatible\_data\_type< Givaro::ZRing< float > > Struct Reference

### Static Public Attributes

- static constexpr bool [value](#) = false



## 16.36.1 Field Documentation

### 16.36.1.1 value

constexpr bool value = false [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.37 Compose< H1, H2 > Struct Template Reference

### Public Member Functions

- [Compose](#) ()
- [Compose](#) (const [Compose](#) &other)
- [Compose](#) (const [Sequential](#) &S)
- [Compose](#) (size\_t th1, size\_t th2)
- [Compose](#) (const H1 &o1, const H2 &o2)
- H1 [first\\_component](#) () const
- H2 [second\\_component](#) () const

### Friends

- std::ostream & [operator<<](#) (std::ostream &o, const [Compose](#) &c)

## 16.37.1 Constructor & Destructor Documentation

### 16.37.1.1 Compose() [1/5]

[Compose](#) ( ) [inline]

### 16.37.1.2 Compose() [2/5]

[Compose](#) (   
 const [Compose](#)< H1, H2 > & other ) [inline]

### 16.37.1.3 Compose() [3/5]

[Compose](#) (   
 const [Sequential](#) & S ) [inline]

### 16.37.1.4 Compose() [4/5]

[Compose](#) (   
 size\_t th1,   
 size\_t th2 ) [inline]

### 16.37.1.5 Compose() [5/5]

[Compose](#) (   
 const H1 & o1,   
 const H2 & o2 ) [inline]

## 16.37.2 Member Function Documentation

### 16.37.2.1 first\_component()

```
H1 first_component ( ) const [inline]
```

### 16.37.2.2 second\_component()

```
H2 second_component ( ) const [inline]
```

## 16.37.3 Friends And Related Function Documentation

### 16.37.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & o,
    const Compose< H1, H2 > & c ) [friend]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.38 Const\_int\_t< n > Class Template Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

## 16.39 Const\_uint\_t< n > Class Template Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

## 16.40 Simd128\_impl< true, true, false, 2 >::Converter Union Reference

### Data Fields

- [vect\\_t](#) v
- [scalar\\_t](#) t [vect\_size]

### 16.40.1 Field Documentation

#### 16.40.1.1 v

```
vect\_t v
```

**16.40.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int16.inl](#)

**16.41 Simd128\_impl< true, true, false, 4 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

**16.41.1 Field Documentation****16.41.1.1 v**

`vect_t v`

**16.41.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int32.inl](#)

**16.42 Simd128\_impl< true, true, false, 8 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

**16.42.1 Field Documentation****16.42.1.1 v**

`vect_t v`

**16.42.1.2 t**

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int64.inl](#)

**16.43 Simd128\_impl< true, true, true, 2 >::Converter Union Reference****Data Fields**

- [vect\\_t v](#)
- [scalar\\_t t\[vect\\_size\]](#)

### 16.43.1 Field Documentation

#### 16.43.1.1 v

[vect\\_t](#) v

#### 16.43.1.2 t

[scalar\\_t](#) t[[vect\\_size](#)]

The documentation for this union was generated from the following file:

- [simd128\\_int16.inl](#)

## 16.44 Simd128\_impl< true, true, true, 4 >::Converter Union Reference

### Data Fields

- [vect\\_t](#) v
- [scalar\\_t](#) t[[vect\\_size](#)]

### 16.44.1 Field Documentation

#### 16.44.1.1 v

[vect\\_t](#) v

#### 16.44.1.2 t

[scalar\\_t](#) t[[vect\\_size](#)]

The documentation for this union was generated from the following file:

- [simd128\\_int32.inl](#)

## 16.45 Simd128\_impl< true, true, true, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t](#) v
- [scalar\\_t](#) t[[vect\\_size](#)]

### 16.45.1 Field Documentation

#### 16.45.1.1 v

[vect\\_t](#) v

#### 16.45.1.2 t

[scalar\\_t](#) t[[vect\\_size](#)]

The documentation for this union was generated from the following file:

- [simd128\\_int64.inl](#)

## 16.46 Simd256\_impl< true, true, false, 2 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.46.1 Field Documentation

##### 16.46.1.1 v

[vect\\_t v](#)

##### 16.46.1.2 t

[scalar\\_t t \[vect\\_size\]](#)

The documentation for this union was generated from the following file:

- [simd256\\_int16.inl](#)

## 16.47 Simd256\_impl< true, true, false, 4 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.47.1 Field Documentation

##### 16.47.1.1 v

[vect\\_t v](#)

##### 16.47.1.2 t

[scalar\\_t t](#)

The documentation for this union was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.48 Simd256\_impl< true, true, false, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.48.1 Field Documentation

**16.48.1.1 v**

`vect_t` v

**16.48.1.2 t**

`scalar_t` t[`vect_size`]

The documentation for this union was generated from the following file:

- [simd256\\_int64.inl](#)

**16.49 Simd256\_impl< true, true, true, 2 >::Converter Union Reference****Data Fields**

- [vect\\_t](#) v
- [scalar\\_t](#) t[`vect_size`]

**16.49.1 Field Documentation****16.49.1.1 v**

`vect_t` v

**16.49.1.2 t**

`scalar_t` t[`vect_size`]

The documentation for this union was generated from the following file:

- [simd256\\_int16.inl](#)

**16.50 Simd256\_impl< true, true, true, 4 >::Converter Union Reference****Data Fields**

- [vect\\_t](#) v
- [scalar\\_t](#) t[`vect_size`]

**16.50.1 Field Documentation****16.50.1.1 v**

`vect_t` v

**16.50.1.2 t**

`scalar_t` t

The documentation for this union was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.51 Simd256\_impl< true, true, true, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.51.1 Field Documentation

##### 16.51.1.1 v

[vect\\_t v](#)

##### 16.51.1.2 t

[scalar\\_t t \[vect\\_size\]](#)

The documentation for this union was generated from the following file:

- [simd256\\_int64.inl](#)

## 16.52 Simd512\_impl< true, true, false, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.52.1 Field Documentation

##### 16.52.1.1 v

[vect\\_t v](#)

##### 16.52.1.2 t

[scalar\\_t t \[vect\\_size\]](#)

The documentation for this union was generated from the following file:

- [simd512\\_int64.inl](#)

## 16.53 Simd512\_impl< true, true, true, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t \[vect\\_size\]](#)

#### 16.53.1 Field Documentation

**16.53.1.1 v**

`vect_t` [v](#)

**16.53.1.2 t**

`scalar_t` [t](#)[`vect_size`]

The documentation for this union was generated from the following file:

- [simd512\\_int64.inl](#)

**16.54 ConvertTo< T > Struct Template Reference**

Force conversion to appropriate element type of ElementCategory T.

`#include <field-traits.h>`

**16.54.1 Detailed Description**

`template<class T>`

`struct FFLAS::ModeCategories::ConvertTo< T >`

Force conversion to appropriate element type of ElementCategory T.

e.g.

- `ConvertTo<ElementCategories::MachineFloatTag>` tries conversion of `Modular<int>` to `Modular<double>`
- `ConvertTo<ElementCategories::FixedPrecIntTag>` tries conversion of `Modular<Integer>` to `Modular<RecInt<K>>`
- `ConvertTo<ElementCategories::ArbitraryPrecIntTag>` tries conversion of `Modular<Integer>` to `RNSInteger`

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.55 Coo< ValT, IdxT > Struct Template Reference****Public Types**

- using `Self` = `Coo< ValT, IdxT >`

**Public Member Functions**

- `Coo` (`ValT` v, `IdxT` r, `IdxT` c)
- `Coo` ()=default
- `Coo` (const `Self` &)=default
- `Coo` (`Self` &&)=default
- `Self` & `operator=` (const `Self` &)=default
- `Self` & `operator=` (`Self` &&)=default

**Data Fields**

- `ValT` `val` = 0
- `IdxT` `row` = 0
- `IdxT` `col` = 0

**16.55.1 Member Typedef Documentation**



### 16.55.1.1 Self

```
using Self = Coo<ValT, IdxT>
```

## 16.55.2 Constructor & Destructor Documentation

### 16.55.2.1 `Coo()` [1/4]

```
Coo (
    ValT v,
    IdxT r,
    IdxT c ) [inline]
```

### 16.55.2.2 `Coo()` [2/4]

```
Coo ( ) [default]
```

### 16.55.2.3 `Coo()` [3/4]

```
Coo (
    const Self & ) [default]
```

### 16.55.2.4 `Coo()` [4/4]

```
Coo (
    Self && ) [default]
```

## 16.55.3 Member Function Documentation

### 16.55.3.1 `operator=()` [1/2]

```
Self& operator= (
    const Self & ) [default]
```

### 16.55.3.2 `operator=()` [2/2]

```
Self& operator= (
    Self && ) [default]
```

## 16.55.4 Field Documentation

### 16.55.4.1 `val`

```
ValT val = 0
```

### 16.55.4.2 `row`

```
IdxT row = 0
```

### 16.55.4.3 col

```
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- [csr\\_hyb\\_utils.inl](#)

## 16.56 **Coo**< **Field** > Struct Template Reference

```
#include <read_sparse.h>
```

### Public Member Functions

- **Coo** ()=default
- **Coo** (typename [Field::Element](#) v, [index\\_t](#) r, [index\\_t](#) c)
- **Coo** (const [Self](#) &)=default
- **Coo** ([Self](#) &&)=default
- [Self](#) & **operator=** (const [Self](#) &)=default
- [Self](#) & **operator=** ([Self](#) &&)=default

### Data Fields

- [Field::Element](#) val = 0
- [index\\_t](#) col = 0
- [index\\_t](#) row = 0
- bool [deleted](#) = false

## 16.56.1 Constructor & Destructor Documentation

### 16.56.1.1 **Coo**() [1/4]

```
Coo ( ) [default]
```

### 16.56.1.2 **Coo**() [2/4]

```
Coo (
    typename Field::Element v,
    index\_t r,
    index\_t c ) [inline]
```

### 16.56.1.3 **Coo**() [3/4]

```
Coo (
    const Self & ) [default]
```

### 16.56.1.4 **Coo**() [4/4]

```
Coo (
    Self && ) [default]
```

## 16.56.2 Member Function Documentation

**16.56.2.1 `operator=()` [1/2]**

```
Self& operator= (
    const Self & ) [default]
```

**16.56.2.2 `operator=()` [2/2]**

```
Self& operator= (
    Self && ) [default]
```

**16.56.3 Field Documentation****16.56.3.1 `val`**

```
Field::Element val = 0
```

**16.56.3.2 `col`**

```
index_t col = 0
```

**16.56.3.3 `row`**

```
index_t row = 0
```

**16.56.3.4 `deleted`**

```
bool deleted = false
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

**16.57 `Coo< ValT, IdxT >` Struct Template Reference****Public Types**

- using `Self` = `Coo< ValT, IdxT >`

**Public Member Functions**

- `Coo` (`ValT` v, `IdxT` r, `IdxT` c)
- `Coo` ()=default
- `Coo` (const `Self` &)=default
- `Coo` (`Self` &&)=default
- `Self` & `operator=` (const `Self` &)=default
- `Self` & `operator=` (`Self` &&)=default

**Data Fields**

- `ValT` `val` = 0
- `IdxT` `row` = 0
- `IdxT` `col` = 0

## 16.57.1 Member Typedef Documentation

### 16.57.1.1 Self

```
using Self = Coo<ValT, IdxT>
```

## 16.57.2 Constructor & Destructor Documentation

### 16.57.2.1 Coo() [1/4]

```
Coo (
    ValT v,
    IdxT r,
    IdxT c ) [inline]
```

### 16.57.2.2 Coo() [2/4]

```
Coo ( ) [default]
```

### 16.57.2.3 Coo() [3/4]

```
Coo (
    const Self & ) [default]
```

### 16.57.2.4 Coo() [4/4]

```
Coo (
    Self && ) [default]
```

## 16.57.3 Member Function Documentation

### 16.57.3.1 operator=() [1/2]

```
Self& operator= (
    const Self & ) [default]
```

### 16.57.3.2 operator=() [2/2]

```
Self& operator= (
    Self && ) [default]
```

## 16.57.4 Field Documentation

### 16.57.4.1 val

```
ValT val = 0
```

**16.57.4.2 row**

```
IdxT row = 0
```

**16.57.4.3 col**

```
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- [sell\\_utils.inl](#)

**16.58 CooMat< Field > Struct Template Reference**

```
#include <fflas_sparse.h>
```

**Data Fields**

- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int16\\_t > \\* \\_coo16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int32\\_t > \\* \\_coo32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int64\\_t > \\* \\_coo64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int16\\_t > \\* \\_coo16\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int32\\_t > \\* \\_coo32\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int64\\_t > \\* \\_coo64\\_zo](#) = nullptr

**16.58.1 Field Documentation****16.58.1.1 \_coo16**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int16_t>* _coo16 = nullptr
```

**16.58.1.2 \_coo32**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int32_t>* _coo32 = nullptr
```

**16.58.1.3 \_coo64**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int64_t>* _coo64 = nullptr
```

**16.58.1.4 \_coo16\_zo**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int16_t>* _coo16_zo = nullptr
```

**16.58.1.5 \_coo32\_zo**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int32_t>* _coo32_zo = nullptr
```

**16.58.1.6 \_coo64\_zo**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int64_t>* _coo64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.59 CsrMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

### Data Fields

- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR, int16\\_t > \\* \\_csr16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR, int32\\_t > \\* \\_csr32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR, int64\\_t > \\* \\_csr64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR\\_ZO, int16\\_t > \\* \\_csr16\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR\\_ZO, int32\\_t > \\* \\_csr32\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR\\_ZO, int64\\_t > \\* \\_csr64\\_zo](#) = nullptr

### 16.59.1 Field Documentation

#### 16.59.1.1 \_csr16

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int16_t>* _csr16 = nullptr
```

#### 16.59.1.2 \_csr32

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int32_t>* _csr32 = nullptr
```

#### 16.59.1.3 \_csr64

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int64_t>* _csr64 = nullptr
```

#### 16.59.1.4 \_csr16\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int16_t>* _csr16_zo = nullptr
```

#### 16.59.1.5 \_csr32\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int32_t>* _csr32_zo = nullptr
```

#### 16.59.1.6 \_csr64\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int64_t>* _csr64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.60 DefaultBoundedTag Struct Reference

Use standard field operations, but keeps track of bounds on input and output.

```
#include <field-traits.h>
```

### 16.60.1 Detailed Description

Use standard field operations, but keeps track of bounds on input and output.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.61 DefaultTag Struct Reference

No specific mode of action: use standard field operations.

```
#include <field-traits.h>
```

### 16.61.1 Detailed Description

No specific mode of action: use standard field operations.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.62 DelayedTag Struct Reference

Performs field operations with delayed mod reductions. Ensures result is reduced.

```
#include <field-traits.h>
```

### 16.62.1 Detailed Description

Performs field operations with delayed mod reductions. Ensures result is reduced.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.63 ElementTraits< Element > Struct Template Reference

[ElementTraits](#).

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::GenericTag](#) value

### 16.63.1 Detailed Description

```
template<class Element>
```

```
struct FFLAS::ElementTraits< Element >
```

[ElementTraits](#).

### 16.63.2 Member Typedef Documentation

#### 16.63.2.1 value

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.64 ElementTraits< double > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineFloatTag](#) value

## 16.64.1 Member Typedef Documentation

### 16.64.1.1 value

typedef [ElementCategories::MachineFloatTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.65 ElementTraits< FFPACK::rns\_double\_elt > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::RNSElementTag](#) value

## 16.65.1 Member Typedef Documentation

### 16.65.1.1 value

typedef [ElementCategories::RNSElementTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.66 ElementTraits< float > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineFloatTag](#) value

## 16.66.1 Member Typedef Documentation

### 16.66.1.1 value

typedef [ElementCategories::MachineFloatTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.67 ElementTraits< Givaro::Integer > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::ArbitraryPrecIntTag](#) value



## 16.67.1 Member Typedef Documentation

### 16.67.1.1 value

typedef [ElementCategories::ArbitraryPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.68 ElementTraits< int16\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.68.1 Member Typedef Documentation

### 16.68.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.69 ElementTraits< int32\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.69.1 Member Typedef Documentation

### 16.69.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.70 ElementTraits< int64\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.70.1 Member Typedef Documentation

### 16.70.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.71 ElementTraits< int8\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.71.1 Member Typedef Documentation

### 16.71.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.72 ElementTraits< RecInt::rint< K > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

## 16.72.1 Member Typedef Documentation

### 16.72.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.73 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

### 16.73.1 Member Typedef Documentation

#### 16.73.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.74 ElementTraits< RecInt::ruint< K > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

### 16.74.1 Member Typedef Documentation

#### 16.74.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.75 ElementTraits< uint16\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.75.1 Member Typedef Documentation

#### 16.75.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.76 ElementTraits< uint32\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.76.1 Member Typedef Documentation

### 16.76.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.77 ElementTraits< uint64\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.77.1 Member Typedef Documentation

### 16.77.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.78 ElementTraits< uint8\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.78.1 Member Typedef Documentation

### 16.78.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.79 EII Mat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

## Data Fields

- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#), [int16\\_t](#) > \* [\\_ell16](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#), [int32\\_t](#) > \* [\\_ell32](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#), [int64\\_t](#) > \* [\\_ell64](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#), [int16\\_t](#) > \* [\\_ell16\\_zo](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#), [int32\\_t](#) > \* [\\_ell32\\_zo](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#), [int64\\_t](#) > \* [\\_ell64\\_zo](#) = nullptr

## 16.79.1 Field Documentation

### 16.79.1.1 [\\_ell16](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int16_t>* _ell16 = nullptr
```

### 16.79.1.2 [\\_ell32](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int32_t>* _ell32 = nullptr
```

### 16.79.1.3 [\\_ell64](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int64_t>* _ell64 = nullptr
```

### 16.79.1.4 [\\_ell16\\_zo](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int16_t>* _ell16_zo = nullptr
```

### 16.79.1.5 [\\_ell32\\_zo](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int32_t>* _ell32_zo = nullptr
```

### 16.79.1.6 [\\_ell64\\_zo](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int64_t>* _ell64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.80 Failure Class Reference

A precondition failed.

```
#include <debug.h>
```

## Public Member Functions

- [Failure](#) ()
- void [operator\(\)](#) (const char \*function, int line, const char \*check)
- void [operator\(\)](#) (const char \*function, const char \*file, int line, const char \*check)
- void [setErrorStream](#) (std::ostream &stream)
- std::ostream & [print](#) (std::ostream &o) const

## Protected Attributes

- `std::ostream * _errorStream`

### 16.80.1 Detailed Description

A precondition failed.

The `throw` mechanism is usually used here as in

```
if (!check)
    failure(__func__, __LINE__, "this check just failed");
```

The parameters of the constructor help debugging.

### 16.80.2 Constructor & Destructor Documentation

#### 16.80.2.1 Failure()

```
Failure ( ) [inline]
```

### 16.80.3 Member Function Documentation

#### 16.80.3.1 operator>() [1/2]

```
void operator() (
    const char * function,
    int line,
    const char * check ) [inline]
```

A precondition failed.

##### Parameters

<i>function</i>	usually <b>func</b> , the function that threw the error
<i>line</i>	usually <b>LINE</b> , the line where it happened
<i>check</i>	a string telling what failed.

#### 16.80.3.2 operator>() [2/2]

```
void operator() (
    const char * function,
    const char * file,
    int line,
    const char * check ) [inline]
```

A precondition failed. The parameter help debugging. This is not much different from the previous except we can digg faster in the file where the exception was triggered.

##### Parameters

<i>function</i>	usually <b>func</b> , the function that threw the error
<i>file</i>	usually <b>FILE</b> , the file where this function is
<i>line</i>	usually <b>LINE</b> , the line where it happened
<i>check</i>	a string telling what failed.

**16.80.3.3 setErrorMessage()**

```
void setErrorMessage (
    std::ostream & stream )
```

**16.80.3.4 print()**

```
std::ostream& print (
    std::ostream & o ) const [inline]
```

overload the virtual print of LinboxError.

**Parameters**

<i>o</i>	output stream
----------	---------------

**16.80.4 Field Documentation****16.80.4.1 \_errorMessage**

```
std::ostream* _errorMessage [protected]
```

The documentation for this class was generated from the following file:

- [debug.h](#)

**16.81 FailureCharpolyCheck Class Reference**

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

**16.82 FailureDetCheck Class Reference**

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

**16.83 FailureFgemmCheck Class Reference**

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_fflas.h](#)

**16.84 FailureInvertCheck Class Reference**

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

## 16.85 FailurePLUQCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

## 16.86 FailureTrsmCheck Class Reference

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_fflas.h](#)

## 16.87 FieldSimd< \_Field > Class Template Reference

### Public Types

- using [Field](#) = [\\_Field](#)
- using [Element](#) = typename [Field::Element](#)
- using [simd](#) = [Simd](#)< typename [\\_Field::Element](#) >
- using [vect\\_t](#) = typename [simd::vect\\_t](#)
- using [scalar\\_t](#) = typename [simd::scalar\\_t](#)

### Public Member Functions

- [FieldSimd](#) (const [Field](#) &f)
- [FieldSimd](#) (const [Self](#) &)=default
- [FieldSimd](#) ([Self](#) &&)=default
- [Self](#) & operator= (const [Self](#) &)=default
- [Self](#) & operator= ([Self](#) &&)=default
- [INLINE vect\\_t init](#) ([vect\\_t](#) &x, const [vect\\_t](#) a) const
- [INLINE vect\\_t init](#) (const [vect\\_t](#) a) const
- [INLINE vect\\_t add](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t add\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t add\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t addin\\_r](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t subin\\_r](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t zero](#) ([vect\\_t](#) &x) const
- [INLINE vect\\_t zero](#) () const
- [INLINE vect\\_t mod](#) ([vect\\_t](#) &c) const
- [INLINE vect\\_t mul](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mulin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t axpy](#) ([vect\\_t](#) &r, const [vect\\_t](#) a, const [vect\\_t](#) b, const [vect\\_t](#) c) const
- [INLINE vect\\_t axpy](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t axpyin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const



- `INLINE vect_t axpy_r (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const`
- `INLINE vect_t axpy_r (const vect_t c, const vect_t a, const vect_t b) const`
- `INLINE vect_t axpyin_r (vect_t &c, const vect_t a, const vect_t b) const`
- `INLINE vect_t maxpy (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const`
- `INLINE vect_t maxpy (const vect_t c, const vect_t a, const vect_t b) const`
- `INLINE vect_t maxpyin (vect_t &c, const vect_t a, const vect_t b) const`

## Static Public Attributes

- static constexpr const size\_t `vect_size` = `simd::vect_size`
- static constexpr const size\_t `alignment` = `simd::alignment`

## 16.87.1 Member Typedef Documentation

### 16.87.1.1 Field

using `Field` = `_Field`

### 16.87.1.2 Element

using `Element` = `typename Field::Element`

### 16.87.1.3 simd

using `simd` = `Simd<typename _Field::Element>`

### 16.87.1.4 vect\_t

using `vect_t` = `typename simd::vect_t`

### 16.87.1.5 scalar\_t

using `scalar_t` = `typename simd::scalar_t`

## 16.87.2 Constructor & Destructor Documentation

### 16.87.2.1 FieldSimd() [1/3]

```
FieldSimd (
    const Field & f ) [inline]
```

### 16.87.2.2 FieldSimd() [2/3]

```
FieldSimd (
    const Self & ) [default]
```

### 16.87.2.3 FieldSimd() [3/3]

```
FieldSimd (
    Self && ) [default]
```

## 16.87.3 Member Function Documentation

### 16.87.3.1 operator=() [1/2]

```
Self& operator= (
    const Self & ) [default]
```

### 16.87.3.2 operator=() [2/2]

```
Self& operator= (
    Self && ) [default]
```

### 16.87.3.3 init() [1/2]

```
INLINE vect_t init (
    vect_t & x,
    const vect_t a ) const [inline]
```

### 16.87.3.4 init() [2/2]

```
INLINE vect_t init (
    const vect_t a ) const [inline]
```

### 16.87.3.5 add() [1/2]

```
INLINE vect_t add (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline]
```

### 16.87.3.6 add() [2/2]

```
INLINE vect_t add (
    const vect_t a,
    const vect_t b ) [inline]
```

### 16.87.3.7 addin()

```
INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) const [inline]
```

### 16.87.3.8 add\_r() [1/2]

```
INLINE vect_t add_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.9 add\_r() [2/2]**

```
INLINE vect_t add_r (  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.87.3.10 addin\_r()**

```
INLINE vect_t addin_r (  
    vect_t & a,  
    const vect_t b ) const [inline]
```

**16.87.3.11 sub() [1/2]**

```
INLINE vect_t sub (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline]
```

**16.87.3.12 sub() [2/2]**

```
INLINE vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline]
```

**16.87.3.13 subin()**

```
INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) const [inline]
```

**16.87.3.14 sub\_r() [1/2]**

```
INLINE vect_t sub_r (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.87.3.15 sub\_r() [2/2]**

```
INLINE vect_t sub_r (  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.87.3.16 subin\_r()**

```
INLINE vect_t subin_r (  
    vect_t & a,  
    const vect_t b ) const [inline]
```

**16.87.3.17 zero() [1/2]**

```
INLINE vect_t zero (
    vect_t & x ) const [inline]
```

**16.87.3.18 zero() [2/2]**

```
INLINE vect_t zero ( ) const [inline]
```

**16.87.3.19 mod()**

```
INLINE vect_t mod (
    vect_t & c ) const [inline]
```

**16.87.3.20 mul() [1/2]**

```
INLINE vect_t mul (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.21 mul() [2/2]**

```
INLINE vect_t mul (
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.22 mulin()**

```
INLINE vect_t mulin (
    vect_t & a,
    const vect_t b ) const [inline]
```

**16.87.3.23 mul\_r() [1/2]**

```
INLINE vect_t mul_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.24 mul\_r() [2/2]**

```
INLINE vect_t mul_r (
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.25 axpy() [1/2]**

```
INLINE vect_t axpy (
    vect_t & r,
    const vect_t a,
```

```
const vect_t b,  
const vect_t c ) const [inline]
```

#### 16.87.3.26 axpy() [2/2]

```
INLINE vect_t axpy (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

#### 16.87.3.27 axpyin()

```
INLINE vect_t axpyin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

#### 16.87.3.28 axpy\_r() [1/2]

```
INLINE vect_t axpy_r (  
    vect_t & r,  
    const vect_t a,  
    const vect_t b,  
    const vect_t c ) const [inline]
```

#### 16.87.3.29 axpy\_r() [2/2]

```
INLINE vect_t axpy_r (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

#### 16.87.3.30 axpyin\_r()

```
INLINE vect_t axpyin_r (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

#### 16.87.3.31 maxpy() [1/2]

```
INLINE vect_t maxpy (  
    vect_t & r,  
    const vect_t a,  
    const vect_t b,  
    const vect_t c ) const [inline]
```

#### 16.87.3.32 maxpy() [2/2]

```
INLINE vect_t maxpy (  
    const vect_t c,
```

```
const vect_t a,
const vect_t b ) const [inline]
```

### 16.87.3.33 maxpyin()

```
INLINE vect_t maxpyin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

## 16.87.4 Field Documentation

### 16.87.4.1 vect\_size

```
constexpr const size_t vect_size = simd::vect_size [static], [constexpr]
```

### 16.87.4.2 alignment

```
constexpr const size_t alignment = simd::alignment [static], [constexpr]
```

The documentation for this class was generated from the following file:

- [simd\\_modular.inl](#)

## 16.88 FieldTraits< Field > Struct Template Reference

FieldTrait.

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::GenericTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.88.1 Detailed Description

```
template<class Field>
struct FFLAS::FieldTraits< Field >
```

FieldTrait.

### 16.88.2 Member Typedef Documentation

#### 16.88.2.1 category

```
typedef FieldCategories::GenericTag category
```

### 16.88.3 Field Documentation

### 16.88.3.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.89 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.89.1 Member Typedef Documentation

#### 16.89.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.89.2 Field Documentation

#### 16.89.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.90 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::ModularTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.90.1 Member Typedef Documentation

**16.90.1.1 category**

```
typedef FieldCategories::ModularTag category
```

**16.90.2 Field Documentation****16.90.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.91 FieldTraits< Givaro::Modular< Element > > Struct Template Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::ModularTag](#) category

**Static Public Attributes**

- static const bool [balanced](#) = false

**16.91.1 Member Typedef Documentation****16.91.1.1 category**

```
typedef FieldCategories::ModularTag category
```

**16.91.2 Field Documentation****16.91.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.92 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::ModularTag](#) category



## Static Public Attributes

- static const bool [balanced](#) = true

### 16.92.1 Member Typedef Documentation

#### 16.92.1.1 category

```
typedef FieldCategories::ModularTag category
```

### 16.92.2 Field Documentation

#### 16.92.2.1 balanced

```
const bool balanced = true [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.93 FieldTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.93.1 Member Typedef Documentation

#### 16.93.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.93.2 Field Documentation

#### 16.93.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.94 FieldTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.94.1 Member Typedef Documentation

#### 16.94.1.1 category

typedef [FieldCategories::UnparametricTag](#) category

### 16.94.2 Field Documentation

#### 16.94.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.95 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.95.1 Member Typedef Documentation

#### 16.95.1.1 category

typedef [FieldCategories::UnparametricTag](#) category

### 16.95.2 Field Documentation

#### 16.95.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.96 FieldTraits< Givaro::ZRing< int16\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.96.1 Member Typedef Documentation

#### 16.96.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.96.2 Field Documentation

#### 16.96.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.97 FieldTraits< Givaro::ZRing< int32\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.97.1 Member Typedef Documentation

#### 16.97.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.97.2 Field Documentation

**16.97.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.98 FieldTraits< Givaro::ZRing< int64\_t > > Struct Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::UnparametricTag](#) category

**Static Public Attributes**

- static const bool [balanced](#) = false

**16.98.1 Member Typedef Documentation****16.98.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

**16.98.2 Field Documentation****16.98.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.99 FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > Struct Template Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::UnparametricTag](#) category

**Static Public Attributes**

- static const bool [balanced](#) = false

**16.99.1 Member Typedef Documentation****16.99.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

## 16.99.2 Field Documentation

### 16.99.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.100 FieldTraits< Givaro::ZRing< uint16\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

## 16.100.1 Member Typedef Documentation

### 16.100.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

## 16.100.2 Field Documentation

### 16.100.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.101 FieldTraits< Givaro::ZRing< uint32\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

## 16.101.1 Member Typedef Documentation

**16.101.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

**16.101.2 Field Documentation****16.101.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.102 FieldTraits< Givaro::ZRing< uint64\_t > > Struct Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::UnparametricTag](#) category

**Static Public Attributes**

- static const bool [balanced](#) = false

**16.102.1 Member Typedef Documentation****16.102.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

**16.102.2 Field Documentation****16.102.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.103 Fixed Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.104 FixedPrecIntTag Struct Reference**

Fixed precision integers above machine precision: Givaro::reclnt.

```
#include <field-traits.h>
```

### 16.104.1 Detailed Description

Fixed precision integers above machine precision: Givaro::reclnt.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.105 ForStrategy1D< blocksize\_t, Cut, Param > Struct Template Reference

### Public Member Functions

- [ForStrategy1D](#) (const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- [ForStrategy1D](#) (const blocksize\_t b, const blocksize\_t e, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- void [build](#) (const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- blocksize\_t [initialize](#) ()
- bool [isTerminated](#) () const
- blocksize\_t [begin](#) () const
- blocksize\_t [end](#) () const
- blocksize\_t [numblocks](#) () const
- blocksize\_t [blockindex](#) () const
- blocksize\_t [operator++](#) ()

### Protected Attributes

- blocksize\_t [ibeg](#)
- blocksize\_t [iend](#)
- blocksize\_t [current](#)
- blocksize\_t [firstBlockSize](#)
- blocksize\_t [lastBlockSize](#)
- blocksize\_t [changeBS](#)
- blocksize\_t [numBlock](#)

### 16.105.1 Constructor & Destructor Documentation

#### 16.105.1.1 ForStrategy1D() [1/2]

```
ForStrategy1D (
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

#### 16.105.1.2 ForStrategy1D() [2/2]

```
ForStrategy1D (
    const blocksize_t b,
    const blocksize_t e,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

### 16.105.2 Member Function Documentation

#### 16.105.2.1 build()

```
void build (
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

#### 16.105.2.2 initialize()

```
blocksize_t initialize ( ) [inline]
```

#### 16.105.2.3 isTerminated()

```
bool isTerminated ( ) const [inline]
```

#### 16.105.2.4 begin()

```
blocksize_t begin ( ) const [inline]
```

#### 16.105.2.5 end()

```
blocksize_t end ( ) const [inline]
```

#### 16.105.2.6 numblocks()

```
blocksize_t numblocks ( ) const [inline]
```

#### 16.105.2.7 blockindex()

```
blocksize_t blockindex ( ) const [inline]
```

#### 16.105.2.8 operator++()

```
blocksize_t operator++ ( ) [inline]
```

### 16.105.3 Field Documentation

#### 16.105.3.1 ibeg

```
blocksize_t ibeg [protected]
```

#### 16.105.3.2 iend

```
blocksize_t iend [protected]
```

#### 16.105.3.3 current

```
blocksize_t current [protected]
```



**16.105.3.4 firstBlockSize**

```
blocksize_t firstBlockSize [protected]
```

**16.105.3.5 lastBlockSize**

```
blocksize_t lastBlockSize [protected]
```

**16.105.3.6 changeBS**

```
blocksize_t changeBS [protected]
```

**16.105.3.7 numBlock**

```
blocksize_t numBlock [protected]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.106 ForStrategy2D< blocksize\_t, Cut, Param > Struct Template Reference

### Public Member Functions

- [ForStrategy2D](#) (const blocksize\_t m, const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- blocksize\_t [initialize](#) ()
- bool [isTerminated](#) () const
- blocksize\_t [ibegin](#) () const
- blocksize\_t [jbegin](#) () const
- blocksize\_t [iend](#) () const
- blocksize\_t [jend](#) () const
- blocksize\_t [operator++](#) ()
- blocksize\_t [rownumblocks](#) () const
- blocksize\_t [colnumblocks](#) () const
- blocksize\_t [blockindex](#) () const
- blocksize\_t [rowblockindex](#) () const
- blocksize\_t [colblockindex](#) () const

### Protected Attributes

- blocksize\_t [\\_ibeg](#)
- blocksize\_t [\\_iend](#)
- blocksize\_t [\\_jbeg](#)
- blocksize\_t [\\_jend](#)
- blocksize\_t [rowBlockSize](#)
- blocksize\_t [colBlockSize](#)
- blocksize\_t [current](#)
- blocksize\_t [lastRBS](#)
- blocksize\_t [lastCBS](#)
- blocksize\_t [changeRBS](#)
- blocksize\_t [changeCBS](#)
- blocksize\_t [numRowBlock](#)
- blocksize\_t [numColBlock](#)
- blocksize\_t [BLOCKS](#)

## Friends

- `std::ostream & operator<< (std::ostream &out, const ForStrategy2D &FS2D)`

## 16.106.1 Constructor & Destructor Documentation

### 16.106.1.1 ForStrategy2D()

```
ForStrategy2D (
    const blocksize_t m,
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

## 16.106.2 Member Function Documentation

### 16.106.2.1 initialize()

```
blocksize_t initialize ( ) [inline]
```

### 16.106.2.2 isTerminated()

```
bool isTerminated ( ) const [inline]
```

### 16.106.2.3 ibegin()

```
blocksize_t ibegin ( ) const [inline]
```

### 16.106.2.4 jbegin()

```
blocksize_t jbegin ( ) const [inline]
```

### 16.106.2.5 iend()

```
blocksize_t iend ( ) const [inline]
```

### 16.106.2.6 jend()

```
blocksize_t jend ( ) const [inline]
```

### 16.106.2.7 operator++()

```
blocksize_t operator++ ( ) [inline]
```

### 16.106.2.8 rownumblocks()

```
blocksize_t rownumblocks ( ) const [inline]
```

#### 16.106.2.9 colnumblocks()

```
blocksize_t colnumblocks ( ) const [inline]
```

#### 16.106.2.10 blockindex()

```
blocksize_t blockindex ( ) const [inline]
```

#### 16.106.2.11 rowblockindex()

```
blocksize_t rowblockindex ( ) const [inline]
```

#### 16.106.2.12 colblockindex()

```
blocksize_t colblockindex ( ) const [inline]
```

### 16.106.3 Friends And Related Function Documentation

#### 16.106.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const ForStrategy2D< blocksize_t, Cut, Param > & FS2D ) [friend]
```

### 16.106.4 Field Documentation

#### 16.106.4.1 \_ibeg

```
blocksize_t _ibeg [protected]
```

#### 16.106.4.2 \_iend

```
blocksize_t _iend [protected]
```

#### 16.106.4.3 \_jbeg

```
blocksize_t _jbeg [protected]
```

#### 16.106.4.4 \_jend

```
blocksize_t _jend [protected]
```

#### 16.106.4.5 rowBlockSize

```
blocksize_t rowBlockSize [protected]
```

**16.106.4.6 colBlockSize**

blocksize\_t colBlockSize [protected]

**16.106.4.7 current**

blocksize\_t current [protected]

**16.106.4.8 lastRBS**

blocksize\_t lastRBS [protected]

**16.106.4.9 lastCBS**

blocksize\_t lastCBS [protected]

**16.106.4.10 changeRBS**

blocksize\_t changeRBS [protected]

**16.106.4.11 changeCBS**

blocksize\_t changeCBS [protected]

**16.106.4.12 numRowsBlock**

blocksize\_t numRowsBlock [protected]

**16.106.4.13 numColBlock**

blocksize\_t numColBlock [protected]

**16.106.4.14 BLOCKS**

blocksize\_t BLOCKS [protected]

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.107 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

**16.108 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.109 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.110 ftrmmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.111 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.112 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.113 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.114 ftrmmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.115 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.116 ftrmmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.117 **ftrmmRightLowerTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.118 **ftrmmRightLowerTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.119 **ftrmmRightUpperNoTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.120 **ftrmmRightUpperNoTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.121 **ftrmmRightUpperTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.122 **ftrmmRightUpperTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.123 **ftrsmLeftLowerNoTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.124 **ftrsmLeftLowerNoTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.125 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.126 ftrsmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.127 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

### 16.127.1 Detailed Description

```
template<class Element>
```

```
class FFLAS::Protected::ftrsmLeftUpperNoTransNonUnit< Element >
```

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

Compute the maximal dimension  $k$ , such that a unit diagonal triangular system of dimension  $k$  can be solved over  $\mathbb{Z}$  without overflow of the underlying floating point representation.

**Bibliography** • Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

#### Parameters

$F$	Finite Field/Ring of the computation
-----	--------------------------------------

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.128 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.129 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.130 ftrsmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.131 **ftrsmRightLowerNoTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.132 **ftrsmRightLowerNoTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.133 **ftrsmRightLowerTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.134 **ftrsmRightLowerTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.135 **ftrsmRightUpperNoTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.136 **ftrsmRightUpperNoTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.137 **ftrsmRightUpperTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.138 **ftrsmRightUpperTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)



## 16.139 GenericTag Struct Reference

default is generic

```
#include <field-traits.h>
```

### 16.139.1 Detailed Description

default is generic

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.140 GenericTag Struct Reference

generic ring.

```
#include <field-traits.h>
```

### 16.140.1 Detailed Description

generic ring.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.141 Grain Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.142 has\_minus\_eq\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.142.1 Field Documentation

#### 16.142.1.1 value

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.143 has\_minus\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.143.1 Field Documentation

#### 16.143.1.1 value

`constexpr bool value = type::value [static], [constexpr]`

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.144 has\_mul\_eq\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.144.1 Field Documentation

#### 16.144.1.1 value

`constexpr bool value = type::value [static], [constexpr]`

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.145 has\_mul\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.145.1 Field Documentation

#### 16.145.1.1 value

`constexpr bool value = type::value [static], [constexpr]`

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.146 has\_operation< T > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#)

## 16.146.1 Field Documentation

### 16.146.1.1 value

constexpr bool value [static], [constexpr]

**Initial value:**

```
= (has_plus<T>::value && has_minus<T>::value && has_equal<T>::value &&
    has_plus_eq<T>::value && has_minus_eq<T>::value && has_mul<T>::value
    && has_mul_eq<T>::value)
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.147 has\_plus\_eq\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

## 16.147.1 Field Documentation

### 16.147.1.1 value

constexpr bool value = type::value [static], [constexpr]

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.148 has\_plus\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

## 16.148.1 Field Documentation

### 16.148.1.1 value

constexpr bool value = type::value [static], [constexpr]

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.149 HelperFlag Struct Reference

```
#include <fflas_sparse.h>
```

## Static Public Attributes

- static constexpr [uint64\\_t none](#) = 0\_ui64
- static constexpr [uint64\\_t coo](#) = 1\_ui64
- static constexpr [uint64\\_t csr](#) = 1\_ui64 << 1
- static constexpr [uint64\\_t ell](#) = 1\_ui64 << 2
- static constexpr [uint64\\_t aut](#) = 1\_ui64 << 32
- static constexpr [uint64\\_t pm1](#) = 1\_ui64 << 33

### 16.149.1 Field Documentation

#### 16.149.1.1 none

constexpr [uint64\\_t none](#) = 0\_ui64 [static], [constexpr]

#### 16.149.1.2 coo

constexpr [uint64\\_t coo](#) = 1\_ui64 [static], [constexpr]

#### 16.149.1.3 csr

constexpr [uint64\\_t csr](#) = 1\_ui64 << 1 [static], [constexpr]

#### 16.149.1.4 ell

constexpr [uint64\\_t ell](#) = 1\_ui64 << 2 [static], [constexpr]

#### 16.149.1.5 aut

constexpr [uint64\\_t aut](#) = 1\_ui64 << 32 [static], [constexpr]

#### 16.149.1.6 pm1

constexpr [uint64\\_t pm1](#) = 1\_ui64 << 33 [static], [constexpr]

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

### 16.150 HelperMod< Field, ElementTraits > Struct Template Reference

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

### 16.151 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference

#### Public Member Functions

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)

## Data Fields

- [Field::Element p](#)
- double [invp](#)
- double [min](#)
- double [max](#)
- [int64\\_t pow50rem](#)

## 16.151.1 Constructor & Destructor Documentation

### 16.151.1.1 HelperMod() [1/2]

```
HelperMod ( ) [inline]
```

### 16.151.1.2 HelperMod() [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

## 16.151.2 Field Documentation

### 16.151.2.1 p

```
Field::Element p
```

### 16.151.2.2 invp

```
double invp
```

### 16.151.2.3 min

```
double min
```

### 16.151.2.4 max

```
double max
```

### 16.151.2.5 pow50rem

```
int64_t pow50rem
```

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.152 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference

### Public Member Functions

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)

### Data Fields

- [Field::Element](#) p

### 16.152.1 Constructor & Destructor Documentation

#### 16.152.1.1 HelperMod() [1/2]

[HelperMod](#) ( ) [inline]

#### 16.152.1.2 HelperMod() [2/2]

[HelperMod](#) (   
 const [Field](#) & F ) [inline]

### 16.152.2 Field Documentation

#### 16.152.2.1 p

[Field::Element](#) p

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.153 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference

### Public Member Functions

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)

### Data Fields

- [Field::Element](#) p

### 16.153.1 Constructor & Destructor Documentation

#### 16.153.1.1 HelperMod() [1/2]

[HelperMod](#) ( ) [inline]

**16.153.1.2 HelperMod()** [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

**16.153.2 Field Documentation****16.153.2.1 p**

Field::Element p

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.154 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference

**Public Member Functions**

- [HelperMod\(\)](#)
- [HelperMod](#) (const [Field](#) &F)

**Data Fields**

- [Field::Element](#) p
- [Field::Element](#) invp
- [Field::Element](#) min
- [Field::Element](#) max

**16.154.1 Constructor & Destructor Documentation****16.154.1.1 HelperMod()** [1/2]

```
HelperMod ( ) [inline]
```

**16.154.1.2 HelperMod()** [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

**16.154.2 Field Documentation****16.154.2.1 p**

Field::Element p

**16.154.2.2 invp**

Field::Element invp

**16.154.2.3 min**

`Field::Element` min

**16.154.2.4 max**

`Field::Element` max

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

**16.155 Hybrid Struct Reference**

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

**16.156 Info Struct Reference****Public Member Functions**

- [Info](#) ([uint64\\_t](#) it, [uint64\\_t](#) s, [uint64\\_t](#) p)
- [Info](#) ()=default
- [Info](#) (const [Info](#) &)=default
- [Info](#) ([Info](#) &&)=default
- [Info](#) & [operator=](#) (const [Info](#) &)=default
- [Info](#) & [operator=](#) ([Info](#) &&)=default

**Data Fields**

- [uint64\\_t](#) size = 0
- [uint64\\_t](#) perm = 0
- [uint64\\_t](#) begin = 0

**16.156.1 Constructor & Destructor Documentation****16.156.1.1 Info() [1/4]**

```
Info (
    uint64_t it,
    uint64_t s,
    uint64_t p ) [inline]
```

**16.156.1.2 Info() [2/4]**

```
Info ( ) [default]
```

**16.156.1.3 Info() [3/4]**

```
Info (
    const Info & ) [default]
```



#### 16.156.1.4 Info() [4/4]

```
Info (
    Info && ) [default]
```

### 16.156.2 Member Function Documentation

#### 16.156.2.1 operator=() [1/2]

```
Info& operator= (
    const Info & ) [default]
```

#### 16.156.2.2 operator=() [2/2]

```
Info& operator= (
    Info && ) [default]
```

### 16.156.3 Field Documentation

#### 16.156.3.1 size

```
uint64_t size = 0
```

#### 16.156.3.2 perm

```
uint64_t perm = 0
```

#### 16.156.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- [csr\\_hyb\\_utils.inl](#)

## 16.157 Info Struct Reference

### Public Member Functions

- [Info](#) (uint64\_t it, uint64\_t s, uint64\_t p)
- [Info](#) ()=default
- [Info](#) (const [Info](#) &)=default
- [Info](#) ([Info](#) &&)=default
- [Info](#) & [operator=](#) (const [Info](#) &)=default
- [Info](#) & [operator=](#) ([Info](#) &&)=default

### Data Fields

- [uint64\\_t](#) size = 0
- [uint64\\_t](#) perm = 0
- [uint64\\_t](#) begin = 0

## 16.157.1 Constructor & Destructor Documentation

### 16.157.1.1 Info() [1/4]

```
Info (
    uint64_t it,
    uint64_t s,
    uint64_t p ) [inline]
```

### 16.157.1.2 Info() [2/4]

```
Info ( ) [default]
```

### 16.157.1.3 Info() [3/4]

```
Info (
    const Info & ) [default]
```

### 16.157.1.4 Info() [4/4]

```
Info (
    Info && ) [default]
```

## 16.157.2 Member Function Documentation

### 16.157.2.1 operator=() [1/2]

```
Info& operator= (
    const Info & ) [default]
```

### 16.157.2.2 operator=() [2/2]

```
Info& operator= (
    Info && ) [default]
```

## 16.157.3 Field Documentation

### 16.157.3.1 size

```
uint64_t size = 0
```

### 16.157.3.2 perm

```
uint64_t perm = 0
```

### 16.157.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- [sell\\_utils.inl](#)

## 16.158 is\_simd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

### Public Types

- using [type](#) = std::integral\_constant< bool, false >

### Static Public Attributes

- static constexpr const bool [value](#) = false

### 16.158.1 Member Typedef Documentation

#### 16.158.1.1 type

```
using type = std::integral_constant<bool, false>
```

### 16.158.2 Field Documentation

#### 16.158.2.1 value

```
constexpr const bool value = false [static], [constexpr]
```

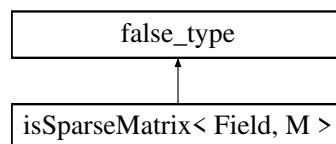
The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.159 isSparseMatrix< Field, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, M >:



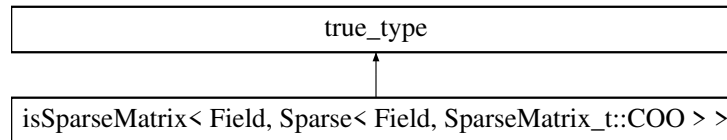
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.160 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO > >** **Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO > >:



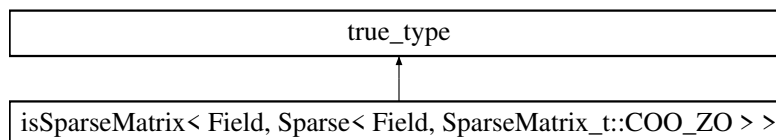
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.161 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >** **Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >:



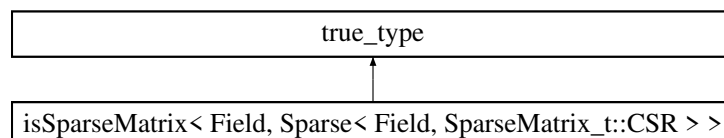
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.162 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR > >** **Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR > >:



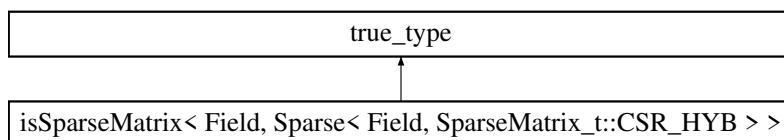
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.163 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > >** **Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > >:



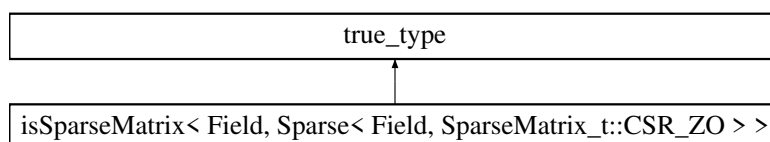
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.164 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >:



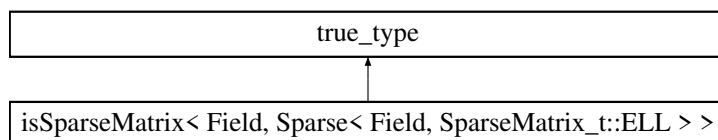
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.165 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > >:



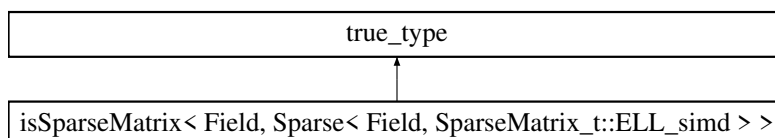
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.166 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >:



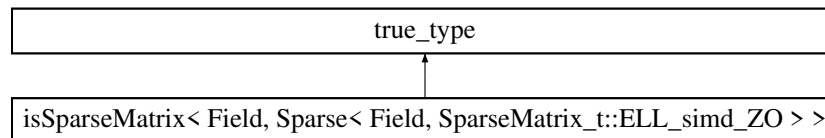
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.167 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >:



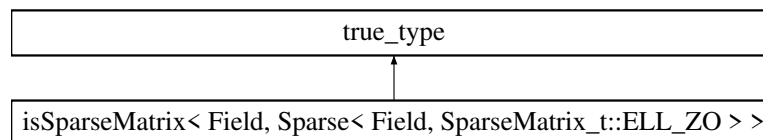
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.168 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >:



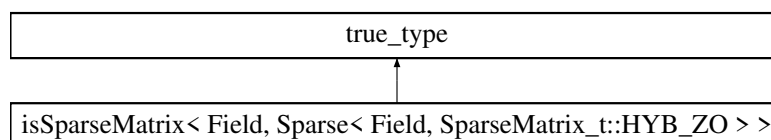
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.169 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >:



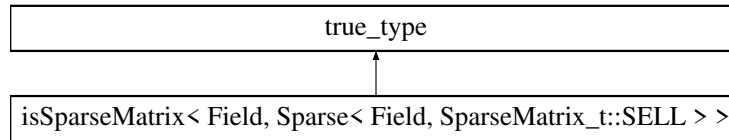
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.170 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > >:



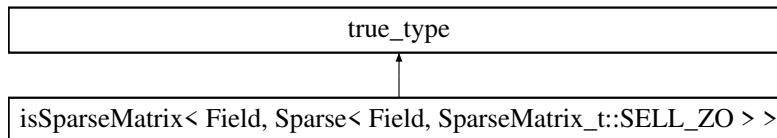
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.171 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >:



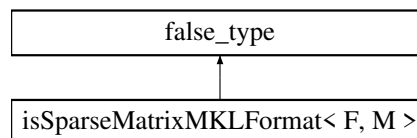
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.172 isSparseMatrixMKLFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixMKLFormat< F, M >:



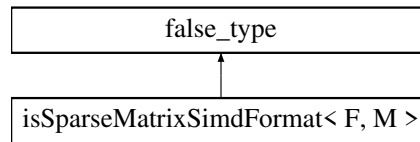
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.173 isSparseMatrixSimdFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixSimdFormat< F, M >:



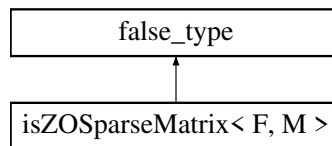
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.174 isZOSparseMatrix< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< F, M >:



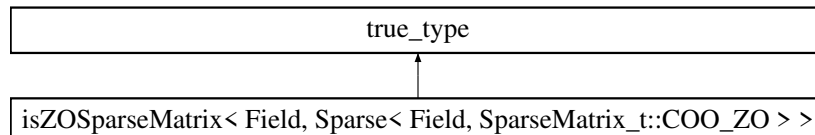
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.175 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >:



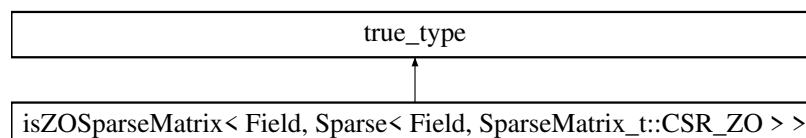
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.176 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >:



The documentation for this struct was generated from the following file:

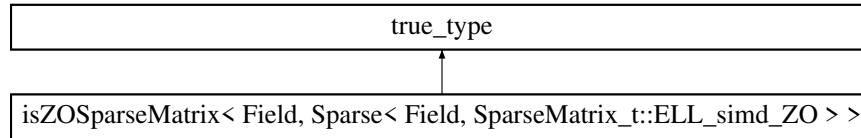
- [sparse\\_matrix\\_traits.h](#)



## 16.177 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >:



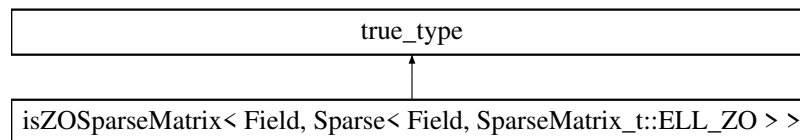
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.178 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >:



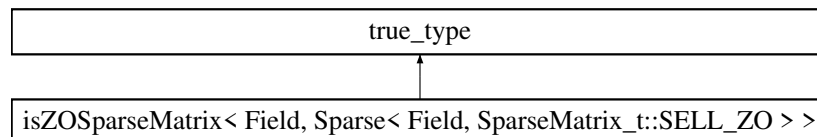
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.179 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >:



The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.180 Iterative Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.181 LazyTag Struct Reference

Performs field operations with delayed mod only when necessary. Result may not be reduced.

```
#include <field-traits.h>
```

### 16.181.1 Detailed Description

Performs field operations with delayed mod only when necessary. Result may not be reduced.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.182 limits< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.183 limits< char > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef char [T](#)

### Static Public Member Functions

- constexpr static char [max](#) () noexcept
- constexpr static char [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

### 16.183.1 Member Typedef Documentation

#### 16.183.1.1 T

```
typedef char T
```

### 16.183.2 Member Function Documentation

#### 16.183.2.1 max()

```
constexpr static char max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.183.2.2 min()

```
constexpr static char min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.183.2.3 digits()

```
constexpr static int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.184 limits< double > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef double [T](#)

### Static Public Member Functions

- constexpr static [int64\\_t](#) [max](#) () noexcept
- constexpr static [int64\\_t](#) [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

### 16.184.1 Member Typedef Documentation

#### 16.184.1.1 T

```
typedef double T
```

### 16.184.2 Member Function Documentation

#### 16.184.2.1 max()

```
constexpr static int64\_t max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.184.2.2 min()

```
constexpr static int64\_t min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.184.2.3 digits()

```
constexpr static int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.185 limits< float > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef float [T](#)

### Static Public Member Functions

- constexpr static [int32\\_t](#) [max](#) () noexcept
- constexpr static [int32\\_t](#) [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

## 16.185.1 Member Typedef Documentation

### 16.185.1.1 T

```
typedef float T
```

## 16.185.2 Member Function Documentation

### 16.185.2.1 max()

```
constexpr static int32_t max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.185.2.2 min()

```
constexpr static int32_t min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.185.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.186 limits< Givaro::Integer > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef Givaro::Integer T

### Static Public Member Functions

- constexpr static int [max](#) () noexcept
- constexpr static int [min](#) () noexcept

## 16.186.1 Member Typedef Documentation

### 16.186.1.1 T

```
typedef Givaro::Integer T
```

## 16.186.2 Member Function Documentation

### 16.186.2.1 max()

```
constexpr static int max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.186.2.2 min()

```
constexpr static int min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.187 limits< int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef int [T](#)

### Static Public Member Functions

- constexpr static int [max](#) () noexcept
- constexpr static int [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

## 16.187.1 Member Typedef Documentation

### 16.187.1.1 T

```
typedef int T
```

## 16.187.2 Member Function Documentation

### 16.187.2.1 max()

```
constexpr static int max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.187.2.2 min()

```
constexpr static int min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.187.2.3 digits()

```
constexpr static int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.188 limits< long > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef long [T](#)

## Static Public Member Functions

- constexpr static long [max](#) () noexcept
- constexpr static long [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

### 16.188.1 Member Typedef Documentation

#### 16.188.1.1 T

typedef long [T](#)

### 16.188.2 Member Function Documentation

#### 16.188.2.1 max()

constexpr static long max ( ) [inline], [static], [constexpr], [noexcept]

#### 16.188.2.2 min()

constexpr static long min ( ) [inline], [static], [constexpr], [noexcept]

#### 16.188.2.3 digits()

constexpr static [int32\\_t](#) digits ( ) [inline], [static], [constexpr], [noexcept]

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.189 limits< long long > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef long long [T](#)

### Static Public Member Functions

- constexpr static long long [max](#) () noexcept
- constexpr static long long [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

### 16.189.1 Member Typedef Documentation

#### 16.189.1.1 T

typedef long long [T](#)

## 16.189.2 Member Function Documentation

### 16.189.2.1 max()

```
constexpr static long long max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.189.2.2 min()

```
constexpr static long long min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.189.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.190 limits< RecInt::rint< K > > Struct Template Reference

```
#include <flimits.h>
```

### Public Types

- typedef [RecInt::ruint](#)< K > [T](#)

### Static Public Member Functions

- constexpr static [RecInt::rint](#)< K > [max](#) ( ) noexcept
- constexpr static [RecInt::rint](#)< K > [min](#) ( ) noexcept

## 16.190.1 Member Typedef Documentation

### 16.190.1.1 T

```
typedef RecInt::ruint<K> T
```

## 16.190.2 Member Function Documentation

### 16.190.2.1 max()

```
constexpr static RecInt::rint<K> max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.190.2.2 min()

```
constexpr static RecInt::rint<K> min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.191 limits< RecInt::ruint< K > > Struct Template Reference

```
#include <flimits.h>
```

### Public Types

- typedef [RecInt::ruint](#)< K > [T](#)

### Static Public Member Functions

- constexpr static [RecInt::ruint](#)< K > [max](#) () noexcept
- constexpr static [RecInt::ruint](#)< K > [min](#) () noexcept

### 16.191.1 Member Typedef Documentation

#### 16.191.1.1 T

```
typedef RecInt::ruint<K> T
```

### 16.191.2 Member Function Documentation

#### 16.191.2.1 max()

```
constexpr static RecInt::ruint<K> max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.191.2.2 min()

```
constexpr static RecInt::ruint<K> min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.192 limits< short int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef short int [T](#)

### Static Public Member Functions

- constexpr static short int [max](#) () noexcept
- constexpr static short int [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

### 16.192.1 Member Typedef Documentation

#### 16.192.1.1 T

```
typedef short int T
```



## 16.192.2 Member Function Documentation

### 16.192.2.1 max()

```
constexpr static short int max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.192.2.2 min()

```
constexpr static short int min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.192.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.193 limits< signed char > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef signed char [T](#)

### Static Public Member Functions

- constexpr static signed char [max](#) () noexcept
- constexpr static signed char [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

## 16.193.1 Member Typedef Documentation

### 16.193.1.1 T

```
typedef signed char T
```

## 16.193.2 Member Function Documentation

### 16.193.2.1 max()

```
constexpr static signed char max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.193.2.2 min()

```
constexpr static signed char min ( ) [inline], [static], [constexpr], [noexcept]
```

**16.193.2.3 digits()**

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

**16.194 limits< unsigned char > Struct Reference**

```
#include <flimits.h>
```

**Public Types**

- typedef unsigned char [T](#)

**Static Public Member Functions**

- constexpr static unsigned char [max](#) () noexcept
- constexpr static unsigned char [min](#) () noexcept
- constexpr static [int32\\_t digits](#) () noexcept

**16.194.1 Member Typedef Documentation****16.194.1.1 T**

```
typedef unsigned char T
```

**16.194.2 Member Function Documentation****16.194.2.1 max()**

```
constexpr static unsigned char max ( ) [inline], [static], [constexpr], [noexcept]
```

**16.194.2.2 min()**

```
constexpr static unsigned char min ( ) [inline], [static], [constexpr], [noexcept]
```

**16.194.2.3 digits()**

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

**16.195 limits< unsigned int > Struct Reference**

```
#include <flimits.h>
```

**Public Types**

- typedef unsigned int [T](#)

## Static Public Member Functions

- constexpr static unsigned int [max](#) () noexcept
- constexpr static unsigned int [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

### 16.195.1 Member Typedef Documentation

#### 16.195.1.1 T

```
typedef unsigned int T
```

### 16.195.2 Member Function Documentation

#### 16.195.2.1 max()

```
constexpr static unsigned int max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.195.2.2 min()

```
constexpr static unsigned int min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.195.2.3 digits()

```
constexpr static int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.196 limits< unsigned long > Struct Reference

```
#include <flimits.h>
```

## Public Types

- typedef unsigned long [T](#)

## Static Public Member Functions

- constexpr static unsigned long [max](#) () noexcept
- constexpr static unsigned long [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

### 16.196.1 Member Typedef Documentation

#### 16.196.1.1 T

```
typedef unsigned long T
```

## 16.196.2 Member Function Documentation

### 16.196.2.1 max()

```
constexpr static unsigned long max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.196.2.2 min()

```
constexpr static unsigned long min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.196.2.3 digits()

```
constexpr static int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.197 limits< unsigned long long > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned long long [T](#)

### Static Public Member Functions

- constexpr static unsigned long long [max](#) () noexcept
- constexpr static unsigned long long [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

## 16.197.1 Member Typedef Documentation

### 16.197.1.1 T

```
typedef unsigned long long T
```

## 16.197.2 Member Function Documentation

### 16.197.2.1 max()

```
constexpr static unsigned long long max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.197.2.2 min()

```
constexpr static unsigned long long min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.197.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.198 limits< unsigned short int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned short int [T](#)

### Static Public Member Functions

- constexpr static unsigned short int [max](#) () noexcept
- constexpr static unsigned short int [min](#) () noexcept
- constexpr static [int32\\_t](#) [digits](#) () noexcept

### 16.198.1 Member Typedef Documentation

#### 16.198.1.1 T

```
typedef unsigned short int T
```

### 16.198.2 Member Function Documentation

#### 16.198.2.1 max()

```
constexpr static unsigned short int max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.198.2.2 min()

```
constexpr static unsigned short int min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.198.2.3 digits()

```
constexpr static int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.199 MachineFloatTag Struct Reference

float or double

```
#include <field-traits.h>
```

### 16.199.1 Detailed Description

float or double

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.200 MachineIntTag Struct Reference

short, int, long, long long, and unsigned variants

```
#include <field-traits.h>
```

### 16.200.1 Detailed Description

short, int, long, long long, and unsigned variants

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.201 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeTrait](#), [ParSeqTrait](#) > [Self\\_t](#)
- typedef [associatedDelayedField](#)< const [Field](#) >::type [DelayedField\\_t](#)
- typedef [associatedDelayedField](#)< const [Field](#) >::field [DelayedField](#)
- typedef [DelayedField::Element](#) [DFElt](#)

### Public Member Functions

- void [initC](#) ()
- void [initA](#) ()
- void [initB](#) ()
- void [initOut](#) ()
- [size\\_t](#) [MaxDelayedDim](#) ([DFElt](#) beta)
- bool [Aunfit](#) ()
- bool [Bunfit](#) ()
- void [setOutBounds](#) (const [size\\_t](#) k, const [DFElt](#) alpha, const [DFElt](#) beta)
- bool [checkA](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_TRANSPOSE](#) ta, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) A, const [size\\_t](#) lda)
- bool [checkB](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_TRANSPOSE](#) tb, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) B, const [size\\_t](#) ldb)
- bool [checkOut](#) (const [Field](#) &F, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) A, const [size\\_t](#) lda)
- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, [size\\_t](#) m, [size\\_t](#) k, [size\\_t](#) n, [ParSeqTrait](#) \_PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())
- template<class F2, typename AlgoT2, typename FT2, typename PS2 >  
  [MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)
- [MMHelper](#) (const [Field](#) &F, int w, [DFElt](#) \_Amin, [DFElt](#) \_Amax, [DFElt](#) \_Bmin, [DFElt](#) \_Bmax, [DFElt](#) \_Cmin, [DFElt](#) \_Cmax, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())

## Data Fields

- int [recLevel](#)
- [DFelt](#) [FieldMin](#)
- [DFelt](#) [FieldMax](#)
- [DFelt](#) [Amin](#)
- [DFelt](#) [Amax](#)
- [DFelt](#) [Bmin](#)
- [DFelt](#) [Bmax](#)
- [DFelt](#) [Cmin](#)
- [DFelt](#) [Cmax](#)
- [DFelt](#) [Outmin](#)
- [DFelt](#) [Outmax](#)
- [DFelt](#) [MaxStorableValue](#)
- const [DelayedField\\_t](#) [delayedField](#)
- [ParSeqTrait](#) [parseq](#)

## Friends

- [std::ostream](#) & [operator<<](#) ([std::ostream](#) &out, const [Self\\_t](#) &M)

## 16.201.1 Member Typedef Documentation

### 16.201.1.1 Self\_t

```
typedef MMHelper<Field,AlgoTrait,ModeTrait,ParSeqTrait> Self_t
```

### 16.201.1.2 DelayedField\_t

```
typedef associatedDelayedField<const Field>::type DelayedField_t
```

### 16.201.1.3 DelayedField

```
typedef associatedDelayedField<const Field>::field DelayedField
```

### 16.201.1.4 DFElt

```
typedef DelayedField::Element DFElt
```

## 16.201.2 Constructor & Destructor Documentation

### 16.201.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

**16.201.2.2 MMHelper() [2/5]**

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

**16.201.2.3 MMHelper() [3/5]**

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

**16.201.2.4 MMHelper() [4/5]**

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

**16.201.2.5 MMHelper() [5/5]**

```
MMHelper (
    const Field & F,
    int w,
    DFelt _Amin,
    DFelt _Amax,
    DFelt _Bmin,
    DFelt _Bmax,
    DFelt _Cmin,
    DFelt _Cmax,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

**16.201.3 Member Function Documentation****16.201.3.1 initC()**

```
void initC ( ) [inline]
```

**16.201.3.2 initA()**

```
void initA ( ) [inline]
```

**16.201.3.3 initB()**

```
void initB ( ) [inline]
```

**16.201.3.4 initOut()**

```
void initOut ( ) [inline]
```



**16.201.3.5 MaxDelayedDim()**

```
size_t MaxDelayedDim (
    DFelt beta ) [inline]
```

**16.201.3.6 Aunfit()**

```
bool Aunfit ( ) [inline]
```

**16.201.3.7 Bunfit()**

```
bool Bunfit ( ) [inline]
```

**16.201.3.8 setOutBounds()**

```
void setOutBounds (
    const size_t k,
    const DFelt alpha,
    const DFelt beta ) [inline]
```

**16.201.3.9 checkA()**

```
bool checkA (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

**16.201.3.10 checkB()**

```
bool checkB (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

**16.201.3.11 checkOut()**

```
bool checkOut (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

**16.201.4 Friends And Related Function Documentation**

**16.201.4.1 operator<<**

```
std::ostream& operator<< (  
    std::ostream & out,  
    const Self_t & M ) [friend]
```

**16.201.5 Field Documentation****16.201.5.1 recLevel**

```
int recLevel
```

**16.201.5.2 FieldMin**

```
DFElt FieldMin
```

**16.201.5.3 FieldMax**

```
DFElt FieldMax
```

**16.201.5.4 Amin**

```
DFElt Amin
```

**16.201.5.5 Amax**

```
DFElt Amax
```

**16.201.5.6 Bmin**

```
DFElt Bmin
```

**16.201.5.7 Bmax**

```
DFElt Bmax
```

**16.201.5.8 Cmin**

```
DFElt Cmin
```

**16.201.5.9 Cmax**

```
DFElt Cmax
```

**16.201.5.10 Outmin**

```
DFElt Outmin
```

### 16.201.5.11 Outmax

[DFelt](#) Outmax

### 16.201.5.12 MaxStorableValue

[DFelt](#) MaxStorableValue

### 16.201.5.13 delayedField

const [DelayedField\\_t](#) delayedField

### 16.201.5.14 parseq

[ParSeqTrait](#) parseq

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.202 MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [FFPACK::RNSInteger](#)< E >, [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (Givaro::Integer [Amax](#), Givaro::Integer [Bmax](#))
- [MMHelper](#) (const [FFPACK::RNSInteger](#)< E > &F, size\_t m, size\_t n, size\_t k, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- [MMHelper](#) (const [FFPACK::RNSInteger](#)< E > &F, int wino, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- template<class F2 , class A2 , class M2 , class PS2 > [MMHelper](#) ([MMHelper](#)< F2, A2, M2, PS2 > H2)
- void [setNorm](#) (Givaro::Integer p)

### Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

### 16.202.1 Member Typedef Documentation

### 16.202.1.1 Self\_t

```
typedef MMHelper<FFPACK::RNSInteger<E>, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait>
Self_t
```

## 16.202.2 Constructor & Destructor Documentation

### 16.202.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

### 16.202.2.2 MMHelper() [2/5]

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

### 16.202.2.3 MMHelper() [3/5]

```
MMHelper (
    const FFPACK::RNSInteger< E > & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

### 16.202.2.4 MMHelper() [4/5]

```
MMHelper (
    const FFPACK::RNSInteger< E > & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

### 16.202.2.5 MMHelper() [5/5]

```
MMHelper (
    MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

## 16.202.3 Member Function Documentation

### 16.202.3.1 setNorm()

```
void setNorm (
    Givaro::Integer p ) [inline]
```

## 16.202.4 Friends And Related Function Documentation

#### 16.202.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

### 16.202.5 Field Documentation

#### 16.202.5.1 normA

Givaro::Integer normA

#### 16.202.5.2 normB

Givaro::Integer normB

#### 16.202.5.3 recLevel

int recLevel

#### 16.202.5.4 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.203 MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

### Public Types

- typedef MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, [ModeCategories::DefaultTag](#), ParSeqTrait > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (Givaro::Integer [Amax](#), Givaro::Integer [Bmax](#))
- [MMHelper](#) (const [FFPACK::RNSIntegerMod](#)< E > &F, size\_t m, size\_t n, size\_t k, ParSeqTrait PS=[ParSeqTrait](#)())
- [MMHelper](#) (const [FFPACK::RNSIntegerMod](#)< E > &F, int wino, ParSeqTrait PS=[ParSeqTrait](#)())
- template<class F2, typename AlgoT2, typename FT2, typename PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)
- void [setNorm](#) (Givaro::Integer p)

### Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- ParSeqTrait [parseq](#)

## Friends

- `std::ostream & operator<< (std::ostream &out, const Self_t &M)`

### 16.203.1 Member Typedef Documentation

#### 16.203.1.1 Self\_t

```
typedef MMHelper<FFPACK::RNSIntegerMod<E>, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait>
Self_t
```

### 16.203.2 Constructor & Destructor Documentation

#### 16.203.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

#### 16.203.2.2 MMHelper() [2/5]

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

#### 16.203.2.3 MMHelper() [3/5]

```
MMHelper (
    const FFPACK::RNSIntegerMod< E > & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

#### 16.203.2.4 MMHelper() [4/5]

```
MMHelper (
    const FFPACK::RNSIntegerMod< E > & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

#### 16.203.2.5 MMHelper() [5/5]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

### 16.203.3 Member Function Documentation

#### 16.203.3.1 setNorm()

```
void setNorm (
    Givaro::Integer p ) [inline]
```

## 16.203.4 Friends And Related Function Documentation

### 16.203.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

## 16.203.5 Field Documentation

### 16.203.5.1 normA

Givaro::Integer normA

### 16.203.5.2 normB

Givaro::Integer normB

### 16.203.5.3 recLevel

int recLevel

### 16.203.5.4 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.204 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< [Dest](#) >, [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, [ParSeqTrait](#) \_PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())
- template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)

### Data Fields

- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.204.1 Member Typedef Documentation

### 16.204.1.1 Self\_t

```
typedef MMHelper<Field,AlgoTrait, ModeCategories::ConvertTo<Dest>,ParSeqTrait> Self_t
```

## 16.204.2 Constructor & Destructor Documentation

### 16.204.2.1 MMHelper() [1/4]

```
MMHelper ( ) [inline]
```

### 16.204.2.2 MMHelper() [2/4]

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

### 16.204.2.3 MMHelper() [3/4]

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

### 16.204.2.4 MMHelper() [4/4]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

## 16.204.3 Friends And Related Function Documentation

### 16.204.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

## 16.204.4 Field Documentation

### 16.204.4.1 recLevel

```
int recLevel
```



#### 16.204.4.2 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.205 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- template<class F2 , class A2 , class M2 , class PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, A2, M2, PS2 > H2)
- [MMHelper](#) (Givaro::Integer [Amax](#), Givaro::Integer [Bmax](#))
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t n, size\_t k, ParSeqTrait PS=ParSeqTrait())
- [MMHelper](#) (const [Field](#) &F, int wino, ParSeqTrait PS=ParSeqTrait())
- void [setNorm](#) (Givaro::Integer p)

### Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- ParSeqTrait [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.205.1 Member Typedef Documentation

### 16.205.1.1 Self\_t

```
typedef MMHelper<Field, AlgoTrait,ModeCategories::ConvertTo<ElementCategories::RNSElementTag>,  
ParSeqTrait> Self\_t
```

## 16.205.2 Constructor & Destructor Documentation

### 16.205.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

**16.205.2.2 MMHelper() [2/5]**

```
MMHelper (
    MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

**16.205.2.3 MMHelper() [3/5]**

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

**16.205.2.4 MMHelper() [4/5]**

```
MMHelper (
    const Field & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

**16.205.2.5 MMHelper() [5/5]**

```
MMHelper (
    const Field & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

**16.205.3 Member Function Documentation****16.205.3.1 setNorm()**

```
void setNorm (
    Givaro::Integer p ) [inline]
```

**16.205.4 Friends And Related Function Documentation****16.205.4.1 operator<<**

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

**16.205.5 Field Documentation****16.205.5.1 normA**

```
Givaro::Integer normA
```

### 16.205.5.2 normB

```
Givaro::Integer normB
```

### 16.205.5.3 recLevel

```
int recLevel
```

### 16.205.5.4 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.206 MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

FGEMM Helper for Default and ConvertTo modes of operation.

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, [ParSeqTrait](#) \_PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())
- template<class F2, typename AlgoT2, typename FT2, typename PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)

### Data Fields

- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

### 16.206.1 Detailed Description

```
template<class Field, typename AlgoTrait, typename ParSeqTrait>
struct FFLAS::MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >
```

FGEMM Helper for Default and ConvertTo modes of operation.

### 16.206.2 Member Typedef Documentation

#### 16.206.2.1 Self\_t

```
typedef MMHelper<Field,AlgoTrait, ModeCategories::DefaultTag,ParSeqTrait> Self\_t
```

### 16.206.3 Constructor & Destructor Documentation

#### 16.206.3.1 MMHelper() [1/4]

```
MMHelper ( ) [inline]
```

#### 16.206.3.2 MMHelper() [2/4]

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

#### 16.206.3.3 MMHelper() [3/4]

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

#### 16.206.3.4 MMHelper() [4/4]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

### 16.206.4 Friends And Related Function Documentation

#### 16.206.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

### 16.206.5 Field Documentation

#### 16.206.5.1 recLevel

```
int recLevel
```

#### 16.206.5.2 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.207 ModeTraits< Field > Struct Template Reference

[ModeTraits.](#)

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultTag](#) value

#### 16.207.1 Detailed Description

```
template<class Field>
```

```
struct FFLAS::ModeTraits< Field >
```

[ModeTraits.](#)

#### 16.207.2 Member Typedef Documentation

##### 16.207.2.1 value

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.208 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DelayedTag](#) value

#### 16.208.1 Member Typedef Documentation

##### 16.208.1.1 value

```
typedef ModeCategories::DelayedTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.209 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

## 16.209.1 Member Typedef Documentation

### 16.209.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.210 ModeTraits< Givaro::Modular< int16\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.210.1 Member Typedef Documentation

#### 16.210.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.211 ModeTraits< Givaro::Modular< int32\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.211.1 Member Typedef Documentation

#### 16.211.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.212 ModeTraits< Givaro::Modular< int8\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.212.1 Member Typedef Documentation

#### 16.212.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.213 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value](#)

### 16.213.1 Member Typedef Documentation

#### 16.213.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.214 ModeTraits< Givaro::Modular< uint16\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.214.1 Member Typedef Documentation

#### 16.214.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.215 ModeTraits< Givaro::Modular< uint32\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.215.1 Member Typedef Documentation

#### 16.215.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.216 ModeTraits< Givaro::Modular< uint8\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.216.1 Member Typedef Documentation

#### 16.216.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.217 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DelayedTag value](#)

### 16.217.1 Member Typedef Documentation



### 16.217.1.1 value

typedef [ModeCategories::DelayedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.218 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

### 16.218.1 Member Typedef Documentation

#### 16.218.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.219 ModeTraits< Givaro::ModularBalanced< int16\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.219.1 Member Typedef Documentation

#### 16.219.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.220 ModeTraits< Givaro::ModularBalanced< int32\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

## 16.220.1 Member Typedef Documentation

### 16.220.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.221 ModeTraits< Givaro::ModularBalanced< int8\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

## 16.221.1 Member Typedef Documentation

### 16.221.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.222 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

## 16.222.1 Member Typedef Documentation

### 16.222.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.223 ModeTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

### 16.223.1 Member Typedef Documentation

#### 16.223.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.224 ModeTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

### 16.224.1 Member Typedef Documentation

#### 16.224.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.225 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

### 16.225.1 Member Typedef Documentation

#### 16.225.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.226 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

## 16.227 ModularTag Struct Reference

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`  
`#include <field-traits.h>`

### 16.227.1 Detailed Description

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`  
 The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.228 Montgomery< T > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

## 16.229 need\_field\_characteristic< Field > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = false

### 16.229.1 Field Documentation

#### 16.229.1.1 value

```
constexpr bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.230 need\_field\_characteristic< Givaro::Modular< Field > > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = true

### 16.230.1 Field Documentation

#### 16.230.1.1 value

```
constexpr bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.231 need\_field\_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = true

### 16.231.1 Field Documentation

#### 16.231.1.1 value

constexpr bool value = true [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.232 NoSimd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

### Public Types

- using [vect\\_t](#) = T \*
- using [scalar\\_t](#) = T

### Static Public Member Functions

- static const std::string [type\\_string](#) ()
- template<class TT >  
static constexpr bool [valid](#) (TT p)
- template<class TT >  
static constexpr bool [compliant](#) (TT n)

### Static Public Attributes

- static constexpr const size\_t [vect\\_size](#) = 1

### 16.232.1 Member Typedef Documentation

#### 16.232.1.1 vect\_t

```
using vect\_t = T*
```

#### 16.232.1.2 scalar\_t

```
using scalar\_t = T
```

### 16.232.2 Member Function Documentation

#### 16.232.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

#### 16.232.2.2 valid()

```
static constexpr bool valid (
    TT p ) [inline], [static], [constexpr]
```

**16.232.2.3 compliant()**

```
static constexpr bool compliant (
    TT n ) [inline], [static], [constexpr]
```

**16.232.3 Field Documentation****16.232.3.1 vect\_size**

```
constexpr const size_t vect_size = 1 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

**16.233 Parallel< C, P > Struct Template Reference****Public Types**

- typedef C [Cut](#)
- typedef P [Param](#)

**Public Member Functions**

- [Parallel](#) (size\_t n=[NUM\\_THREADS](#))
- size\_t [numthreads](#) () const
- size\_t & [set\\_numthreads](#) (size\_t n)

**Friends**

- std::ostream & [operator<<](#) (std::ostream &out, const [Parallel](#) &p)

**16.233.1 Member Typedef Documentation****16.233.1.1 Cut**

```
typedef C Cut
```

**16.233.1.2 Param**

```
typedef P Param
```

**16.233.2 Constructor & Destructor Documentation****16.233.2.1 Parallel()**

```
Parallel (
    size_t n = NUM\_THREADS ) [inline]
```

**16.233.3 Member Function Documentation**

### 16.233.3.1 numthreads()

```
size_t numthreads ( ) const [inline]
```

### 16.233.3.2 set\_numthreads()

```
size_t& set_numthreads (
    size_t n ) [inline]
```

## 16.233.4 Friends And Related Function Documentation

### 16.233.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Parallel< C, P > & p ) [friend]
```

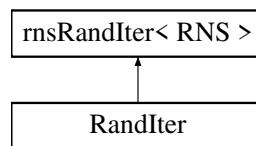
The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.234 RNSInteger< RNS >::RandIter Class Reference

```
#include <rns-integer.h>
```

Inheritance diagram for RNSInteger< RNS >::RandIter:



## Public Member Functions

- [RandIter](#) (const [RNSInteger< RNS >](#) &F, size\_t size=0, uint64\_t seed=0)
- [RNS::Element & random](#) (typename [RNS::Element](#) &elt) const  
*RNS ring Element random assignement.*
- [RNS::Element random](#) () const
- [RNS::Element & operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element operator\(\)](#) () const
- const [RNS](#) & [ring](#) () const

## 16.234.1 Constructor & Destructor Documentation

### 16.234.1.1 RandIter()

```
RandIter (
    const RNSInteger< RNS > & F,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]
```

## 16.234.2 Member Function Documentation

**16.234.2.1 random() [1/2]**

```
RNS::Element& random (
    typename RNS::Element & elt ) const [inline], [inherited]
```

RNS ring Element random assignment.

Element is supposed to be initialized

Returns

random ring Element

**16.234.2.2 random() [2/2]**

```
RNS::Element random ( ) const [inline], [inherited]
```

**16.234.2.3 operator>() [1/2]**

```
RNS::Element& operator() (
    typename RNS::Element & elt ) const [inline], [inherited]
```

**16.234.2.4 operator>() [2/2]**

```
RNS::Element operator() ( ) const [inline], [inherited]
```

**16.234.2.5 ring()**

```
const RNS& ring ( ) const [inline], [inherited]
```

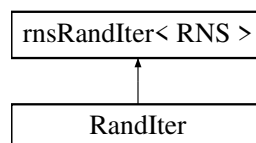
The documentation for this class was generated from the following file:

- [rns-integer.h](#)

**16.235 RNSIntegerMod< RNS >::RandIter Class Reference**

```
#include <rns-integer-mod.h>
```

Inheritance diagram for RNSIntegerMod< RNS >::RandIter:

**Public Member Functions**

- [RandIter](#) (const [RNSIntegerMod](#)< [RNS](#) > &F, size\_t size=0, uint64\_t seed=0)
- [RNS::Element](#) & [random](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element](#) [random](#) () const
- [RNS::Element](#) & [operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element](#) [operator\(\)](#) () const
- const [RNS](#) & [ring](#) () const

**16.235.1 Constructor & Destructor Documentation**



**16.235.1.1 RandIter()**

```
RandIter (
    const RNSIntegerMod< RNS > & F,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]
```

**16.235.2 Member Function Documentation****16.235.2.1 random() [1/2]**

```
RNS::Element& random (
    typename RNS::Element & elt ) const [inline]
```

**16.235.2.2 random() [2/2]**

```
RNS::Element random ( ) const [inline], [inherited]
```

**16.235.2.3 operator>() [1/2]**

```
RNS::Element& operator() (
    typename RNS::Element & elt ) const [inline], [inherited]
```

**16.235.2.4 operator>() [2/2]**

```
RNS::Element operator() ( ) const [inline], [inherited]
```

**16.235.2.5 ring()**

```
const RNS& ring ( ) const [inline], [inherited]
```

The documentation for this class was generated from the following file:

- [rns-integer-mod.h](#)

**16.236 readMyMachineType< Field, T > Struct Template Reference**

```
#include <read_sparse.h>
```

**Public Types**

- typedef [Field::Element](#) Element
- typedef [Field::Element\\_ptr](#) Element\_ptr

**Public Member Functions**

- void [operator\(\)](#) (const [Field](#) &F, [Element](#) &modulo, [Element\\_ptr](#) val, std::ifstream &file, const [uint64\\_t](#) dims, const [mask\\_t](#) data\_type, const [mask\\_t](#) field\_desc)

**16.236.1 Member Typedef Documentation**

### 16.236.1.1 Element

```
typedef Field::Element Element
```

### 16.236.1.2 Element\_ptr

```
typedef Field::Element_ptr Element_ptr
```

## 16.236.2 Member Function Documentation

### 16.236.2.1 operator>()

```
void operator() (
    const Field & F,
    Element & modulo,
    Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

## 16.237 readMyMachineType< Field, mpz\_t > Struct Template Reference

```
#include <read_sparse.h>
```

### Public Types

- typedef [Field::Element](#) [Element](#)
- typedef [Field::Element\\_ptr](#) [Element\\_ptr](#)

### Public Member Functions

- void [operator\(\)](#) (const [Field](#) &F, [Element](#) &modulo, [Element\\_ptr](#) val, std::ifstream &file, const [uint64\\_t](#) dims, const [mask\\_t](#) data\_type, const [mask\\_t](#) field\_desc)

## 16.237.1 Member Typedef Documentation

### 16.237.1.1 Element

```
typedef Field::Element Element
```

### 16.237.1.2 Element\_ptr

```
typedef Field::Element_ptr Element_ptr
```

## 16.237.2 Member Function Documentation

**16.237.2.1 operator()()**

```
void operator() (
    const Field & F,
    typename Field::Element & modulo,
    typename Field::Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

**16.238 Recursive Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.239 Recursive Struct Reference**

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

**16.240 rint< K > Class Template Reference**

The documentation for this class was generated from the following file:

- [field-traits.h](#)

**16.241 rns\_double Struct Reference**

```
#include <rns-double.h>
```

**Public Types**

- typedef Givaro::Integer [integer](#)
- typedef Givaro::Modular< double > [ModField](#)
- typedef double [BasisElement](#)
- typedef [rns\\_double\\_elt](#) [Element](#)
- typedef [rns\\_double\\_elt\\_ptr](#) [Element\\_ptr](#)
- typedef [rns\\_double\\_elt\\_cstptr](#) [ConstElement\\_ptr](#)

**Public Member Functions**

- [rns\\_double](#) (const [integer](#) &bound, size\_t pbits, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double](#) (size\_t pbits, size\_t size, long seed=time(NULL))
- template<typename Vect >  
  [rns\\_double](#) (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double](#) (const [RNSIntegerMod](#)< [rns\\_double](#) > &basis, bool rnsmod=false, long seed=time(NULL))
- void [precompute\\_cst](#) (size\_t K=0)
- template<typename T >  
  void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const T \*A, size\_t lda, const [integer](#) &maxA, bool RNS\_MAJOR=false) const

- void `init` (size\_t m, size\_t n, double \*Arns, size\_t rda, const integer \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- void `init_transpose` (size\_t m, size\_t n, double \*Arns, size\_t rda, const integer \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- void `convert` (size\_t m, size\_t n, integer gamma, integer \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- void `convert_transpose` (size\_t m, size\_t n, integer gamma, integer \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- void `reduce` (size\_t n, double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- template<size\_t K>  
void `init` (size\_t m, size\_t n, double \*Arns, size\_t rda, const Reclnt::ruint< K > \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- template<size\_t K>  
void `convert` (size\_t m, size\_t n, integer gamma, Reclnt::ruint< K > \*A, size\_t lda, const double \*Arns, size\_t rda, integer p=0, bool RNS\_MAJOR=false) const

## Data Fields

- std::vector< double, AlignedAllocator< double, Alignment::CACHE\_LINE > > `_basis`
- std::vector< double, AlignedAllocator< double, Alignment::CACHE\_LINE > > `_basisMax`
- std::vector< double, AlignedAllocator< double, Alignment::CACHE\_LINE > > `_negbasis`
- std::vector< double, AlignedAllocator< double, Alignment::CACHE\_LINE > > `_invbasis`
- std::vector< ModField > `_field_rns`
- integer `_M`
- std::vector< integer > `_Mi`
- std::vector< double > `_MMi`
- std::vector< double > `_crt_in`
- std::vector< double > `_crt_out`
- size\_t `_size`
- size\_t `_pbits`
- size\_t `_ldm`
- integer `_mi_sum`

## 16.241.1 Member Typedef Documentation

### 16.241.1.1 integer

```
typedef Givaro::Integer integer
```

### 16.241.1.2 ModField

```
typedef Givaro::Modular<double> ModField
```

### 16.241.1.3 BasisElement

```
typedef double BasisElement
```

### 16.241.1.4 Element

```
typedef rns_double_elt Element
```

### 16.241.1.5 Element\_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

### 16.241.1.6 ConstElement\_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

## 16.241.2 Constructor & Destructor Documentation

### 16.241.2.1 rns\_double() [1/4]

```
rns_double (
    const integer & bound,
    size_t pbits,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

### 16.241.2.2 rns\_double() [2/4]

```
rns_double (
    size_t pbits,
    size_t size,
    long seed = time(NULL) ) [inline]
```

### 16.241.2.3 rns\_double() [3/4]

```
rns_double (
    const Vect & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

### 16.241.2.4 rns\_double() [4/4]

```
rns_double (
    const RNSIntegerMod< rns_double > & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

## 16.241.3 Member Function Documentation

### 16.241.3.1 precompute\_cst()

```
void precompute_cst (
    size_t K = 0 ) [inline]
```

### 16.241.3.2 init() [1/3]

```
void init (
    size_t m,
    size_t n,
```

```
double * Arns,  
size_t rda,  
const T * A,  
size_t lda,  
const integer & maxA,  
bool RNS_MAJOR = false ) const [inline]
```

#### 16.241.3.3 init() [2/3]

```
void init (  
    size_t m,  
    size_t n,  
    double * Arns,  
    size_t rda,  
    const integer * A,  
    size_t lda,  
    size_t k,  
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.241.3.4 init\_transpose()

```
void init_transpose (  
    size_t m,  
    size_t n,  
    double * Arns,  
    size_t rda,  
    const integer * A,  
    size_t lda,  
    size_t k,  
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.241.3.5 convert() [1/2]

```
void convert (  
    size_t m,  
    size_t n,  
    integer gamma,  
    integer * A,  
    size_t lda,  
    const double * Arns,  
    size_t rda,  
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.241.3.6 convert\_transpose()

```
void convert_transpose (  
    size_t m,  
    size_t n,  
    integer gamma,  
    integer * A,  
    size_t lda,  
    const double * Arns,  
    size_t rda,  
    bool RNS_MAJOR = false ) const [inline]
```

**16.241.3.7 reduce()**

```
void reduce (
    size_t n,
    double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

**16.241.3.8 init() [3/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const RecInt::ruint< K > * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) const [inline]
```

**16.241.3.9 convert() [2/2]**

```
void convert (
    size_t m,
    size_t n,
    integer gamma,
    RecInt::ruint< K > * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    integer p = 0,
    bool RNS_MAJOR = false ) const [inline]
```

**16.241.4 Field Documentation****16.241.4.1 \_basis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

**16.241.4.2 \_basisMax**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

**16.241.4.3 \_negbasis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

**16.241.4.4 \_invbasis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

#### 16.241.4.5 `_field_rns`

```
std::vector<ModField> _field_rns
```

#### 16.241.4.6 `_M`

```
integer _M
```

#### 16.241.4.7 `_Mi`

```
std::vector<integer> _Mi
```

#### 16.241.4.8 `_MMi`

```
std::vector<double> _MMi
```

#### 16.241.4.9 `_crt_in`

```
std::vector<double> _crt_in
```

#### 16.241.4.10 `_crt_out`

```
std::vector<double> _crt_out
```

#### 16.241.4.11 `_size`

```
size_t _size
```

#### 16.241.4.12 `_pbits`

```
size_t _pbits
```

#### 16.241.4.13 `_ldm`

```
size_t _ldm
```

#### 16.241.4.14 `_mi_sum`

```
integer _mi_sum
```

The documentation for this struct was generated from the following files:

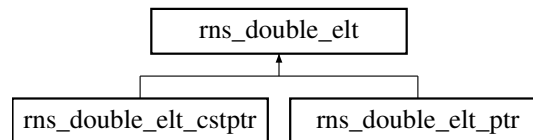
- [rns-double.h](#)
- [rns-double-recint.inl](#)
- [rns-double.inl](#)

## 16.242 `rns_double_elt` Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for `rns_double_elt`:





## Public Member Functions

- [rns\\_double\\_elt](#) ()
- [~rns\\_double\\_elt](#) ()
- [rns\\_double\\_elt](#) (double \*p, size\_t r, size\_t a=false)
- [rns\\_double\\_elt\\_ptr](#) operator& ()
- [rns\\_double\\_elt\\_cstptr](#) operator& () const
- [rns\\_double\\_elt](#) (const [rns\\_double\\_elt](#) &x)

## Data Fields

- double \* [\\_ptr](#)
- size\_t [\\_stride](#)
- bool [\\_alloc](#)

## 16.242.1 Constructor & Destructor Documentation

### 16.242.1.1 [rns\\_double\\_elt](#)() [1/3]

```
rns\_double\_elt ( ) [inline]
```

### 16.242.1.2 [~rns\\_double\\_elt](#)()

```
~rns\_double\_elt ( ) [inline]
```

### 16.242.1.3 [rns\\_double\\_elt](#)() [2/3]

```
rns\_double\_elt (
    double * p,
    size_t r,
    size_t a = false ) [inline]
```

### 16.242.1.4 [rns\\_double\\_elt](#)() [3/3]

```
rns\_double\_elt (
    const rns\_double\_elt & x ) [inline]
```

## 16.242.2 Member Function Documentation

### 16.242.2.1 [operator&\(\)](#) [1/2]

```
rns\_double\_elt\_ptr operator& ( ) [inline]
```

### 16.242.2.2 operator&() [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline]
```

## 16.242.3 Field Documentation

### 16.242.3.1 \_ptr

```
double* _ptr
```

### 16.242.3.2 \_stride

```
size_t _stride
```

### 16.242.3.3 \_alloc

```
bool _alloc
```

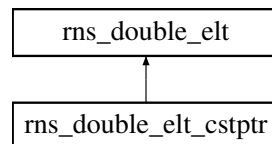
The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

## 16.243 rns\_double\_elt\_cstptr Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for rns\_double\_elt\_cstptr:



## Public Member Functions

- [rns\\_double\\_elt\\_cstptr](#) ()
- [rns\\_double\\_elt\\_cstptr](#) (double \*p, size\_t r)
- [rns\\_double\\_elt\\_cstptr](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) ([rns\\_double\\_elt\\_cstptr](#) &&)=default
- [rns\\_double\\_elt\\_cstptr](#) \* [operator&](#) ()
- [rns\\_double\\_elt](#) & [operator\\*](#) () const
- [rns\\_double\\_elt](#) [operator\[\]](#) (size\_t i) const
- [rns\\_double\\_elt](#) & [operator\[\]](#) (size\_t i)
- [rns\\_double\\_elt\\_cstptr](#) [operator++](#) ()
- [rns\\_double\\_elt\\_cstptr](#) [operator--](#) ()
- [rns\\_double\\_elt\\_cstptr](#) [operator+](#) (size\_t inc) const
- [rns\\_double\\_elt\\_cstptr](#) [operator-](#) (size\_t inc) const
- [rns\\_double\\_elt\\_cstptr](#) & [operator+=](#) (size\_t inc)
- [rns\\_double\\_elt\\_cstptr](#) & [operator-=](#) (size\_t inc)
- [rns\\_double\\_elt\\_cstptr](#) & [operator=](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- bool [operator<](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- bool [operator!=](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) [operator&](#) () const

## Data Fields

- [rns\\_double\\_elt](#) other
- `double * \_ptr`
- `size_t \_stride`
- `bool \_alloc`

## 16.243.1 Constructor & Destructor Documentation

### 16.243.1.1 `rns_double_elt_cstptr()` [1/5]

```
rns_double_elt_cstptr ( ) [inline]
```

### 16.243.1.2 `rns_double_elt_cstptr()` [2/5]

```
rns_double_elt_cstptr (
    double * p,
    size_t r ) [inline]
```

### 16.243.1.3 `rns_double_elt_cstptr()` [3/5]

```
rns_double_elt_cstptr (
    const rns\_double\_elt\_ptr & x ) [inline]
```

### 16.243.1.4 `rns_double_elt_cstptr()` [4/5]

```
rns_double_elt_cstptr (
    const rns\_double\_elt\_cstptr & x ) [inline]
```

### 16.243.1.5 `rns_double_elt_cstptr()` [5/5]

```
rns_double_elt_cstptr (
    rns\_double\_elt\_cstptr && ) [default]
```

## 16.243.2 Member Function Documentation

### 16.243.2.1 `operator&()` [1/2]

```
rns\_double\_elt\_cstptr* operator& ( ) [inline]
```

### 16.243.2.2 `operator*()`

```
rns\_double\_elt& operator* ( ) const [inline]
```

### 16.243.2.3 `operator[]()` [1/2]

```
rns\_double\_elt operator[] (
    size_t i ) const [inline]
```

**16.243.2.4 operator[]()** [2/2]

```
rns_double_elt& operator[] (
    size_t i ) [inline]
```

**16.243.2.5 operator++()**

```
rns_double_elt_cstptr operator++ ( ) [inline]
```

**16.243.2.6 operator--()**

```
rns_double_elt_cstptr operator-- ( ) [inline]
```

**16.243.2.7 operator+()**

```
rns_double_elt_cstptr operator+ (
    size_t inc ) const [inline]
```

**16.243.2.8 operator-()**

```
rns_double_elt_cstptr operator- (
    size_t inc ) const [inline]
```

**16.243.2.9 operator+=()**

```
rns_double_elt_cstptr& operator+= (
    size_t inc ) [inline]
```

**16.243.2.10 operator-=()**

```
rns_double_elt_cstptr& operator-= (
    size_t inc ) [inline]
```

**16.243.2.11 operator=()**

```
rns_double_elt_cstptr & operator= (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.243.2.12 operator<()**

```
bool operator< (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.243.2.13 operator"!="()**

```
bool operator!= (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.243.2.14 operator&()** [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```

### 16.243.3 Field Documentation

#### 16.243.3.1 other

`rns_double_elt` other

#### 16.243.3.2 \_ptr

`double* _ptr` [inherited]

#### 16.243.3.3 \_stride

`size_t _stride` [inherited]

#### 16.243.3.4 \_alloc

`bool _alloc` [inherited]

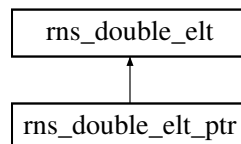
The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

## 16.244 rns\_double\_elt\_ptr Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for `rns_double_elt_ptr`:



### Public Member Functions

- [rns\\_double\\_elt\\_ptr](#) ()
- [rns\\_double\\_elt\\_ptr](#) (double \*p, size\_t r)
- [rns\\_double\\_elt\\_ptr](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_ptr](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_ptr](#) ([rns\\_double\\_elt\\_ptr](#) &&)=default
- [rns\\_double\\_elt\\_ptr](#) \* [operator&](#) ()
- [rns\\_double\\_elt](#) & [operator\\*](#) ()
- [rns\\_double\\_elt](#) [operator\[\]](#) (size\_t i) const
- [rns\\_double\\_elt](#) & [operator\[\]](#) (size\_t i)
- [rns\\_double\\_elt\\_ptr](#) [operator++](#) ()
- [rns\\_double\\_elt\\_ptr](#) [operator--](#) ()
- [rns\\_double\\_elt\\_ptr](#) [operator+](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) [operator-](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator+=](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator-=](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator=](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- `bool` [operator<](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- `bool` [operator!=](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) [operator&](#) () const

## Data Fields

- [rns\\_double\\_elt](#) other
- double \* [\\_ptr](#)
- size\_t [\\_stride](#)
- bool [\\_alloc](#)

## 16.244.1 Constructor & Destructor Documentation

### 16.244.1.1 [rns\\_double\\_elt\\_ptr\(\)](#) [1/5]

```
rns\_double\_elt\_ptr ( ) [inline]
```

### 16.244.1.2 [rns\\_double\\_elt\\_ptr\(\)](#) [2/5]

```
rns\_double\_elt\_ptr (
    double * p,
    size_t r ) [inline]
```

### 16.244.1.3 [rns\\_double\\_elt\\_ptr\(\)](#) [3/5]

```
rns\_double\_elt\_ptr (
    const rns\_double\_elt\_ptr & x ) [inline]
```

### 16.244.1.4 [rns\\_double\\_elt\\_ptr\(\)](#) [4/5]

```
rns\_double\_elt\_ptr (
    const rns\_double\_elt\_cstptr & x ) [inline]
```

### 16.244.1.5 [rns\\_double\\_elt\\_ptr\(\)](#) [5/5]

```
rns\_double\_elt\_ptr (
    rns\_double\_elt\_ptr && ) [default]
```

## 16.244.2 Member Function Documentation

### 16.244.2.1 [operator&\(\)](#) [1/2]

```
rns\_double\_elt\_ptr* operator& ( ) [inline]
```

### 16.244.2.2 [operator\\*\(\)](#)

```
rns\_double\_elt& operator\* ( ) [inline]
```

### 16.244.2.3 [operator\[\]\(\)](#) [1/2]

```
rns\_double\_elt operator\[\] (
    size_t i ) const [inline]
```

**16.244.2.4 operator[]()** [2/2]

```
rns_double_elt& operator[] (
    size_t i ) [inline]
```

**16.244.2.5 operator++()**

```
rns_double_elt_ptr operator++ ( ) [inline]
```

**16.244.2.6 operator--()**

```
rns_double_elt_ptr operator-- ( ) [inline]
```

**16.244.2.7 operator+()**

```
rns_double_elt_ptr operator+ (
    size_t inc ) [inline]
```

**16.244.2.8 operator-()**

```
rns_double_elt_ptr operator- (
    size_t inc ) [inline]
```

**16.244.2.9 operator+=()**

```
rns_double_elt_ptr& operator+= (
    size_t inc ) [inline]
```

**16.244.2.10 operator-=()**

```
rns_double_elt_ptr& operator-= (
    size_t inc ) [inline]
```

**16.244.2.11 operator=()**

```
rns_double_elt_ptr & operator= (
    const rns_double_elt_ptr & x ) [inline]
```

**16.244.2.12 operator<()**

```
bool operator< (
    const rns_double_elt_ptr & x ) [inline]
```

**16.244.2.13 operator"!="()**

```
bool operator!= (
    const rns_double_elt_ptr & x ) [inline]
```

**16.244.2.14 operator&()** [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```

### 16.244.3 Field Documentation

#### 16.244.3.1 other

`rns_double_elt` other

#### 16.244.3.2 \_ptr

`double* _ptr` [inherited]

#### 16.244.3.3 \_stride

`size_t _stride` [inherited]

#### 16.244.3.4 \_alloc

`bool _alloc` [inherited]

The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

## 16.245 rns\_double\_extended Struct Reference

```
#include <rns-double.h>
```

### Public Types

- typedef Givaro::Integer [integer](#)
- typedef Givaro::ModularExtended< double > [ModField](#)
- typedef double [BasisElement](#)
- typedef [rns\\_double\\_elt](#) [Element](#)
- typedef [rns\\_double\\_elt\\_ptr](#) [Element\\_ptr](#)
- typedef [rns\\_double\\_elt\\_cstptr](#) [ConstElement\\_ptr](#)

### Public Member Functions

- [rns\\_double\\_extended](#) (const [integer](#) &bound, size\_t pbits, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double\\_extended](#) (size\_t pbits, size\_t size, long seed=time(NULL))
- template<typename Vect >  
  [rns\\_double\\_extended](#) (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- void [precompute\\_cst](#) ()
- void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [integer](#) \*A, size\_t lda, const [integer](#) &maxA, bool RNS\_MAJOR=false) const
- void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [integer](#) \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false)
- void [convert](#) (size\_t m, size\_t n, [integer](#) gamma, [integer](#) \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false)
- void [init](#) (size\_t m, double \*Arns, const [integer](#) \*A, size\_t lda) const
- void [convert](#) (size\_t m, [integer](#) \*A, const double \*Arns) const
- void [reduce](#) (size\_t n, double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const



## Data Fields

- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basis`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basisMax`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _negbasis`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _invbasis`
- `std::vector< ModField > _field_rns`
- `integer _M`
- `std::vector< integer > _Mi`
- `std::vector< double > _MMi`
- `std::vector< double > _crt_in`
- `std::vector< double > _crt_out`
- `size_t _size`
- `size_t _pbits`
- `size_t _ldm`

## 16.245.1 Member Typedef Documentation

### 16.245.1.1 integer

```
typedef Givaro::Integer integer
```

### 16.245.1.2 ModField

```
typedef Givaro::ModularExtended<double> ModField
```

### 16.245.1.3 BasisElement

```
typedef double BasisElement
```

### 16.245.1.4 Element

```
typedef rns\_double\_elt Element
```

### 16.245.1.5 Element\_ptr

```
typedef rns\_double\_elt\_ptr Element\_ptr
```

### 16.245.1.6 ConstElement\_ptr

```
typedef rns\_double\_elt\_cstptr ConstElement\_ptr
```

## 16.245.2 Constructor & Destructor Documentation

**16.245.2.1 rns\_double\_extended() [1/3]**

```
rns_double_extended (
    const integer & bound,
    size_t pbits,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.245.2.2 rns\_double\_extended() [2/3]**

```
rns_double_extended (
    size_t pbits,
    size_t size,
    long seed = time(NULL) ) [inline]
```

**16.245.2.3 rns\_double\_extended() [3/3]**

```
rns_double_extended (
    const Vect & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.245.3 Member Function Documentation****16.245.3.1 precompute\_cst()**

```
void precompute_cst ( ) [inline]
```

**16.245.3.2 init() [1/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    const integer & maxA,
    bool RNS_MAJOR = false ) const [inline]
```

**16.245.3.3 init() [2/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) [inline]
```

**16.245.3.4 convert() [1/2]**

```
void convert (
    size_t m,
    size_t n,
    integer gamma,
    integer * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) [inline]
```

**16.245.3.5 init() [3/3]**

```
void init (
    size_t m,
    double * Arns,
    const integer * A,
    size_t lda ) const [inline]
```

**16.245.3.6 convert() [2/2]**

```
void convert (
    size_t m,
    integer * A,
    const double * Arns ) const [inline]
```

**16.245.3.7 reduce()**

```
void reduce (
    size_t n,
    double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

**16.245.4 Field Documentation****16.245.4.1 \_basis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

**16.245.4.2 \_basisMax**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

**16.245.4.3 \_negbasis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

#### 16.245.4.4 `_invbasis`

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

#### 16.245.4.5 `_field_rns`

```
std::vector<ModField> _field_rns
```

#### 16.245.4.6 `_M`

```
integer _M
```

#### 16.245.4.7 `_Mi`

```
std::vector<integer> _Mi
```

#### 16.245.4.8 `_MMi`

```
std::vector<double> _MMi
```

#### 16.245.4.9 `_crt_in`

```
std::vector<double> _crt_in
```

#### 16.245.4.10 `_crt_out`

```
std::vector<double> _crt_out
```

#### 16.245.4.11 `_size`

```
size_t _size
```

#### 16.245.4.12 `_pbits`

```
size_t _pbits
```

#### 16.245.4.13 `_ldm`

```
size_t _ldm
```

The documentation for this struct was generated from the following files:

- [rns-double.h](#)
- [rns-double.inl](#)

## 16.246 RNSElementTag Struct Reference

Representation in a Residue Number System.

```
#include <field-traits.h>
```

### 16.246.1 Detailed Description

Representation in a Residue Number System.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.247 RNSInteger< RNS > Class Template Reference

```
#include <rns-integer.h>
```

### Data Structures

- class [RandIter](#)

### Public Types

- typedef [RNS::Element](#) [Element](#)
- typedef [RNS::Element\\_ptr](#) [Element\\_ptr](#)
- typedef [RNS::ConstElement\\_ptr](#) [ConstElement\\_ptr](#)

### Public Member Functions

- [RNSInteger](#) (const [RNS](#) &myrns)
- template<typename T >  
  [RNSInteger](#) (const T &F)
- const [RNS](#) & [rns](#) () const
- size\_t [size](#) () const
- bool [isOne](#) (const [Element](#) &x) const
- bool [isMOne](#) (const [Element](#) &x) const
- bool [isZero](#) (const [Element](#) &x) const
- integer [characteristic](#) ([integer](#) &p) const
- integer [cardinality](#) ([integer](#) &p) const
- [Element](#) & [init](#) ([Element](#) &x) const
- [Element](#) & [init](#) ([Element](#) &x, const Givaro::Integer &y) const
- [Element](#) & [reduce](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) & [reduce](#) ([Element](#) &x) const
- Givaro::Integer [convert](#) (Givaro::Integer &x, const [Element](#) &y) const
- [Element](#) & [assign](#) ([Element](#) &x, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os) const

### Data Fields

- [Element](#) [one](#)
- [Element](#) [mOne](#)
- [Element](#) [zero](#)

### Protected Types

- typedef [RNS::BasisElement](#) [BasisElement](#)
- typedef Givaro::Integer [integer](#)

### Protected Attributes

- const [RNS](#) \* [\\_rns](#)

## 16.247.1 Member Typedef Documentation

### 16.247.1.1 BasisElement

```
typedef RNS::BasisElement BasisElement [protected]
```

### 16.247.1.2 integer

```
typedef Givaro::Integer integer [protected]
```

### 16.247.1.3 Element

```
typedef RNS::Element Element
```

### 16.247.1.4 Element\_ptr

```
typedef RNS::Element_ptr Element_ptr
```

### 16.247.1.5 ConstElement\_ptr

```
typedef RNS::ConstElement_ptr ConstElement_ptr
```

## 16.247.2 Constructor & Destructor Documentation

### 16.247.2.1 RNSInteger() [1/2]

```
RNSInteger (
    const RNS & myrns ) [inline]
```

### 16.247.2.2 RNSInteger() [2/2]

```
RNSInteger (
    const T & F ) [inline]
```

## 16.247.3 Member Function Documentation

### 16.247.3.1 rns()

```
const RNS& rns ( ) const [inline]
```

### 16.247.3.2 size()

```
size_t size ( ) const [inline]
```

**16.247.3.3 isOne()**

```
bool isOne (
    const Element & x ) const [inline]
```

**16.247.3.4 isMOne()**

```
bool isMOne (
    const Element & x ) const [inline]
```

**16.247.3.5 isZero()**

```
bool isZero (
    const Element & x ) const [inline]
```

**16.247.3.6 characteristic()**

```
integer characteristic (
    integer & p ) const [inline]
```

**16.247.3.7 cardinality()**

```
integer cardinality (
    integer & p ) const [inline]
```

**16.247.3.8 init() [1/2]**

```
Element& init (
    Element & x ) const [inline]
```

**16.247.3.9 init() [2/2]**

```
Element& init (
    Element & x,
    const Givaro::Integer & y ) const [inline]
```

**16.247.3.10 reduce() [1/2]**

```
Element& reduce (
    Element & x,
    const Element & y ) const [inline]
```

**16.247.3.11 reduce() [2/2]**

```
Element& reduce (
    Element & x ) const [inline]
```

**16.247.3.12 convert()**

```
Givaro::Integer convert (
    Givaro::Integer & x,
    const Element & y ) const [inline]
```

**16.247.3.13 assign()**

```
Element& assign (
    Element & x,
    const Element & y ) const [inline]
```

**16.247.3.14 write() [1/2]**

```
std::ostream& write (
    std::ostream & os,
    const Element & y ) const [inline]
```

**16.247.3.15 write() [2/2]**

```
std::ostream& write (
    std::ostream & os ) const [inline]
```

**16.247.4 Field Documentation****16.247.4.1 \_rns**

```
const RNS* _rns [protected]
```

**16.247.4.2 one**

```
Element one
```

**16.247.4.3 mOne**

```
Element mOne
```

**16.247.4.4 zero**

```
Element zero
```

The documentation for this class was generated from the following files:

- [field-traits.h](#)
- [rns-integer.h](#)

**16.248 RNSIntegerMod< RNS > Class Template Reference**

```
#include <rns-integer-mod.h>
```

**Data Structures**

- class [RandIter](#)



## Public Types

- typedef [RNS::Element](#) [Element](#)
- typedef [RNS::Element\\_ptr](#) [Element\\_ptr](#)
- typedef [RNS::ConstElement\\_ptr](#) [ConstElement\\_ptr](#)

## Public Member Functions

- [RNSIntegerMod](#) (const [integer](#) &p, const [RNS](#) &myrns)
- const [rns\\_double](#) &[rns](#) () const
- const [RNSInteger](#)< [RNS](#) > &[delayed](#) () const
- [size\\_t](#) [size](#) () const
- bool [isOne](#) (const [Element](#) &x) const
- bool [isMOne](#) (const [Element](#) &x) const
- bool [isZero](#) (const [Element](#) &x) const
- [integer](#) &[characteristic](#) ([integer](#) &p) const
- [integer](#) [characteristic](#) () const
- [integer](#) &[cardinality](#) ([integer](#) &p) const
- [integer](#) [cardinality](#) () const
- [integer](#) [minElement](#) () const
- [integer](#) [maxElement](#) () const
- [Element](#) &[init](#) ([Element](#) &x) const
- [Element](#) &[init](#) ([Element](#) &x, const Givaro::Integer &y) const
- [Element](#) &[reduce](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[reduce](#) ([Element](#) &x) const
- [Element](#) &[init](#) ([Element](#) &x, const [Element](#) &y) const
- Givaro::Integer [convert](#) (Givaro::Integer &x, const [Element](#) &y) const
- [Element](#) &[assign](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[add](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[sub](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[neg](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[mul](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[axpyin](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[inv](#) ([Element](#) &x, const [Element](#) &y) const
- bool [areEqual](#) (const [Element](#) &x, const [Element](#) &y) const
- std::ostream &[write](#) (std::ostream &os, const [Element](#) &y) const
- std::ostream &[write](#) (std::ostream &os) const
- void [reduce\\_modp](#) ([size\\_t](#) n, [Element\\_ptr](#) B) const
- std::ostream &[write\\_matrix](#) (std::ostream &c, const double \*E, int n, int m, int lda) const
- std::ostream &[write\\_matrix\\_long](#) (std::ostream &c, const double \*E, int n, int m, int lda) const
- void [reduce\\_modp](#) ([size\\_t](#) m, [size\\_t](#) n, [Element\\_ptr](#) B, [size\\_t](#) lda) const
- void [reduce\\_modp\\_rnsmajor](#) ([size\\_t](#) n, [Element\\_ptr](#) B) const

## Data Fields

- [Element](#) [one](#)
- [Element](#) [mOne](#)
- [Element](#) [zero](#)

## Protected Types

- typedef [RNS::BasisElement](#) [BasisElement](#)
- typedef Givaro::Modular< [BasisElement](#) > [ModField](#)
- typedef Givaro::Integer [integer](#)

## Protected Attributes

- `integer _p`
- `std::vector< BasisElement, AlignedAllocator< BasisElement, Alignment::CACHE_LINE > > _Mi_modp_rns`
- `std::vector< BasisElement, AlignedAllocator< BasisElement, Alignment::CACHE_LINE > > _iM_modp_rns`
- `const RNS * _rns`
- `Givaro::Modular< Givaro::Integer > _F`
- `RNSInteger< RNS > _RNSdelayed`

## 16.248.1 Member Typedef Documentation

### 16.248.1.1 Element

```
typedef RNS::Element Element
```

### 16.248.1.2 Element\_ptr

```
typedef RNS::Element\_ptr Element_ptr
```

### 16.248.1.3 ConstElement\_ptr

```
typedef RNS::ConstElement\_ptr ConstElement_ptr
```

### 16.248.1.4 BasisElement

```
typedef RNS::BasisElement BasisElement [protected]
```

### 16.248.1.5 ModField

```
typedef Givaro::Modular<BasisElement> ModField [protected]
```

### 16.248.1.6 integer

```
typedef Givaro::Integer integer [protected]
```

## 16.248.2 Constructor & Destructor Documentation

### 16.248.2.1 RNSIntegerMod()

```
RNSIntegerMod (
    const integer & p,
    const RNS & myrns ) [inline]
```

## 16.248.3 Member Function Documentation

### 16.248.3.1 rns()

```
const rns\_double& rns ( ) const [inline]
```

**16.248.3.2 delayed()**

```
const RNSInteger<RNS>& delayed ( ) const [inline]
```

**16.248.3.3 size()**

```
size_t size ( ) const [inline]
```

**16.248.3.4 isOne()**

```
bool isOne (
    const Element & x ) const [inline]
```

**16.248.3.5 isMOne()**

```
bool isMOne (
    const Element & x ) const [inline]
```

**16.248.3.6 isZero()**

```
bool isZero (
    const Element & x ) const [inline]
```

**16.248.3.7 characteristic() [1/2]**

```
integer& characteristic (
    integer & p ) const [inline]
```

**16.248.3.8 characteristic() [2/2]**

```
integer characteristic ( ) const [inline]
```

**16.248.3.9 cardinality() [1/2]**

```
integer& cardinality (
    integer & p ) const [inline]
```

**16.248.3.10 cardinality() [2/2]**

```
integer cardinality ( ) const [inline]
```

**16.248.3.11 minElement()**

```
integer minElement ( ) const [inline]
```

**16.248.3.12 maxElement()**

```
integer maxElement ( ) const [inline]
```

**16.248.3.13 init() [1/3]**

```
Element& init (
    Element & x ) const [inline]
```

**16.248.3.14 init() [2/3]**

```
Element& init (
    Element & x,
    const Givaro::Integer & y ) const [inline]
```

**16.248.3.15 reduce() [1/2]**

```
Element& reduce (
    Element & x,
    const Element & y ) const [inline]
```

**16.248.3.16 reduce() [2/2]**

```
Element& reduce (
    Element & x ) const [inline]
```

**16.248.3.17 init() [3/3]**

```
Element& init (
    Element & x,
    const Element & y ) const [inline]
```

**16.248.3.18 convert()**

```
Givaro::Integer convert (
    Givaro::Integer & x,
    const Element & y ) const [inline]
```

**16.248.3.19 assign()**

```
Element& assign (
    Element & x,
    const Element & y ) const [inline]
```

**16.248.3.20 add()**

```
Element& add (
    Element & x,
    const Element & y,
    const Element & z ) const [inline]
```

**16.248.3.21 sub()**

```
Element& sub (
    Element & x,
```

```
const Element & y,  
const Element & z ) const [inline]
```

#### 16.248.3.22 neg()

```
Element& neg (  
    Element & x,  
    const Element & y ) const [inline]
```

#### 16.248.3.23 mul()

```
Element& mul (  
    Element & x,  
    const Element & y,  
    const Element & z ) const [inline]
```

#### 16.248.3.24 axpyin()

```
Element& axpyin (  
    Element & x,  
    const Element & y,  
    const Element & z ) const [inline]
```

#### 16.248.3.25 inv()

```
Element& inv (  
    Element & x,  
    const Element & y ) const [inline]
```

#### 16.248.3.26 areEqual()

```
bool areEqual (  
    const Element & x,  
    const Element & y ) const [inline]
```

#### 16.248.3.27 write() [1/2]

```
std::ostream& write (  
    std::ostream & os,  
    const Element & y ) const [inline]
```

#### 16.248.3.28 write() [2/2]

```
std::ostream& write (  
    std::ostream & os ) const [inline]
```

#### 16.248.3.29 reduce\_modp() [1/2]

```
void reduce_modp (  
    size_t n,  
    Element_ptr B ) const [inline]
```

**16.248.3.30 write\_matrix()**

```
std::ostream& write_matrix (
    std::ostream & c,
    const double * E,
    int n,
    int m,
    int lda ) const [inline]
```

**16.248.3.31 write\_matrix\_long()**

```
std::ostream& write_matrix_long (
    std::ostream & c,
    const double * E,
    int n,
    int m,
    int lda ) const [inline]
```

**16.248.3.32 reduce\_modp() [2/2]**

```
void reduce_modp (
    size_t m,
    size_t n,
    Element_ptr B,
    size_t lda ) const [inline]
```

**16.248.3.33 reduce\_modp\_rnsmajor()**

```
void reduce_modp_rnsmajor (
    size_t n,
    Element_ptr B ) const [inline]
```

**16.248.4 Field Documentation****16.248.4.1 \_p**

```
integer _p [protected]
```

**16.248.4.2 \_Mi\_modp\_rns**

```
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _Mi_modp_↵
rns [protected]
```

**16.248.4.3 \_iM\_modp\_rns**

```
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _iM_modp_↵
rns [protected]
```

**16.248.4.4 \_rns**

```
const RNS* _rns [protected]
```

16.248.4.5 `_F`

```
Givaro::Modular<Givaro::Integer> _F [protected]
```

16.248.4.6 `_RNSdelayed`

```
RNSInteger<RNS> _RNSdelayed [protected]
```

16.248.4.7 `one`

```
Element one
```

16.248.4.8 `mOne`

```
Element mOne
```

16.248.4.9 `zero`

```
Element zero
```

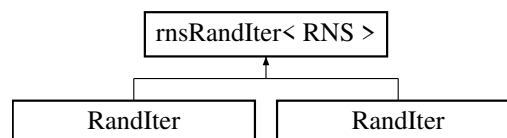
The documentation for this class was generated from the following files:

- [field-traits.h](#)
- [rns-integer-mod.h](#)

16.249 `rnsRandIter< RNS >` Class Template Reference

```
#include <rns-double.h>
```

Inheritance diagram for `rnsRandIter< RNS >`:



## Public Member Functions

- `rnsRandIter` (const [RNS](#) &R, `size_t` size=0, `uint64_t` seed=0)
- `RNS::Element & random` (typename [RNS::Element](#) &elt) const  
*RNS ring Element random assignement.*
- `RNS::Element & operator()` (typename [RNS::Element](#) &elt) const
- `RNS::Element operator()` () const
- `RNS::Element random` () const
- `const RNS & ring` () const

## 16.249.1 Constructor &amp; Destructor Documentation

16.249.1.1 `rnsRandIter()`

```
rnsRandIter (
    const RNS & R,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]
```

## 16.249.2 Member Function Documentation

### 16.249.2.1 random() [1/2]

```
RNS::Element& random (
    typename RNS::Element & elt ) const [inline]
```

RNS ring Element random assignment.

Element is supposed to be initialized

Returns

random ring Element

### 16.249.2.2 operator>() [1/2]

```
RNS::Element& operator() (
    typename RNS::Element & elt ) const [inline]
```

### 16.249.2.3 operator>() [2/2]

```
RNS::Element operator() ( ) const [inline]
```

### 16.249.2.4 random() [2/2]

```
RNS::Element random ( ) const [inline]
```

### 16.249.2.5 ring()

```
const RNS& ring ( ) const [inline]
```

The documentation for this class was generated from the following file:

- [rns-double.h](#)

## 16.250 Row Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.251 ruint< K > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

## 16.252 ScalFunctions< Element, Enable > Struct Template Reference

The documentation for this struct was generated from the following file:

- [test-simd.C](#)



## 16.253 ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type > Struct Template Reference

### Static Public Member Functions

- static Element [zero](#) ()
- static Element [vand](#) (Element x1, Element x2)
- static Element [vor](#) (Element x1, Element x2)
- static Element [vxor](#) (Element x1, Element x2)
- static Element [vandnot](#) (Element x1, Element x2)
- static Element [ceil](#) (Element x)
- static Element [floor](#) (Element x)
- static Element [round](#) (Element x)
- static Element [add](#) (Element x1, Element x2)
- static Element [addin](#) (Element &x1, Element x2)
- static Element [sub](#) (Element x1, Element x2)
- static Element [subin](#) (Element &x1, Element x2)
- static Element [mul](#) (Element x1, Element x2)
- static Element [mulin](#) (Element &x1, Element x2)
- static Element [div](#) (Element x1, Element x2)
- static Element [fmadd](#) (Element x1, Element x2, Element x3)
- static Element [fmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsub](#) (Element x1, Element x2, Element x3)
- static Element [fmsubin](#) (Element &x1, Element x2, Element x3)
- static Element [fnmadd](#) (Element x1, Element x2, Element x3)
- static Element [fnmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [lesser](#) (Element x1, Element x2)
- static Element [lesser\\_eq](#) (Element x1, Element x2)
- static Element [greater](#) (Element x1, Element x2)
- static Element [greater\\_eq](#) (Element x1, Element x2)
- static Element [eq](#) (Element x1, Element x2)

### 16.253.1 Member Function Documentation

#### 16.253.1.1 zero()

```
static Element zero ( ) [inline], [static]
```

#### 16.253.1.2 vand()

```
static Element vand (
    Element x1,
    Element x2 ) [inline], [static]
```

#### 16.253.1.3 vor()

```
static Element vor (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.4 vxor()**

```
static Element vxor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.5 vandnot()**

```
static Element vandnot (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.6 ceil()**

```
static Element ceil (  
    Element x ) [inline], [static]
```

**16.253.1.7 floor()**

```
static Element floor (  
    Element x ) [inline], [static]
```

**16.253.1.8 round()**

```
static Element round (  
    Element x ) [inline], [static]
```

**16.253.1.9 add()**

```
static Element add (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.10 addin()**

```
static Element addin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.11 sub()**

```
static Element sub (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.12 subin()**

```
static Element subin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.13 mul()**

```
static Element mul (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.14 mulin()**

```
static Element mulin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.15 div()**

```
static Element div (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.16 fmadd()**

```
static Element fmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.17 fmaddin()**

```
static Element fmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.18 fmsub()**

```
static Element fmsub (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.19 fmsubin()**

```
static Element fmsubin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.20 fnmadd()**

```
static Element fnmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.21 fnmaddin()**

```
static Element fnmaddin (
    Element & x1,
    Element x2,
    Element x3 ) [inline], [static]
```

**16.253.1.22 lesser()**

```
static Element lesser (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.23 lesser\_eq()**

```
static Element lesser_eq (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.24 greater()**

```
static Element greater (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.25 greater\_eq()**

```
static Element greater_eq (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.26 eq()**

```
static Element eq (
    Element x1,
    Element x2 ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.254 ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type > Struct Template Reference

### Static Public Member Functions

- static Element [zero](#) ()
- static Element [round](#) (Element x)
- static Element [vand](#) (Element x1, Element x2)
- static Element [vor](#) (Element x1, Element x2)
- static Element [vxor](#) (Element x1, Element x2)
- static Element [vandnot](#) (Element x1, Element x2)
- static Element [add](#) (Element x1, Element x2)
- static Element [addin](#) (Element &x1, Element x2)

- static Element [sub](#) (Element x1, Element x2)
- static Element [subin](#) (Element &x1, Element x2)
- static Element [mul](#) (Element x1, Element x2)
- static Element [mullo](#) (Element x1, Element x2)
- static Element [mulhi](#) (Element x1, Element x2)
- static Element [mulx](#) (Element x1, Element x2)
- static Element [fmadd](#) (Element x1, Element x2, Element x3)
- static Element [fmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fmaddx](#) (Element x1, Element x2, Element x3)
- static Element [fmaddxin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsub](#) (Element x1, Element x2, Element x3)
- static Element [fmsubin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsubx](#) (Element x1, Element x2, Element x3)
- static Element [fmsubxin](#) (Element &x1, Element x2, Element x3)
- static Element [fnmadd](#) (Element x1, Element x2, Element x3)
- static Element [fnmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fnmaddx](#) (Element x1, Element x2, Element x3)
- static Element [fnmaddxin](#) (Element &x1, Element x2, Element x3)
- template<int s, bool EnableTrue = true>  
static enable\_if<lis\_signed< Element >::value &&EnableTrue, Element >::type [sra](#) (Element x1)
- template<int s, bool EnableTrue = true>  
static enable\_if< is\_signed< Element >::value &&EnableTrue, Element >::type [sra](#) (Element x1)
- template<int s>  
static Element [srl](#) (Element x1)
- template<int s>  
static Element [sll](#) (Element x1)
- static Element [lesser](#) (Element x1, Element x2)
- static Element [lesser\\_eq](#) (Element x1, Element x2)
- static Element [greater](#) (Element x1, Element x2)
- static Element [greater\\_eq](#) (Element x1, Element x2)
- static Element [eq](#) (Element x1, Element x2)

## 16.254.1 Member Function Documentation

### 16.254.1.1 zero()

```
static Element zero ( ) [inline], [static]
```

### 16.254.1.2 round()

```
static Element round (  
    Element x ) [inline], [static]
```

### 16.254.1.3 vand()

```
static Element vand (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.4 vor()**

```
static Element vor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.5 vxor()**

```
static Element vxor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.6 vandnot()**

```
static Element vandnot (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.7 add()**

```
static Element add (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.8 addin()**

```
static Element addin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.9 sub()**

```
static Element sub (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.10 subin()**

```
static Element subin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.11 mul()**

```
static Element mul (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.12 mullo()**

```
static Element mullo (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.13 mulhi()**

```
static Element mulhi (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.14 mulx()**

```
static Element mulx (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.15 fmadd()**

```
static Element fmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.16 fmaddin()**

```
static Element fmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.17 fmaddx()**

```
static Element fmaddx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.18 fmaddxin()**

```
static Element fmaddxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.19 fmsub()**

```
static Element fmsub (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.20 fmsubin()**

```
static Element fmsubin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.21 fmsubx()**

```
static Element fmsubx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.22 fmsubxin()**

```
static Element fmsubxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.23 fnmadd()**

```
static Element fnmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.24 fnmaddin()**

```
static Element fnmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.25 fnmaddx()**

```
static Element fnmaddx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.26 fnmaddxin()**

```
static Element fnmaddxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.27 sra() [1/2]**

```
static enable_if<!is_signed<Element>::value && EnableTrue, Element>::type sra (  
    Element x1 ) [inline], [static]
```



**16.254.1.28 sra()** [2/2]

```
static enable_if<is_signed<Element>::value && EnableTrue, Element>::type sra (  
    Element x1 ) [inline], [static]
```

**16.254.1.29 srl()**

```
static Element srl (  
    Element x1 ) [inline], [static]
```

**16.254.1.30 sll()**

```
static Element sll (  
    Element x1 ) [inline], [static]
```

**16.254.1.31 lesser()**

```
static Element lesser (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.32 lesser\_eq()**

```
static Element lesser_eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.33 greater()**

```
static Element greater (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.34 greater\_eq()**

```
static Element greater_eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.35 eq()**

```
static Element eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.255 Sequential Struct Reference

### Public Member Functions

- [Sequential](#) ()
- [Sequential](#) (size\_t nth)
- template<class Cut, class Param >  
[Sequential](#) ([Parallel](#)< Cut, Param > &)
- size\_t [numthreads](#) () const

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Sequential](#) &)

## 16.255.1 Constructor & Destructor Documentation

### 16.255.1.1 Sequential() [1/3]

```
Sequential ( ) [inline]
```

### 16.255.1.2 Sequential() [2/3]

```
Sequential (
    size_t nth ) [inline]
```

### 16.255.1.3 Sequential() [3/3]

```
Sequential (
    Parallel< Cut, Param > & ) [inline]
```

## 16.255.2 Member Function Documentation

### 16.255.2.1 numthreads()

```
size_t numthreads ( ) const [inline]
```

## 16.255.3 Friends And Related Function Documentation

### 16.255.3.1 operator<<

```
std::ostream& operator<< (
    std::ostream & out,
    const Sequential & ) [friend]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

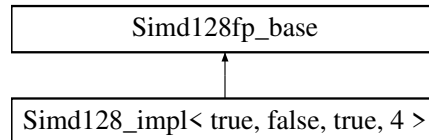
## 16.256 Simd128\_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

## 16.257 Simd128\_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd128\_impl< true, false, true, 4 >:



### Static Public Member Functions

- static const std::string [type\\_string](#) ()

### 16.257.1 Member Function Documentation

#### 16.257.1.1 type\_string()

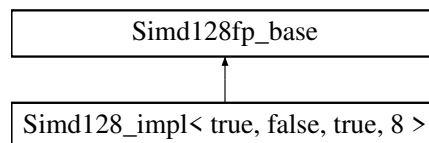
```
static const std::string type_string ( ) [inline], [static], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_float.inl](#)

## 16.258 Simd128\_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd128\_impl< true, false, true, 8 >:



### Static Public Member Functions

- static const std::string [type\\_string](#) ()

### 16.258.1 Member Function Documentation

#### 16.258.1.1 type\_string()

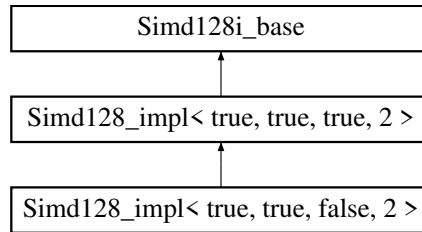
```
static const std::string type_string ( ) [inline], [static], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_double.inl](#)

## 16.259 Simd128\_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, false, 2 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [scalar\\_t](#) = [uint16\\_t](#)
- using [vect\\_t](#) = [\\_\\_m128i](#)

### Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)

- `template<uint32_t s>`  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- `template<uint8_t s>`  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `__m64` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- `template<uint8_t s>`  
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- `template<uint8_t s>`  
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 16

## 16.259.1 Member Typedef Documentation

### 16.259.1.1 scalar\_t

```
using scalar_t = uint16_t
```

### 16.259.1.2 vect\_t

```
using vect_t = __m128i [inherited]
```

## 16.259.2 Member Function Documentation

#### 16.259.2.1 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

#### 16.259.2.2 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

#### 16.259.2.3 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.259.2.4 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.259.2.5 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.259.2.6 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.259.2.7 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.259.2.8 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.259.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.259.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.259.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.259.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.14 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.17 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.259.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.22 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.259.2.23 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

#### 16.259.2.24 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

#### 16.259.2.25 sll()

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```



**16.259.2.26 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.27 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.28 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.29 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.30 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.31 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.32 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.33 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.34 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.35 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.36 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.37 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.38 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.39 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.40 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.41 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.42 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.43 eq()**

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.44 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.45 mod()**

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const __m64 & INV_P,
    const vect_t & NEG_P,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]
```

**16.259.2.46 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.259.2.47 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.259.2.48 sll128()**

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.49 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.50 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.51 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.52 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.53 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.3 Field Documentation****16.259.3.1 vect\_size**

```
constexpr const size_t vect_size = 8 [static], [constexpr], [inherited]
```

**16.259.3.2 alignment**

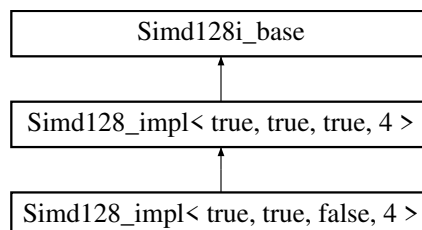
```
constexpr const size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int16.inl](#)

**16.260 Simd128\_impl< true, true, false, 4 > Struct Reference**

Inheritance diagram for Simd128\_impl< true, true, false, 4 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [scalar\\_t](#) = uint32\_t
- using [vect\\_t](#) = \_\_m128i

## Static Public Member Functions

- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3)
- template<class T >
  - static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>
  - static `INLINE CONST vect_t sra` (const `vect_t` a)
- static `INLINE CONST vect_t greater` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t lesser` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >
  - static constexpr bool `valid` (T \*p)
- template<class T >
  - static constexpr bool `compliant` (T n)
- template<int s>
  - static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
  - static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8\_t s>
  - static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>
  - static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)

- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- template<uint8\_t s>  
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 16

## 16.260.1 Member Typedef Documentation

### 16.260.1.1 scalar\_t

```
using scalar_t = uint32_t
```

### 16.260.1.2 vect\_t

```
using vect_t = __m128i [inherited]
```

## 16.260.2 Member Function Documentation

### 16.260.2.1 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.260.2.2 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

### 16.260.2.3 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.260.2.4 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

### 16.260.2.5 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

### 16.260.2.6 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

### 16.260.2.7 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

### 16.260.2.8 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

### 16.260.2.9 sra()

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

### 16.260.2.10 greater()

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

### 16.260.2.11 lesser()

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

### 16.260.2.12 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

### 16.260.2.13 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.14 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.17 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.18 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.19 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.20 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.21 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```



**16.260.2.22 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.260.2.23 valid()**

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.260.2.24 compliant()**

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.260.2.25 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.26 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.27 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.28 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.29 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.30 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.31 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.32 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.33 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.34 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.35 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.36 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.37 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.38 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.39 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.40 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.41 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.42 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.43 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.44 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.45 mod()**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

#### 16.260.2.46 type\_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

#### 16.260.2.47 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

#### 16.260.2.48 sll128()

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.260.2.49 srl128()

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.260.2.50 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.260.2.51 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.260.2.52 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.260.2.53 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

### 16.260.3 Field Documentation

#### 16.260.3.1 vect\_size

```
constexpr const size_t vect_size = 4 [static], [constexpr], [inherited]
```

### 16.260.3.2 alignment

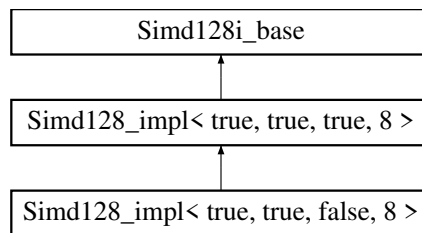
```
constexpr const size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int32.inl](#)

## 16.261 Simd128\_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, false, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [scalar\\_t](#) = [uint64\\_t](#)
- using [vect\\_t](#) = [\\_\\_m128i](#)

### Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) (const [vect\\_t](#) x0, const [vect\\_t](#) x1)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubxin](#) ([vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >  
static constexpr bool [valid](#) (T \*p)

- `template<class T >`  
`static constexpr bool compliant (T n)`
- `template<int idx>`  
`static INLINE CONST scalar_t get (vect_t v)`
- `template<int s>`  
`static INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`  
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE CONST vect_t mask_high ()`
- `static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- `static INLINE vect_t mod (vect_t &C, const __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)`
- `static const std::string type_string ()`
- `static INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
`static INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t srl128 (const vect_t a)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- `static constexpr const size_t vect_size = 2`
- `static constexpr const size_t alignment = 16`

## Static Protected Member Functions

- `static INLINE CONST vect_t signbits (const vect_t x)`

## 16.261.1 Member Typedef Documentation

### 16.261.1.1 scalar\_t

```
using scalar_t = uint64_t
```

### 16.261.1.2 vect\_t

```
using vect_t = __m128i [inherited]
```

## 16.261.2 Member Function Documentation

### 16.261.2.1 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.261.2.2 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1 ) [inline], [static]
```

### 16.261.2.3 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.261.2.4 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

### 16.261.2.5 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

### 16.261.2.6 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

### 16.261.2.7 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.261.2.8 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.261.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.261.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.261.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.261.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.14 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t x0,  
    const vect_t x1 ) [inline], [static]
```

**16.261.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,
```



```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.261.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.261.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.261.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.261.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.261.2.21 fmsubxin() [1/2]

```
static INLINE CONST vect_t fmsubxin (  
    vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.261.2.22 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.261.2.23 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.261.2.24 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.261.2.25 get()**

```
static INLINE CONST scalar_t get (
    vect_t v ) [inline], [static], [inherited]
```

**16.261.2.26 sll()**

```
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.27 srl()**

```
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.28 shuffle()**

```
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.29 unpacklo()**

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.30 unpackhi()**

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.31 blend()**

```
static INLINE CONST vect_t blend (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.32 add()**

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.33 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.34 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.35 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.36 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.37 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.38 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.39 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.40 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.41 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.42 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.43 fmsubxin() [2/2]**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.44 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.45 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.46 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.261.2.47 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (  
    vect_t x,  
    vect_t y ) [static], [inherited]
```

**16.261.2.48 mod()**

```
INLINE vect_t mod (  
    vect_t & C,  
    const __m128d & P,  
    const __m128d & INVP,  
    const __m128d & NEGP,  
    const vect_t & POW50REM,  
    const __m128d & MIN,  
    const __m128d & MAX,
```

```
__m128d & Q,  
__m128d & T ) [static], [inherited]
```

#### 16.261.2.49 signbits()

```
static INLINE CONST vect_t signbits (  
    const vect_t x ) [inline], [static], [protected], [inherited]
```

#### 16.261.2.50 type\_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

#### 16.261.2.51 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

#### 16.261.2.52 sll128()

```
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.261.2.53 srl128()

```
static INLINE CONST vect_t srl128 (  
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.261.2.54 vand()

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.261.2.55 vor()

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.261.2.56 vxor()

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.261.2.57 vandnot()

```
static INLINE CONST vect_t vandnot (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

### 16.261.3 Field Documentation

#### 16.261.3.1 vect\_size

```
constexpr const size_t vect_size = 2 [static], [constexpr], [inherited]
```

#### 16.261.3.2 alignment

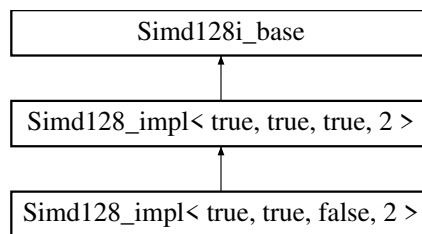
```
constexpr const size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int64.inl](#)

## 16.262 Simd128\_impl< true, true, true, 2 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, true, 2 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int16\_t

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)

- `template<int s>`  
`static INLINE CONST vect_t sra (const vect_t a)`
- `template<uint32_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmadxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE vect_t mod (vect_t &C, const vect_t &P, const __m64 &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- `static const std::string type_string ()`
- `static INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
`static INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t srl128 (const vect_t a)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- `static constexpr const size_t vect_size = 8`
- `static constexpr const size_t alignment = 16`

### 16.262.1 Member Typedef Documentation

### 16.262.1.1 vect\_t

```
using vect_t = __m128i
```

### 16.262.1.2 scalar\_t

```
using scalar_t = int16_t
```

## 16.262.2 Member Function Documentation

### 16.262.2.1 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

### 16.262.2.2 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

### 16.262.2.3 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.262.2.4 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

### 16.262.2.5 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.262.2.6 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```



**16.262.2.7 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.262.2.8 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.262.2.9 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.262.2.10 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.262.2.11 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.262.2.12 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.262.2.13 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.262.2.14 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.262.2.15 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.16 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.17 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.18 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.19 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.20 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.21 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.22 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.23 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.24 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.25 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.26 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.27 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.28 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.29 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.30 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.31 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.32 fmmaddx()**

```
static INLINE CONST vect_t fmmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.33 fmmaddxin()**

```
static INLINE vect_t fmmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.34 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.35 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.36 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.37 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.38 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.39 greater()**

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.40 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.41 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.42 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.43 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.262.2.44 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.262.2.45 mod()**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const __m64 & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static]
```

**16.262.2.46 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.262.2.47 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.262.2.48 sll128()**

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.262.2.49 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.262.2.50 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.262.2.51 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.262.2.52 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.262.2.53 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.262.3 Field Documentation****16.262.3.1 vect\_size**

```
constexpr const size_t vect_size = 8 [static], [constexpr]
```

**16.262.3.2 alignment**

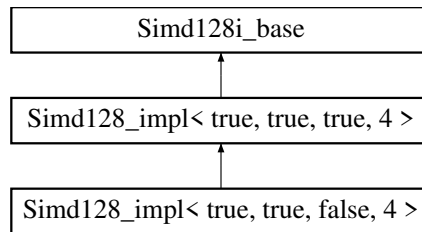
```
constexpr const size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int16.inl](#)

## 16.263 Simd128\_impl< true, true, true, 4 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, true, 4 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int32\_t

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- template<uint8\_t s>  
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 16

## 16.263.1 Member Typedef Documentation

### 16.263.1.1 `vect_t`

```
using vect_t = __m128i
```

### 16.263.1.2 `scalar_t`

```
using scalar_t = int32_t
```

## 16.263.2 Member Function Documentation



**16.263.2.1 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.263.2.2 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.263.2.3 set1()**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.263.2.4 set()**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

**16.263.2.5 gather()**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

**16.263.2.6 load()**

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

**16.263.2.7 loadu()**

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

**16.263.2.8 store()**

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

**16.263.2.9 storeu()**

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

#### 16.263.2.10 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.263.2.11 sll()

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

#### 16.263.2.12 srl()

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

#### 16.263.2.13 sra()

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

#### 16.263.2.14 shuffle()

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

#### 16.263.2.15 unpacklo()

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.16 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.17 blend()

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.18 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.19 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.20 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.21 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.22 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.23 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.24 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.25 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.26 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.27 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.28 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.29 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.30 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.31 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.32 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.33 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.34 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.263.2.35 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.36 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.38 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.39 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.40 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.263.2.41 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.42 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.43 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.263.2.44 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.263.2.45 mod()**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static]
```

**16.263.2.46 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.263.2.47 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.263.2.48 sll128()**

```
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.263.2.49 srl128()**

```
static INLINE CONST vect_t srl128 (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.263.2.50 vand()**

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.263.2.51 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.263.2.52 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.263.2.53 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.263.3 Field Documentation****16.263.3.1 vect\_size**

```
constexpr const size_t vect_size = 4 [static], [constexpr]
```

**16.263.3.2 alignment**

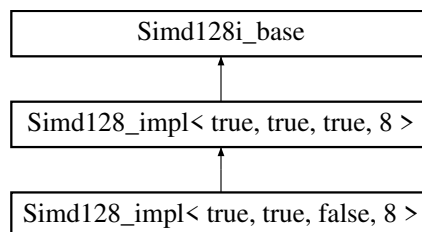
```
constexpr const size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int32.inl](#)

**16.264 Simd128\_impl< true, true, true, 8 > Struct Reference**

Inheritance diagram for Simd128\_impl< true, true, true, 8 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int64\_t

## Static Public Member Functions

- `template<class T >`  
`static constexpr bool valid (T *p)`
- `template<class T >`  
`static constexpr bool compliant (T n)`
- `static INLINE CONST vect_t set1 (const scalar_t x)`
- `static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1)`
- `template<class T >`  
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `template<int idx>`  
`static INLINE CONST scalar_t get (vect_t v)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`  
`static INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`  
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<int s>`  
`static INLINE CONST vect_t sra (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t x0, const vect_t x1)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE CONST vect_t mask_high ()`



- static `INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- static `INLINE vect_t mod (vect_t &C, const __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)`
- static const std::string `type_string ()`
- static `INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
static `INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
static `INLINE CONST vect_t srl128 (const vect_t a)`
- static `INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 2
- static constexpr const size\_t `alignment` = 16

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits (const vect_t x)`

## 16.264.1 Member Typedef Documentation

### 16.264.1.1 vect\_t

```
using vect_t = __m128i
```

### 16.264.1.2 scalar\_t

```
using scalar_t = int64_t
```

## 16.264.2 Member Function Documentation

### 16.264.2.1 valid()

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.264.2.2 compliant()

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.264.2.3 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

#### 16.264.2.4 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1 ) [inline], [static]
```

#### 16.264.2.5 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.264.2.6 get()

```
static INLINE CONST scalar_t get (  
    vect_t v ) [inline], [static]
```

#### 16.264.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.264.2.8 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.264.2.9 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.264.2.10 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.264.2.11 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.264.2.12 sll()

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.264.2.13 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.264.2.14 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.264.2.15 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.264.2.16 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.17 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.18 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.19 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.20 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.21 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.22 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.23 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t x0,  
    const vect_t x1 ) [inline], [static]
```

**16.264.2.24 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.25 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.26 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.27 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.28 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.29 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.30 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.31 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.32 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.33 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.34 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.35 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.36 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.37 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.264.2.38 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.39 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.40 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.41 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.42 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.43 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.264.2.44 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

#### 16.264.2.45 mask\_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.264.2.46 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]
```

**16.264.2.47 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m128d & P,
    const __m128d & INVP,
    const __m128d & NEGP,
    const vect_t & POW50REM,
    const __m128d & MIN,
    const __m128d & MAX,
    __m128d & Q,
    __m128d & T ) [static]
```

**16.264.2.48 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]
```

**16.264.2.49 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.264.2.50 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.264.2.51 sll128()**

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.264.2.52 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.264.2.53 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.264.2.54 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.264.2.55 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.264.2.56 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.264.3 Field Documentation****16.264.3.1 vect\_size**

```
constexpr const size_t vect_size = 2 [static], [constexpr]
```

**16.264.3.2 alignment**

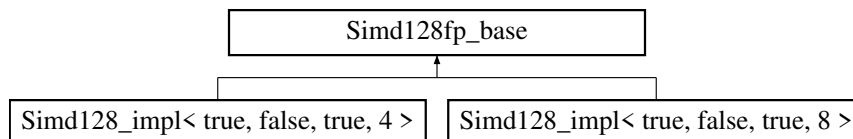
```
constexpr const size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int64.inl](#)

**16.265 Simd128fp\_base Struct Reference**

Inheritance diagram for Simd128fp\_base:

**Static Public Member Functions**

- static const std::string [type\\_string](#) ()

**16.265.1 Member Function Documentation****16.265.1.1 type\_string()**

```
static const std::string type_string ( ) [inline], [static]
```

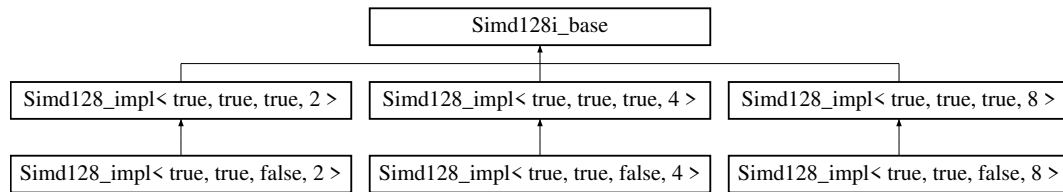
The documentation for this struct was generated from the following file:

- [simd128.inl](#)



## 16.266 Simd128i\_base Struct Reference

Inheritance diagram for Simd128i\_base:



### Public Types

- using `vect_t` = `__m128i`

### Static Public Member Functions

- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- template<uint8\_t s>  
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

### 16.266.1 Member Typedef Documentation

#### 16.266.1.1 vect\_t

using `vect_t` = `__m128i`

### 16.266.2 Member Function Documentation

#### 16.266.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

#### 16.266.2.2 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

#### 16.266.2.3 sll128()

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static]
```

**16.266.2.4 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static]
```

**16.266.2.5 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.266.2.6 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.266.2.7 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.266.2.8 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

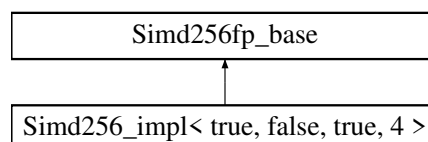
**16.267 Simd256\_impl< ArithType, Int, Signed, Size > Struct Template Reference**

The documentation for this struct was generated from the following file:

- [simd256.inl](#)

**16.268 Simd256\_impl< true, false, true, 4 > Struct Reference**

Inheritance diagram for Simd256\_impl< true, false, true, 4 >:

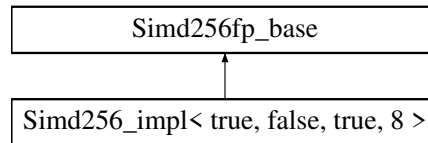


The documentation for this struct was generated from the following file:

- [simd256\\_float.inl](#)

## 16.269 Simd256\_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd256\_impl< true, false, true, 8 >:



### Public Types

- using `vect_t` = `__m256d`
- using `scalar_t` = `double`

### Static Public Member Functions

- `template<class T >`  
static constexpr bool `valid` (T \*p)
- `template<class T >`  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4)
- `template<class T >`  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE void storeu` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE void stream` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- `template<uint8_t s>`  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t blendv` (const `vect_t` a, const `vect_t` b, const `vect_t` mask)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t div` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)

- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t floor` (const `vect_t` a)
- static `INLINE CONST vect_t ceil` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t hadd` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 32

## 16.269.1 Member Typedef Documentation

### 16.269.1.1 `vect_t`

```
using vect_t = __m256d
```

### 16.269.1.2 `scalar_t`

```
using scalar_t = double
```

## 16.269.2 Member Function Documentation

### 16.269.2.1 `valid()`

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.269.2.2 `compliant()`

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.269.2.3 `zero()`

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

### 16.269.2.4 `set1()`

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

#### 16.269.2.5 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4 ) [inline], [static]
```

#### 16.269.2.6 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.269.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.269.2.8 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.269.2.9 store()

```
static INLINE void store (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.269.2.10 storeu()

```
static INLINE void storeu (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.269.2.11 stream()

```
static INLINE void stream (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.269.2.12 unpacklo\_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.13 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.14 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.15 blendv()**

```
static INLINE CONST vect_t blendv (  
    const vect_t a,  
    const vect_t b,  
    const vect_t mask ) [inline], [static]
```

**16.269.2.16 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.17 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.18 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.19 subin()**

```
static INLINE CONST vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.20 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.21 mulin()**

```
static INLINE CONST vect_t mulin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.22 div()**

```
static INLINE CONST vect_t div (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.23 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.24 fmaddin()**

```
static INLINE CONST vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.25 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.26 fnmaddin()**

```
static INLINE CONST vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.27 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.28 fmsubin()**

```
static INLINE CONST vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.29 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.30 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.31 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.32 greater()**

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.33 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.34 vand()**

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.35 vor()**

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.36 vxor()**

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```



**16.269.2.37 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.269.2.38 floor()**

```
static INLINE CONST vect_t floor (
    const vect_t a ) [inline], [static]
```

**16.269.2.39 ceil()**

```
static INLINE CONST vect_t ceil (
    const vect_t a ) [inline], [static]
```

**16.269.2.40 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

**16.269.2.41 hadd()**

```
static INLINE CONST vect_t hadd (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.269.2.42 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.269.2.43 mod()**

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

**16.269.3 Field Documentation****16.269.3.1 vect\_size**

```
constexpr const size_t vect_size = 4 [static], [constexpr]
```

### 16.269.3.2 alignment

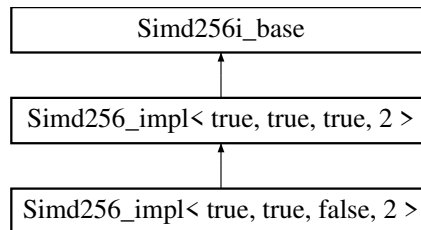
```
constexpr const size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd256\\_double.inl](#)

## 16.270 Simd256\_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, false, 2 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [scalar\\_t](#) = [uint16\\_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >
- using [vect\\_t](#) = [\\_\\_m256i](#)
- using [half\\_t](#) = [\\_\\_m128i](#)

### Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7, const [scalar\\_t](#) x8, const [scalar\\_t](#) x9, const [scalar\\_t](#) x10, const [scalar\\_t](#) x11, const [scalar\\_t](#) x12, const [scalar\\_t](#) x13, const [scalar\\_t](#) x14, const [scalar\\_t](#) x15)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint64\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &s1, `vect_t` &s2, const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 16
- static constexpr const size\_t `alignment` = 32

## 16.270.1 Member Typedef Documentation

### 16.270.1.1 scalar\_t

```
using scalar_t = uint16_t
```

### 16.270.1.2 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

### 16.270.1.3 vect\_t

```
using vect_t = __m256i [inherited]
```

### 16.270.1.4 half\_t

```
using half_t = __m128i [inherited]
```

## 16.270.2 Member Function Documentation

### 16.270.2.1 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.270.2.2 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7,  
    const scalar_t x8,  
    const scalar_t x9,  
    const scalar_t x10,  
    const scalar_t x11,  
    const scalar_t x12,  
    const scalar_t x13,  
    const scalar_t x14,  
    const scalar_t x15 ) [inline], [static]
```

### 16.270.2.3 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.270.2.4 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

### 16.270.2.5 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.270.2.6 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.270.2.7 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.270.2.8 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.270.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.270.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.270.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.270.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.14 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.270.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.17 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.18 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.19 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.20 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.21 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.22 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.270.2.23 valid()**

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.270.2.24 compliant()**

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.270.2.25 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.270.2.26 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.270.2.27 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.270.2.28 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.29 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.30 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.31 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.32 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.33 blend\_twice()**

```
static INLINE CONST vect_t blend_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.34 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.35 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.36 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.37 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.38 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```



**16.270.2.39 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.40 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.41 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.42 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.43 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.44 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.45 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.46 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

### 16.270.2.47 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

### 16.270.2.48 mod()

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]
```

### 16.270.2.49 type\_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

### 16.270.2.50 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

## 16.270.3 Field Documentation

### 16.270.3.1 vect\_size

```
constexpr const size_t vect_size = 16 [static], [constexpr], [inherited]
```

### 16.270.3.2 alignment

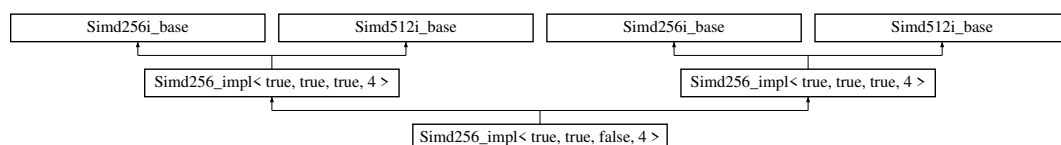
```
constexpr const size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int16.inl](#)

## 16.271 Simd256\_impl< true, true, false, 4 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, false, 4 >:



## Data Structures

- union [Converter](#)

## Public Types

- using `scalar_t` = `uint32_t`
- using `simdHalf` = `Simd128< scalar_t >`
- using `scalar_t` = `uint32_t`
- using `simdHalf` = `Simd128< scalar_t >`
- using `vect_t` = `__m256i`
- using `vect_t` = `__m512i`
- using `half_t` = `__m128i`
- using `half_t` = `__m256i`

## Static Public Member Functions

- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- static `INLINE CONST vect_t greater` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t lesser` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- static `INLINE CONST vect_t greater` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t lesser` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)

- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- template<class T >  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8, const `scalar_t` x9, const `scalar_t` x10, const `scalar_t` x11, const `scalar_t` x12, const `scalar_t` x13, const `scalar_t` x14, const `scalar_t` x15)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint32\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- template<uint64\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &s1, `vect_t` &s2, const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)

- static `INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- static `INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- static const std::string `type_string ()`
- static const std::string `type_string ()`
- static `INLINE CONST vect_t zero ()`
- static `INLINE CONST vect_t zero ()`
- static `INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 32

## 16.271.1 Member Typedef Documentation

### 16.271.1.1 scalar\_t [1/2]

```
using scalar_t = uint32_t
```

### 16.271.1.2 simdHalf [1/2]

```
using simdHalf = Simd128<scalar_t>
```

### 16.271.1.3 scalar\_t [2/2]

```
using scalar_t = uint32_t
```

### 16.271.1.4 simdHalf [2/2]

```
using simdHalf = Simd128<scalar_t>
```

**16.271.1.5 vect\_t [1/2]**

```
using vect_t = __m256i [inherited]
```

**16.271.1.6 vect\_t [2/2]**

```
using vect_t = __m512i [inherited]
```

**16.271.1.7 half\_t [1/2]**

```
using half_t = __m128i [inherited]
```

**16.271.1.8 half\_t [2/2]**

```
using half_t = __m256i [inherited]
```

**16.271.2 Member Function Documentation****16.271.2.1 set1() [1/2]**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.271.2.2 set() [1/3]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

**16.271.2.3 gather() [1/2]**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

**16.271.2.4 load() [1/2]**

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

**16.271.2.5 loadu() [1/2]**

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

**16.271.2.6 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.271.2.7 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.271.2.8 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.271.2.9 sra()** [1/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.271.2.10 greater()** [1/2]

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.11 lesser()** [1/2]

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.12 greater\_eq()** [1/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.13 lesser\_eq()** [1/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.14 mulhi()** [1/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.15 mulx() [1/2]**

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.16 fmaddx() [1/2]**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.17 fmaddxin() [1/2]**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.18 fnmaddx() [1/2]**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.19 fnmaddxin() [1/2]**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.20 fmsubx() [1/2]**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.21 fmsubxin() [1/2]**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```



**16.271.2.22 hadd\_to\_scal()** [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.271.2.23 set1()** [2/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

**16.271.2.24 set()** [2/3]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

**16.271.2.25 gather()** [2/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.271.2.26 load()** [2/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.271.2.27 loadu()** [2/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.271.2.28 store()** [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.271.2.29 storeu()** [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.271.2.30 stream()** [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.271.2.31 sra()** [2/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.271.2.32 greater()** [2/2]

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.33 lesser()** [2/2]

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.34 greater\_eq()** [2/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.35 lesser\_eq()** [2/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.36 mulhi()** [2/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.37 mulx()** [2/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.38 fmaddx()** [2/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.271.2.39 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.271.2.40 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.271.2.41 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.271.2.42 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.271.2.43 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.271.2.44 hadd\_to\_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.271.2.45 valid() [1/2]

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.271.2.46 valid()** [2/2]

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.271.2.47 compliant()** [1/2]

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.271.2.48 compliant()** [2/2]

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.271.2.49 set()** [3/3]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7,  
    const scalar_t x8,  
    const scalar_t x9,  
    const scalar_t x10,  
    const scalar_t x11,  
    const scalar_t x12,  
    const scalar_t x13,  
    const scalar_t x14,  
    const scalar_t x15 ) [inline], [static], [inherited]
```

**16.271.2.50 sll()** [1/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.51 sll()** [2/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.52 srl()** [1/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.53 srl()** [2/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.54 shuffle\_twice()** [1/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.55 shuffle\_twice()** [2/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.56 shuffle()** [1/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.57 shuffle()** [2/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.58 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.59 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.60 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.61 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.62 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.63 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.64 add() [1/2]**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.65 add() [2/2]**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.66 addin() [1/2]**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.67 addin() [2/2]**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.68 sub() [1/2]**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.69 sub() [2/2]**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.70 subin() [1/2]**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.71 subin() [2/2]**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.72 mullo() [1/2]**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.73 mullo() [2/2]**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.74 mul() [1/2]**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.75 mul() [2/2]**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.76 fmadd() [1/2]**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.77 fmadd() [2/2]**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.78 fmaddin() [1/2]**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.79 fmaddin() [2/2]**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.80 fnmadd() [1/2]**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.81 fnmadd() [2/2]**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.82 fnmaddin() [1/2]**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.83 fnmaddin() [2/2]**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.84 fmsub() [1/2]**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.85 fmsub() [2/2]**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,
```



```
const vect_t a,  
const vect_t b ) [inline], [static], [inherited]
```

#### 16.271.2.86 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.271.2.87 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.271.2.88 eq() [1/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.271.2.89 eq() [2/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.271.2.90 round() [1/2]

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.271.2.91 round() [2/2]

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.271.2.92 mod() [1/2]

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

**16.271.2.93 mod()** [2/2]

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]
```

**16.271.2.94 type\_string()** [1/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.271.2.95 type\_string()** [2/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.271.2.96 zero()** [1/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.271.2.97 zero()** [2/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.271.2.98 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.99 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.100 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.101 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

### 16.271.3 Field Documentation

#### 16.271.3.1 vect\_size

```
static constexpr const size_t vect_size = 8 [static], [constexpr], [inherited]
```

#### 16.271.3.2 alignment

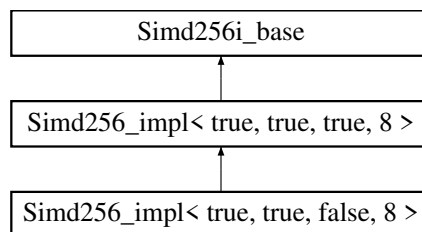
```
static constexpr const size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.272 Simd256\_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, false, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [scalar\\_t](#) = [uint64\\_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >
- using [vect\\_t](#) = [\\_\\_m256i](#)
- using [half\\_t](#) = [\\_\\_m128i](#)

### Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- template<int idx>  
static `INLINE CONST scalar_t get` (`vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &l, `vect_t` &h, const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const \_\_m256d &P, const \_\_m256d &INVP, const \_\_m256d &NEGP, const `vect_t` &POW50REM, const \_\_m256d &MIN, const \_\_m256d &MAX, \_\_m256d &Q, \_\_m256d &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 32

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

### 16.272.1 Member Typedef Documentation

#### 16.272.1.1 scalar\_t

```
using scalar_t = uint64_t
```

#### 16.272.1.2 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

#### 16.272.1.3 vect\_t

```
using vect_t = __m256i [inherited]
```

#### 16.272.1.4 half\_t

```
using half_t = __m128i [inherited]
```

### 16.272.2 Member Function Documentation

#### 16.272.2.1 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

#### 16.272.2.2 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

#### 16.272.2.3 gather()

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

#### 16.272.2.4 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

**16.272.2.5 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.272.2.6 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.272.2.7 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.272.2.8 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.272.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.272.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.272.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.272.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.14 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.272.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.17 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.18 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.19 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.20 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.21 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.22 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.272.2.23 valid()**

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.272.2.24 compliant()**

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.272.2.25 get()**

```
static INLINE CONST scalar_t get (  
    vect_t v ) [inline], [static], [inherited]
```

**16.272.2.26 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.272.2.27 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.272.2.28 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.272.2.29 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.30 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```



**16.272.2.31 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.32 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.33 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.34 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.35 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.36 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.37 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.38 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.39 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.40 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.41 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.42 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.43 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.44 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.45 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.46 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.47 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

**16.272.2.48 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.272.2.49 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

**16.272.2.50 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m256d & P,
    const __m256d & INV_P,
    const __m256d & NEG_P,
    const vect_t & POW50REM,
    const __m256d & MIN,
    const __m256d & MAX,
    __m256d & Q,
    __m256d & T ) [static], [inherited]
```

**16.272.2.51 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected], [inherited]
```

**16.272.2.52 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.272.2.53 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.272.3 Field Documentation****16.272.3.1 vect\_size**

```
constexpr const size_t vect_size = 4 [static], [constexpr], [inherited]
```

### 16.272.3.2 alignment

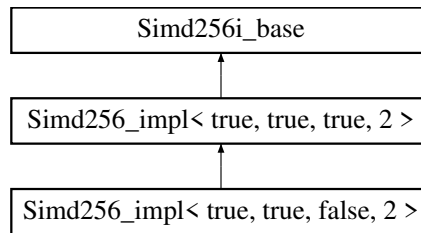
```
constexpr const size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int64.inl](#)

## 16.273 Simd256\_impl< true, true, true, 2 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, true, 2 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m256i
- using [half\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = [int16\\_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST](#) [vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST](#) [vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7, const [scalar\\_t](#) x8, const [scalar\\_t](#) x9, const [scalar\\_t](#) x10, const [scalar\\_t](#) x11, const [scalar\\_t](#) x12, const [scalar\\_t](#) x13, const [scalar\\_t](#) x14, const [scalar\\_t](#) x15)
- template<class T >  
static [INLINE PURE](#) [vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE](#) [vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE](#) [vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST](#) [vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST](#) [vect\\_t srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST](#) [vect\\_t sra](#) (const [vect\\_t](#) a)
- template<uint64\_t s>  
static [INLINE CONST](#) [vect\\_t shuffle](#) (const [vect\\_t](#) a)

- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &s1, `vect_t` &s2, const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>
  - static `INLINE CONST vect_t blend_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmadxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 16
- static constexpr const size\_t `alignment` = 32

## 16.273.1 Member Typedef Documentation

### 16.273.1.1 vect\_t

```
using vect_t = __m256i
```

### 16.273.1.2 half\_t

```
using half_t = __m128i
```

**16.273.1.3 scalar\_t**

```
using scalar_t = int16_t
```

**16.273.1.4 simdHalf**

```
using simdHalf = Simd128<scalar_t>
```

**16.273.2 Member Function Documentation****16.273.2.1 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.273.2.2 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.273.2.3 set1()**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.273.2.4 set()**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static]
```

**16.273.2.5 gather()**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

**16.273.2.6 load()**

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.273.2.7 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.273.2.8 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.273.2.9 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.273.2.10 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.273.2.11 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.273.2.12 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.273.2.13 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.273.2.14 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.273.2.15 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.16 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.17 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.18 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.19 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.20 blend\_twice()**

```
static INLINE CONST vect_t blend_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.21 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.22 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```



**16.273.2.23 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.24 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.25 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.26 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.27 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.28 mulx()**

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.273.2.29 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.30 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.31 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.32 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.33 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.34 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.35 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.36 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.37 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.38 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.273.2.39 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.40 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.41 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.42 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.43 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.44 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.45 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.46 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.273.2.47 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

**16.273.2.48 mod()**

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

**16.273.2.49 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.273.2.50 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.273.3 Field Documentation****16.273.3.1 vect\_size**

```
constexpr const size_t vect_size = 16 [static], [constexpr]
```

**16.273.3.2 alignment**

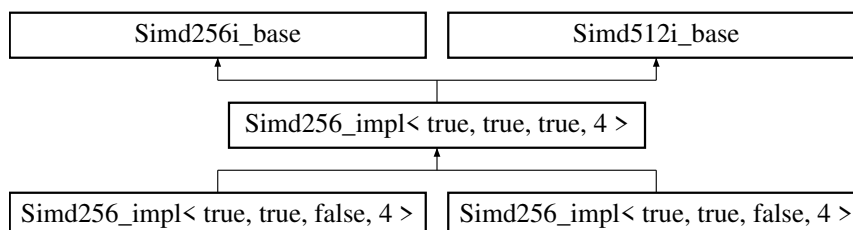
```
constexpr const size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int16.inl](#)

**16.274 Simd256\_impl< true, true, true, 4 > Struct Reference**

Inheritance diagram for Simd256\_impl< true, true, true, 4 >:

**Data Structures**

- union [Converter](#)

## Public Types

- using `vect_t` = `__m256i`
- using `half_t` = `__m128i`
- using `scalar_t` = `int32_t`
- using `simdHalf` = `Simd128`< `scalar_t` >
- using `vect_t` = `__m512i`
- using `half_t` = `__m256i`
- using `scalar_t` = `int32_t`
- using `simdHalf` = `Simd256`< `scalar_t` >

## Static Public Member Functions

- `template<class T >`  
static constexpr bool `valid` (T \*p)
- `template<class T >`  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- `template<class T >`  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- `template<int s>`  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- `template<int s>`  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- `template<int s>`  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- `template<uint8_t s>`  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- `template<uint32_t s>`  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &s1, `vect_t` &s2, const `vect_t` a, const `vect_t` b)
- `template<uint8_t s>`  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)

- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8, const `scalar_t` x9, const `scalar_t` x10, const `scalar_t` x11, const `scalar_t` x12, const `scalar_t` x13, const `scalar_t` x14, const `scalar_t` x15)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint64\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)

- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 32

## 16.274.1 Member Typedef Documentation

### 16.274.1.1 `vect_t` [1/2]

```
using vect_t = __m256i
```

### 16.274.1.2 `half_t` [1/2]

```
using half_t = __m128i
```

### 16.274.1.3 `scalar_t` [1/2]

```
using scalar_t = int32_t
```

### 16.274.1.4 `simdHalf` [1/2]

```
using simdHalf = Simd128<scalar_t>
```

**16.274.1.5 vect\_t [2/2]**

```
using vect_t = __m512i
```

**16.274.1.6 half\_t [2/2]**

```
using half_t = __m256i
```

**16.274.1.7 scalar\_t [2/2]**

```
using scalar_t = int32_t
```

**16.274.1.8 simdHalf [2/2]**

```
using simdHalf = Simd256<scalar_t>
```

**16.274.2 Member Function Documentation****16.274.2.1 valid() [1/2]**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.274.2.2 compliant() [1/2]**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.274.2.3 set1() [1/2]**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.274.2.4 set() [1/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

**16.274.2.5 gather() [1/2]**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```



**16.274.2.6 load()** [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.274.2.7 loadu()** [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.274.2.8 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.274.2.9 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.274.2.10 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.274.2.11 sll()** [1/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.274.2.12 srl()** [1/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.274.2.13 sra()** [1/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.274.2.14 shuffle\_twice()** [1/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static]
```

**16.274.2.15 shuffle()** [1/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.274.2.16 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.17 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.18 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.19 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.20 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.21 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.22 add()** [1/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.23 addin()** [1/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.24 sub()** [1/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.25 subin()** [1/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.26 mullo()** [1/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.27 mul()** [1/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.28 mulhi()** [1/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.29 mulx()** [1/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.274.2.30 fmadd()** [1/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.31 fmaddin() [1/2]**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.32 fmaddx() [1/2]**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.33 fmaddxin() [1/2]**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.34 fnmadd() [1/2]**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.35 fnmaddin() [1/2]**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.36 fnmaddx() [1/2]**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.37 fnmaddxin() [1/2]**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.38 fmsub() [1/2]**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.274.2.39 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.40 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.41 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.42 eq() [1/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.43 greater() [1/2]

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.44 lesser() [1/2]

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.45 greater\_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.46 lesser\_eq()** [1/2]

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.274.2.47 hadd\_to\_scal()** [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.274.2.48 round()** [1/2]

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

**16.274.2.49 mod()** [1/2]

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

**16.274.2.50 valid()** [2/2]

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.274.2.51 compliant()** [2/2]

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.274.2.52 set1()** [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.274.2.53 set()** [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
```

```
const scalar_t x7,  
const scalar_t x8,  
const scalar_t x9,  
const scalar_t x10,  
const scalar_t x11,  
const scalar_t x12,  
const scalar_t x13,  
const scalar_t x14,  
const scalar_t x15 ) [inline], [static]
```

#### 16.274.2.54 gather() [2/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.274.2.55 load() [2/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.274.2.56 loadu() [2/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.274.2.57 store() [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.274.2.58 storeu() [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.274.2.59 stream() [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.274.2.60 sll() [2/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.274.2.61 srl()** [2/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.274.2.62 sra()** [2/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.274.2.63 shuffle\_twice()** [2/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static]
```

**16.274.2.64 shuffle()** [2/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.274.2.65 add()** [2/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.66 addin()** [2/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.67 sub()** [2/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.68 subin()** [2/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.69 mullo()** [2/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```



**16.274.2.70 mul()** [2/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.71 mulhi()** [2/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.72 mulx()** [2/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.274.2.73 fmadd()** [2/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.74 fmaddin()** [2/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.75 fmaddx()** [2/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.76 fmaddxin()** [2/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.77 fnmadd()** [2/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.78 fnmaddin() [2/2]**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.79 fnmaddx() [2/2]**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.80 fnmaddxin() [2/2]**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.81 fmsub() [2/2]**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.82 fmsubin() [2/2]**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.83 fmsubx() [2/2]**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.84 fmsubxin() [2/2]**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.85 eq() [2/2]**

```
static INLINE CONST vect_t eq (  
    const vect_t a,
```

```
const vect_t b ) [inline], [static]
```

#### 16.274.2.86 greater() [2/2]

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.87 lesser() [2/2]

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.88 greater\_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.89 lesser\_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.90 hadd\_to\_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.274.2.91 round() [2/2]

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

#### 16.274.2.92 mod() [2/2]

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static]
```

#### 16.274.2.93 type\_string() [1/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.274.2.94 zero()** [1/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.274.2.95 type\_string()** [2/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.274.2.96 zero()** [2/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.274.2.97 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.274.2.98 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.274.2.99 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.274.2.100 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.274.3 Field Documentation****16.274.3.1 vect\_size**

```
static constexpr const size_t vect_size = 8 [static], [constexpr]
```

**16.274.3.2 alignment**

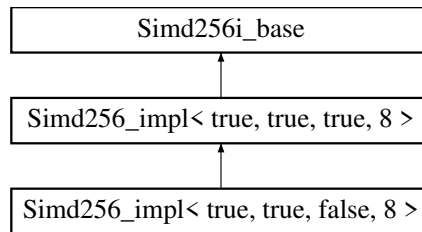
```
static constexpr const size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.275 Simd256\_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, true, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m256i
- using [half\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int64\_t
- using [simdHalf](#) = Simd128< [scalar\\_t](#) >

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- template<int idx>  
static [INLINE CONST scalar\\_t get](#) ([vect\\_t](#) v)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [unpacklohi](#) ([vect\\_t](#) &l, [vect\\_t](#) &h, const [vect\\_t](#) a, const [vect\\_t](#) b)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmadddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const `__m256d` &P, const `__m256d` &INVP, const `__m256d` &NEGP, const `vect_t` &POW50REM, const `__m256d` &MIN, const `__m256d` &MAX, `__m256d` &Q, `__m256d` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 4
- static constexpr const size\_t `alignment` = 32

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.275.1 Member Typedef Documentation

### 16.275.1.1 `vect_t`

```
using vect_t = __m256i
```

### 16.275.1.2 `half_t`

```
using half_t = __m128i
```

### 16.275.1.3 scalar\_t

```
using scalar_t = int64_t
```

### 16.275.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

## 16.275.2 Member Function Documentation

### 16.275.2.1 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

### 16.275.2.2 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

### 16.275.2.3 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.275.2.4 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

### 16.275.2.5 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.275.2.6 get()

```
static INLINE CONST scalar_t get (  
    vect_t v ) [inline], [static]
```

### 16.275.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.275.2.8 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.275.2.9 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.275.2.10 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.275.2.11 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.275.2.12 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.275.2.13 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.275.2.14 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.275.2.15 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.275.2.16 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```



**16.275.2.17 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.18 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.19 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.20 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.21 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.22 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.23 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.24 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.25 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.26 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.275.2.27 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.28 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.29 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.30 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.31 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.32 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.33 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.34 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.35 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.36 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.37 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.38 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.39 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.40 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,
```

```
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.41 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.42 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.43 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.44 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.45 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.46 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.275.2.47 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

#### 16.275.2.48 mask\_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.275.2.49 mulhi\_fast()**

```

INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]

```

**16.275.2.50 mod()**

```

INLINE vect_t mod (
    vect_t & C,
    const __m256d & P,
    const __m256d & INV_P,
    const __m256d & NEG_P,
    const vect_t & POW50REM,
    const __m256d & MIN,
    const __m256d & MAX,
    __m256d & Q,
    __m256d & T ) [static]

```

**16.275.2.51 signbits()**

```

static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]

```

**16.275.2.52 type\_string()**

```

static const std::string type_string ( ) [inline], [static], [inherited]

```

**16.275.2.53 zero()**

```

static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]

```

**16.275.3 Field Documentation****16.275.3.1 vect\_size**

```

constexpr const size_t vect_size = 4 [static], [constexpr]

```

**16.275.3.2 alignment**

```

constexpr const size_t alignment = 32 [static], [constexpr]

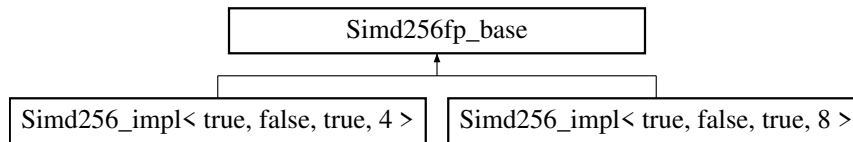
```

The documentation for this struct was generated from the following file:

- [simd256\\_int64.inl](#)

**16.276 Simd256fp\_base Struct Reference**

Inheritance diagram for Simd256fp\_base:

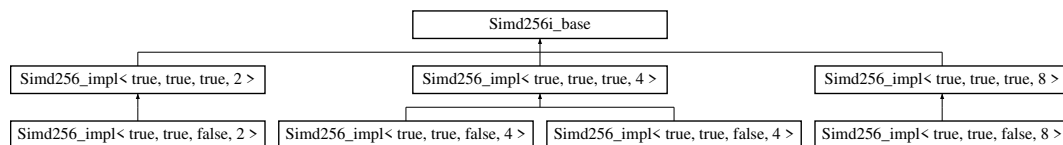


The documentation for this struct was generated from the following file:

- [simd256.inl](#)

## 16.277 Simd256i\_base Struct Reference

Inheritance diagram for Simd256i\_base:



### Public Types

- using [vect\\_t](#) = \_\_m256i

### Static Public Member Functions

- static const std::string [type\\_string](#) ()
- static [INLINE CONST vect\\_t zero](#) ()

## 16.277.1 Member Typedef Documentation

### 16.277.1.1 vect\_t

using [vect\\_t](#) = \_\_m256i

## 16.277.2 Member Function Documentation

### 16.277.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.277.2.2 zero()

```
static INLINE CONST vect\_t zero ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd256.inl](#)

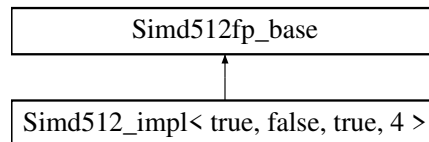
## 16.278 Simd512\_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

## 16.279 Simd512\_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd512\_impl< true, false, true, 4 >:



### Static Public Member Functions

- static const std::string [type\\_string](#) ()

### 16.279.1 Member Function Documentation

#### 16.279.1.1 type\_string()

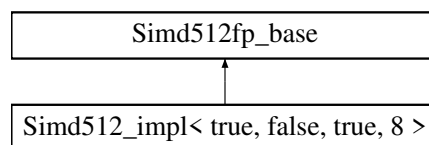
```
static const std::string type_string ( ) [inline], [static], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd512\\_float.inl](#)

## 16.280 Simd512\_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd512\_impl< true, false, true, 8 >:



### Public Types

- using [vect\\_t](#) = \_\_m512d
- using [scalar\\_t](#) = double

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t zero](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)

- static `INLINE CONST vect_t set` (const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE void storeu` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE void stream` (const `scalar_t` \*p, const `vect_t` v)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t blendv` (const `vect_t` a, const `vect_t` b, const `vect_t` mask)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t div` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t floor` (const `vect_t` a)
- static `INLINE CONST vect_t ceil` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t hadd` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static const std::string `type_string` ()

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 64

## 16.280.1 Member Typedef Documentation

### 16.280.1.1 vect\_t

using `vect_t` = \_\_m512d



### 16.280.1.2 scalar\_t

```
using scalar_t = double
```

## 16.280.2 Member Function Documentation

### 16.280.2.1 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

### 16.280.2.2 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

### 16.280.2.3 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

### 16.280.2.4 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.280.2.5 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7,  
    const scalar_t x8 ) [inline], [static]
```

### 16.280.2.6 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.280.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.280.2.8 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.280.2.9 store()

```
static INLINE void store (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.280.2.10 storeu()

```
static INLINE void storeu (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.280.2.11 stream()

```
static INLINE void stream (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.280.2.12 shuffle()

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

#### 16.280.2.13 unpacklo\_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.14 unpackhi\_twice()

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.15 blend()

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.280.2.16 blendv()

```
static INLINE CONST vect_t blendv (  
    const vect_t a,  
    const vect_t b,  
    const vect_t mask ) [inline], [static]
```

**16.280.2.17 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.18 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.19 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.20 subin()**

```
static INLINE CONST vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.21 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.22 mulin()**

```
static INLINE CONST vect_t mulin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.23 div()**

```
static INLINE CONST vect_t div (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.24 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.25 fmaddin()**

```
static INLINE CONST vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.26 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.27 fnmaddin()**

```
static INLINE CONST vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.28 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.29 fmsubin()**

```
static INLINE CONST vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.30 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.31 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.32 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.33 greater()**

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.34 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.35 floor()**

```
static INLINE CONST vect_t floor (  
    const vect_t a ) [inline], [static]
```

**16.280.2.36 ceil()**

```
static INLINE CONST vect_t ceil (  
    const vect_t a ) [inline], [static]
```

**16.280.2.37 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.280.2.38 hadd()**

```
static INLINE CONST vect_t hadd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.39 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.280.2.40 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.280.3 Field Documentation****16.280.3.1 vect\_size**

```
constexpr const size_t vect_size = 8 [static], [constexpr]
```

### 16.280.3.2 alignment

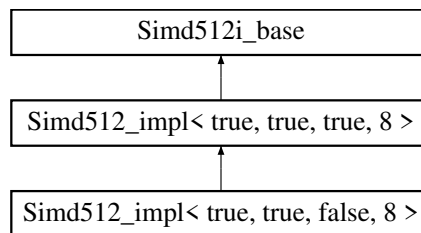
```
constexpr const size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd512\\_double.inl](#)

## 16.281 Simd512\_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd512\_impl< true, true, false, 8 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [scalar\\_t](#) = [uint64\\_t](#)
- using [simdHalf](#) = [Simd256](#)< [scalar\\_t](#) >
- using [vect\\_t](#) = [\\_\\_m512i](#)
- using [half\\_t](#) = [\\_\\_m256i](#)

## Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- template<uint8\_t k>  
static [INLINE void maskstore](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &l, `vect_t` &h, const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmadin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const `__m512d` &P, const `__m512d` &INVP, const `__m512d` &NEGP, const `vect_t` &POW50REM, const `__m512d` &MIN, const `__m512d` &MAX, `__m512d` &Q, `__m512d` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 64

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.281.1 Member Typedef Documentation

### 16.281.1.1 scalar\_t

```
using scalar_t = uint64_t
```

### 16.281.1.2 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

### 16.281.1.3 vect\_t

```
using vect_t = __m512i [inherited]
```

### 16.281.1.4 half\_t

```
using half_t = __m256i [inherited]
```

## 16.281.2 Member Function Documentation

### 16.281.2.1 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.281.2.2 set() [1/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

### 16.281.2.3 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.281.2.4 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```



**16.281.2.5 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.281.2.6 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.281.2.7 maskstore()**

```
static INLINE void maskstore (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.281.2.8 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.281.2.9 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.281.2.10 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.281.2.11 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.281.2.12 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.281.2.13 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.14 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.15 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.281.2.16 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.17 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.18 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.19 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.20 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.21 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.22 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.23 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.281.2.24 valid()**

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.281.2.25 compliant()**

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.281.2.26 set() [2/2]**

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static], [inherited]
```

**16.281.2.27 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.281.2.28 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.281.2.29 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.281.2.30 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.31 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.32 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.33 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.34 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.35 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.36 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.37 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.38 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.39 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.40 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.41 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.42 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.43 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.44 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.45 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.46 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.47 eq()**

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.48 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

**16.281.2.49 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.281.2.50 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

**16.281.2.51 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m512d & P,
    const __m512d & INV_P,
    const __m512d & NEG_P,
    const vect_t & POW50REM,
    const __m512d & MIN,
    const __m512d & MAX,
    __m512d & Q,
    __m512d & T ) [static], [inherited]
```

**16.281.2.52 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected], [inherited]
```

**16.281.2.53 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.281.2.54 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.281.2.55 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.56 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.57 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.58 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.3 Field Documentation****16.281.3.1 vect\_size**

```
constexpr const size_t vect_size = 8 [static], [constexpr], [inherited]
```

**16.281.3.2 alignment**

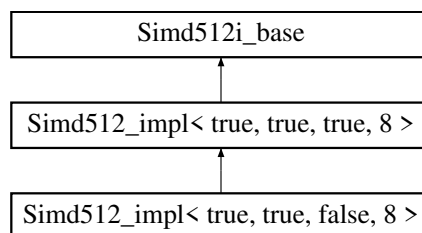
```
constexpr const size_t alignment = 64 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd512\\_int64.inl](#)

**16.282 Simd512\_impl< true, true, true, 8 > Struct Reference**

Inheritance diagram for Simd512\_impl< true, true, true, 8 >:

**Data Structures**

- union [Converter](#)

## Public Types

- using `vect_t` = `__m512i`
- using `half_t` = `__m256i`
- using `scalar_t` = `int64_t`
- using `simdHalf` = `Simd256< scalar_t >`

## Static Public Member Functions

- `template<class T >`  
static constexpr bool `valid` (T \*p)
- `template<class T >`  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3)
- `template<class T >`  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- `template<uint8_t k>`  
static `INLINE void maskstore` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- `template<int s>`  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- `template<int s>`  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- `template<int s>`  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- `template<uint8_t s>`  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &l, `vect_t` &h, const `vect_t` a, const `vect_t` b)
- `template<uint8_t s>`  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)



- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const `__m512d` &P, const `__m512d` &INVP, const `__m512d` &NEGP, const `vect_t` &POW50REM, const `__m512d` &MIN, const `__m512d` &MAX, `__m512d` &Q, `__m512d` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static constexpr const size\_t `vect_size` = 8
- static constexpr const size\_t `alignment` = 64

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.282.1 Member Typedef Documentation

### 16.282.1.1 vect\_t

```
using vect_t = __m512i
```

### 16.282.1.2 half\_t

```
using half_t = __m256i
```

### 16.282.1.3 scalar\_t

```
using scalar_t = int64_t
```

### 16.282.1.4 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

## 16.282.2 Member Function Documentation

**16.282.2.1 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.282.2.2 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.282.2.3 set1()**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.282.2.4 set() [1/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

**16.282.2.5 set() [2/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

**16.282.2.6 gather()**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

**16.282.2.7 load()**

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

**16.282.2.8 loadu()**

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

**16.282.2.9 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.282.2.10 maskstore()**

```
static INLINE void maskstore (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.282.2.11 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.282.2.12 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.282.2.13 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.282.2.14 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.282.2.15 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.282.2.16 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.282.2.17 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.18 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.19 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.20 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.21 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.22 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.23 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.24 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.25 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.26 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.27 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.282.2.28 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.29 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.30 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.31 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.32 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.33 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.34 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.35 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.36 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.37 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.38 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.39 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.40 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.41 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.282.2.42 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.43 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.44 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.45 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.46 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.47 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.282.2.48 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

#### 16.282.2.49 mask\_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.282.2.50 mulhi\_fast()**

```

INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]

```

**16.282.2.51 mod()**

```

INLINE vect_t mod (
    vect_t & C,
    const __m512d & P,
    const __m512d & INVP,
    const __m512d & NEGP,
    const vect_t & POW50REM,
    const __m512d & MIN,
    const __m512d & MAX,
    __m512d & Q,
    __m512d & T ) [static]

```

**16.282.2.52 signbits()**

```

static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]

```

**16.282.2.53 type\_string()**

```

static const std::string type_string ( ) [inline], [static], [inherited]

```

**16.282.2.54 zero()**

```

static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]

```

**16.282.2.55 vor()**

```

static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

**16.282.2.56 vxor()**

```

static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

**16.282.2.57 vand()**

```

static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```



**16.282.2.58 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.3 Field Documentation****16.282.3.1 vect\_size**

```
constexpr const size_t vect_size = 8 [static], [constexpr]
```

**16.282.3.2 alignment**

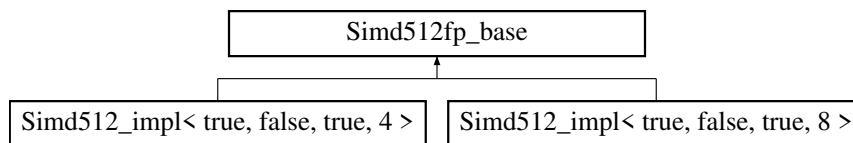
```
constexpr const size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd512\\_int64.inl](#)

**16.283 Simd512fp\_base Struct Reference**

Inheritance diagram for Simd512fp\_base:

**Static Public Member Functions**

- static const std::string [type\\_string](#) ()

**16.283.1 Member Function Documentation****16.283.1.1 type\_string()**

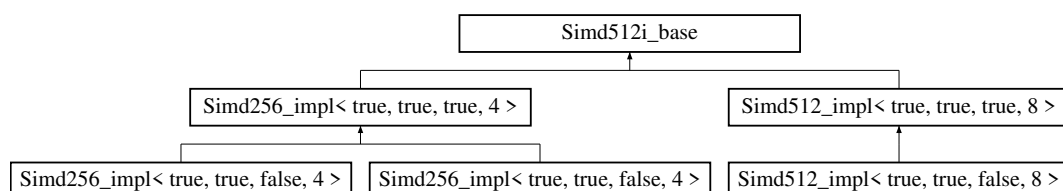
```
static const std::string type_string ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

**16.284 Simd512i\_base Struct Reference**

Inheritance diagram for Simd512i\_base:



## Public Types

- using `vect_t` = `__m512i`

## Static Public Member Functions

- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## 16.284.1 Member Typedef Documentation

### 16.284.1.1 `vect_t`

using `vect_t` = `__m512i`

## 16.284.2 Member Function Documentation

### 16.284.2.1 `type_string()`

```
static const std::string type_string ( ) [inline], [static]
```

### 16.284.2.2 `zero()`

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

### 16.284.2.3 `vor()`

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

### 16.284.2.4 `vxor()`

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

### 16.284.2.5 `vand()`

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.284.2.6 vandnot()

```
static INLINE CONST vect_t vandnot (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

### 16.285 SimdChooser< T, bool, bool > Struct Template Reference

```
#include <fflas_simd.h>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

### 16.286 SimdChooser< T, false, b > Struct Template Reference

```
#include <fflas_simd.h>
```

#### Public Types

- using [value](#) = [NoSimd](#)< T >

#### 16.286.1 Member Typedef Documentation

##### 16.286.1.1 value

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

### 16.287 SimdChooser< T, true, false > Struct Template Reference

```
#include <fflas_simd.h>
```

#### Public Types

- using [value](#) = [NoSimd](#)< T >

#### 16.287.1 Member Typedef Documentation

##### 16.287.1.1 value

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

### 16.288 SimdChooser< T, true, true > Struct Template Reference

```
#include <fflas_simd.h>
```

## Public Types

- using [value](#) = [NoSimd](#)< T >

### 16.288.1 Member Typedef Documentation

#### 16.288.1.1 value

using [value](#) = [NoSimd](#)<T>

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

### 16.289 [simdToType](#)< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

### 16.290 Single Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.291 [Sparse](#)< [Field](#), [SparseMatrix\\_t](#), [IdxT](#), [PtrT](#) > Struct Template Reference

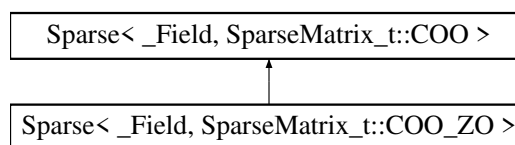
The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

### 16.292 [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::COO](#) > Struct Template Reference

```
#include <coo.h>
```

Inheritance diagram for [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::COO](#) >:



## Public Types

- using [Field](#) = [\\_Field](#)

## Data Fields

- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [row](#) = nullptr
- [\\_Field::Element\\_ptr](#) dat

- bool `delayed` = false
- `uint64_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0

## 16.292.1 Member Typedef Documentation

### 16.292.1.1 Field

using `Field` = `_Field`

## 16.292.2 Field Documentation

### 16.292.2.1 col

`index_t*` `col` = `nullptr`

### 16.292.2.2 row

`index_t*` `row` = `nullptr`

### 16.292.2.3 dat

`_Field::Element_ptr` `dat`

### 16.292.2.4 delayed

bool `delayed` = false

### 16.292.2.5 kmax

`uint64_t` `kmax` = 0

### 16.292.2.6 m

`index_t` `m` = 0

### 16.292.2.7 n

`index_t` `n` = 0

### 16.292.2.8 nnz

`uint64_t` `nnz` = 0

### 16.292.2.9 nElements

```
uint64_t nElements = 0
```

### 16.292.2.10 maxrow

```
uint64_t maxrow = 0
```

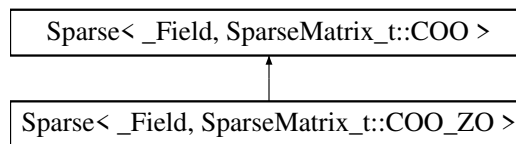
The documentation for this struct was generated from the following file:

- [coo.h](#)

## 16.293 Sparse<\_Field, SparseMatrix\_t::COO\_ZO > Struct Template Reference

```
#include <coo.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::COO\_ZO >:



### Public Types

- using [Field](#) = \_Field

### Data Fields

- \_Field::Element [cst](#) = 1
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [row](#) = nullptr
- \_Field::Element\_ptr [dat](#)
- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0

## 16.293.1 Member Typedef Documentation

### 16.293.1.1 Field

```
using Field = _Field
```

## 16.293.2 Field Documentation

### 16.293.2.1 cst

```
_Field::Element cst = 1
```

**16.293.2.2 col**

```
index_t* col = nullptr [inherited]
```

**16.293.2.3 row**

```
index_t* row = nullptr [inherited]
```

**16.293.2.4 dat**

```
_Field::Element_ptr dat [inherited]
```

**16.293.2.5 delayed**

```
bool delayed = false [inherited]
```

**16.293.2.6 kmax**

```
uint64_t kmax = 0 [inherited]
```

**16.293.2.7 m**

```
index_t m = 0 [inherited]
```

**16.293.2.8 n**

```
index_t n = 0 [inherited]
```

**16.293.2.9 nnz**

```
uint64_t nnz = 0 [inherited]
```

**16.293.2.10 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.293.2.11 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

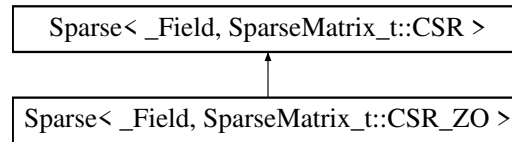
The documentation for this struct was generated from the following file:

- [coo.h](#)

## 16.294 Sparse< \_Field, SparseMatrix\_t::CSR > Struct Template Reference

```
#include <csr.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::CSR >:



## Public Types

- using `Field` = `_Field`

## Data Fields

- bool `delayed` = false
- `uint64_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0
- `index_t` \* `col` = nullptr
- `index_t` \* `st` = nullptr
- `index_t` \* `stend` = nullptr
- `_Field::Element_ptr` `dat`

## 16.294.1 Member Typedef Documentation

### 16.294.1.1 Field

using `Field` = `_Field`

## 16.294.2 Field Documentation

### 16.294.2.1 delayed

bool `delayed` = false

### 16.294.2.2 kmax

`uint64_t` `kmax` = 0

### 16.294.2.3 m

`index_t` `m` = 0

### 16.294.2.4 n

`index_t` `n` = 0



**16.294.2.5 nnz**

```
uint64_t nnz = 0
```

**16.294.2.6 nElements**

```
uint64_t nElements = 0
```

**16.294.2.7 maxrow**

```
uint64_t maxrow = 0
```

**16.294.2.8 col**

```
index_t* col = nullptr
```

**16.294.2.9 st**

```
index_t* st = nullptr
```

**16.294.2.10 stend**

```
index_t* stend = nullptr
```

**16.294.2.11 dat**

```
_Field::Element_ptr dat
```

The documentation for this struct was generated from the following file:

- [csr.h](#)

## 16.295 Sparse< \_Field, SparseMatrix\_t::CSR\_HYB > Struct Template Reference

```
#include <csr_hyb.h>
```

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- bool [delayed](#) = false
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [st](#) = nullptr
- [\\_Field::Element\\_ptr](#) [dat](#)
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0

- `uint64_t nOnes = 0`
- `uint64_t nMOnes = 0`
- `uint64_t nOthers = 0`

## 16.295.1 Member Typedef Documentation

### 16.295.1.1 Field

```
using Field = _Field
```

## 16.295.2 Field Documentation

### 16.295.2.1 delayed

```
bool delayed = false
```

### 16.295.2.2 col

```
index_t* col = nullptr
```

### 16.295.2.3 st

```
index_t* st = nullptr
```

### 16.295.2.4 dat

```
_Field::Element_ptr dat
```

### 16.295.2.5 kmax

```
uint64_t kmax = 0
```

### 16.295.2.6 m

```
index_t m = 0
```

### 16.295.2.7 n

```
index_t n = 0
```

### 16.295.2.8 nnz

```
uint64_t nnz = 0
```

### 16.295.2.9 nElements

```
uint64_t nElements = 0
```

**16.295.2.10 maxrow**

```
uint64_t maxrow = 0
```

**16.295.2.11 nOnes**

```
uint64_t nOnes = 0
```

**16.295.2.12 nMOnes**

```
uint64_t nMOnes = 0
```

**16.295.2.13 nOthers**

```
uint64_t nOthers = 0
```

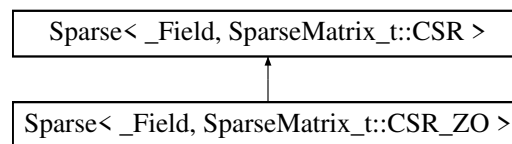
The documentation for this struct was generated from the following file:

- [csr\\_hyb.h](#)

## 16.296 Sparse< \_Field, SparseMatrix\_t::CSR\_ZO > Struct Template Reference

```
#include <csr.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >:

**Public Types**

- using [Field](#) = [\\_Field](#)

**Data Fields**

- [int64\\_t](#) [cst](#) = 1
- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [st](#) = nullptr
- [index\\_t](#) \* [stend](#) = nullptr
- [\\_Field::Element\\_ptr](#) [dat](#)

**16.296.1 Member Typedef Documentation**

### 16.296.1.1 Field

```
using Field = _Field
```

## 16.296.2 Field Documentation

### 16.296.2.1 cst

```
int64_t cst = 1
```

### 16.296.2.2 delayed

```
bool delayed = false
```

### 16.296.2.3 kmax

```
uint64_t kmax = 0 [inherited]
```

### 16.296.2.4 m

```
index_t m = 0 [inherited]
```

### 16.296.2.5 n

```
index_t n = 0 [inherited]
```

### 16.296.2.6 nnz

```
uint64_t nnz = 0 [inherited]
```

### 16.296.2.7 nElements

```
uint64_t nElements = 0 [inherited]
```

### 16.296.2.8 maxrow

```
uint64_t maxrow = 0 [inherited]
```

### 16.296.2.9 col

```
index_t* col = nullptr [inherited]
```

### 16.296.2.10 st

```
index_t* st = nullptr [inherited]
```

**16.296.2.11 stend**

```
index_t* stend = nullptr [inherited]
```

**16.296.2.12 dat**

```
_Field::Element_ptr dat [inherited]
```

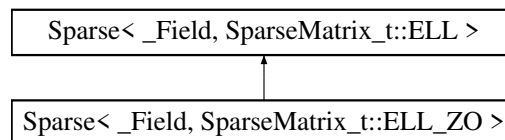
The documentation for this struct was generated from the following file:

- [csr.h](#)

## 16.297 Sparse< \_Field, SparseMatrix\_t::ELL > Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::ELL >:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [ld](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [index\\_t](#)\* [col](#) = nullptr
- [\\_Field::Element\\_ptr](#) [dat](#)

**16.297.1 Member Typedef Documentation****16.297.1.1 Field**

```
using Field = _Field
```

**16.297.2 Field Documentation****16.297.2.1 delayed**

```
bool delayed = false
```

**16.297.2.2 kmax**

```
uint64_t kmax = 0
```

**16.297.2.3 m**

```
index_t m = 0
```

**16.297.2.4 n**

```
index_t n = 0
```

**16.297.2.5 ld**

```
index_t ld = 0
```

**16.297.2.6 nnz**

```
uint64_t nnz = 0
```

**16.297.2.7 nElements**

```
uint64_t nElements = 0
```

**16.297.2.8 maxrow**

```
uint64_t maxrow = 0
```

**16.297.2.9 col**

```
index_t* col = nullptr
```

**16.297.2.10 dat**

```
_Field::Element_ptr dat
```

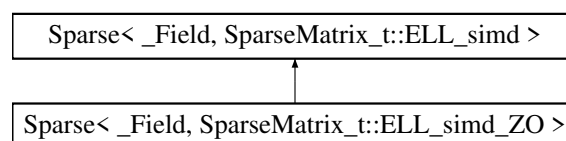
The documentation for this struct was generated from the following file:

- [ell.h](#)

## 16.298 Sparse<\_Field, SparseMatrix\_t::ELL\_simd> Struct Template Reference

```
#include <ell_simd.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::ELL\_simd>:



## Data Fields

- bool `delayed` = false
- int `chunk` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `index_t` `ld` = 0
- `uint64_t` `kmax` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0
- `uint64_t` `nChunks` = 0
- `index_t` \* `col` = nullptr
- `_Field::Element_ptr` `dat`

### 16.298.1 Field Documentation

#### 16.298.1.1 `delayed`

```
bool delayed = false
```

#### 16.298.1.2 `chunk`

```
int chunk = 0
```

#### 16.298.1.3 `m`

```
index_t m = 0
```

#### 16.298.1.4 `n`

```
index_t n = 0
```

#### 16.298.1.5 `ld`

```
index_t ld = 0
```

#### 16.298.1.6 `kmax`

```
uint64_t kmax = 0
```

#### 16.298.1.7 `nnz`

```
uint64_t nnz = 0
```

#### 16.298.1.8 `nElements`

```
uint64_t nElements = 0
```

**16.298.1.9 maxrow**

```
uint64_t maxrow = 0
```

**16.298.1.10 nChunks**

```
uint64_t nChunks = 0
```

**16.298.1.11 col**

```
index_t* col = nullptr
```

**16.298.1.12 dat**

```
_Field::Element_ptr dat
```

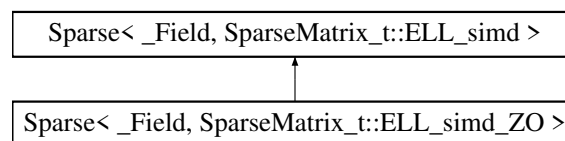
The documentation for this struct was generated from the following file:

- [ell\\_simd.h](#)

## 16.299 Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO > Struct Template Reference

```
#include <ell_simd.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >:

**Data Fields**

- \_Field::Element [cst](#) = 1
- bool [delayed](#) = false
- int [chunk](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [ld](#) = 0
- [uint64\\_t](#) [kmax](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [uint64\\_t](#) [nChunks](#) = 0
- [index\\_t](#) \* [col](#) = nullptr
- \_Field::Element\_ptr [dat](#)

**16.299.1 Field Documentation****16.299.1.1 cst**

```
_Field::Element cst = 1
```



**16.299.1.2 delayed**

```
bool delayed = false [inherited]
```

**16.299.1.3 chunk**

```
int chunk = 0 [inherited]
```

**16.299.1.4 m**

```
index_t m = 0 [inherited]
```

**16.299.1.5 n**

```
index_t n = 0 [inherited]
```

**16.299.1.6 ld**

```
index_t ld = 0 [inherited]
```

**16.299.1.7 kmax**

```
uint64_t kmax = 0 [inherited]
```

**16.299.1.8 nnz**

```
uint64_t nnz = 0 [inherited]
```

**16.299.1.9 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.299.1.10 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

**16.299.1.11 nChunks**

```
uint64_t nChunks = 0 [inherited]
```

**16.299.1.12 col**

```
index_t* col = nullptr [inherited]
```

**16.299.1.13 dat**

```
_Field::Element_ptr dat [inherited]
```

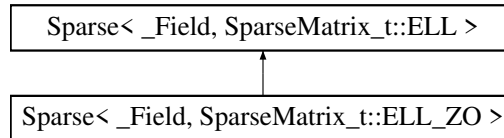
The documentation for this struct was generated from the following file:

- [ell\\_simd.h](#)

## 16.300 Sparse< \_Field, SparseMatrix\_t::ELL\_ZO > Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >:



### Public Types

- using [Field](#) = \_Field

### Data Fields

- \_Field::Element [cst](#) = 1
- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [ld](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [index\\_t](#) \* [col](#) = nullptr
- \_Field::Element\_ptr [dat](#)

### 16.300.1 Member Typedef Documentation

#### 16.300.1.1 Field

```
using Field = _Field
```

### 16.300.2 Field Documentation

#### 16.300.2.1 cst

```
_Field::Element cst = 1
```

#### 16.300.2.2 delayed

```
bool delayed = false [inherited]
```

#### 16.300.2.3 kmax

```
uint64\_t kmax = 0 [inherited]
```

**16.300.2.4 m**

```
index_t m = 0 [inherited]
```

**16.300.2.5 n**

```
index_t n = 0 [inherited]
```

**16.300.2.6 ld**

```
index_t ld = 0 [inherited]
```

**16.300.2.7 nnz**

```
uint64_t nnz = 0 [inherited]
```

**16.300.2.8 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.300.2.9 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

**16.300.2.10 col**

```
index_t* col = nullptr [inherited]
```

**16.300.2.11 dat**

```
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- [ell.h](#)

## 16.301 Sparse<\_Field, SparseMatrix\_t::HYB\_ZO > Struct Template Reference

```
#include <hyb_zo.h>
```

### Public Types

- using [Field](#) = [\\_Field](#)
- typedef [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) > [Self\\_t](#)

### Data Fields

- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0

- `uint64_t nnz = 0`
- `uint64_t maxrow = 0`
- `uint64_t nElements = 0`
- `Sparse<_Field, SparseMatrix_t::CSR> * dat = nullptr`
- `Sparse<_Field, SparseMatrix_t::CSR_ZO> * one = nullptr`
- `Sparse<_Field, SparseMatrix_t::CSR_ZO> * mone = nullptr`

### 16.301.1 Member Typedef Documentation

#### 16.301.1.1 Field

using `Field` = `_Field`

#### 16.301.1.2 Self\_t

typedef `Sparse<_Field, SparseMatrix_t::HYB_ZO>` `Self_t`

### 16.301.2 Field Documentation

#### 16.301.2.1 delayed

`bool delayed = false`

#### 16.301.2.2 kmax

`uint64_t kmax = 0`

#### 16.301.2.3 m

`index_t m = 0`

#### 16.301.2.4 n

`index_t n = 0`

#### 16.301.2.5 nnz

`uint64_t nnz = 0`

#### 16.301.2.6 maxrow

`uint64_t maxrow = 0`

#### 16.301.2.7 nElements

`uint64_t nElements = 0`

**16.301.2.8 dat**

```
Sparse<_Field, SparseMatrix_t::CSR>* dat = nullptr
```

**16.301.2.9 one**

```
Sparse<_Field, SparseMatrix_t::CSR_ZO>* one = nullptr
```

**16.301.2.10 mone**

```
Sparse<_Field, SparseMatrix_t::CSR_ZO>* mone = nullptr
```

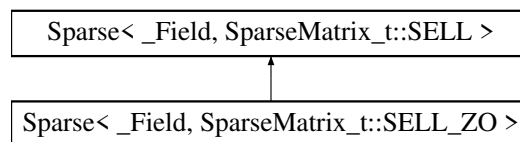
The documentation for this struct was generated from the following file:

- [hyb\\_zo.h](#)

## 16.302 Sparse< \_Field, SparseMatrix\_t::SELL > Struct Template Reference

```
#include <sell.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::SELL >:

**Public Types**

- using [Field](#) = [\\_Field](#)

**Data Fields**

- bool [delayed](#) = false
- int [chunk](#) = 0
- [index\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [maxrow](#) = 0
- [index\\_t](#) [sigma](#) = 0
- [index\\_t](#) [nChunks](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [index\\_t](#) \* [perm](#) = nullptr
- [uint64\\_t](#) \* [st](#) = nullptr
- [index\\_t](#) \* [chunkSize](#) = nullptr
- [index\\_t](#) \* [col](#) = nullptr
- [\\_Field::Element\\_ptr](#) [dat](#)

**16.302.1 Member Typedef Documentation**

#### 16.302.1.1 Field

```
using Field = _Field
```

### 16.302.2 Field Documentation

#### 16.302.2.1 delayed

```
bool delayed = false
```

#### 16.302.2.2 chunk

```
int chunk = 0
```

#### 16.302.2.3 kmax

```
index_t kmax = 0
```

#### 16.302.2.4 m

```
index_t m = 0
```

#### 16.302.2.5 n

```
index_t n = 0
```

#### 16.302.2.6 maxrow

```
index_t maxrow = 0
```

#### 16.302.2.7 sigma

```
index_t sigma = 0
```

#### 16.302.2.8 nChunks

```
index_t nChunks = 0
```

#### 16.302.2.9 nnz

```
uint64_t nnz = 0
```

#### 16.302.2.10 nElements

```
uint64_t nElements = 0
```

**16.302.2.11 perm**

```
index_t* perm = nullptr
```

**16.302.2.12 st**

```
uint64_t* st = nullptr
```

**16.302.2.13 chunkSize**

```
index_t* chunkSize = nullptr
```

**16.302.2.14 col**

```
index_t* col = nullptr
```

**16.302.2.15 dat**

```
_Field::Element_ptr dat
```

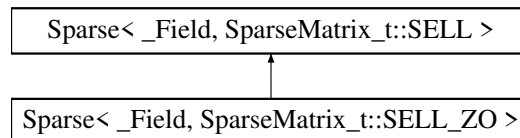
The documentation for this struct was generated from the following file:

- [sell.h](#)

## 16.303 Sparse< \_Field, SparseMatrix\_t::SELL\_ZO > Struct Template Reference

```
#include <sell.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- \_Field::Element [cst](#) = 1
- bool [delayed](#) = false
- int [chunk](#) = 0
- [index\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [maxrow](#) = 0
- [index\\_t](#) [sigma](#) = 0
- [index\\_t](#) [nChunks](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [index\\_t](#) \* [perm](#) = nullptr

- `uint64_t * st = nullptr`
- `index_t * chunkSize = nullptr`
- `index_t * col = nullptr`
- `_Field::Element_ptr dat`

### 16.303.1 Member Typedef Documentation

#### 16.303.1.1 Field

using `Field` = `_Field`

### 16.303.2 Field Documentation

#### 16.303.2.1 cst

`_Field::Element` `cst` = 1

#### 16.303.2.2 delayed

`bool` `delayed` = `false` [inherited]

#### 16.303.2.3 chunk

`int` `chunk` = 0 [inherited]

#### 16.303.2.4 kmax

`index_t` `kmax` = 0 [inherited]

#### 16.303.2.5 m

`index_t` `m` = 0 [inherited]

#### 16.303.2.6 n

`index_t` `n` = 0 [inherited]

#### 16.303.2.7 maxrow

`index_t` `maxrow` = 0 [inherited]

#### 16.303.2.8 sigma

`index_t` `sigma` = 0 [inherited]



**16.303.2.9 nChunks**

```
index_t nChunks = 0 [inherited]
```

**16.303.2.10 nnz**

```
uint64_t nnz = 0 [inherited]
```

**16.303.2.11 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.303.2.12 perm**

```
index_t* perm = nullptr [inherited]
```

**16.303.2.13 st**

```
uint64_t* st = nullptr [inherited]
```

**16.303.2.14 chunkSize**

```
index_t* chunkSize = nullptr [inherited]
```

**16.303.2.15 col**

```
index_t* col = nullptr [inherited]
```

**16.303.2.16 dat**

```
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- [sell.h](#)

**16.304 SpMat< Field, flag > Struct Template Reference**

```
#include <fflas_sparse.h>
```

**Data Fields**

- [FFLAS::CooMat< Field > \\* \\_coo](#) = nullptr
- [FFLAS::CsrMat< Field > \\* \\_csr](#) = nullptr
- [FFLAS::EliMat< Field > \\* \\_eli](#) = nullptr

**16.304.1 Field Documentation****16.304.1.1 \_coo**

```
FFLAS::CooMat<Field>* _coo = nullptr
```

**16.304.1.2 \_csr**

```
FFLAS::CsrMat<Field>* _csr = nullptr
```

**16.304.1.3 \_ell**

```
FFLAS::EllMat<Field>* _ell = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

**16.305 Static\_error\_check< bool > Class Template Reference**

```
#include <instrset.h>
```

**Public Member Functions**

- [Static\\_error\\_check\(\)](#)

**16.305.1 Constructor & Destructor Documentation****16.305.1.1 Static\_error\_check()**

```
Static_error_check ( ) [inline]
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

**16.306 Static\_error\_check< false > Class Reference**

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

**16.307 StatsMatrix Struct Reference**

```
#include <utils.h>
```

**Data Fields**

- [uint64\\_t rowdim](#) = 0
- [uint64\\_t coldim](#) = 0
- [uint64\\_t nOnes](#) = 0
- [uint64\\_t nMOnes](#) = 0
- [uint64\\_t nOthers](#) = 0
- [uint64\\_t nnz](#) = 0
- [uint64\\_t maxRow](#) = 0
- [uint64\\_t minRow](#) = 0
- [uint64\\_t averageRow](#) = 0
- [uint64\\_t deviationRow](#) = 0
- [uint64\\_t maxCol](#) = 0
- [uint64\\_t minCol](#) = 0
- [uint64\\_t averageCol](#) = 0

- `uint64_t` `deviationCol` = 0
- `uint64_t` `minColDifference` = 0
- `uint64_t` `maxColDifference` = 0
- `uint64_t` `averageColDifference` = 0
- `uint64_t` `deviationColDifference` = 0
- `uint64_t` `minRowDifference` = 0
- `uint64_t` `maxRowDifference` = 0
- `uint64_t` `averageRowDifference` = 0
- `uint64_t` `deviationRowDifference` = 0
- `uint64_t` `nDenseRows` = 0
- `uint64_t` `nDenseCols` = 0
- `uint64_t` `nEmptyRows` = 0
- `uint64_t` `nEmptyCols` = 0
- `uint64_t` `nEmptyColsEnd` = 0
- `std::vector< uint64_t >` `denseRows`
- `std::vector< uint64_t >` `denseCols`

## 16.307.1 Field Documentation

### 16.307.1.1 rowdim

`uint64_t` `rowdim` = 0

### 16.307.1.2 coldim

`uint64_t` `coldim` = 0

### 16.307.1.3 nOnes

`uint64_t` `nOnes` = 0

### 16.307.1.4 nMOnes

`uint64_t` `nMOnes` = 0

### 16.307.1.5 nOthers

`uint64_t` `nOthers` = 0

### 16.307.1.6 nnz

`uint64_t` `nnz` = 0

### 16.307.1.7 maxRow

`uint64_t` `maxRow` = 0

**16.307.1.8 minRow**

```
uint64_t minRow = 0
```

**16.307.1.9 averageRow**

```
uint64_t averageRow = 0
```

**16.307.1.10 deviationRow**

```
uint64_t deviationRow = 0
```

**16.307.1.11 maxCol**

```
uint64_t maxCol = 0
```

**16.307.1.12 minCol**

```
uint64_t minCol = 0
```

**16.307.1.13 averageCol**

```
uint64_t averageCol = 0
```

**16.307.1.14 deviationCol**

```
uint64_t deviationCol = 0
```

**16.307.1.15 minColDifference**

```
uint64_t minColDifference = 0
```

**16.307.1.16 maxColDifference**

```
uint64_t maxColDifference = 0
```

**16.307.1.17 averageColDifference**

```
uint64_t averageColDifference = 0
```

**16.307.1.18 deviationColDifference**

```
uint64_t deviationColDifference = 0
```

**16.307.1.19 minRowDifference**

```
uint64_t minRowDifference = 0
```

**16.307.1.20 maxRowDifference**

```
uint64_t maxRowDifference = 0
```

**16.307.1.21 averageRowDifference**

```
uint64_t averageRowDifference = 0
```

**16.307.1.22 deviationRowDifference**

```
uint64_t deviationRowDifference = 0
```

**16.307.1.23 nDenseRows**

```
uint64_t nDenseRows = 0
```

**16.307.1.24 nDenseCols**

```
uint64_t nDenseCols = 0
```

**16.307.1.25 nEmptyRows**

```
uint64_t nEmptyRows = 0
```

**16.307.1.26 nEmptyCols**

```
uint64_t nEmptyCols = 0
```

**16.307.1.27 nEmptyColsEnd**

```
uint64_t nEmptyColsEnd = 0
```

**16.307.1.28 denseRows**

```
std::vector<uint64_t> denseRows
```

**16.307.1.29 denseCols**

```
std::vector<uint64_t> denseCols
```

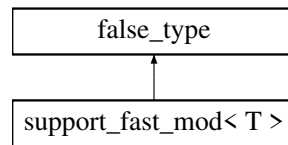
The documentation for this struct was generated from the following file:

- [utils.h](#)

**16.308 support\_fast\_mod< T > Struct Template Reference**

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< T >:



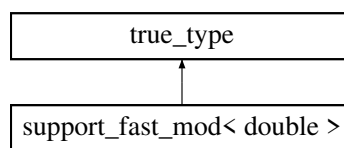
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

### 16.309 support\_fast\_mod< double > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< double >:



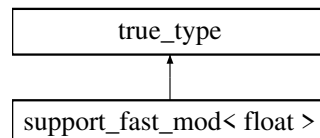
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

### 16.310 support\_fast\_mod< float > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< float >:



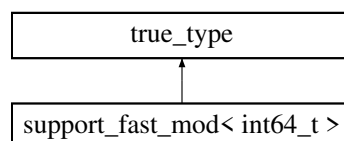
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

### 16.311 support\_fast\_mod< int64\_t > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< int64\_t >:



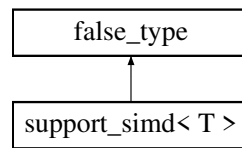
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.312 support\_simd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

Inheritance diagram for support\_simd< T >:



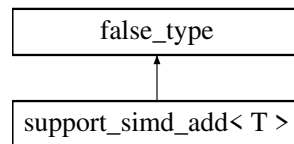
The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.313 support\_simd\_add< T > Struct Template Reference

```
#include <fflas_fadd.h>
```

Inheritance diagram for support\_simd\_add< T >:



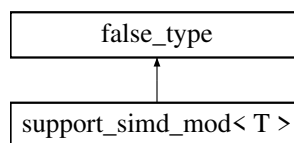
The documentation for this struct was generated from the following file:

- [fflas\\_fadd.h](#)

## 16.314 support\_simd\_mod< T > Struct Template Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_simd\_mod< T >:



The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.315 tfn\_minus Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\)(Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

## 16.315.1 Member Function Documentation

### 16.315.1.1 operator>()()

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.316 tfn\_minus\_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

## 16.316.1 Member Function Documentation

### 16.316.1.1 operator>()()

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.317 tfn\_mul Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

## 16.317.1 Member Function Documentation

### 16.317.1.1 operator>()()

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)



## 16.318 tfn\_mul\_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.318.1 Member Function Documentation

#### 16.318.1.1 `operator>()()`

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.319 tfn\_plus Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.319.1 Member Function Documentation

#### 16.319.1.1 `operator>()()`

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.320 tfn\_plus\_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.320.1 Member Function Documentation

### 16.320.1.1 operator>()

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.321 Threads Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.322 ThreeD Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.323 ThreeDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.324 ThreeDInPlace Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.325 TRSMHelper< ReclterTrait, ParSeqTrait > Struct Template Reference

TRSM Helper.

### Public Member Functions

- `template<class Cut , class Param >`  
[TRSMHelper](#) ([ParSeqHelper::Parallel](#)< Cut, Param > \_PS)
- `TRSMHelper` ([ParSeqHelper::Sequential](#) \_PS)
- `template<typename RIT , typename PST >`  
[TRSMHelper](#) ([TRSMHelper](#)< RIT, PST > &\_TH)
- `template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value>`  
[FFLAS::MMHelper](#)< Dom, Algo, ModeT, ParSeqTrait > [pMMH](#) (Dom &D, size\_t m, size\_t k, size\_t n, ParSeqTrait p) const
- `template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value>`  
[FFLAS::MMHelper](#)< Dom, Algo, ModeT, ParSeqTrait > [pMMH](#) (Dom &D, size\_t m, size\_t k, size\_t n) const

### Data Fields

- ParSeqTrait [parseq](#)

### 16.325.1 Detailed Description

```
template<typename ReclterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
struct FFLAS::TRSMHelper< ReclterTrait, ParSeqTrait >
```

TRSM Helper.

### 16.325.2 Constructor & Destructor Documentation

#### 16.325.2.1 TRSMHelper() [1/3]

```
TRSMHelper (
    ParSeqHelper::Parallel< Cut, Param > _PS ) [inline]
```

#### 16.325.2.2 TRSMHelper() [2/3]

```
TRSMHelper (
    ParSeqHelper::Sequential _PS ) [inline]
```

#### 16.325.2.3 TRSMHelper() [3/3]

```
TRSMHelper (
    TRSMHelper< RIT, PST > & _TH ) [inline]
```

### 16.325.3 Member Function Documentation

#### 16.325.3.1 pMMH() [1/2]

```
FFLAS::MMHelper<Dom, Algo, ModeT, ParSeqTrait> pMMH (
    Dom & D,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait p ) const [inline]
```

#### 16.325.3.2 pMMH() [2/2]

```
FFLAS::MMHelper<Dom, Algo, ModeT, ParSeqTrait> pMMH (
    Dom & D,
    size_t m,
    size_t k,
    size_t n ) const [inline]
```

### 16.325.4 Field Documentation

#### 16.325.4.1 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.326 TwoD Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.327 TwoDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.328 UnparametricTag Struct Reference

If the field uses a representation with infix operators.

```
#include <field-traits.h>
```

### 16.328.1 Detailed Description

If the field uses a representation with infix operators.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.329 Winograd Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.330 WinogradPar Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

# Chapter 17

## File Documentation

### 17.1 101-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.1.1 Function Documentation

##### 17.1.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.2 2x2-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.2.1 Function Documentation

#### 17.2.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.3 2x2-fftrsv.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
#include <array>
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.3.1 Function Documentation

##### 17.3.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.4 2x2-pluq.C File Reference

```
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.4.1 Function Documentation

##### 17.4.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.5 align-allocator.h File Reference

```
#include "fflas-ffpack/config.h"
```

## 17.6 args-parser.h File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/givinteger.h>
#include <givaro/givprint.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <cstring>
#include <list>
#include <stdlib.h>
```

### Data Structures

- struct [Argument](#)

### Namespaces

- [FFLAS](#)

### Macros

- #define [TYPE\\_BOOL](#) [TYPE\\_NONE](#)
- #define [END\\_OF\\_ARGUMENTS](#) { '\0', "\0", "\0", [TYPE\\_NONE](#), NULL }
- #define [type\\_integer](#) long int

### Enumerations

- enum [ArgumentType](#) {  
[TYPE\\_NONE](#) , [TYPE\\_INT](#) , [TYPE\\_UINT64](#) , [TYPE\\_LONGLONG](#) ,  
[TYPE\\_INTEGER](#) , [TYPE\\_DOUBLE](#) , [TYPE\\_INTLIST](#) , [TYPE\\_STR](#) }

### Functions

- void [parseArguments](#) (int argc, char \*\*argv, [Argument](#) \*args, bool printDefaults=true)
- void [printHelpMessage](#) (const char \*program, [Argument](#) \*args, bool printDefaults=false)
- [Argument](#) \* [findArgument](#) ([Argument](#) \*args, char c)
- int [getListArgs](#) (std::list< int > &outlist, std::string &instring)  
*transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}*
- std::ostream & [writeCommandString](#) (std::ostream &os, [Argument](#) \*args, const char \*programName=nullptr)  
*writes the values of all arguments, preceded by the programName*

### 17.6.1 Macro Definition Documentation

#### 17.6.1.1 TYPE\_BOOL

```
#define TYPE_BOOL TYPE\_NONE
```

### 17.6.1.2 END\_OF\_ARGUMENTS

```
#define END_OF_ARGUMENTS { '\0', "\0", "\0", TYPE_NONE, NULL }
```

### 17.6.1.3 type\_integer

```
#define type_integer long int
```

## 17.6.2 Enumeration Type Documentation

### 17.6.2.1 ArgumentType

```
enum ArgumentType
```

Enumerator

TYPE_NONE	
TYPE_INT	
TYPE_UINT64	
TYPE_LONGLONG	
TYPE_INTEGER	
TYPE_DOUBLE	
TYPE_INTLIST	
TYPE_STR	

## 17.6.3 Function Documentation

### 17.6.3.1 printHelpMessage()

```
void printHelpMessage (
    const char * program,
    Argument * args,
    bool printDefaults = false )
```

### 17.6.3.2 findArgument()

```
Argument* findArgument (
    Argument * args,
    char c )
```

### 17.6.3.3 getListArgs()

```
int getListArgs (
    std::list< int > & outlist,
    std::string & instring )
```

transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}

Parameters

<i>outlist</i>	list once converted
----------------	---------------------



## Parameters

<i>instring</i>	list to be converted
-----------------	----------------------

## Returns

status message.

## 17.7 arithprog.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))
- #define [GFOPS](#)(m, n, r, t) (2.7\*CUBE(double(n)/1000.0))/t

### Typedefs

- typedef Givaro::Timer [TTimer](#)

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.7.1 Macro Definition Documentation

### 17.7.1.1 CUBE

```
#define CUBE(
    x )  ((x)*(x)*(x))
```

### 17.7.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
    r,
    t )  (2.7*CUBE(double(n)/1000.0))/t
```

## 17.7.2 Typedef Documentation

### 17.7.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

### 17.7.3 Function Documentation

#### 17.7.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.8 benchmark-charpoly-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/Matio.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define \_\_FFLASFFPACK\_FORCE\_SEQ`

### Functions

- `int main (int argc, char **argv)`

### 17.8.1 Macro Definition Documentation

#### 17.8.1.1 [\\_\\_FFLASFFPACK\\_FORCE\\_SEQ](#)

```
#define \_\_FFLASFFPACK\_FORCE\_SEQ
```

### 17.8.2 Function Documentation

#### 17.8.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.9 benchmark-charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givpoly1.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

```
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `template<class Field >`  
`void run_with_field (int q, size_t bits, size_t n, size_t d, size_t iter, std::string file, int variant)`
- `int main (int argc, char **argv)`

### 17.9.1 Macro Definition Documentation

#### 17.9.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.9.2 Function Documentation

#### 17.9.2.1 run\_with\_field()

```
void run_with_field (
    int q,
    size_t bits,
    size_t n,
    size_t d,
    size_t iter,
    std::string file,
    int variant )
```

#### 17.9.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.10 benchmark-checkers.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include <fstream>
```

## Macros

- #define `ENABLE_ALL_CHECKINGS` 1
- #define `_NR_TESTS` 5
- #define `_MAX_SIZE_MATRICES` 1000
- #define `CUBE(x)`  $((x)*(x)*(x))$

## Functions

- int `main` (int argc, char \*\*argv)

## 17.10.1 Macro Definition Documentation

### 17.10.1.1 `ENABLE_ALL_CHECKINGS`

```
#define ENABLE_ALL_CHECKINGS 1
```

### 17.10.1.2 `_NR_TESTS`

```
#define _NR_TESTS 5
```

### 17.10.1.3 `_MAX_SIZE_MATRICES`

```
#define _MAX_SIZE_MATRICES 1000
```

### 17.10.1.4 `CUBE`

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

## 17.10.2 Function Documentation

### 17.10.2.1 `main()`

```
int main (  
    int argc,  
    char ** argv )
```

## 17.11 benchmark-dgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/config-blas.h"  
#include "fflas-ffpack/fflas/fflas.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/fflas_io.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- #define [CBLAS\\_GEMM](#) `cblas_dgemm`

## Typedefs

- typedef [FFLAS::Timer](#) `TTimer`
- typedef double [Floats](#)

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.11.1 Macro Definition Documentation

#### 17.11.1.1 CBLAS\_GEMM

```
#define CBLAS_GEMM cblas_dgemm
```

### 17.11.2 Typedef Documentation

#### 17.11.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

#### 17.11.2.2 Floats

```
typedef double Floats
```

### 17.11.3 Function Documentation

#### 17.11.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.12 benchmark-dgetrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- #define [\\_\\_FFLASFFPACK\\_HAVE\\_DGETRF](#) 1

## Typedefs

- typedef [FFLAS::Timer](#) TTimer

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.12.1 Macro Definition Documentation

#### 17.12.1.1 \_\_FFLASFFPACK\_HAVE\_DGETRF

```
#define __FFLASFFPACK_HAVE_DGETRF 1
```

### 17.12.2 Typedef Documentation

#### 17.12.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

### 17.12.3 Function Documentation

#### 17.12.3.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.13 benchmark-dgetri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <vector>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/fflas_io.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

## Typedefs

- typedef [FFLAS::Timer](#) TTimer

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.13.1 Typedef Documentation

### 17.13.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

## 17.13.2 Function Documentation

### 17.13.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.14 benchmark-dsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [EFFGFF](#)(n, t, i) ( (double(n)/1000.\*double(n)/1000.\*double(n)/1000.0) / double(t) \* double(i) / 3.)

### Typedefs

- typedef [FFLAS::Timer](#) TTimer

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.14.1 Macro Definition Documentation

### 17.14.1.1 EFFGFF

```
#define EFFGFF(
    n,
    t,
    i ) ( (double(n)/1000.*double(n)/1000.*double(n)/1000.0) / double(t) * double(i)
/ 3.)
```

## 17.14.2 Typedef Documentation

### 17.14.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

### 17.14.3 Function Documentation

#### 17.14.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.15 benchmark-dtrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Typedefs

- typedef [FFLAS::Timer](#) TTimer

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.15.1 Typedef Documentation

#### 17.15.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

### 17.15.2 Function Documentation

#### 17.15.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.16 benchmark-dtrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```



## Macros

- #define `__FFLASFFPACK_HAVE_DTRTRI` 1

## Typedefs

- typedef `FFLAS::Timer` `TTimer`

## Functions

- int `main` (int argc, char \*\*argv)

### 17.16.1 Macro Definition Documentation

#### 17.16.1.1 `__FFLASFFPACK_HAVE_DTRTRI`

```
#define __FFLASFFPACK_HAVE_DTRTRI 1
```

### 17.16.2 Typedef Documentation

#### 17.16.2.1 `TTimer`

```
typedef FFLAS::Timer TTimer
```

### 17.16.3 Function Documentation

#### 17.16.3.1 `main()`

```
int main (  
    int argc,  
    char ** argv )
```

## 17.17 benchmark-fadd-lvl2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/fflas_io.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- #define `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET` 1

## Functions

- int `main` (int argc, char \*\*argv)

## 17.17.1 Macro Definition Documentation

### 17.17.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.17.2 Function Documentation

### 17.17.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.18 benchmark-fdot.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givrational.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `template<class Field >`  
[Field::Element run\\_with\\_field](#) (int q, size\_t iter, size\_t N, const size\_t BS, const size\_t p, const size\_t threads)
- `int main` (int argc, char \*\*argv)

## 17.18.1 Macro Definition Documentation

### 17.18.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.18.2 Function Documentation

**17.18.2.1 run\_with\_field()**

```
Field::Element run_with_field (
    int q,
    size_t iter,
    size_t N,
    const size_t BS,
    const size_t p,
    const size_t threads )
```

**17.18.2.2 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.19 benchmark-fgemm-mp.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

**Macros**

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

**Functions**

- `template<typename Ints >`  
`int tmain ()`
- `int main (int argc, char **argv)`

**17.19.1 Macro Definition Documentation****17.19.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

**17.19.2 Function Documentation****17.19.2.1 tmain()**

```
int tmain ( )
```

### 17.19.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.20 benchmark-fgemm-rns.C File Reference

```
#include "fflas-ffpack/fflas/fflas.h"
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

### Typedefs

- typedef `FFPACK::rns_double` `RNS`
- typedef `FFPACK::RNSInteger< RNS >` `Field`
- typedef `Field::Element_ptr` `Element_ptr`
- typedef `Field::ConstElement_ptr` `ConstElement_ptr`
- typedef `StrategyParameter::Threads` `THREADS`
- typedef `StrategyParameter::Grain` `GRAIN`
- typedef `StrategyParameter::TwoD` `TWOD`
- typedef `StrategyParameter::TwoDAdaptive` `TWODA`
- typedef `StrategyParameter::ThreeD` `THREED`
- typedef `StrategyParameter::ThreeDAdaptive` `THREEDA`
- typedef `StrategyParameter::ThreeDInPlace` `THREEDIP`
- typedef `ParSeqHelper::Sequential` `PSeq`

### Functions

- int `main` (int argc, char \*argv[])

## 17.20.1 Macro Definition Documentation

### 17.20.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.20.2 Typedef Documentation

### 17.20.2.1 RNS

```
typedef FFPACK::rns_double RNS
```

### 17.20.2.2 Field

```
typedef FFPACK::RNSInteger<RNS> Field
```

### 17.20.2.3 Element\_ptr

```
typedef Field::Element_ptr Element_ptr
```

### 17.20.2.4 ConstElement\_ptr

```
typedef Field::ConstElement_ptr ConstElement_ptr
```

### 17.20.2.5 THREADS

```
typedef StrategyParameter::Threads THREADS
```

### 17.20.2.6 GRAIN

```
typedef StrategyParameter::Grain GRAIN
```

### 17.20.2.7 TWOD

```
typedef StrategyParameter::TwoD TWOD
```

### 17.20.2.8 TWODA

```
typedef StrategyParameter::TwoDAdaptive TWODA
```

### 17.20.2.9 THREED

```
typedef StrategyParameter::ThreeD THREED
```

### 17.20.2.10 THREEDA

```
typedef StrategyParameter::ThreeDAdaptive THREEDA
```

### 17.20.2.11 THREEDIP

```
typedef StrategyParameter::ThreeDInPlace THREEDIP
```

### 17.20.2.12 PSeq

```
typedef ParSeqHelper::Sequential PSeq
```

## 17.20.3 Function Documentation

### 17.20.3.1 main()

```
int main (  
    int argc,  
    char * argv[] )
```

## 17.21 benchmark-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [CLASSIC\\_HYBRID](#)

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.21.1 Macro Definition Documentation

### 17.21.1.1 CLASSIC\_HYBRID

```
#define CLASSIC_HYBRID
```

## 17.21.2 Function Documentation

### 17.21.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.22 benchmark-fgemv-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_OPENBLAS\\_NT\\_ALREADY\\_SET](#) 1

## Functions

- `template<typename T>`  
`std::ostream & write\_matrix (std::ostream &out, Givaro::Integer p, size_t m, size_t n, T *C, size_t ldc)`

### 17.22.1 Macro Definition Documentation

#### 17.22.1.1 `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET`

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.22.2 Function Documentation

#### 17.22.2.1 `write_matrix()`

```
std::ostream& write_matrix (
    std::ostream & out,
    Givaro::Integer p,
    size_t m,
    size_t n,
    T * C,
    size_t ldc )
```

## 17.23 benchmark-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
```

## Data Structures

- struct [need\\_field\\_characteristic](#)< Field >
- struct [need\\_field\\_characteristic](#)< Givaro::Modular< Field > >
- struct [need\\_field\\_characteristic](#)< Givaro::ModularBalanced< Field > >
- struct [compatible\\_data\\_type](#)< Field >
- struct [compatible\\_data\\_type](#)< Givaro::ZRing< float > >
- struct [compatible\\_data\\_type](#)< Givaro::ZRing< double > >

## Macros

- `#define \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET 1`

## Functions

- `template<class Field , class RandIter , class Matrix , class Vector >`  
`void fill_value (Field &F, RandIter &Rand, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK)`
- `template<class Field , class Matrix , class Vector >`  
`void genData (Field &F, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK, int bitsize, uint64_t seed)`
- `template<class Field , class Matrix , class Vector >`  
`bool check_result (Field &F, size_t m, size_t lda, Matrix &A, Vector &X, size_t incX, Vector &Y, size_t incY)`
- `template<class Field , class Matrix , class Vector >`  
`bool benchmark_with_timer (Field &F, int p, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, size_t iters, int t, double &time, size_t GrainSize)`
- `template<class Field , class arg >`  
`void benchmark_disp (Field &F, bool pass, double &time, size_t iters, int p, size_t m, size_t k, arg &as)`
- `template<class Field , class arg >`  
`void benchmark_in_Field (Field &F, int p, size_t m, size_t k, int NBK, int bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `template<class Field , class arg >`  
`void benchmark_with_field (int p, size_t m, size_t k, int NBK, int bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `template<class Field , class arg >`  
`void benchmark_with_field (const Givaro::Integer &q, int p, size_t m, size_t k, int NBK, int bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `int main (int argc, char **argv)`

## 17.23.1 Macro Definition Documentation

### 17.23.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.23.2 Function Documentation

### 17.23.2.1 fill\_value()

```
void fill_value (
    Field & F,
    RandIter & Rand,
    Matrix & A,
    Vector & X,
    Vector & Y,
    size_t m,
    size_t k,
    size_t incX,
    size_t incY,
    size_t lda,
    int NBK )
```

### 17.23.2.2 genData()

```
void genData (
    Field & F,
    Matrix & A,
```



```
Vector & X,  
Vector & Y,  
size_t m,  
size_t k,  
size_t incX,  
size_t incY,  
size_t lda,  
int NBK,  
int bitsize,  
uint64_t seed )
```

### 17.23.2.3 check\_result()

```
bool check_result (   
    Field & F,  
    size_t m,  
    size_t lda,  
    Matrix & A,  
    Vector & X,  
    size_t incX,  
    Vector & Y,  
    size_t incY )
```

### 17.23.2.4 benchmark\_with\_timer()

```
bool benchmark_with_timer (   
    Field & F,  
    int p,  
    Matrix & A,  
    Vector & X,  
    Vector & Y,  
    size_t m,  
    size_t k,  
    size_t incX,  
    size_t incY,  
    size_t lda,  
    size_t iters,  
    int t,  
    double & time,  
    size_t GrainSize )
```

### 17.23.2.5 benchmark\_disp()

```
void benchmark_disp (   
    Field & F,  
    bool pass,  
    double & time,  
    size_t iters,  
    int p,  
    size_t m,  
    size_t k,  
    arg & as )
```

### 17.23.2.6 benchmark\_in\_Field()

```
void benchmark_in_Field (
    Field & F,
    int p,
    size_t m,
    size_t k,
    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

### 17.23.2.7 benchmark\_with\_field() [1/2]

```
void benchmark_with_field (
    int p,
    size_t m,
    size_t k,
    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

### 17.23.2.8 benchmark\_with\_field() [2/2]

```
void benchmark_with_field (
    const Givaro::Integer & q,
    int p,
    size_t m,
    size_t k,
    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

### 17.23.2.9 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.24 benchmark-fgesv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
```

```
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.24.1 Macro Definition Documentation

#### 17.24.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.24.2 Function Documentation

#### 17.24.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.25 benchmark-fsyrrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`
- `#define CUBE(x) ((x)*(x)*(x))`

## Functions

- `int main (int argc, char **argv)`

### 17.25.1 Macro Definition Documentation

**17.25.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

**17.25.1.2 CUBE**

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

**17.25.2 Function Documentation****17.25.2.1 main()**

```
int main (  
    int argc,  
    char ** argv )
```

**17.26 benchmark-fsytrf.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/fflas_io.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

**Macros**

- [#define \\_\\_FFPACK\\_FSYTRF\\_BC\\_CROUT](#)
- [#define \\_\\_FFLASFFPACK\\_OPENBLAS\\_NT\\_ALREADY\\_SET 1](#)
- [#define CUBE\(x\) \(\(x\)\\*\(x\)\\*\(x\)\)](#)

**Functions**

- int [main](#) (int argc, char \*\*argv)

**17.26.1 Macro Definition Documentation****17.26.1.1 \_\_FFPACK\_FSYTRF\_BC\_CROUT**

```
#define __FFPACK_FSYTRF_BC_CROUT
```

**17.26.1.2 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

**17.26.1.3 CUBE**

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

## 17.26.2 Function Documentation

### 17.26.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.27 benchmark-fftrsm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

### Functions

- `int main (int argc, char **argv)`

## 17.27.1 Macro Definition Documentation

### 17.27.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.27.2 Function Documentation

### 17.27.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.28 benchmark-fftrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.28.1 Macro Definition Documentation

#### 17.28.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.28.2 Function Documentation

#### 17.28.2.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.29 benchmark-fftrsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.29.1 Macro Definition Documentation

#### 17.29.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.29.2 Function Documentation

**17.29.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.30 benchmark-fftrtri.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

**Macros**

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`
- `#define CUBE(x) ((x)*(x)*(x))`

**Functions**

- int `main` (int argc, char \*\*argv)

**17.30.1 Macro Definition Documentation****17.30.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

**17.30.1.2 CUBE**

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

**17.30.2 Function Documentation****17.30.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.31 benchmark-inverse.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
```

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define CUBE(x) ((x)*(x)*(x))`

## Functions

- `int main (int argc, char **argv)`

### 17.31.1 Macro Definition Documentation

#### 17.31.1.1 CUBE

```
#define CUBE(  
    x )  ( (x)*(x)*(x) )
```

### 17.31.2 Function Documentation

#### 17.31.2.1 main()

```
int main (  
    int  argc,  
    char ** argv )
```

## 17.32 benchmark-lqup-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <vector>  
#include <string>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "fflas-ffpack/utils/args-parser.h"  
#include "givaro/modular-integer.h"
```

## Functions

- `int main (int argc, char **argv)`

### 17.32.1 Function Documentation

#### 17.32.1.1 main()

```
int main (  
    int  argc,  
    char ** argv )
```



## 17.33 benchmark-lqup.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.33.1 Macro Definition Documentation

### 17.33.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

## 17.33.2 Function Documentation

### 17.33.2.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.34 benchmark-pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include <givaro/givranditer.h>
#include <iostream>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_OPENBLAS\\_NT\\_ALREADY\\_SET](#) 1
- #define [CUBE](#)(x) ((x)\*(x)\*(x))

## Typedefs

- typedef [Givaro::ModularBalanced](#)< double > [Field](#)

## Functions

- void [verification\\_PLUQ](#) (const [Field](#) &F, typename [Field::Element](#) \*B, typename [Field::Element](#) \*A, size\_t \*P, size\_t \*Q, size\_t m, size\_t n, size\_t R)
- void [Rec\\_Initialize](#) ([Field](#) &F, [Field::Element](#) \*C, size\_t m, size\_t n, size\_t ldc)
- int [main](#) (int argc, char \*\*argv)

## 17.34.1 Macro Definition Documentation

### 17.34.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.34.1.2 CUBE

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

## 17.34.2 Typedef Documentation

### 17.34.2.1 Field

```
typedef Givaro::ModularBalanced<double> Field
```

## 17.34.3 Function Documentation

### 17.34.3.1 verification\_PLUQ()

```
void verification_PLUQ (  
    const Field & F,  
    typename Field::Element * B,  
    typename Field::Element * A,  
    size_t * P,  
    size_t * Q,  
    size_t m,  
    size_t n,  
    size_t R )
```

### 17.34.3.2 Rec\_Initialize()

```
void Rec_Initialize (  
    Field & F,  
    Field::Element * C,  
    size_t m,  
    size_t n,  
    size_t ldc )
```

### 17.34.3.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.35 benchmark-wino.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))

### Functions

- template<class Field >  
void [launch\\_wino](#) (const [Field](#) &F, const size\_t &n, const size\_t &NB, const size\_t &wino, const bool &asmax,  
const size\_t &seed, const bool compare)
- int [main](#) (int argc, char \*\*argv)

## 17.35.1 Macro Definition Documentation

### 17.35.1.1 CUBE

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

## 17.35.2 Function Documentation

### 17.35.2.1 launch\_wino()

```
void launch_wino (
    const Field & F,
    const size_t & n,
    const size_t & NB,
    const size_t & wino,
    const bool & asmax,
    const size_t & seed,
    const bool compare )
```

### 17.35.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.36 bit\_manipulation.h File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
```

### Macros

- `#define __has_builtin(x) 0`

### Functions

- `int32_t clz (uint64_t val)`
- `int32_t clz (uint32_t val)`
- `int32_t ctz (uint32_t val)`
- `int32_t ctz (uint64_t val)`

### 17.36.1 Macro Definition Documentation

#### 17.36.1.1 \_\_has\_builtin

```
#define __has_builtin(  
    x ) 0
```

### 17.36.2 Function Documentation

#### 17.36.2.1 clz() [1/2]

```
int32_t clz (  
    uint64_t val ) [inline]
```

#### 17.36.2.2 clz() [2/2]

```
int32_t clz (  
    uint32_t val ) [inline]
```

#### 17.36.2.3 ctz() [1/2]

```
int32_t ctz (  
    uint32_t val ) [inline]
```

#### 17.36.2.4 ctz() [2/2]

```
int32_t ctz (  
    uint64_t val ) [inline]
```

## 17.37 blockcuts.inl File Reference

```
#include <fflas-ffpack/fflas/fflas_enum.h>
#include <math.h>
#include <cassert>
```

## Data Structures

- struct [Single](#)
- struct [Row](#)
- struct [Column](#)
- struct [Block](#)
- struct [Recursive](#)
- struct [Fixed](#)
- struct [Threads](#)
- struct [Grain](#)
- struct [TwoD](#)
- struct [TwoDAdaptive](#)
- struct [ThreeD](#)
- struct [ThreeDInPlace](#)
- struct [ThreeDAdaptive](#)
- struct [Parallel< C, P >](#)
- struct [Sequential](#)
- struct [Compose< H1, H2 >](#)
- struct [ForStrategy1D< blocksize\\_t, Cut, Param >](#)
- struct [ForStrategy2D< blocksize\\_t, Cut, Param >](#)

## Namespaces

- [FFLAS](#)
- [FFLAS::CuttingStrategy](#)
- [FFLAS::StrategyParameter](#)
- [FFLAS::ParSeqHelper](#)

*ParSeqHelper* for both *fgemm* and *ftsm*.

## Macros

- `#define __FFLASFFPACK_fflas_blockcuts_INL`
- `#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)`

## Typedefs

- typedef Row [RNSModulus](#)

## Functions

- `template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>  
void BlockCuts (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`

- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>  
void BlockCuts (size_t &rowBlockSize, size_t &colBlockSize, size_t &lastRBS, size_t &lastCBS, size_t &changeRBS, size_t &changeCBS, size_t &numRowBlock, size_t &numColBlock, size_t m, size_t n, const size_t numthreads)`

## 17.37.1 Macro Definition Documentation

### 17.37.1.1 `__FFLASFFPACK_fflas_blockcuts_INL`

```
#define __FFLASFFPACK_fflas_blockcuts_INL
```

### 17.37.1.2 `__FFLASFFPACK_MINBLOCKCUTS`

```
#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)
```

## 17.38 `cast.h` File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- `template<class T, class CT = const T>  
T fflas\_const\_cast (CT x)`

## 17.39 `cblas.C` File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- `#define \_\_FFLASFFPACK\_CONFIGURATION`
- `#define \_\_FFLASFFPACK\_HAVE\_CBLAS 1`

### Functions

- `int main ()`

## 17.39.1 Macro Definition Documentation

### 17.39.1.1 `__FFLASFFPACK_CONFIGURATION`

```
#define __FFLASFFPACK_CONFIGURATION
```

### 17.39.1.2 \_\_FFLASFFPACK\_HAVE\_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

## 17.39.2 Function Documentation

### 17.39.2.1 main()

```
int main (
    void )
```

## 17.40 charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))
- #define [GFOPS](#)(m, n, r, t) (2.7\*CUBE(double(n)/1000.0))/t

### Typedefs

- typedef Givaro::Timer [TTimer](#)

### Functions

- int [main](#) ()

## 17.40.1 Macro Definition Documentation

### 17.40.1.1 CUBE

```
#define CUBE(
    x ) ( (x) * (x) * (x) )
```

### 17.40.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
    r,
    t ) (2.7*CUBE(double(n)/1000.0))/t
```

## 17.40.2 Typedef Documentation

### 17.40.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.40.3 Function Documentation

### 17.40.3.1 main()

```
int main (
    void )
```

## 17.41 charpoly.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

*This example computes the characteristic polynomial of a matrix over a defined finite field.*

### 17.41.1 Function Documentation

#### 17.41.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the characteristic polynomial of a matrix over a defined finite field.  
Outputs the characteristic polynomial.

## 17.42 checker\_charpoly.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Data Structures

- class [CheckerImplem\\_charpoly](#)< Field, Polynomial >

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_charpoly\\_INL](#)

### 17.42.1 Macro Definition Documentation



#### 17.42.1.1 \_\_FFLASFFPACK\_checker\_charpoly\_INL

```
#define __FFLASFFPACK_checker_charpoly_INL
```

## 17.43 checker\_det.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Data Structures

- class [CheckerImplem\\_Det< Field >](#)

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_det\\_INL](#)

## 17.43.1 Macro Definition Documentation

#### 17.43.1.1 \_\_FFLASFFPACK\_checker\_det\_INL

```
#define __FFLASFFPACK_checker_det_INL
```

## 17.44 checker\_empty.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
```

### Data Structures

- struct [Checker\\_Empty< Field >](#)

### Namespaces

- [FFLAS](#)

## 17.45 checker\_fgemm.inl File Reference

### Data Structures

- class [CheckerImplem\\_fgemm< Field >](#)

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_fgemm\\_INL](#)

## 17.45.1 Macro Definition Documentation

### 17.45.1.1 \_\_FFLASFFPACK\_checker\_fgemm\_INL

```
#define __FFLASFFPACK_checker_fgemm_INL
```

## 17.46 checker\_ftrsm.inl File Reference

### Data Structures

- class [CheckerImplem\\_ftrsm< Field >](#)

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_ftrsm\\_INL](#)

## 17.46.1 Macro Definition Documentation

### 17.46.1.1 \_\_FFLASFFPACK\_checker\_ftrsm\_INL

```
#define __FFLASFFPACK_checker_ftrsm_INL
```

## 17.47 checker\_invert.inl File Reference

### Data Structures

- class [CheckerImplem\\_invert< Field >](#)

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_invert\\_INL](#)

## 17.47.1 Macro Definition Documentation

### 17.47.1.1 \_\_FFLASFFPACK\_checker\_invert\_INL

```
#define __FFLASFFPACK_checker_invert_INL
```

## 17.48 checker\_pluq.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

## Data Structures

- class [CheckerImplem\\_PLUQ< Field >](#)

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_checker\\_pluq\\_INL](#)

## 17.48.1 Macro Definition Documentation

### 17.48.1.1 \_\_FFLASFFPACK\_checker\_pluq\_INL

```
#define __FFLASFFPACK_checker_pluq_INL
```

## 17.49 checkers.doxy File Reference

## 17.50 checkers\_fflas.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/fflas/fflas_memory.h"
```

## Data Structures

- class [FailureFgemmCheck](#)
- class [FailureTrsmCheck](#)

## Namespaces

- [FFLAS](#)

## Typedefs

- `template<class Field >`  
using [Checker\\_fgemm](#) = [FFLAS::Checker\\_Empty< Field >](#)
- `template<class Field >`  
using [Checker\\_ftrsm](#) = [FFLAS::Checker\\_Empty< Field >](#)

## 17.51 checkers\_fflas.inl File Reference

```
#include "checker_fgemm.inl"
#include "checker_ftrsm.inl"
```

## Namespaces

- [FFLAS](#)

## Macros

- `#define FFLASFFPACK_checkers_fflas_inl_H`

## Typedefs

- `template<class Field >`  
using `ForceCheck_fgemm` = `CheckerImplem_fgemm< Field >`
- `template<class Field >`  
using `ForceCheck_ftrsm` = `CheckerImplem_ftrsm< Field >`

## 17.51.1 Macro Definition Documentation

### 17.51.1.1 FFLASFFPACK\_checkers\_fflas\_inl\_H

```
#define FFLASFFPACK_checkers_fflas_inl_H
```

## 17.52 checkers\_ffpack.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Data Structures

- class `FailurePLUQCheck`
- class `FailureDetCheck`
- class `FailureInvertCheck`
- class `FailureCharpolyCheck`

## Namespaces

- `FFPACK`

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Typedefs

- `template<class Field >`  
using `Checker_PLUQ` = `FFLAS::Checker_Empty< Field >`
- `template<class Field >`  
using `Checker_Det` = `FFLAS::Checker_Empty< Field >`
- `template<class Field >`  
using `Checker_invert` = `FFLAS::Checker_Empty< Field >`
- `template<class Field , class Polynomial >`  
using `Checker_charpoly` = `FFLAS::Checker_Empty< Field >`

## 17.53 checkers\_ffpack.inl File Reference

```
#include "checker_pluq.inl"
#include "checker_det.inl"
#include "checker_invert.inl"
#include "checker_charpoly.inl"
```

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define FFLASFFPACK_checkers_ffpack_inl_H`

## Typedefs

- `template<class Field >`  
using [ForceCheck\\_PLUQ](#) = CheckerImplem\_PLUQ< [Field](#) >
- `template<class Field >`  
using [ForceCheck\\_Det](#) = CheckerImplem\_Det< [Field](#) >
- `template<class Field >`  
using [ForceCheck\\_invert](#) = CheckerImplem\_invert< [Field](#) >
- `template<class Field , class Polynomial >`  
using [ForceCheck\\_charpoly](#) = CheckerImplem\_charpoly< [Field](#), Polynomial >

### 17.53.1 Macro Definition Documentation

#### 17.53.1.1 FFLASFFPACK\_checkers\_ffpack\_inl\_H

```
#define FFLASFFPACK_checkers_ffpack_inl_H
```

## 17.54 clapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

## Macros

- `#define __FFLASFFPACK_CONFIGURATION`
- `#define __FFLASFFPACK_HAVE_LAPACK 1`
- `#define __FFLASFFPACK_HAVE_CLAPACK 1`

## Functions

- `int main ()`

### 17.54.1 Macro Definition Documentation

#### 17.54.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

#### 17.54.1.2 \_\_FFLASFFPACK\_HAVE\_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

### 17.54.1.3 \_\_FFLASFFPACK\_HAVE\_CLAPACK

```
#define __FFLASFFPACK_HAVE_CLAPACK 1
```

## 17.54.2 Function Documentation

### 17.54.2.1 main()

```
int main (
    void )
```

## 17.55 config-blas.h File Reference

### Macros

- #define [CBLAS\\_INT](#) int
- #define [CBLAS\\_ENUM\\_DEFINED\\_H](#)
- #define [CBLAS\\_EXTERNALS](#)
- #define [blas\\_enum](#) enum

### Enumerations

- enum [CBLAS\\_ORDER](#) { [CblasRowMajor](#) =101 , [CblasColMajor](#) =102 }
- enum [CBLAS\\_TRANSPOSE](#) { [CblasNoTrans](#) =111 , [CblasTrans](#) =112 , [CblasConjTrans](#) =113 , [AtlasConj](#) =114 }
- enum [CBLAS\\_UPLO](#) { [CblasUpper](#) =121 , [CblasLower](#) =122 }
- enum [CBLAS\\_DIAG](#) { [CblasNonUnit](#) =131 , [CblasUnit](#) =132 }
- enum [CBLAS\\_SIDE](#) { [CblasLeft](#) =141 , [CblasRight](#) =142 }

### Functions

- void [daxpy\\_](#) (const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void [saxpy\\_](#) (const int \*, const float \*, const float \*, const int \*, float \*, const int \*)
- double [ddot\\_](#) (const int \*, const double \*, const int \*, const double \*, const int \*)
- float [sdot\\_](#) (const int \*, const float \*, const int \*, const float \*, const int \*)
- double [dasum\\_](#) (const int \*, const double \*, const int \*)
- int [idamax\\_](#) (const int \*, const double \*, const int \*)
- double [dnrm2\\_](#) (const int \*, const double \*, const int \*)
- void [dgemv\\_](#) (const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)
- void [sgemv\\_](#) (const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, const float \*, float \*, const int \*)
- void [dger\\_](#) (const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, double \*, const int \*)
- void [sger\\_](#) (const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, float \*, const int \*)
- void [dcopy\\_](#) (const int \*, const double \*, const int \*, double \*, const int \*)
- void [scopy\\_](#) (const int \*, const float \*, const int \*, float \*, const int \*)
- void [dscal\\_](#) (const int \*, const double \*, double \*, const int \*)
- void [sscal\\_](#) (const int \*, const float \*, float \*, const int \*)
- void [dtrsm\\_](#) (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void [strsm\\_](#) (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, float \*, const int \*)

- void [dtrmm\\_](#) (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void [strmm\\_](#) (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, float \*, const int \*)
- void [sgemm\\_](#) (const char \*, const char \*, const int \*, const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, const float \*, float \*, const int \*)
- void [dgemm\\_](#) (const char \*, const char \*, const int \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)

## 17.55.1 Macro Definition Documentation

### 17.55.1.1 CBLAS\_INT

```
#define CBLAS_INT int
```

### 17.55.1.2 CBLAS\_ENUM\_DEFINED\_H

```
#define CBLAS_ENUM_DEFINED_H
```

### 17.55.1.3 CBLAS\_EXTERNALS

```
#define CBLAS_EXTERNALS
```

### 17.55.1.4 blas\_enum

```
#define blas_enum enum
```

## 17.55.2 Enumeration Type Documentation

### 17.55.2.1 CBLAS\_ORDER

```
enum CBLAS\_ORDER
```

Enumerator

CblasRowMajor	
CblasColMajor	

### 17.55.2.2 CBLAS\_TRANSPOSE

```
enum CBLAS\_TRANSPOSE
```

Enumerator

CblasNoTrans	
CblasTrans	
CblasConjTrans	
AtlasConj	

### 17.55.2.3 CBLAS\_UPLO

enum [CBLAS\\_UPLO](#)

Enumerator

CblasUpper	
CblasLower	

### 17.55.2.4 CBLAS\_DIAG

enum [CBLAS\\_DIAG](#)

Enumerator

CblasNonUnit	
CblasUnit	

### 17.55.2.5 CBLAS\_SIDE

enum [CBLAS\\_SIDE](#)

Enumerator

CblasLeft	
CblasRight	

## 17.55.3 Function Documentation

### 17.55.3.1 daxpy\_()

```
void daxpy_ (
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

### 17.55.3.2 saxpy\_()

```
void saxpy_ (
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```



### 17.55.3.3 ddot\_()

```
double ddot_ (
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    const int * )
```

### 17.55.3.4 sdot\_()

```
float sdot_ (
    const int * ,
    const float * ,
    const int * ,
    const float * ,
    const int * )
```

### 17.55.3.5 dasum\_()

```
double dasum_ (
    const int * ,
    const double * ,
    const int * )
```

### 17.55.3.6 idamax\_()

```
int idamax_ (
    const int * ,
    const double * ,
    const int * )
```

### 17.55.3.7 dnorm2\_()

```
double dnorm2_ (
    const int * ,
    const double * ,
    const int * )
```

### 17.55.3.8 dgemv\_()

```
void dgemv_ (
    const char * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
```

```
double * ,  
const int * )
```

### 17.55.3.9 sgemv\_()

```
void sgemv_ (  
    const char * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    const float * ,  
    const int * ,  
    const float * ,  
    float * ,  
    const int * )
```

### 17.55.3.10 dger\_()

```
void dger_ (  
    const int * ,  
    const int * ,  
    const double * ,  
    const double * ,  
    const int * ,  
    const double * ,  
    const int * ,  
    double * ,  
    const int * )
```

### 17.55.3.11 sger\_()

```
void sger_ (  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    const float * ,  
    const int * ,  
    float * ,  
    const int * )
```

### 17.55.3.12 dcopy\_()

```
void dcopy_ (  
    const int * ,  
    const double * ,  
    const int * ,  
    double * ,  
    const int * )
```

**17.55.3.13 scopy\_()**

```
void scopy_ (
    const int * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

**17.55.3.14 dscal\_()**

```
void dscal_ (
    const int * ,
    const double * ,
    double * ,
    const int * )
```

**17.55.3.15 sscal\_()**

```
void sscal_ (
    const int * ,
    const float * ,
    float * ,
    const int * )
```

**17.55.3.16 dtrsm\_()**

```
void dtrsm_ (
    const char * ,
    const char * ,
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

**17.55.3.17 strsm\_()**

```
void strsm_ (
    const char * ,
    const char * ,
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

**17.55.3.18 dtrmm\_()**

```
void dtrmm_ (
    const char * ,
    const char * ,
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

**17.55.3.19 strmm\_()**

```
void strmm_ (
    const char * ,
    const char * ,
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

**17.55.3.20 sgemm\_()**

```
void sgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    const float * ,
    const int * ,
    const float * ,
    float * ,
    const int * )
```

**17.55.3.21 dgemm\_()**

```
void dgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const double * ,
```

```

const double * ,
const int * ,
const double * ,
const int * ,
const double * ,
double * ,
const int * )

```

## 17.56 config.h File Reference

### Macros

- #define [HAVE\\_BLAS](#) 1
- #define [HAVE\\_CBLAS](#) 1
- #define [HAVE\\_CXX11](#) 1
- #define [HAVE\\_DLFCN\\_H](#) 1
- #define [HAVE\\_FLOAT\\_H](#) 1
- #define [HAVE\\_INT128](#) 1
- #define [HAVE\\_INTPYPES\\_H](#) 1
- #define [HAVE\\_LAPACK](#) 1
- #define [HAVE\\_LIMITS\\_H](#) 1
- #define [HAVE\\_LITTLE\\_ENDIAN](#) 1
- #define [HAVE\\_PTHREAD\\_H](#) 1
- #define [HAVE\\_STDDEF\\_H](#) 1
- #define [HAVE\\_STDINT\\_H](#) 1
- #define [HAVE\\_STDIO\\_H](#) 1
- #define [HAVE\\_STDLIB\\_H](#) 1
- #define [HAVE\\_STRINGS\\_H](#) 1
- #define [HAVE\\_STRING\\_H](#) 1
- #define [HAVE\\_SYS\\_STAT\\_H](#) 1
- #define [HAVE\\_SYS\\_TIME\\_H](#) 1
- #define [HAVE\\_SYS\\_TYPES\\_H](#) 1
- #define [HAVE\\_UNISTD\\_H](#) 1
- #define [LT\\_OBJDIR](#) ".libs/"
- #define [OPENBLAS\\_NUM\\_THREADS](#) 1
- #define [PACKAGE](#) "fflas-ffpack"
- #define [PACKAGE\\_BUGREPORT](#) "ffpack-devel@googlegroups.com"
- #define [PACKAGE\\_NAME](#) "FFLAS-FFPACK"
- #define [PACKAGE\\_STRING](#) "FFLAS-FFPACK 2.4.3"
- #define [PACKAGE\\_TARNAME](#) "fflas-ffpack"
- #define [PACKAGE\\_URL](#) "https://github.com/linbox-team/fflas-ffpack"
- #define [PACKAGE\\_VERSION](#) "2.4.3"
- #define [SIZEOF\\_CHAR](#) 1
- #define [SIZEOF\\_INT](#) 4
- #define [SIZEOF\\_LONG](#) 8
- #define [SIZEOF\\_LONG\\_LONG](#) 8
- #define [SIZEOF\\_SHORT](#) 2
- #define [SIZEOF\\_\\_\\_INT64](#) 0
- #define [STDC\\_HEADERS](#) 1
- #define [USE\\_OPENMP](#) 1
- #define [VERSION](#) "2.4.3"

### 17.56.1 Macro Definition Documentation

**17.56.1.1 HAVE\_BLAS**

```
#define HAVE_BLAS 1
```

**17.56.1.2 HAVE\_CBLAS**

```
#define HAVE_CBLAS 1
```

**17.56.1.3 HAVE\_CXX11**

```
#define HAVE_CXX11 1
```

**17.56.1.4 HAVE\_DLFCN\_H**

```
#define HAVE_DLFCN_H 1
```

**17.56.1.5 HAVE\_FLOAT\_H**

```
#define HAVE_FLOAT_H 1
```

**17.56.1.6 HAVE\_INT128**

```
#define HAVE_INT128 1
```

**17.56.1.7 HAVE\_INTPYPES\_H**

```
#define HAVE_INTPYPES_H 1
```

**17.56.1.8 HAVE\_LAPACK**

```
#define HAVE_LAPACK 1
```

**17.56.1.9 HAVE\_LIMITS\_H**

```
#define HAVE_LIMITS_H 1
```

**17.56.1.10 HAVE\_LITTLE\_ENDIAN**

```
#define HAVE_LITTLE_ENDIAN 1
```

**17.56.1.11 HAVE\_PTHREAD\_H**

```
#define HAVE_PTHREAD_H 1
```

**17.56.1.12 HAVE\_STDDEF\_H**

```
#define HAVE_STDDEF_H 1
```

**17.56.1.13 HAVE\_STDINT\_H**

```
#define HAVE_STDINT_H 1
```

**17.56.1.14 HAVE\_STDIO\_H**

```
#define HAVE_STDIO_H 1
```

**17.56.1.15 HAVE\_STDLIB\_H**

```
#define HAVE_STDLIB_H 1
```

**17.56.1.16 HAVE\_STRINGS\_H**

```
#define HAVE_STRINGS_H 1
```

**17.56.1.17 HAVE\_STRING\_H**

```
#define HAVE_STRING_H 1
```

**17.56.1.18 HAVE\_SYS\_STAT\_H**

```
#define HAVE_SYS_STAT_H 1
```

**17.56.1.19 HAVE\_SYS\_TIME\_H**

```
#define HAVE_SYS_TIME_H 1
```

**17.56.1.20 HAVE\_SYS\_TYPES\_H**

```
#define HAVE_SYS_TYPES_H 1
```

**17.56.1.21 HAVE\_UNISTD\_H**

```
#define HAVE_UNISTD_H 1
```

**17.56.1.22 LT\_OBJDIR**

```
#define LT_OBJDIR ".libs/"
```

**17.56.1.23 OPENBLAS\_NUM\_THREADS**

```
#define OPENBLAS_NUM_THREADS 1
```

**17.56.1.24 PACKAGE**

```
#define PACKAGE "fflas-ffpack"
```

**17.56.1.25 PACKAGE\_BUGREPORT**

```
#define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

**17.56.1.26 PACKAGE\_NAME**

```
#define PACKAGE_NAME "FFLAS-FFPACK"
```

**17.56.1.27 PACKAGE\_STRING**

```
#define PACKAGE_STRING "FFLAS-FFPACK 2.4.3"
```

**17.56.1.28 PACKAGE\_TARNAME**

```
#define PACKAGE_TARNAME "fflas-ffpack"
```

**17.56.1.29 PACKAGE\_URL**

```
#define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

**17.56.1.30 PACKAGE\_VERSION**

```
#define PACKAGE_VERSION "2.4.3"
```

**17.56.1.31 SIZEOF\_CHAR**

```
#define SIZEOF_CHAR 1
```

**17.56.1.32 SIZEOF\_INT**

```
#define SIZEOF_INT 4
```

**17.56.1.33 SIZEOF\_LONG**

```
#define SIZEOF_LONG 8
```

**17.56.1.34 SIZEOF\_LONG\_LONG**

```
#define SIZEOF_LONG_LONG 8
```

**17.56.1.35 SIZEOF\_SHORT**

```
#define SIZEOF_SHORT 2
```

**17.56.1.36 SIZEOF\_\_INT64**

```
#define SIZEOF__INT64 0
```



**17.56.1.37 STDC\_HEADERS**

```
#define STDC_HEADERS 1
```

**17.56.1.38 USE\_OPENMP**

```
#define USE_OPENMP 1
```

**17.56.1.39 VERSION**

```
#define VERSION "2.4.3"
```

**17.57 config.h File Reference****Macros**

- [#define \\_\\_FFLASFFPACK\\_HAVE\\_BLAS 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_CBLAS 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_CXX11 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_DLFCN\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_FLOAT\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_INT128 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_INTPTR\\_T 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LAPACK 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LIMITS\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LITTLE\\_ENDIAN 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_PTHREAD\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STDDEF\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STDINT\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STDIO\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STDLIB\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STRINGS\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STRING\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_STAT\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_TIME\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_TYPES\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_UNISTD\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_LT\\_OBJDIR ".libs/"](#)
- [#define \\_\\_FFLASFFPACK\\_OPENBLAS\\_NUM\\_THREADS 1](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE "fflas-ffpack"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_BUGREPORT "ffpack-devel@googlegroups.com"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_NAME "FFLAS-FFPACK"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_STRING "FFLAS-FFPACK 2.4.3"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_TARNAME "fflas-ffpack"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_URL "https://github.com/linbox-team/fflas-ffpack"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_VERSION "2.4.3"](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_CHAR 1](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_INT 4](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_LONG 8](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_LONG\\_LONG 8](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_SHORT 2](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_\\_INT64 0](#)
- [#define \\_\\_FFLASFFPACK\\_STDC\\_HEADERS 1](#)
- [#define \\_\\_FFLASFFPACK\\_USE\\_OPENMP 1](#)
- [#define \\_\\_FFLASFFPACK\\_VERSION "2.4.3"](#)

## 17.57.1 Macro Definition Documentation

### 17.57.1.1 `__FFLASFFPACK_HAVE_BLAS`

```
#define __FFLASFFPACK_HAVE_BLAS 1
```

### 17.57.1.2 `__FFLASFFPACK_HAVE_CBLAS`

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

### 17.57.1.3 `__FFLASFFPACK_HAVE_CXX11`

```
#define __FFLASFFPACK_HAVE_CXX11 1
```

### 17.57.1.4 `__FFLASFFPACK_HAVE_DLFCN_H`

```
#define __FFLASFFPACK_HAVE_DLFCN_H 1
```

### 17.57.1.5 `__FFLASFFPACK_HAVE_FLOAT_H`

```
#define __FFLASFFPACK_HAVE_FLOAT_H 1
```

### 17.57.1.6 `__FFLASFFPACK_HAVE_INT128`

```
#define __FFLASFFPACK_HAVE_INT128 1
```

### 17.57.1.7 `__FFLASFFPACK_HAVE_INTTYPES_H`

```
#define __FFLASFFPACK_HAVE_INTTYPES_H 1
```

### 17.57.1.8 `__FFLASFFPACK_HAVE_LAPACK`

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

### 17.57.1.9 `__FFLASFFPACK_HAVE_LIMITS_H`

```
#define __FFLASFFPACK_HAVE_LIMITS_H 1
```

### 17.57.1.10 `__FFLASFFPACK_HAVE_LITTLE_ENDIAN`

```
#define __FFLASFFPACK_HAVE_LITTLE_ENDIAN 1
```

### 17.57.1.11 `__FFLASFFPACK_HAVE_PTHREAD_H`

```
#define __FFLASFFPACK_HAVE_PTHREAD_H 1
```

**17.57.1.12 \_\_FFLASFFPACK\_HAVE\_STDDEF\_H**

```
#define __FFLASFFPACK_HAVE_STDDEF_H 1
```

**17.57.1.13 \_\_FFLASFFPACK\_HAVE\_STDINT\_H**

```
#define __FFLASFFPACK_HAVE_STDINT_H 1
```

**17.57.1.14 \_\_FFLASFFPACK\_HAVE\_STDIO\_H**

```
#define __FFLASFFPACK_HAVE_STDIO_H 1
```

**17.57.1.15 \_\_FFLASFFPACK\_HAVE\_STDLIB\_H**

```
#define __FFLASFFPACK_HAVE_STDLIB_H 1
```

**17.57.1.16 \_\_FFLASFFPACK\_HAVE\_STRINGS\_H**

```
#define __FFLASFFPACK_HAVE_STRINGS_H 1
```

**17.57.1.17 \_\_FFLASFFPACK\_HAVE\_STRING\_H**

```
#define __FFLASFFPACK_HAVE_STRING_H 1
```

**17.57.1.18 \_\_FFLASFFPACK\_HAVE\_SYS\_STAT\_H**

```
#define __FFLASFFPACK_HAVE_SYS_STAT_H 1
```

**17.57.1.19 \_\_FFLASFFPACK\_HAVE\_SYS\_TIME\_H**

```
#define __FFLASFFPACK_HAVE_SYS_TIME_H 1
```

**17.57.1.20 \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H**

```
#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1
```

**17.57.1.21 \_\_FFLASFFPACK\_HAVE\_UNISTD\_H**

```
#define __FFLASFFPACK_HAVE_UNISTD_H 1
```

**17.57.1.22 \_\_FFLASFFPACK\_LT\_OBJDIR**

```
#define __FFLASFFPACK_LT_OBJDIR ".libs/"
```

**17.57.1.23 \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS**

```
#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1
```

**17.57.1.24 \_\_FFLASFFPACK\_PACKAGE**

```
#define __FFLASFFPACK_PACKAGE "fflas-ffpack"
```

**17.57.1.25 \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT**

```
#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

**17.57.1.26 \_\_FFLASFFPACK\_PACKAGE\_NAME**

```
#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"
```

**17.57.1.27 \_\_FFLASFFPACK\_PACKAGE\_STRING**

```
#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.4.3"
```

**17.57.1.28 \_\_FFLASFFPACK\_PACKAGE\_TARNAME**

```
#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
```

**17.57.1.29 \_\_FFLASFFPACK\_PACKAGE\_URL**

```
#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

**17.57.1.30 \_\_FFLASFFPACK\_PACKAGE\_VERSION**

```
#define __FFLASFFPACK_PACKAGE_VERSION "2.4.3"
```

**17.57.1.31 \_\_FFLASFFPACK\_SIZEOF\_CHAR**

```
#define __FFLASFFPACK_SIZEOF_CHAR 1
```

**17.57.1.32 \_\_FFLASFFPACK\_SIZEOF\_INT**

```
#define __FFLASFFPACK_SIZEOF_INT 4
```

**17.57.1.33 \_\_FFLASFFPACK\_SIZEOF\_LONG**

```
#define __FFLASFFPACK_SIZEOF_LONG 8
```

**17.57.1.34 \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG**

```
#define __FFLASFFPACK_SIZEOF_LONG_LONG 8
```

**17.57.1.35 \_\_FFLASFFPACK\_SIZEOF\_SHORT**

```
#define __FFLASFFPACK_SIZEOF_SHORT 2
```

**17.57.1.36 \_\_FFLASFFPACK\_SIZEOF\_\_INT64**

```
#define __FFLASFFPACK_SIZEOF__INT64 0
```

**17.57.1.37 \_\_FFLASFFPACK\_STDC\_HEADERS**

```
#define __FFLASFFPACK_STDC_HEADERS 1
```

**17.57.1.38 \_\_FFLASFFPACK\_USE\_OPENMP**

```
#define __FFLASFFPACK_USE_OPENMP 1
```

**17.57.1.39 \_\_FFLASFFPACK\_VERSION**

```
#define __FFLASFFPACK_VERSION "2.4.3"
```

**17.58 coo.h File Reference**

```
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmv.inl"
```

**Data Structures**

- struct [Sparse< \\_Field, SparseMatrix\\_t::COO >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::COO\\_ZO >](#)

**Namespaces**

- [FFLAS](#)

**Functions**

- template<class Field , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::COO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class Field , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::COO > &A)
- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A)

**17.59 coo\_spmv.inl File Reference****Namespaces**

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

**Macros**

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_coo\\_spmv\\_INL](#)

## Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

## 17.59.1 Macro Definition Documentation

### 17.59.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

## 17.60 coo\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)

- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_coo\\_spmv\\_INL](#)

## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 17.60.1 Macro Definition Documentation

#### 17.60.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

## 17.61 coo\_utils.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_coo\\_utils\\_INL](#)

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::COO > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::COO_ZO > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.61.1 Macro Definition Documentation

### 17.61.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_utils_INL
```

## 17.62 csr.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
```

### Data Structures

- struct `Sparse< _Field, SparseMatrix_t::CSR >`
- struct `Sparse< _Field, SparseMatrix_t::CSR_ZO >`

### Namespaces

- `FFLAS`

### Functions

- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)`

## 17.63 csr\_hyb.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmv.inl"
```

### Data Structures

- struct `Sparse< _Field, SparseMatrix_t::CSR_HYB >`

### Namespaces

- `FFLAS`



## Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.64 csr\_hyb\_pspmm.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL`

### Functions

- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`

### 17.64.1 Macro Definition Documentation

#### 17.64.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL
```

## 17.65 csr\_hyb\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL`

## Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`

## 17.65.1 Macro Definition Documentation

### 17.65.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL
```

## 17.66 csr\_hyb\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`

## 17.66.1 Macro Definition Documentation

### 17.66.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

## 17.67 csr\_hyb\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_spmv\\_INL](#)

### Functions

- [template<class Field >  
void fspmv \(const \[Field\]\(#\) &F, const Sparse< \[Field\]\(#\), SparseMatrix\\_t::CSR\\_HYB > &A, typename \[Field::ConstElement\\\_ptr\]\(#\) x\\_, typename \[Field::Element\\\_ptr\]\(#\) y\\_, FieldCategories::GenericTag\)](#)
- [template<class Field >  
void fspmv \(const \[Field\]\(#\) &F, const Sparse< \[Field\]\(#\), SparseMatrix\\_t::CSR\\_HYB > &A, typename \[Field::ConstElement\\\_ptr\]\(#\) x\\_, typename \[Field::Element\\\_ptr\]\(#\) y\\_, FieldCategories::UnparametricTag\)](#)
- [template<class Field >  
void fspmv \(const \[Field\]\(#\) &F, const Sparse< \[Field\]\(#\), SparseMatrix\\_t::CSR\\_HYB > &A, typename \[Field::ConstElement\\\_ptr\]\(#\) x\\_, typename \[Field::Element\\\_ptr\]\(#\) y\\_, const \[uint64\\\_t\]\(#\) kmax\)](#)

### 17.67.1 Macro Definition Documentation

#### 17.67.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

## 17.68 csr\_hyb\_utils.inl File Reference

### Data Structures

- [struct Info](#)
- [struct Coo< ValT, IdxT >](#)

### Namespaces

- [FFLAS](#)
- [FFLAS::csr\\_hyb\\_details](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_utils\\_INL](#)

### Functions

- [template<class Field >  
void sparse\\_delete \(const Sparse< \[Field\]\(#\), SparseMatrix\\_t::CSR\\_HYB > &A\)](#)
- [template<class Field, class IndexT >  
void sparse\\_init \(const \[Field\]\(#\) &F, Sparse< \[Field\]\(#\), SparseMatrix\\_t::CSR\\_HYB > &A, const IndexT \\*row, const IndexT \\*col, typename \[Field::ConstElement\\\_ptr\]\(#\) dat, \[uint64\\\_t\]\(#\) rowdim, \[uint64\\\_t\]\(#\) coldim, \[uint64\\\_t\]\(#\) nnz\)](#)

### 17.68.1 Macro Definition Documentation

### 17.68.1.1 \_\_FflasFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL

```
#define __FflasFFPACK_fflas_sparse_CSR_HYB_utils_INL
```

## 17.69 csr\_pspmm.inl File Reference

### Namespaces

- [Fflas](#)
- [Fflas::sparse\\_details\\_impl](#)

### Macros

- `#define __FflasFFPACK_fflas_sparse_CSR_pspmm_INL`

### Functions

- `template<class Field >`  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::GenericTag)
- `template<class Field >`  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::UnparametricTag)
- `template<class Field >`  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, const [int64\\_t](#) kmax)
- `template<class Field >`  
void [pfspmm\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR\_ZO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::GenericTag)
- `template<class Field >`  
void [pfspmm\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR\_ZO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::GenericTag)
- `template<class Field >`  
void [pfspmm\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR\_ZO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::UnparametricTag)
- `template<class Field >`  
void [pfspmm\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR\_ZO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::UnparametricTag)

### 17.69.1 Macro Definition Documentation

#### 17.69.1.1 \_\_FflasFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL

```
#define __FflasFFPACK_fflas_sparse_CSR_pspmm_INL
```

## 17.70 csr\_pspmv.inl File Reference

```
#include <thread>
```

## Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_pspmv\\_INL](#)

## Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.70.1 Macro Definition Documentation

### 17.70.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL
```

## 17.71 csr\_spmv.inl File Reference

## Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_spmv\\_INL](#)

## Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t blockSize, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

### 17.71.1 Macro Definition Documentation

#### 17.71.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

## 17.72 csr\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)

- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL`

## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 17.72.1 Macro Definition Documentation

#### 17.72.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

## 17.73 csr\_utils.inl File Reference

### Namespaces

- [FFLAS](#)

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)`
- `template<class Field >`  
`std::ostream & sparse_print (std::ostream &os, const Sparse< Field, SparseMatrix_t::CSR > &A)`
- `template<class IndexT >`  
`void sparse_init (const Givaro::Modular< Givaro::Integer > &F, Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

- `template<class IndexT >`  
`void sparse_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >> &F, Sparse< Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >> &F, Sparse< Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< Reclnt::rmint< RECINT_SIZE >>::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.74 cuda.C File Reference

```
#include <stdio.h>
#include <cuda_runtime.h>
#include <cusparse.h>
```

### Functions

- `int main ()`

### 17.74.1 Function Documentation

#### 17.74.1.1 main()

```
int main (
    void )
```

## 17.75 debug.h File Reference

Various utilities for debugging.

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <iostream>
#include <sstream>
#include <cmath>
#include <stdexcept>
```

### Data Structures

- class `Failure`  
*A precondition failed.*



## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define FFLASFFPACK_check(check)`
- `#define FFLASFFPACK_abort(msg)`

## Functions

- Failure & [failure](#) ()
- `template<class T >`  
`bool isOdd (const T &a)`
- `bool isOdd (const float &a)`
- `bool isOdd (const double &a)`

### 17.75.1 Detailed Description

Various utilities for debugging.

**Todo** we should put vector printing elsewhere.

### 17.75.2 Macro Definition Documentation

#### 17.75.2.1 FFLASFFPACK\_check

```
#define FFLASFFPACK_check(  
    check )
```

**Value:**

```
if (!(check)) {\n    FFPACK::failure() (__func__, __FILE__, __LINE__, #check); \n    throw std::runtime_error(#check); \n}
```

#### 17.75.2.2 FFLASFFPACK\_abort

```
#define FFLASFFPACK_abort(  
    msg )
```

**Value:**

```
{\n    FFPACK::failure() (__func__, __FILE__, __LINE__, msg); \n    throw std::runtime_error(msg); \n}
```

## 17.76 det.C File Reference

```
#include <givaro/modular.h>\n#include <iostream>\n#include "fflas-ffpack/fflas-ffpack-config.h"\n#include "fflas-ffpack/fflas-ffpack.h"\n#include "fflas-ffpack/utils/fflas_io.h"
```

## Functions

- `int main (int argc, char **argv)`

*This example computes the determinant of a matrix over a defined finite field.*

## 17.76.1 Function Documentation

### 17.76.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the determinant of a matrix over a defined finite field.  
Outputs the determinant.

## 17.77 ell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmmm.inl"
```

## Data Structures

- struct `Sparse< _Field, SparseMatrix_t::ELL >`
- struct `Sparse< _Field, SparseMatrix_t::ELL_ZO >`

## Namespaces

- `FFLAS`

## Functions

- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)`

## 17.78 ell\_pspmm.inl File Reference

## Namespaces

- `FFLAS`
- `FFLAS::sparse_details_impl`

## Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL`

## Functions

- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`
- `template<class Field, class Func >`  
`void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, Func &&func)`
- `template<class Field, class Func >`  
`void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)`

### 17.78.1 Macro Definition Documentation

#### 17.78.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL
```

## 17.79 ell\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL`

### Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`

- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.79.1 Macro Definition Documentation

### 17.79.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL
```

## 17.80 ell\_simd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_spmv.inl"
```

### Data Structures

- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#)

### Namespaces

- [FFLAS](#)

### Functions

- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)`

## 17.81 ell\_simd\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

## Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL`

## Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 17.81.1 Macro Definition Documentation

#### 17.81.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL
```

## 17.82 ell\_simd\_spmv.inl File Reference

### Namespaces

- `FFLAS`
- `FFLAS::sparse_details_impl`

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.82.1 Macro Definition Documentation

### 17.82.1.1 \_\_FflasFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL

```
#define __FflasFPACK_fflas_sparse_ELL_simd_spmv_INL
```

## 17.83 ell\_simd\_utils.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FflasFPACK_fflas_sparse_ELL_simd_utils_INL`

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.83.1 Macro Definition Documentation

### 17.83.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL
```

## 17.84 ell\_spmmm.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_spmmm\\_INL](#)

### Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A,`

```
size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy,
FieldCategories::UnparametricTag)
```

- `template<class Field >`  
`void fspmm\_mone\_simd\_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A,`  
`size_t blockSize, typename Field::ConstElement\_ptr x_, int ldx, typename Field::Element\_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`

## 17.84.1 Macro Definition Documentation

### 17.84.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_spmv\\_INL](#)

```
#define \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL
```

## 17.85 [ell\\_spmv.inl](#) File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_spmv\\_INL](#)

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement\_ptr`  
`x_, typename Field::Element\_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement\_ptr`  
`x_, typename Field::Element\_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement\_ptr`  
`x_, typename Field::Element\_ptr y_, const uint64\_t kmax)`
- `template<class Field >`  
`void fspmv\_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement\_ptr x_, typename Field::Element\_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv\_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement\_ptr x_, typename Field::Element\_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv\_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement\_ptr x_, typename Field::Element\_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv\_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename`  
`Field::ConstElement\_ptr x_, typename Field::Element\_ptr y_, FieldCategories::UnparametricTag)`

## 17.85.1 Macro Definition Documentation

### 17.85.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_spmv\\_INL](#)

```
#define \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL
```



## 17.86 ell\_utils.inl File Reference

```
#include <vector>
```

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_utils\\_INL](#)

### Functions

- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL > &A)
- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_ZO > &A)
- template<class Field , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class Field , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

### 17.86.1 Macro Definition Documentation

#### 17.86.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_utils\\_INL](#)

```
#define __FFLASFFPACK_fflas_sparse_ELL_utils_INL
```

## 17.87 fblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)

### Functions

- void [dgemm\\_](#) (const char \*, const char \*, const int \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)
- int [main](#) ()

### 17.87.1 Macro Definition Documentation

#### 17.87.1.1 [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)

```
#define __FFLASFFPACK_CONFIGURATION
```

## 17.87.2 Function Documentation

### 17.87.2.1 dgemm\_()

```
void dgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    double * ,
    const int * )
```

### 17.87.2.2 main()

```
int main (
    void )
```

## 17.88 fflas-101\_1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.88.1 Function Documentation

### 17.88.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.89 fflas-101\_3.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

```
#include <iostream>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.89.1 Function Documentation

#### 17.89.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.90 fflas-ffpack-config.h File Reference

Defaults for optimised values.

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/fflas-ffpack-thresholds.h"
#include "fflas-ffpack/fflas-ffpack-default-thresholds.h"
#include "givaro/givconfig.h"
```

## Macros

- #define [GCC\\_VERSION](#) (\_\_GNUC\_\_ \* 10000 + \_\_GNUC\_MINOR\_\_ \* 100 + \_\_GNUC\_PATCHLEVEL\_\_)

### 17.90.1 Detailed Description

Defaults for optimised values.

While `fflas-ffpack-optimize.h` is created by `configure` script, (either left blank or filled by optimiser), this file produces the defaults for the optimised values. If `fflas-ffpack-optimize.h` is not empty, then its values preceeds the defaults here.

### 17.90.2 Macro Definition Documentation

#### 17.90.2.1 GCC\_VERSION

```
#define GCC_VERSION (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)
```

## 17.91 fflas-ffpack-default-thresholds.h File Reference

## Macros

- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD](#) 1000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_FLT](#) 2000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_BAL](#) 1000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_BAL\\_FLT](#) 2000
- #define [\\_\\_FFLASFFPACK\\_PLUQ\\_THRESHOLD](#) 256
- #define [\\_\\_FFLASFFPACK\\_CHARPOLY\\_LUKrylov\\_ArithProg\\_THRESHOLD](#) 1000
- #define [\\_\\_FFLASFFPACK\\_CHARPOLY\\_Danilevskii\\_LUKrylov\\_THRESHOLD](#) 16
- #define [\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#) 30

- `#define __FFLASFFPACK_FTRTRI_THRESHOLD 32`
- `#define __FFLASFFPACK_FSYTRF_THRESHOLD 64`
- `#define __FFLASFFPACK_FSYRK_THRESHOLD 3000`

## 17.91.1 Macro Definition Documentation

### 17.91.1.1 `__FFLASFFPACK_WINOTHRESHOLD`

```
#define __FFLASFFPACK_WINOTHRESHOLD 1000
```

### 17.91.1.2 `__FFLASFFPACK_WINOTHRESHOLD_FLT`

```
#define __FFLASFFPACK_WINOTHRESHOLD_FLT 2000
```

### 17.91.1.3 `__FFLASFFPACK_WINOTHRESHOLD_BAL`

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL 1000
```

### 17.91.1.4 `__FFLASFFPACK_WINOTHRESHOLD_BAL_FLT`

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT 2000
```

### 17.91.1.5 `__FFLASFFPACK_PLUQ_THRESHOLD`

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 256
```

### 17.91.1.6 `__FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD`

```
#define __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD 1000
```

### 17.91.1.7 `__FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD`

```
#define __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD 16
```

### 17.91.1.8 `__FFLASFFPACK_ARITHPROG_THRESHOLD`

```
#define __FFLASFFPACK_ARITHPROG_THRESHOLD 30
```

### 17.91.1.9 `__FFLASFFPACK_FTRTRI_THRESHOLD`

```
#define __FFLASFFPACK_FTRTRI_THRESHOLD 32
```

### 17.91.1.10 `__FFLASFFPACK_FSYTRF_THRESHOLD`

```
#define __FFLASFFPACK_FSYTRF_THRESHOLD 64
```

**17.91.1.11 \_\_FFLASFFPACK\_FSYRK\_THRESHOLD**

```
#define __FFLASFFPACK_FSYRK_THRESHOLD 3000
```

**17.92 fflas-ffpack-thresholds.h File Reference****17.93 fflas-ffpack.doxy File Reference****17.94 fflas-ffpack.h File Reference**

Includes [FFLAS](#) and [FFPACK](#).

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas/fflas.h"
#include "ffpack/ffpack.h"
```

**17.94.1 Detailed Description**

Includes [FFLAS](#) and [FFPACK](#).

**17.95 fflas.doxy File Reference****17.96 fflas.h File Reference**

**Finite Field Linear Algebra Subroutines**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include <cmath>
#include <cstring>
#include <float.h>
#include <algorithm>
#include "fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas_level1.inl"
#include "fflas_level2.inl"
#include "fflas_level3.inl"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas_freduce.h"
#include "fflas_fadd.h"
#include "fflas_fscal.h"
#include "fflas_fassign.h"
#include "fflas_fgemm.inl"
#include "fflas_pfgemm.inl"
#include "fflas_fgemv.inl"
#include "fflas-ffpack/paladin/pfgemv.inl"
#include "fflas_freivalds.inl"
#include "fflas_fger.inl"
#include "fflas_fsyk.inl"
#include "fflas_fsyk2k.inl"
#include "fflas_ftrsm.inl"
#include "fflas_pftrsm.inl"
#include "fflas_ftrmm.inl"
#include "fflas_ftrsv.inl"
#include "fflas_faxpy.inl"
```

```
#include "fflas_fdot.inl"
#include "fflas-ffpack/field/rns.h"
#include "fflas_fscal_mp.inl"
#include "fflas_freduce_mp.inl"
#include "fflas-ffpack/fflas/fflas_fger_mp.inl"
#include "fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas_ftrsm_mp.inl"
#include "fflas_fgemv_mp.inl"
#include "fflas-ffpack/field/rns.inl"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas_sparse.h"
#include "fflas-ffpack/checkers/checkers_fflas.inl"
```

## Macros

- `#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD`
- `#define DOUBLE_TO_FLOAT_CROSSOVER 800`

*Thresholds determining which floating point representation to use, depending on the cardinality of the finite field.*

### 17.96.1 Detailed Description

Finite Field Linear Algebra Subroutines

Author

Clément Pernet.

### 17.96.2 Macro Definition Documentation

#### 17.96.2.1 WINOTHRESHOLD

```
#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD
```

#### 17.96.2.2 DOUBLE\_TO\_FLOAT\_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 800
```

Thresholds determining which floating point representation to use, depending on the cardinality of the finite field. This is only used when the element representation is not a floating point type.

**Bug** to be benchmarked.

## 17.97 fflas\_101.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
```

## Functions

- `int main` (int argc, char \*\*argv)

### 17.97.1 Function Documentation

### 17.97.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.98 fflas\_101\_lvl1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.98.1 Function Documentation

#### 17.98.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.99 fflas\_bounds.inl File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/flimits.h"
#include <givaro/udl.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_bounds\\_INL](#)
- `#define` [FFLAS\\_INT\\_TYPE](#) [uint64\\_t](#)

### Functions

- `template<class Field >`  
double [computeFactorClassic](#) (const [Field](#) &F)
- `template<>` double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< double > &F)
- `template<>` double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< float > &F)
- `template<class Field >`  
size\_t [DotProdBoundClassic](#) (const [Field](#) &F, const typename [Field::Element](#) &beta)
- [Givaro::Integer](#) [InfNorm](#) (const size\_t M, const size\_t N, const [Givaro::Integer](#) \*A, const size\_t lda)

- template<class Field >  
size\_t [TRSMBound](#) (const [Field](#) &)  
*TRSMBound.*
- template<class Element >  
size\_t [TRSMBound](#) (const Givaro::Modular< Element > &F)  
*Specialization for positive modular representation over float.*
- template<class Element >  
size\_t [TRSMBound](#) (const [Givaro::ModularBalanced](#)< Element > &F)  
*Specialization for balanced modular representation over double.*

## 17.99.1 Macro Definition Documentation

### 17.99.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_bounds\\_INL](#)

```
#define __FFLASFFPACK_fflas_bounds_INL
```

### 17.99.1.2 [FFLAS\\_INT\\_TYPE](#)

```
#define FFLAS_INT_TYPE uint64\_t
```

## 17.100 [fflas\\_c.h](#) File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>
```

### Macros

- #define [FFLAS\\_COMPILED](#)

### Enumerations

- enum [FFLAS\\_C\\_ORDER](#) { [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 , [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 }  
*Storage by row or col ?*
- enum [FFLAS\\_C\\_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 , [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }  
*Is matrix transposed ?*
- enum [FFLAS\\_C\\_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 , [FflasUpper](#) = 121 , [FflasLower](#) = 122 }  
*Is triangular matrix's shape upper ?*
- enum [FFLAS\\_C\\_DIAG](#) { [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 , [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 }  
*Is the triangular matrix implicitly unit diagonal ?*
- enum [FFLAS\\_C\\_SIDE](#) { [FflasLeft](#) = 141 , [FflasRight](#) = 142 , [FflasLeft](#) = 141 , [FflasRight](#) = 142 }  
*On what side ?*
- enum [FFLAS\\_C\\_BASE](#) { [FflasDouble](#) = 151 , [FflasFloat](#) = 152 , [FflasGeneric](#) = 153 }  
*FFLAS\_C\_BASE determines the type of the element representation for Matrix Mult kernel.*



## Functions

- void [freducein\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [freduce\\_1\\_modular\\_double](#) (const double F, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fnegin\\_1\\_modular\\_double](#) (const double F, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [fneg\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fzero\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- bool [fiszero\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, bool positive)
- bool [fequal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fassign\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, double \*X, const size\_t incX, bool positive)
- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [faxpy\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- double [fdot\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fswap\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [fadd\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsub\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [faddin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsubin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fassign\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*A, const size\_t incA, bool positive)
- void [fzero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- bool [fequal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, bool positive)
- bool [fiszero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, bool positive)
- void [fidentity\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, const double d, bool positive)
- void [freducein\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- void [freduce\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*A, const size\_t incA, bool positive)
- void [fnegin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- void [fneg\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*A, const size\_t incA, bool positive)
- void [fscal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t incA, bool positive)
- void [fscal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t incA, double \*B, const size\_t incB, bool positive)
- void [faxpy\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)

- void [fmove\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- void [fadd\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, double \*C, const size\_t ldC, bool positive)
- void [fsub\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, double \*C, const size\_t ldC, bool positive)
- void [fsubin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldB, double \*C, const size\_t ldC, bool positive)
- void [faddin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldB, double \*C, const size\_t ldC, bool positive)
- double \* [fgemv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double \*X, const size\_t incX, const double beta, double \*Y, const size\_t incY, bool positive)
- void [fger\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*x, const size\_t incX, const double \*y, const size\_t incY, double \*A, const size\_t ldA, bool positive)
- void [ftrsv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t n, const double \*A, const size\_t ldA, double \*X, int incX, bool positive)
- void [ftrsm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- void [ftrmm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- double \* [fgemm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_TRANSPOSE](#) tB, const size\_t m, const size\_t n, const size\_t k, const double alpha, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, const double beta, double \*C, const size\_t ldC, bool positive)
- double \* [fsquare\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double beta, double \*C, const size\_t ldC, bool positive)

## 17.100.1 Macro Definition Documentation

### 17.100.1.1 FFLAS\_COMPILED

```
#define FFLAS_COMPILED
```

## 17.100.2 Enumeration Type Documentation

### 17.100.2.1 FFLAS\_C\_ORDER

```
enum FFLAS\_C\_ORDER
```

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major
FflasRowMajor	
FflasColMajor	

**17.100.2.2 FFLAS\_C\_TRANSPOSE**enum [FFLAS\\_C\\_TRANSPOSE](#)

Is matrix transposed ?

**Enumerator**

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.
FflasNoTrans	
FflasTrans	

**17.100.2.3 FFLAS\_C\_UPLO**enum [FFLAS\\_C\\_UPLO](#)

Is triangular matrix's shape upper ?

**Enumerator**

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ )
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ )
FflasUpper	
FflasLower	

**17.100.2.4 FFLAS\_C\_DIAG**enum [FFLAS\\_C\\_DIAG](#)

Is the triangular matrix implicitly unit diagonal ?

**Enumerator**

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ )
FflasNonUnit	
FflasUnit	

**17.100.2.5 FFLAS\_C\_SIDE**enum [FFLAS\\_C\\_SIDE](#)

On what side ?

**Enumerator**

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the righth.
FflasLeft	
FflasRight	

### 17.100.2.6 FFLAS\_C\_BASE

enum [FFLAS\\_C\\_BASE](#)

FFLAS\_C\_BASE determines the type of the element representation for Matrix Mult kernel.  
(deprecated, should not be used)

#### Enumerator

FflasDouble	to use the double precision BLAS
FflasFloat	to use the single precison BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

## 17.100.3 Function Documentation

### 17.100.3.1 `freducein_1_modular_double()`

```
void freducein_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

### 17.100.3.2 `freduce_1_modular_double()`

```
void freduce_1_modular_double (
    const double F,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

### 17.100.3.3 `fnegin_1_modular_double()`

```
void fnegin_1_modular_double (
    const double F,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

### 17.100.3.4 `fneg_1_modular_double()`

```
void fneg_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
```

```
double * X,  
const size_t incX,  
bool positive )
```

#### 17.100.3.5 fzero\_1\_modular\_double()

```
void fzero_1_modular_double (   
    const double p,  
    const size_t n,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.100.3.6 fiszero\_1\_modular\_double()

```
bool fiszero_1_modular_double (   
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.100.3.7 fequal\_1\_modular\_double()

```
bool fequal_1_modular_double (   
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.100.3.8 fassign\_1\_modular\_double()

```
void fassign_1_modular_double (   
    const double p,  
    const size_t n,  
    const double * Y,  
    const size_t incY,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.100.3.9 fscaln\_1\_modular\_double()

```
void fscaln_1_modular_double (   
    const double p,  
    const size_t n,  
    const double alpha,  
    double * X,  
    const size_t incX,  
    bool positive )
```

**17.100.3.10 fscal\_1\_modular\_double()**

```
void fscal_1_modular_double (
    const double p,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t incX,
    double * Y,
    const size_t incY,
    bool positive )
```

**17.100.3.11 faxpy\_1\_modular\_double()**

```
void faxpy_1_modular_double (
    const double p,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t incX,
    double * Y,
    const size_t incY,
    bool positive )
```

**17.100.3.12 fdot\_1\_modular\_double()**

```
double fdot_1_modular_double (
    const double p,
    const size_t n,
    const double * X,
    const size_t incX,
    const double * Y,
    const size_t incY,
    bool positive )
```

**17.100.3.13 fswap\_1\_modular\_double()**

```
void fswap_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    double * Y,
    const size_t incY,
    bool positive )
```

**17.100.3.14 fadd\_1\_modular\_double()**

```
void fadd_1_modular_double (
    const double p,
    const size_t n,
    const double * A,
    const size_t incA,
    const double * B,
    const size_t incB,
```

```
double * C,  
const size_t incC,  
bool positive )
```

#### 17.100.3.15 fsub\_1\_modular\_double()

```
void fsub_1_modular_double (   
    const double p,  
    const size_t n,  
    const double * A,  
    const size_t incA,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

#### 17.100.3.16 faddin\_1\_modular\_double()

```
void faddin_1_modular_double (   
    const double p,  
    const size_t n,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

#### 17.100.3.17 fsubin\_1\_modular\_double()

```
void fsubin_1_modular_double (   
    const double p,  
    const size_t n,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

#### 17.100.3.18 fassign\_2\_modular\_double()

```
void fassign_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.100.3.19 fzero\_2\_modular\_double()

```
void fzero_2_modular_double (
```

```
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t ldA,
    bool positive )
```

#### 17.100.3.20 fequal\_2\_modular\_double()

```
bool fequal_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    bool positive )
```

#### 17.100.3.21 fiszero\_2\_modular\_double()

```
bool fiszero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t ldA,
    bool positive )
```

#### 17.100.3.22 fidentity\_2\_modular\_double()

```
void fidentity_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t ldA,
    const double d,
    bool positive )
```

#### 17.100.3.23 freducein\_2\_modular\_double()

```
void freducein_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t ldA,
    bool positive )
```

#### 17.100.3.24 freduce\_2\_modular\_double()

```
void freduce_2_modular_double (
    const double p,
```



```
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.100.3.25 fnegin\_2\_modular\_double()

```
void fnegin_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.100.3.26 fneg\_2\_modular\_double()

```
void fneg_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.100.3.27 fscaln\_2\_modular\_double()

```
void fscaln_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.100.3.28 fscal\_2\_modular\_double()

```
void fscal_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * A,  
    const size_t ldA,  
    double * B,  
    const size_t ldB,  
    bool positive )
```

**17.100.3.29 faxpy\_2\_modular\_double()**

```
void faxpy_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t ldX,
    double * Y,
    const size_t ldY,
    bool positive )
```

**17.100.3.30 fmove\_2\_modular\_double()**

```
void fmove_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t ldA,
    double * B,
    const size_t ldB,
    bool positive )
```

**17.100.3.31 fadd\_2\_modular\_double()**

```
void fadd_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    double * C,
    const size_t ldC,
    bool positive )
```

**17.100.3.32 fsub\_2\_modular\_double()**

```
void fsub_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    double * C,
    const size_t ldC,
    bool positive )
```

**17.100.3.33 fsubin\_2\_modular\_double()**

```
void fsubin_2_modular_double (
```

```
const double p,  
const size_t m,  
const size_t n,  
const double * B,  
const size_t ldB,  
double * C,  
const size_t ldC,  
bool positive )
```

#### 17.100.3.34 faddin\_2\_modular\_double()

```
void faddin_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * C,  
    const size_t ldC,  
    bool positive )
```

#### 17.100.3.35 fgemv\_2\_modular\_double()

```
double* fgemv_2_modular_double (  
    const double p,  
    const enum FFLAS_C_TRANSPOSE TransA,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * A,  
    const size_t ldA,  
    const double * X,  
    const size_t incX,  
    const double betA,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.100.3.36 fger\_2\_modular\_double()

```
void fger_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * x,  
    const size_t incX,  
    const double * y,  
    const size_t incY,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

**17.100.3.37 ftrsv\_2\_modular\_double()**

```
void ftrsv_2_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_TRANSPOSE TransA,
    const enum FFLAS_C_DIAG Diag,
    const size_t n,
    const double * A,
    const size_t ldA,
    double * X,
    int incX,
    bool positive )
```

**17.100.3.38 ftrsm\_3\_modular\_double()**

```
void ftrsm_3_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_TRANSPOSE TransA,
    const enum FFLAS_C_DIAG Diag,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,
    double * B,
    const size_t ldB,
    bool positive )
```

**17.100.3.39 ftrmm\_3\_modular\_double()**

```
void ftrmm_3_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_TRANSPOSE TransA,
    const enum FFLAS_C_DIAG Diag,
    const size_t m,
    const size_t n,
    const double alpha,
    double * A,
    const size_t ldA,
    double * B,
    const size_t ldB,
    bool positive )
```

**17.100.3.40 fgemm\_3\_modular\_double()**

```
double* fgemm_3_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE tA,
    const enum FFLAS_C_TRANSPOSE tB,
    const size_t m,
    const size_t n,
```

```

    const size_t k,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    const double betA,
    double * C,
    const size_t ldC,
    bool positive )

```

#### 17.100.3.41 fsquare\_3\_modular\_double()

```

double* fsquare_3_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE tA,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double betA,
    double * C,
    const size_t ldC,
    bool positive )

```

## 17.101 fflas\_enum.h File Reference

```
#include <algorithm>
```

### Data Structures

- class [AreEqual< X, Y >](#)
- class [AreEqual< X, X >](#)

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Enumerations

- enum [FFLAS\\_ORDER](#) { [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 }
- Storage by row or col ?*
- enum [FFLAS\\_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }
- Is matrix transposed ?*
- enum [FFLAS\\_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 }
- Is triangular matrix's shape upper ?*
- enum [FFLAS\\_DIAG](#) { [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 }
- Is the triangular matrix implicitly unit diagonal ?*
- enum [FFLAS\\_SIDE](#) { [FflasLeft](#) = 141 , [FflasRight](#) = 142 }
- On what side ?*
- enum [FFLAS\\_BASE](#) { [FflasDouble](#) = 151 , [FflasFloat](#) = 152 , [FflasGeneric](#) = 153 }
- FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.*

## Functions

- `template<class T >`  
`const T & min3 (const T &m, const T &n, const T &k)`
- `template<class T >`  
`const T & max3 (const T &m, const T &n, const T &k)`
- `template<class T >`  
`const T & min4 (const T &m, const T &n, const T &k, const T &l)`
- `template<class T >`  
`const T & max4 (const T &m, const T &n, const T &k, const T &l)`

## 17.102 fflas\_fadd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas_fadd.inl"
```

## Data Structures

- struct [support\\_simd\\_add< T >](#)

## Namespaces

- [FFLAS](#)

## Functions

- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`

- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition.*
- `template<class Field >`  
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsub : matrix subtraction.*
- `template<class Field >`  
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*faddin*
- `template<class Field >`  
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsubin  $C = C - B$*
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition with scaling.*

## 17.103 fflas\_fadd.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::vectorised](#)
- [FFLAS::details](#)

### Macros

- `#define __FFLASFFPACK_fadd_INL`

### Functions

- `template<class SimdT , class Element , bool positive>`  
`std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)`
- `template<bool positive, class Element , class T1 , class T2 >`  
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (Element *T, const Element *TA, const Element *TB, size_t n, Element p, T1 min_, T2 max_)`
- `template<class SimdT , class Element , bool positive>`  
`std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)`
- `template<bool positive, class Element , class T1 , class T2 >`  
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (Element *T, const Element *TA, const Element *TB, const size_t n, const Element p, const T1 min_, const T2 max_)`
- `template<class Element >`  
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (Element *T, const Element *TA, const Element *TB, size_t n)`

- `template<class Element >`  
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (Element *T, const Element`  
`*TA, const Element *TB, size_t n)`
- `template<class Field , bool ADD>`  
`std::enable_if< FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field↵`  
`Categories::ModularTag)`
- `template<class Field , bool ADD>`  
`std::enable_if<!FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field↵`  
`Categories::ModularTag)`
- `template<class Field , bool ADD>`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, type-`  
`name Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field↵`  
`Categories::GenericTag)`
- `template<class Field , bool ADD>`  
`std::enable_if<!FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field↵`  
`Categories::UnparametricTag)`
- `template<class Field , bool ADD>`  
`std::enable_if< FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd`  
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field↵`  
`Categories::UnparametricTag)`

### 17.103.1 Macro Definition Documentation

#### 17.103.1.1 \_\_FFLASFFPACK\_fadd\_INL

```
#define __FFLASFFPACK_fadd_INL
```

## 17.104 fflas\_fassign.h File Reference

```
#include "fflas_fassign.inl"
```

## 17.105 fflas\_fassign.inl File Reference

```
#include <string.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/debug.h"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fassign_INL`



## Functions

- `template<class Field >`  
`void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$fassign : x \leftarrow y.$$
- `template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<class Field >`  
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$fassign : A \leftarrow B.$$

### 17.105.1 Macro Definition Documentation

#### 17.105.1.1 \_\_FFLASFFPACK\_fassign\_INL

```
#define __FFLASFFPACK_fassign_INL
```

## 17.106 fflas\_faxpy.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_faxpy_INL`

## Functions

- `template<class Field >`  
`void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::ConstElement a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$

## 17.106.1 Macro Definition Documentation

### 17.106.1.1 \_\_FFLASFFPACK\_faxpy\_INL

```
#define __FFLASFFPACK_faxpy_INL
```

## 17.107 fflas\_fdot.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fdot\\_INL](#)

### Functions

- `template<class Field >`  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- `template<class Field >`  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DelayedTag &MT)
- `template<>` [Givaro::DoubleDomain::Element](#) [fdot](#) (const [Givaro::DoubleDomain](#) &, const size\_t N, [Givaro::DoubleDomain::ConstElement\\_ptr](#) x, const size\_t incx, [Givaro::DoubleDomain::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- `template<>` [Givaro::FloatDomain::Element](#) [fdot](#) (const [Givaro::FloatDomain](#) &, const size\_t N, [Givaro::FloatDomain::ConstElement\\_ptr](#) x, const size\_t incx, [Givaro::FloatDomain::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- `template<class Field, class T >`  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::ConvertTo< T > &MT)
- `template<class Field >`  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultBoundedTag &dbt)
- `template<class Field >`  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, const [ParSeqHelper::Sequential](#) seq)
- `template<class Field >`  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY)  
*fdot: dot product  $x^T y$ .*

## 17.107.1 Macro Definition Documentation

### 17.107.1.1 \_\_FFLASFFPACK\_fdot\_INL

```
#define __FFLASFFPACK_fdot_INL
```

## 17.108 fflas\_fgemm.inl File Reference

```
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/debug.h"
#include "fflas_fgemm/fgemm_classical.inl"
#include "fflas_fgemm/fgemm_winograd.inl"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fgemm\\_INL](#)

### Functions

- [template<class NewField , class Field , class FieldMode >](#)  
[Field::Element\\_ptr fgemm\\_convert](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode > &H)
- [template<class Field , class Element , class AlgoT , class ParSeqTrait >](#)  
[bool NeedPreAddReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- [template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >](#)  
[bool NeedPreAddReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< [Field](#), AlgoT, ModeT, ParSeqTrait > &WH)
- [template<class Field , class Element , class AlgoT , class ParSeqTrait >](#)  
[bool NeedPreSubReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- [template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >](#)  
[bool NeedPreSubReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< [Field](#), AlgoT, ModeT, ParSeqTrait > &WH)
- [template<class Field , class Element , class AlgoT , class ParSeqTrait >](#)  
[bool NeedDoublePreAddReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- [template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >](#)  
[bool NeedDoublePreAddReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< [Field](#), AlgoT, ModeT, ParSeqTrait > &WH)
- [template<class Field , class AlgoT , class ParSeqTrait >](#)  
[void ScalAndReduce](#) (const [Field](#) &F, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) X, const size\_t incX, const MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)
- [template<class Field , class AlgoT , class ParSeqTrait >](#)  
[void ScalAndReduce](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, const MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)

- `template<class Field >`  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Winograd`, `ModeCategories::ConvertTo`< `ElementCategories::MachineFloatTag` >, `ParSeqHelper::Sequential` > &H)
- `template<typename Field >`  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, const `ParSeqHelper::Sequential` seq)
- `template<typename Field , class Cut , class Param >`  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, const `ParSeqHelper::Parallel`< `Cut`, `Param` > par)
- `template<typename Field >`  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc)  
*fgemm: Field GENERAL Matrix Multiply.*
- `template<typename Field , class ModeT , class ParSeq >`  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Auto`, `ModeT`, `ParSeq` > &H)
- `template<class Field >`  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `MMHelper`< `Field`, `MMHelperAlgo::Winograd`, `ModeCategories::DelayedTag`, `ParSeqHelper::Sequential` > &H)
- `template<class Field >`  
`Field::Element_ptr fsquare` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc)  
*fsquare: Squares a matrix.*
- `template<class Field >`  
`Field::Element_ptr fsquareCommon` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc)
- `template<> double * fsquare` (const `Givaro::ModularBalanced`< `double` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `double` alpha, const `double` \*A, const `size_t` lda, const `double` beta, `double` \*C, const `size_t` ldc)
- `template<> float * fsquare` (const `Givaro::ModularBalanced`< `float` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `float` alpha, const `float` \*A, const `size_t` lda, const `float` beta, `float` \*C, const `size_t` ldc)
- `template<> double * fsquare` (const `Givaro::Modular`< `double` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `double` alpha, const `double` \*A, const `size_t` lda, const `double` beta, `double` \*C, const `size_t` ldc)
- `template<> float * fsquare` (const `Givaro::Modular`< `float` > &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `float` alpha, const `float` \*A, const `size_t` lda, const `float` beta, `float` \*C, const `size_t` ldc)

## 17.108.1 Macro Definition Documentation

### 17.108.1.1 \_\_FFLASFFPACK\_fgemm\_INL

```
#define __FFLASFFPACK_fgemm_INL
```

## 17.109 fflas\_fgemv.inl File Reference

```
#include <givaro/zring.h>
```

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fgemv\\_INL](#)

### Functions

- [template<typename FloatElement , class Field >](#)  
[Field::Element\\_ptr fgemv\\_convert](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY)
- [template<class Field >](#)  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), MMHelperAlgo::Classic, ModeCategories::↵ ConvertTo< ElementCategories::MachineFloatTag > > &H)
- [template<class Field >](#)  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), MMHelperAlgo::Classic, ModeCategories::↵ DelayedTag > &H)
- [template<class Field >](#)  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), MMHelperAlgo::Classic, ModeCategories::↵ DefaultTag > &H)
- [template<class Field >](#)  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), MMHelperAlgo::Classic, ModeCategories::↵ LazyTag > &H)
- [template<class Field >](#)  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE TransA, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY)

*finite prime Field GEneral Matrix Vector multiplication.*

- `Givaro::ZRing< int64_t >::Element_ptr fgemv` (const `Givaro::ZRing< int64_t > &F`, const `FFLAS_TRANSPOSE ta`, const `size_t M`, const `size_t N`, const `int64_t alpha`, const `int64_t *A`, const `size_t lda`, const `int64_t *X`, const `size_t incX`, const `int64_t beta`, `int64_t *Y`, const `size_t incY`, `MMHelper< Givaro::ZRing< int64_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H`)
- `Givaro::DoubleDomain::Element_ptr fgemv` (const `Givaro::DoubleDomain &F`, const `FFLAS_TRANSPOSE ta`, const `size_t M`, const `size_t N`, const `Givaro::DoubleDomain::Element alpha`, const `Givaro::DoubleDomain::ConstElement_ptr A`, const `size_t lda`, const `Givaro::DoubleDomain::ConstElement_ptr X`, const `size_t incX`, const `Givaro::DoubleDomain::Element beta`, `Givaro::DoubleDomain::Element_ptr Y`, const `size_t incY`, `MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H`)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field &F`, const `FFLAS_TRANSPOSE ta`, const `size_t M`, const `size_t N`, const `typename Field::Element alpha`, const `typename Field::ConstElement_ptr A`, const `size_t lda`, const `typename Field::ConstElement_ptr X`, const `size_t incX`, const `typename Field::Element beta`, `typename Field::Element_ptr Y`, const `size_t incY`, `MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H`)
- `Givaro::FloatDomain::Element_ptr fgemv` (const `Givaro::FloatDomain &F`, const `FFLAS_TRANSPOSE ta`, const `size_t M`, const `size_t N`, const `Givaro::FloatDomain::Element alpha`, const `Givaro::FloatDomain::ConstElement_ptr A`, const `size_t lda`, const `Givaro::FloatDomain::ConstElement_ptr X`, const `size_t incX`, const `Givaro::FloatDomain::Element beta`, `Givaro::FloatDomain::Element_ptr Y`, const `size_t incY`, `MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H`)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fgemv` (const `Field &F`, const `FFLAS_TRANSPOSE ta`, const `size_t m`, const `size_t n`, const `typename Field::Element alpha`, const `typename Field::ConstElement_ptr A`, const `size_t lda`, const `typename Field::ConstElement_ptr X`, const `size_t incX`, const `typename Field::Element beta`, `typename Field::Element_ptr Y`, const `size_t incY`, `ParSeqHelper::Parallel< Cut, Param > &parH`)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field &F`, const `FFLAS_TRANSPOSE ta`, const `size_t m`, const `size_t n`, const `typename Field::Element alpha`, const `typename Field::ConstElement_ptr A`, const `size_t lda`, const `typename Field::ConstElement_ptr X`, const `size_t incX`, const `typename Field::Element beta`, `typename Field::Element_ptr Y`, const `size_t incY`, `ParSeqHelper::Sequential &seqH`)

## 17.109.1 Macro Definition Documentation

### 17.109.1.1 \_\_FFLASFFPACK\_fgemv\_INL

```
#define __FFLASFFPACK_fgemv_INL
```

## 17.110 fflas\_fgemv\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

## Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_fgemv_mp_INL`

## Functions

- `FFPACK::rns_double::Element_ptr fgemv` (const `FFPACK::RNSInteger< FFPACK::rns_double > &F`, const `FFLAS_TRANSPOSE ta`, const `size_t M`, const `size_t N`, const `FFPACK::rns_double::Element alpha`,

- [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- [FFPACK::rns\\_double::Element\\_ptr](#) fgemv (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
  - Givaro::Integer \* [fgemv](#) (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const Givaro::Integer alpha, Givaro::Integer \*A, const size\_t lda, Givaro::Integer \*X, const size\_t ldx, Givaro::Integer beta, Givaro::Integer \*Y, const size\_t ldy, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
  - Givaro::Integer \* [fgemv](#) (const Givaro::Modular< Givaro::Integer > &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const Givaro::Integer alpha, Givaro::Integer \*A, const size\_t lda, Givaro::Integer \*X, const size\_t ldx, Givaro::Integer beta, Givaro::Integer \*Y, const size\_t ldy, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
  - template<size\_t K1, size\_t K2, class ParSeq >  
[RecInt::ruint](#)< K1 > \* [fgemv](#) (const Givaro::Modular< [RecInt::ruint](#)< K1 >, [RecInt::ruint](#)< K2 > > &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const [RecInt::ruint](#)< K1 > alpha, const [RecInt::ruint](#)< K1 > \*A, const size\_t lda, const [RecInt::ruint](#)< K1 > \*X, const size\_t incx, [RecInt::ruint](#)< K1 > beta, [RecInt::ruint](#)< K1 > \*Y, const size\_t incy, MMHelper< Givaro::Modular< [RecInt::ruint](#)< K1 >, [RecInt::ruint](#)< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)

## 17.110.1 Macro Definition Documentation

### 17.110.1.1 \_\_FFLASFFPACK\_fgemv\_mp\_INL

```
#define __FFLASFFPACK_fgemv_mp_INL
```

## 17.111 fflas\_fger.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fger\\_INL](#)

### Functions

- template<class Field >  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fger: rank one update of a general matrix*
- template<class FloatElement, class Field >  
void [fger\\_convert](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda)



- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)`
- `template<class Field, class AnyTag >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, AnyTag > &H)`
- `void fger (const Givaro::DoubleDomain &F, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, const Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, Givaro::DoubleDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr x, const size_t incx, const typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)`
- `void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, const Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, Givaro::FloatDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)`

### 17.111.1 Macro Definition Documentation

#### 17.111.1.1 \_\_FFLASFFPACK\_fger\_INL

```
#define __FFLASFFPACK_fger_INL
```

## 17.112 fflas\_fger\_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFPACK_fger_mp_INL`



## Functions

- void [fger](#) (const Givaro::Modular< Givaro::Integer > &F, const size\_t M, const size\_t N, const typename Givaro::Integer alpha, typename Givaro::Integer \*x, const size\_t incx, typename Givaro::Integer \*y, const size\_t incy, typename Givaro::Integer \*A, const size\_t lda, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<typename RNS >  
void [fger](#) (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const size\_t M, const size\_t N, const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr x, const size\_t incx, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr y, const size\_t incy, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr A, const size\_t lda, MMHelper< [FFPACK::RNSInteger](#)< [RNS](#) >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<typename RNS >  
void [fger](#) (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const size\_t M, const size\_t N, const typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr x, const size\_t incx, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr y, const size\_t incy, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr A, const size\_t lda, MMHelper< [FFPACK::RNSIntegerMod](#)< [RNS](#) >, MMHelperAlgo::Classic > &H)

### 17.112.1 Macro Definition Documentation

#### 17.112.1.1 \_\_FFPACK\_fger\_mp\_INL

```
#define __FFPACK_fger_mp_INL
```

## 17.113 fflas\_freduce.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/utils/cast.h"
#include "fflas-ffpack/fflas/fflas_freduce.inl"
```

## Data Structures

- struct [support\\_simd\\_mod](#)< T >
- struct [support\\_fast\\_mod](#)< T >
- struct [support\\_fast\\_mod](#)< float >
- struct [support\\_fast\\_mod](#)< double >
- struct [support\\_fast\\_mod](#)< int64\_t >

## Namespaces

- [FFLAS](#)

## Functions

- template<class Field >  
void [freduce](#) (const [Field](#) &F, const size\_t n, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
$$\text{freduce } x \leftarrow y \bmod F.$$
- template<class Field >  
void [freduce](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
$$\text{freduce } x \leftarrow x \bmod F.$$

- `template<class Field >`  
`void freduce_constoverride (const Field &F, const size_t m, typename Field::ConstElement_ptr A, const size_t incX)`
- `template<class Field, class ConstOtherElement_ptr >`  
`void finit (const Field &F, const size_t n, ConstOtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`
- `template<class Field >`  
`void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  
*finit Initializes X in  $F^S$ .*
- `template<class Field >`  
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*freduce  $A \leftarrow A \bmod F$ .*
- `template<class Field >`  
`void pfreduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const size_t numths)`
- `template<class Field >`  
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
*freduce  $A \leftarrow B \bmod F$ .*
- `template<class Field >`  
`void freduce_constoverride (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda)`
- `template<class Field, class OtherElement_ptr >`  
`void finit (const Field &F, const size_t m, const size_t n, const OtherElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  
*finit  $A \leftarrow B \bmod F$ .*
- `template<class Field >`  
`void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  
*finit Initializes A in  $F^S$ .*

## 17.114 fflas\_freduce.inl File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

### Data Structures

- struct [HelperMod](#)< Field, ElementCategories::MachineIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::MachineFloatTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::FixedPrecIntTag >

### Namespaces

- [FFLAS](#)
- [FFLAS::vectorised](#)
- [FFLAS::vectorised::unswitch](#)
- [FFLAS::details](#)

### Macros

- `#define __FFLASFFPACK_fflas_freduce_INL`
- `#define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCHMARK LATER */`

## Functions

- `template<class T >`  
`std::enable_if< ! std::is_integral< T >::value, T >::type` `reduce` (T A, T B)
- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, T >::type` `reduce` (T A, T B)
- `template<> Givaro::Integer` `reduce` (Givaro::Integer A, Givaro::Integer B)
- `float` `reduce` (float A, float B, float invB, float min, float max)
- `double` `reduce` (double A, double B, double invB, double min, double max)
- `int64_t` `reduce` (`int64_t` A, `int64_t` p, double invp, double min, double max, `int64_t` pow50rem)
- `template<class Field >`  
`Field::Element` `reduce` (typename `Field::Element` A, HelperMod< `Field`, ElementCategories::MachineIntTag > &H)
- `template<class Field >`  
`Field::Element` `reduce` (typename `Field::Element` A, HelperMod< `Field`, ElementCategories::MachineFloatTag > &H)
- `template<class Field >`  
`Field::Element` `reduce` (typename `Field::Element` A, HelperMod< `Field`, ElementCategories::ArbitraryPrecisionIntTag > &H)
- `template<class Field >`  
`std::enable_if<!FFLAS::support_simd_mod< typename` `Field::Element` `>::value &&FFLAS::support_fast_mod<`  
`typename` `Field::Element` `>::value, void >::type` `modp` (const `Field` &F, typename `Field::ConstElement_ptr` U,  
`const size_t` &n, typename `Field::Element_ptr` T, HelperMod< `Field` > &H)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename` `Field::Element` `>::value, void >::type` `modp`  
`(const` `Field` `&F, typename` `Field::ConstElement_ptr` `U, const size_t` &n, `const size_t` &incX, typename  
`Field::Element_ptr` `T, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename` `Field::Element` `>::value, void >::type` `modp` (const  
`Field` &F, typename `Field::ConstElement_ptr` U, `const size_t` &n, typename `Field::Element_ptr` T)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename` `Field::Element` `>::value, void >::type` `modp`  
`(const` `Field` `&F, typename` `Field::ConstElement_ptr` `U, const size_t` &n, `const size_t` &incX, typename  
`Field::Element_ptr` `T)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename` `Field::Element` `>::value, void >::type` `freduce` (const  
`Field` &F, `const size_t` m, typename `Field::Element_ptr` A, `const size_t` incX, FieldCategories::ModularTag)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename` `Field::Element` `>::value, void >::type` `freduce`  
`(const` `Field` `&F, const size_t` m, typename `Field::ConstElement_ptr` B, `const size_t` incY, typename  
`Field::Element_ptr` A, `const size_t` incX, FieldCategories::ModularTag)
- `template<class Field, class FC >`  
`void` `freduce` (const `Field` &F, `const size_t` m, typename `Field::Element_ptr` A, `const size_t` incX, FC)
- `template<class Field, class FC >`  
`void` `freduce` (const `Field` &F, `const size_t` m, typename `Field::ConstElement_ptr` B, `const size_t` incY, type-  
`name` `Field::Element_ptr` A, `const size_t` incX, FC)

### 17.114.1 Macro Definition Documentation

#### 17.114.1.1 \_\_FFLASFFPACK\_fflas\_freduce\_INL

```
#define __FFLASFFPACK_fflas_freduce_INL
```

#### 17.114.1.2 FFLASFFPACK\_COPY\_REDUCE

```
#define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCHMARK LATER */
```

## 17.115 fflas\_freduce\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_freduce\\_mp\\_INL](#)

### Functions

- template<> void [freduce](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element\_ptr A, size\_t inc)
- template<> void [freduce](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, [FFPACK::rns\\_double::Element\\_ptr](#) A, size\_t lda)

### 17.115.1 Macro Definition Documentation

#### 17.115.1.1 \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL

```
#define __FFLASFFPACK_fflas_freduce_mp_INL
```

## 17.116 fflas\_freivalds.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_freivalds\\_INL](#)

### Functions

- template<class Field >  
bool [freivalds](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::ConstElement\\_ptr](#) C, const size\_t ldc)

*freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*

### 17.116.1 Macro Definition Documentation

#### 17.116.1.1 \_\_FFLASFFPACK\_freivalds\_INL

```
#define __FFLASFFPACK_freivalds_INL
```

## 17.117 fflas\_fscal.h File Reference

```
#include "fflas_fscal.inl"
```

## 17.118 fflas\_fscal.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::vectorised](#)
- [FFLAS::vectorised::unswitch](#)
- [FFLAS::details](#)

### Macros

- `#define __FFLASFFPACK_fscal_INL`

### Functions

- `template<class Field >`  
`std::enable_if<!FFLAS::support_simd_mod< typename Field::Element >::value &&FFLAS::support_fast_mod<`  
`typename Field::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const`  
`typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, HelperMod< Field >`  
`&H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type fscaln (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX,`  
`FieldCategories::ModularTag)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typename Field::Element >::value, void >::type fscal (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t`  
`incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`
- `template<class Field, class FC >`  
`void fscaln (const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X,`  
`const size_t incX, FC)`
- `template<class Field, class FC >`  
`void fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`
- `template<class Field >`  
`void fscaln (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr`  
`X, const size_t incX)`  

$$fscaln\ x \leftarrow \alpha \cdot x.$$

- `template<class Field >`  
`void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$fscal\ y \leftarrow \alpha \cdot x.$$
- `template<> void fscal (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↔ Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void fscal (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<> void fscaln (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↔ Element a, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void fscaln (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void fscaln (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda)`  

$$fscaln\ A \leftarrow a \cdot A.$$
- `template<class Field >`  
`void fscal (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  

$$fscal\ B \leftarrow a \cdot A.$$

## 17.118.1 Macro Definition Documentation

### 17.118.1.1 \_\_FFLASFFPACK\_fscal\_INL

```
#define __FFLASFFPACK_fscal_INL
```

## 17.119 fflas\_fscal\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas_fscal.h"
#include "fflas_fgemm.inl"
#include "fflas-ffpack/fflas/fflas_freduce_mp.inl"
```

## Namespaces

- [FFLAS](#)

## Macros

- `#define \_\_FFLASFFPACK\_fscal\_mp\_INL`

## Functions

- `template<> void fscaln (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t inc)`
- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscaln (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)`

- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)`
- `template<> void fscaln (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha, typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, const size_t inc)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscaln (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)`

### 17.119.1 Macro Definition Documentation

#### 17.119.1.1 \_\_FFLASFFPACK\_fscal\_mp\_INL

```
#define __FFLASFFPACK_fscal_mp_INL
```

## 17.120 fflas\_fsyr2k.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_fsyr2k_INL`

### Functions

- `template<class Field > Field::Element_ptr fsyr2k (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`

*fsyr2k: Symmetric Rank 2K update*

### 17.120.1 Macro Definition Documentation

#### 17.120.1.1 \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL

```
#define __FFLASFFPACK_fflas_fsyr2k_INL
```

## 17.121 fflas\_fsyrk.inl File Reference

### Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_fflas_fsyk_INL`

## Functions

- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc)  
*fsyrk: Symmetric Rank K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const ParSeqHelper::Sequential seq, const size\_t threshold)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size\_t threshold)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const std::vector< bool > &two←Block, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*

### 17.121.1 Macro Definition Documentation

#### 17.121.1.1 \_\_FFLASFFPACK\_fflas\_fsyk\_INL

```
#define __FFLASFFPACK_fflas_fsyk_INL
```

## 17.122 fflas\_ftrmm.inl File Reference

### Namespaces

- `FFLAS`

### Macros

- `#define __FFLASFFPACK_ftrmm_INL`



## Functions

- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*ftrmm: **TR**angular **M**atrix **M**ultiply.*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*ftrmm: **TR**angular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .*

### 17.122.1 Macro Definition Documentation

#### 17.122.1.1 \_\_FFLASFFPACK\_ftrmm\_INL

```
#define __FFLASFFPACK_ftrmm_INL
```

## 17.123 fflas\_ftrsm.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_ftrsm_INL`

## Functions

- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Sequential &PSH)`
- `template<class Field , class Cut , class Param >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Parallel< Cut, Param > &PSH)`
- `template<class Field , class ParSeqTrait = ParSeqHelper::Sequential>`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Recursive, ParSeqTrait > &H)`

### 17.123.1 Macro Definition Documentation

### 17.123.1.1 \_\_FFLASFFPACK\_ftrsm\_INL

```
#define __FFLASFFPACK_ftrsm_INL
```

## 17.124 fflas\_ftrsm\_mp.inl File Reference

triangular system with matrix right hand side over multiprecision domain (either over  $\mathbb{Z}$  or over  $\mathbb{Z}/p\mathbb{Z}$ )

```
#include <cmath>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFPACK_ftrsm_mp_INL`

### Functions

- void [ftrsm](#) (const Givaro::Modular< Givaro::Integer > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, Givaro::Integer \*B, const size\_t ldb)
- void [cblas\\_imptrsm](#) (const enum FFLAS\_ORDER Order, const enum FFLAS\_SIDE Side, const enum FFLAS\_UPLO Uplo, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_DIAG Diag, const int M, const int N, const [FFPACK::rns\\_double\\_elt](#) alpha, [FFPACK::rns\\_double\\_elt\\_cstptr](#) A, const int lda, [FFPACK::rns\\_double\\_elt\\_ptr](#) B, const int ldb)

### 17.124.1 Detailed Description

triangular system with matrix right hand side over multiprecision domain (either over  $\mathbb{Z}$  or over  $\mathbb{Z}/p\mathbb{Z}$ )

### 17.124.2 Macro Definition Documentation

#### 17.124.2.1 \_\_FFPACK\_ftrsm\_mp\_INL

```
#define __FFPACK_ftrsm_mp_INL
```

## 17.125 fflas\_ftrsv.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_ftrsv_INL`

## Functions

- template<class Field >  
void [ftrsv](#) (const [Field](#) &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)

*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*

## 17.125.1 Macro Definition Documentation

### 17.125.1.1 \_\_FFLASFFPACK\_ftrsv\_INL

```
#define __FFLASFFPACK_ftrsv_INL
```

## 17.126 fflas\_helpers.inl File Reference

```
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/flimits.h"
#include <algorithm>
```

## Data Structures

- struct [Auto](#)
  - struct [Classic](#)
  - struct [Winograd](#)
  - struct [WinogradPar](#)
  - struct [Bini](#)
  - struct [AlgoChooser](#)< [ModeT](#), [ParSeq](#) >
  - struct [AlgoChooser](#)< [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeq](#) >
  - struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) >
- FGEMM Helper for Default and ConvertTo modes of operation.*
- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< [Dest](#) >, [ParSeqTrait](#) >
  - struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeTrait](#), [ParSeqTrait](#) >
  - struct [Recursive](#)
  - struct [Iterative](#)
  - struct [Hybrid](#)
  - struct [TRSMHelper](#)< [ReclterTrait](#), [ParSeqTrait](#) >

*TRSM Helper.*

## Namespaces

- [FFLAS](#)
  - [FFLAS::Protected](#)
  - [FFLAS::MMHelperAlgo](#)
  - [FFLAS::StructureHelper](#)
- StructureHelper for ftrsm.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_mmhelper\\_INL](#)

## Functions

- `template<class Field >`  
`int WinogradSteps (const Field &F, const size_t &m)`  
*Computes the number of recursive levels to perform.*
- `template<class DFE >`  
`size_t min_types (const DFE &k)`
- `template<> size_t min_types (const RecInt::rint< 6 > &k)`
- `template<> size_t min_types (const RecInt::rint< 7 > &k)`
- `template<> size_t min_types (const RecInt::rint< 8 > &k)`
- `template<> size_t min_types (const RecInt::rint< 9 > &k)`
- `template<> size_t min_types (const RecInt::rint< 10 > &k)`
- `template<> size_t min_types (const Givaro::Integer &k)`
- `template<class T >`  
`bool unfit (T x)`
- `template<> bool unfit (int64_t x)`
- `template<size_t K>`  
`bool unfit (RecInt::rint< K > x)`
- `template<> bool unfit (RecInt::rint< 6 > x)`

### 17.126.1 Macro Definition Documentation

#### 17.126.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL

```
#define __FFLASFFPACK_fflas_fflas_mmhelper_INL
```

## 17.127 fflas\_intrinsic.h File Reference

## 17.128 fflas\_io.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include <fstream>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas_memory.h"
```

## Namespaces

- [FFLAS](#)

## Enumerations

- `enum FFLAS_FORMAT {`  
`FflasAuto = 0 , FflasDense = 1 , FflasSMS = 2 , FflasBinary = 3 ,`  
`FflasMath = 4 , FflasMaple = 5 , FflasSageMath = 6 }`

## Functions

- `template<class Field >`  
`std::ostream & WriteMatrix (std::ostream &c, const Field &F, size_t m, size_t n, typename Field::ConstElement_ptr`  
`A, size_t lda, FFLAS_FORMAT format, bool column_major)`  
*WriteMatrix: write a matrix to an output stream.*

- void [preamble](#) (std::ifstream &ifs, FFLAS\_FORMAT &format)
- template<class Field >  
[Field::Element\\_ptr ReadMatrix](#) (std::ifstream &ifs, [Field](#) &F, size\_t &m, size\_t &n, typename [Field::Element\\_ptr](#) &A, FFLAS\_FORMAT format=FFlasAuto)  
*ReadMatrix: read a matrix from an input stream.*
- template<class Field >  
[Field::Element\\_ptr ReadMatrix](#) (const std::string &matrix\_file, [Field](#) &F, size\_t &m, size\_t &n, typename [Field::Element\\_ptr](#) &A, FFLAS\_FORMAT format=FFlasAuto)  
*ReadMatrix: read a matrix from a file.*
- template<class Field >  
void [WriteMatrix](#) (std::string &matrix\_file, const [Field](#) &F, int m, int n, typename [Field::ConstElement\\_ptr](#) A, size\_t lda, FFLAS\_FORMAT format=FFlasDense, bool column\_major=false)  
*WriteMatrix: write a matrix to a file.*
- std::ostream & [WritePermutation](#) (std::ostream &c, const size\_t \*P, size\_t N)  
*WritePermutation: write a permutation matrix to an output stream.*

## 17.129 fflas\_L1\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"
```

### Macros

- #define [\\_\\_FFLAS\\_L1\\_INST\\_C](#)
- #define [INST\\_OR\\_DECL](#)
- #define [FFLAS\\_FIELD](#) [Givaro::ModularBalanced](#)
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) int64\_t
- #define [FFLAS\\_FIELD](#) [Givaro::Modular](#)
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) int64\_t

### 17.129.1 Macro Definition Documentation

#### 17.129.1.1 [\\_\\_FFLAS\\_L1\\_INST\\_C](#)

```
#define \_\_FFLAS\_L1\_INST\_C
```

#### 17.129.1.2 [INST\\_OR\\_DECL](#)

```
#define INST\_OR\_DECL
```

#### 17.129.1.3 [FFLAS\\_FIELD](#) [1/2]

```
#define FFLAS\_FIELD Givaro::ModularBalanced
```

**17.129.1.4 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**17.129.1.5 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.129.1.6 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**17.129.1.7 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.129.1.8 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.129.1.9 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.129.1.10 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.130 fflas\_L1\_inst.h File Reference**

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"
```

**Macros**

- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

**17.130.1 Macro Definition Documentation**

**17.130.1.1 INST\_OR\_DECL**

```
#define INST_OR_DECL <>
```

**17.130.1.2 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**17.130.1.3 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**17.130.1.4 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.130.1.5 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**17.130.1.6 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.130.1.7 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.130.1.8 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.130.1.9 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.131 fflas\_L1\_inst\_implem.inl File Reference****Namespaces**

- [FFLAS](#)

**Functions**

- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{freduce } x \leftarrow x \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{freduce } x \leftarrow y \bmod F.$$

- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)  

$$finit\ x \leftarrow y_{mod F}.$$
- template `INST_OR_DECL` void `fconvert` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)  

$$fconvert\ x \leftarrow y_{mod F}.$$
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX)  

$$fnegin\ x \leftarrow -x.$$
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)  

$$fneg\ x \leftarrow -y.$$
- template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX)  

$$fzero : A \leftarrow 0.$$
- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*X, const size\_t incX)  

$$fiszero : test\ X = 0.$$
- template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)  

$$fequal : test\ X = Y.$$
- template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)  

$$fassign : x \leftarrow y.$$
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*X, const size\_t incX)  

$$fscaln\ x \leftarrow \alpha \cdot x.$$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  

$$fscal\ y \leftarrow \alpha \cdot x.$$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- template `INST_OR_DECL` `FFLAS_ELT` `fdot` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)  

$$faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- template `INST_OR_DECL` void `fswap` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  

$$fswap : X \leftrightarrow Y.$$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)



## 17.132 fflas\_L2\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

### Macros

- `#define __FFLAS_L2_INST_C`
- `#define INST_OR_DECL`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

### 17.132.1 Macro Definition Documentation

#### 17.132.1.1 \_\_FFLAS\_L2\_INST\_C

```
#define __FFLAS_L2_INST_C
```

#### 17.132.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL
```

#### 17.132.1.3 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

#### 17.132.1.4 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

#### 17.132.1.5 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

#### 17.132.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

#### 17.132.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

#### 17.132.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

#### 17.132.1.9 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

#### 17.132.1.10 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

### 17.133 fflas\_L2\_inst.h File Reference

```
#include "givaro/modular.h"  
#include "givaro/modular-balanced.h"  
#include "fflas-ffpack/fflas/fflas.h"  
#include "fflas-ffpack/fflas/fflas_helpers.inl"  
#include "fflas_L2_inst_implem.inl"
```

#### Macros

- #define [INST\\_OR\\_DECL](#) <>
- #define [FFLAS\\_FIELD](#) Givaro::ModularBalanced
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) int64\_t
- #define [FFLAS\\_FIELD](#) Givaro::Modular
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) int64\_t

#### 17.133.1 Macro Definition Documentation

##### 17.133.1.1 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

##### 17.133.1.2 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

##### 17.133.1.3 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

**17.133.1.4 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.133.1.5 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**17.133.1.6 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.133.1.7 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.133.1.8 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.133.1.9 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.134 fflas\_L2\_inst\_implem.inl File Reference****Namespaces**

- [FFLAS](#)

**Functions**

- template [INST\\_OR\\_DECL](#) void [fassign](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
*fassign* :  $A \leftarrow B$ .
- template [INST\\_OR\\_DECL](#) void [fzero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
*fzero* :  $A \leftarrow 0$ .
- template [INST\\_OR\\_DECL](#) bool [fequal](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
*fequal* : test  $A = B$ .
- template [INST\\_OR\\_DECL](#) bool [fiszero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*A, const size\_t lda)  
*fiszero* : test  $A = 0$ .
- template [INST\\_OR\\_DECL](#) void [fidentity](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) &d)  
*creates a diagonal matrix*
- template [INST\\_OR\\_DECL](#) void [fidentity](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
*creates a diagonal matrix*
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)

- freduce*  $A \leftarrow A \bmod F$ .
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)
- freduce*  $A \leftarrow B \bmod F$ .
- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)
- finit*  $A \leftarrow B \bmod F$ .
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)
- fnegin*  $A \leftarrow -A$ .
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)
- fneg*  $A \leftarrow -B$ .
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*A, const size\_t lda)
- fscaln*  $A \leftarrow a \cdot A$ .
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)
- fscal*  $B \leftarrow a \cdot A$ .
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t ldx, `FFLAS_ELT` \*Y, const size\_t ldy)
- faxpy* :  $y \leftarrow \alpha \cdot x + y$ .
- template `INST_OR_DECL` void `fmove` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)
- faxpy* :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
- fadd* : matrix addition.
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
- fsub* : matrix subtraction.
- template `INST_OR_DECL` void `fsubin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
- fsubin*  $C = C - B$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
- fadd* : matrix addition with scaling.
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
- faddin*
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` TransA, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` beta, `FFLAS_ELT` \*Y, const size\_t incY)
- finite prime* `FFLAS_FIELD`<`FFLAS_ELT`> *GEneral Matrix Vector multiplication.*
- template `INST_OR_DECL` void `fger` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*x, const size\_t incx, const `FFLAS_ELT` \*y, const size\_t incy, `FFLAS_ELT` \*A, const size\_t lda)
- fger*: rank one update of a general matrix

- template [INST\\_OR\\_DECL](#) void [ftrsv](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*X, int incX)

*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*

## 17.135 fflas\_L3\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L3_inst_implem.inl"
```

### Macros

- `#define \_\_FFLAS\_L3\_INST\_C`
- `#define INST\_OR\_DECL`
- `#define FFLAS\_FIELD Givaro::ModularBalanced`
- `#define FFLAS\_ELT double`
- `#define FFLAS\_ELT float`
- `#define FFLAS\_ELT int64_t`
- `#define FFLAS\_FIELD Givaro::Modular`
- `#define FFLAS\_ELT double`
- `#define FFLAS\_ELT float`
- `#define FFLAS\_ELT int64_t`

### 17.135.1 Macro Definition Documentation

#### 17.135.1.1 [\\_\\_FFLAS\\_L3\\_INST\\_C](#)

```
#define \_\_FFLAS\_L3\_INST\_C
```

#### 17.135.1.2 [INST\\_OR\\_DECL](#)

```
#define INST\_OR\_DECL
```

#### 17.135.1.3 [FFLAS\\_FIELD](#) [1/2]

```
#define FFLAS\_FIELD Givaro::ModularBalanced
```

#### 17.135.1.4 [FFLAS\\_ELT](#) [1/6]

```
#define FFLAS\_ELT double
```

#### 17.135.1.5 [FFLAS\\_ELT](#) [2/6]

```
#define FFLAS\_ELT float
```

#### 17.135.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

#### 17.135.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

#### 17.135.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

#### 17.135.1.9 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

#### 17.135.1.10 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

### 17.136 fflas\_L3\_inst.h File Reference

```
#include "givaro/modular.h"  
#include "givaro/modular-balanced.h"  
#include "fflas-ffpack/fflas/fflas.h"  
#include "fflas-ffpack/fflas/fflas_helpers.inl"  
#include "fflas_L3_inst_implem.inl"
```

#### Macros

- #define INST\_OR\_DECL <>
- #define FFLAS\_FIELD Givaro::ModularBalanced
- #define FFLAS\_ELT double
- #define FFLAS\_ELT float
- #define FFLAS\_ELT int64\_t
- #define FFLAS\_FIELD Givaro::Modular
- #define FFLAS\_ELT double
- #define FFLAS\_ELT float
- #define FFLAS\_ELT int64\_t

#### 17.136.1 Macro Definition Documentation

##### 17.136.1.1 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

##### 17.136.1.2 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**17.136.1.3 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**17.136.1.4 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.136.1.5 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64\_t
```

**17.136.1.6 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.136.1.7 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.136.1.8 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.136.1.9 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64\_t
```

**17.137 fflas\_L3\_inst\_implem.inl File Reference****Namespaces**

- [FFLAS](#)

**Macros**

- `#define \_\_FFLAS\_\_TRSM\_READONLY`

**Functions**

- template [INST\\_OR\\_DECL](#) void [ftrsm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)

*ftrsm*: **TRI**angular **S**ystem solve with **M**atrix.

- template [INST\\_OR\\_DECL](#) void [ftrmm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)

*ftrmm*: **TRI**angular **M**atrix **M**ultiply.

- template `INST_OR_DECL FFLAS_ELT * fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc)

*fgemm: Field **GE**neral Matrix Multiply.*

- template `INST_OR_DECL FFLAS_ELT * fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Sequential` seq)
- template `INST_OR_DECL FFLAS_ELT * fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive >` par)
- template `INST_OR_DECL FFLAS_ELT * fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads >` par)
- template `INST_OR_DECL FFLAS_ELT * fsquare` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc)

*fsquare: Squares a matrix.*

## 17.137.1 Macro Definition Documentation

### 17.137.1.1 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

## 17.138 fflas\_level1.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_level1\\_INL](#)

### Functions

- template<class Field >  
void [freduce](#) (const [Field](#) &F, const `size_t` n, typename [Field::Element\\_ptr](#) X, const `size_t` incX)  
*freduce  $x \leftarrow x \bmod F$ .*
- template<class Field >  
void [freduce](#) (const [Field](#) &F, const `size_t` n, typename [Field::ConstElement\\_ptr](#) Y, const `size_t` incY, typename [Field::Element\\_ptr](#) X, const `size_t` incX)  
*freduce  $x \leftarrow y \bmod F$ .*
- template<class Field, class OtherElement\_ptr >  
void [finit](#) (const [Field](#) &F, const `size_t` n, const `OtherElement_ptr` Y, const `size_t` incY, typename [Field::Element\\_ptr](#) X, const `size_t` incX)  
*finit  $x \leftarrow y \bmod F$ .*



- template<class Field >  
void **finit** (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)  
*finit* Initializes  $X$  in  $F$ .
- template<class Field, class OtherElement\_ptr >  
void **fconvert** (const Field &F, const size\_t n, OtherElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)  
*fconvert*  $x \leftarrow y \bmod F$ .
- template<class Field >  
void **fnegin** (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)  
*fnegin*  $x \leftarrow -x$ .
- template<class Field >  
void **fneg** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)  
*fneg*  $x \leftarrow -y$ .
- template<class Field >  
void **fzero** (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)  
*fzero* :  $A \leftarrow 0$ .
- template<class Field, class RandIter >  
void **frand** (const Field &F, RandIter &G, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)  
*frand* :  $A \leftarrow \text{random}$ .
- template<class Field >  
bool **fiszero** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr X, const size\_t incX)  
*fiszero* : test  $X = 0$ .
- template<class Field >  
bool **fequal** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)  
*fequal* : test  $X = Y$ .
- template<class Field >  
void **fassign** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)  
*fassign* :  $x \leftarrow y$ .
- template<class Field >  
void **fscal** (const Field &F, const size\_t n, const typename Field::Element alpha, typename Field::Element\_ptr X, const size\_t incX)  
*fscal*  $x \leftarrow \alpha \cdot x$ .
- template<class Field >  
void **fscal** (const Field &F, const size\_t n, const typename Field::Element alpha, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY)  
*fscal*  $y \leftarrow \alpha \cdot x$ .
- template<class Field >  
void **faxpy** (const Field &F, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY)  
*faxpy* :  $y \leftarrow \alpha \cdot x + y$ .
- template<class Field >  
void **faxpby** (const Field &F, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY)  
*faxpby* :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .
- template<class Field >  
Field::Element **fdot** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)  
*fdot* : dot product  $x^T y$ .
- template<class Field >  
Field::Element **fdot** (const Field &F, const size\_t N, typename Field::ConstElement\_ptr x, const size\_t incx, typename Field::ConstElement\_ptr y, const size\_t incy, const ParSeqHelper::Sequential seq)

- `template<typename Field , class Cut , class Param >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX,`  
`typename Field::ConstElement_ptr Y, const size_t incY, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class Field >`  
`void fswap (const Field &F, const size_t N, typename Field::Element_ptr X, const size_t incX, typename`  
`Field::Element_ptr Y, const size_t incY)`  

$$fswap: X \leftrightarrow Y.$$
- `template<class Field >`  
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t`  
`lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc,`  
`const size_t numths)`
- `template<class Field >`  
`void pfsb (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t`  
`lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc,`  
`const size_t numths)`
- `template<class Field >`  
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const`  
`size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const`  
`size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`  
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename`  
`Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename`  
`Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const`  
`typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename`  
`Field::Element_ptr C, const size_t incc)`

## 17.138.1 Macro Definition Documentation

### 17.138.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL

```
#define __FFLASFFPACK_fflas_fflas_level1_INL
```

## 17.139 fflas\_level2.inl File Reference

```
#include "givaro/zring.h"
```

## Namespaces

- [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_fflas_fflas_level2_INL`

## Functions

- `template<class Field >`  
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$fassign : A \leftarrow B.$$
- `template<class Field >`  
`void fzero (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$fzero : A \leftarrow 0.$$
- `template<class Field , class RandIter >`  
`void frand (const Field &F, RandIter &G, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$frand : A \leftarrow random.$$
- `template<class Field >`  
`bool fequal (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)`  

$$fequal : test A = B.$$
- `template<class Field >`  
`bool fiszero (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda)`  

$$fiszero : test A = 0.$$
- `template<class Field >`  
`void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const typename Field::Element &d)`  

$$creates a diagonal matrix$$
- `template<class Field >`  
`void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$creates a diagonal matrix$$
- `template<class Field >`  
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$freduce A \leftarrow A mod F.$$
- `template<class Field >`  
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$freduce A \leftarrow B mod F.$$
- `template<class Field , class OtherElement_ptr >`  
`void finit (const Field &F, const size_t m, const size_t n, const OtherElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$finit A \leftarrow B mod F.$$
- `template<class Field , class OtherElement_ptr >`  
`void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$finit initializes A in F\mathbb{Z}.$$
- `template<class Field , class OtherElement_ptr >`  
`void fconvert (const Field &F, const size_t m, const size_t n, OtherElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)`  

$$fconvert A \leftarrow B mod F.$$
- `template<class Field >`  
`void fnegin (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$fnegin A \leftarrow -A.$$

- `template<class Field >`  
`void fneg (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$fneg\ A \leftarrow -B.$$
- `template<class Field >`  
`void fscaln (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda)`  

$$fscaln\ A \leftarrow a \cdot A.$$
- `template<class Field >`  
`void fscal (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  

$$fscal\ B \leftarrow a \cdot A.$$
- `template<class Field >`  
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<class Field >`  
`void faxpby (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t ldy)`  

$$faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- `template<class Field >`  
`void fmove (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  

$$fmove : A \leftarrow B\ and\ B \leftarrow 0.$$
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  

$$fadd : matrix\ addition.$$
- `template<class Field >`  
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  

$$fsub : matrix\ subtraction.$$
- `template<class Field >`  
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  

$$fsubin\ C = C - B$$
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  

$$fadd : matrix\ addition\ with\ scaling.$$
- `template<class Field >`  
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  

$$faddin$$
- `template<class Field >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE TransA, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)`  

$$finite\ prime\ Field\ GEneral\ Matrix\ Vector\ multiplication.$$
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`

*fger*: rank one update of a general matrix

- template<class Field >  
void **ftrsv** (const [Field](#) &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)

*ftrsv*: *TR*angular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

- template<class Field >  
size\_t **bitsize** (const [Field](#) &F, size\_t M, size\_t N, const typename [Field::ConstElement\\_ptr](#) A, size\_t lda)  
*bitsize*: Computes the largest bitsize of the matrix' coefficients.
- template<> size\_t **bitsize**< [Givaro::ZRing](#)< [Givaro::Integer](#) > > (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, size\_t M, size\_t N, const [Givaro::Integer](#) \*A, size\_t lda)
- template<class Field >  
void **ftmrv** (const [Field](#) &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)

*ftsm*: *TR*angular Matrix Vector prodcut Computes  $X \leftarrow \text{op}(A)X$

## 17.139.1 Macro Definition Documentation

### 17.139.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL

```
#define __FFLASFFPACK_fflas_fflas_level2_INL
```

## 17.140 fflas\_level3.inl File Reference

```
#include "fflas_bounds.inl"
#include "fflas_helpers.inl"
#include "fflas-ffpack/paladin/parallel.h"
```

## Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_level3\\_INL](#)
- #define [\\_\\_FFLAS\\_\\_TRSM\\_READONLY](#)

## Functions

- template<class Field >  
void **MatF2MatD\_Triangular** (const [Field](#) &F, [Givaro::DoubleDomain::Element\\_ptr](#) S, const size\_t lds, typename [Field::ConstElement\\_ptr](#) const E, const size\_t lde, const size\_t m, const size\_t n)
- template<class Field >  
void **MatF2MatFI\_Triangular** (const [Field](#) &F, [Givaro::FloatDomain::Element\\_ptr](#) S, const size\_t lds, typename [Field::ConstElement\\_ptr](#) const E, const size\_t lde, const size\_t m, const size\_t n)
- template<class Field >  
void **ftsm** (const [Field](#) &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)

*ftsm*: **TR**angular System solve with **M**atrix.

- template<class Field >  
void **ftmrm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb)  
*ftmrm: **TRI**angular **M**atrix **M**ultiply.*
- template<class Field >  
void **ftmrm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)  
*ftmrm: **TRI**angular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .*
- template<class Field >  
Field::Element\_ptr **fsyrk** (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)  
*fsyrk: Symmetric Rank K update*
- template<class Field >  
Field::Element\_ptr **fsyr2k** (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)  
*fsyr2k: Symmetric Rank 2K update*
- template<class Field >  
Field::Element\_ptr **fsyrk** (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- template<class Field >  
Field::Element\_ptr **fsyrk** (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const ParSeqHelper::Sequential seq, const size\_t threshold)
- template<class Field, class Cut, class Param >  
Field::Element\_ptr **fsyrk** (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size\_t threshold)
- template<class Field >  
Field::Element\_ptr **fsyrk** (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const std::vector< bool > &two←Block, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- template<typename Field >  
Field::Element\_ptr **fgemm** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)  
*fgemm: **F**ield **G**eneral **M**atrix **M**ultiply.*
- template<typename Field >  
Field::Element\_ptr **fgemm** (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const

typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const ParSeqHelper::↵  
Sequential seq)

- template<typename Field , class Cut , class Param >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE  
tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename  
[Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const  
typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const ParSeqHelper::↵  
Parallel< Cut, Param > par)
- template<typename Field >  
[Field::Element\\_ptr](#) [pfgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE  
tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename  
[Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const  
typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, size\_t numthreads=0)
- template<class Field >  
[Field::Element](#) \* [pfgemm\\_1D\\_rec](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_↵  
TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha,  
const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t  
ldb, const typename [Field::Element](#) beta, typename [Field::Element](#) \*C, const size\_t ldc, size\_t seuil)
- template<class Field >  
[Field::Element](#) \* [pfgemm\\_2D\\_rec](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_↵  
TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha,  
const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t  
ldb, const typename [Field::Element](#) beta, typename [Field::Element](#) \*C, const size\_t ldc, size\_t seuil)
- template<class Field >  
[Field::Element](#) \* [pfgemm\\_3D\\_rec](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_↵  
TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha,  
const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t  
ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, size\_t seuil,  
size\_t \*x)
- template<class Field >  
[Field::Element\\_ptr](#) [pfgemm\\_3D\\_rec2](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_↵  
TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, const  
typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const  
typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, size\_t seuil, size\_t \*x)
- template<class Field >  
[Field::Element\\_ptr](#) [fsquare](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const typename  
[Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#)  
beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)

*fsquare: Squares a matrix.*

## 17.140.1 Macro Definition Documentation

### 17.140.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL

```
#define __FFLASFFPACK_fflas_fflas_level3_INL
```

### 17.140.1.2 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

## 17.141 fflas\_lvl1.C File Reference

C functions calls for level 1 [FFLAS](#) in fflas-c.h.



```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

## Functions

- void [freducein\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [freduce\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fnegin\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [fneg\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fzero\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- bool [fiszero\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, bool positive)
- bool [fequal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fassign\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, double \*X, const size\_t incX, bool positive)
- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [faxpy\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- double [fdot\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fswap\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [fadd\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsub\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [faddin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsubin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)

### 17.141.1 Detailed Description

C functions calls for level 1 [FFLAS](#) in `fflas-c.h`.

#### Author

Brice Boyer

#### See also

[fflas/fflas\\_level1.inl](#)

### 17.141.2 Function Documentation



**17.141.2.1 freducein\_1\_modular\_double()**

```
void freducein_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.141.2.2 freduce\_1\_modular\_double()**

```
void freduce_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.141.2.3 fnegin\_1\_modular\_double()**

```
void fnegin_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.141.2.4 fneg\_1\_modular\_double()**

```
void fneg_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.141.2.5 fzero\_1\_modular\_double()**

```
void fzero_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.141.2.6 fiszero\_1\_modular\_double()**

```
bool fiszero_1_modular_double (
    const double p,
    const size_t n,
```

```
    const double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.141.2.7 fequal\_1\_modular\_double()

```
bool fequal_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.141.2.8 fassign\_1\_modular\_double()

```
void fassign_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * Y,  
    const size_t incY,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.141.2.9 fscaln\_1\_modular\_double()

```
void fscaln_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.141.2.10 fscal\_1\_modular\_double()

```
void fscal_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    const double * X,  
    const size_t incX,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.141.2.11 faxpy\_1\_modular\_double()

```
void faxpy_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,
```

```
const double * X,  
const size_t incX,  
double * Y,  
const size_t incY,  
bool positive )
```

#### 17.141.2.12 fdot\_1\_modular\_double()

```
double fdot_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.141.2.13 fswap\_1\_modular\_double()

```
void fswap_1_modular_double (  
    const double p,  
    const size_t n,  
    double * X,  
    const size_t incX,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.141.2.14 fadd\_1\_modular\_double()

```
void fadd_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * A,  
    const size_t incA,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

#### 17.141.2.15 fsub\_1\_modular\_double()

```
void fsub_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * A,  
    const size_t incA,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

**17.141.2.16 faddin\_1\_modular\_double()**

```
void faddin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.141.2.17 fsubin\_1\_modular\_double()**

```
void fsubin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.142 fflas\_lvl2.C File Reference**

C functions calls for level 2 [FFLAS](#) in `fflas-c.h`.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

**Functions**

- void [fassign\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fzero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- bool [fequal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, bool positive)
- bool [fiszero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, bool positive)
- void [fidentity\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, const double d, bool positive)
- void [freducein\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [freduce\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fnegin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [fneg\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fscalin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t lda, bool positive)
- void [fscale\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [faxpy\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)

- void [fmove\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fadd\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsub\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsubin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [faddin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- double \* [fgemv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double \*X, const size\_t incX, const double beta, double \*Y, const size\_t incY, bool positive)
- void [fger\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, double \*A, const size\_t lda, bool positive)
- void [ftrsv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t n, const double \*A, const size\_t lda, double \*X, int incX, bool positive)

### 17.142.1 Detailed Description

C functions calls for level 2 [FFLAS](#) in fflas-c.h.

Author

Brice Boyer

See also

[fflas/fflas\\_level2.inl](#)

### 17.142.2 Function Documentation

#### 17.142.2.1 fassign\_2\_modular\_double()

```
void fassign_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

#### 17.142.2.2 fzero\_2\_modular\_double()

```
void fzero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    bool positive )
```

#### 17.142.2.3 fequal\_2\_modular\_double()

```
bool fequal_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    const double * B,
    const size_t ldb,
    bool positive )
```

#### 17.142.2.4 fiszero\_2\_modular\_double()

```
bool fiszero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    bool positive )
```

#### 17.142.2.5 fidentity\_2\_modular\_double()

```
void fidentity_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    const double d,
    bool positive )
```

#### 17.142.2.6 freducein\_2\_modular\_double()

```
void freducein_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    bool positive )
```

#### 17.142.2.7 freduce\_2\_modular\_double()

```
void freduce_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.142.2.8 fnegin\_2\_modular\_double()**

```
void fnegin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    bool positive )
```

**17.142.2.9 fneg\_2\_modular\_double()**

```
void fneg_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.142.2.10 fscaln\_2\_modular\_double()**

```
void fscaln_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    double * A,
    const size_t lda,
    bool positive )
```

**17.142.2.11 fscale\_2\_modular\_double()**

```
void fscale_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.142.2.12 faxpy\_2\_modular\_double()**

```
void faxpy_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
```

```
double * B,  
const size_t ldb,  
bool positive )
```

#### 17.142.2.13 fmove\_2\_modular\_double()

```
void fmove_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,  
    bool positive )
```

#### 17.142.2.14 fadd\_2\_modular\_double()

```
void fadd_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t lda,  
    const double * B,  
    const size_t ldb,  
    double * C,  
    const size_t ldc,  
    bool positive )
```

#### 17.142.2.15 fsub\_2\_modular\_double()

```
void fsub_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t lda,  
    const double * B,  
    const size_t ldb,  
    double * C,  
    const size_t ldc,  
    bool positive )
```

#### 17.142.2.16 fsubin\_2\_modular\_double()

```
void fsubin_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldb,  
    double * C,  
    const size_t ldc,  
    bool positive )
```



**17.142.2.17 faddin\_2\_modular\_double()**

```
void faddin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldb,
    double * C,
    const size_t ldc,
    bool positive )
```

**17.142.2.18 fgemv\_2\_modular\_double()**

```
double* fgemv_2_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE TransA,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double * X,
    const size_t incX,
    const double beta,
    double * Y,
    const size_t incY,
    bool positive )
```

**17.142.2.19 fger\_2\_modular\_double()**

```
void fger_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t incX,
    const double * Y,
    const size_t incY,
    double * A,
    const size_t lda,
    bool positive )
```

**17.142.2.20 ftrsv\_2\_modular\_double()**

```
void ftrsv_2_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_TRANSPOSE TransA,
    const enum FFLAS_C_DIAG Diag,
    const size_t n,
    const double * A,
    const size_t lda,
```

```
double * X,
int incX,
bool positive )
```

## 17.143 fflas\_lvl3.C File Reference

C functions calls for level 3 [FFLAS](#) in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

### Functions

- void [ftrsm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- void [ftrmm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- double \* [fgemm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_TRANSPOSE](#) tB, const size\_t m, const size\_t n, const size\_t k, const double alpha, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, const double betA, double \*C, const size\_t ldC, bool positive)
- double \* [fsquare\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double betA, double \*C, const size\_t ldC, bool positive)

### 17.143.1 Detailed Description

C functions calls for level 3 [FFLAS](#) in fflas-c.h.

Author

Brice Boyer

See also

[fflas/fflas\\_level3.inl](#)

### 17.143.2 Function Documentation

#### 17.143.2.1 ftrsm\_3\_modular\_double()

```
void ftrsm_3_modular_double (
    const double p,
    const enum FFLAS\_C\_SIDE Side,
    const enum FFLAS\_C\_UPLO Uplo,
    const enum FFLAS\_C\_TRANSPOSE tA,
    const enum FFLAS\_C\_DIAG Diag,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
```

```
    const size_t ldA,  
    double * B,  
    const size_t ldB,  
    bool positive )
```

#### 17.143.2.2 ftrmm\_3\_modular\_double()

```
void ftrmm_3_modular_double (   
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const enum FFLAS_C_UPLO Uplo,  
    const enum FFLAS_C_TRANSPOSE tA,  
    const enum FFLAS_C_DIAG Diag,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    double * A,  
    const size_t ldA,  
    double * B,  
    const size_t ldB,  
    bool positive )
```

#### 17.143.2.3 fgemv\_3\_modular\_double()

```
double* fgemv_3_modular_double (   
    const double p,  
    const enum FFLAS_C_TRANSPOSE tA,  
    const enum FFLAS_C_TRANSPOSE tB,  
    const size_t m,  
    const size_t n,  
    const size_t k,  
    const double alpha,  
    const double * A,  
    const size_t ldA,  
    const double * B,  
    const size_t ldB,  
    const double betaA,  
    double * C,  
    const size_t ldC,  
    bool positive )
```

#### 17.143.2.4 fsquare\_3\_modular\_double()

```
double* fsquare_3_modular_double (   
    const double p,  
    const enum FFLAS_C_TRANSPOSE tA,  
    const size_t n,  
    const double alpha,  
    const double * A,  
    const size_t ldA,  
    const double betaA,  
    double * C,  
    const size_t ldC,  
    bool positive )
```

## 17.144 fflas\_memory.h File Reference

```
#include "fflas-ffpack/utils/align-allocator.h"
#include <givaro/givinteger.h>
```

### Namespaces

- [FFLAS](#)

### Functions

- `template<class Element >`  
`bool alignable ()`
- `template<> bool alignable< Givaro::Integer * > ()`
- `template<class Field >`  
`Field::Element_ptr fflas_new (const Field &F, const size_t m, const Alignment align=Alignment::DEFAULT)`
- `template<class Field >`  
`Field::Element_ptr fflas_new (const Field &F, const size_t m, const size_t n, const Alignment align=Alignment::DEFAULT)`
- `template<class Element >`  
`Element * fflas_new (const size_t m, const Alignment align=Alignment::DEFAULT)`
- `template<> void fflas_delete (FFPACK::rns_double_elt_ptr A)`
- `template<class Ptr , class ... Args>`  
`void fflas_delete (Ptr p, Args ... args)`
- `void prefetch (const int64_t *)`
- `void getTLBSize (int &tlb)`
- `void queryCacheSizes (int &l1, int &l2, int &l3)`
- `int queryL1CacheSize ()`
- `int queryTopLevelCacheSize ()`

## 17.145 fflas\_pfgemm.inl File Reference

```
#include "fflas-ffpack/paladin/blockcuts.inl"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/paladin/pfgemm_variants.inl"
```

### Namespaces

- [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_pfgemm_INL`
- `#define __FFLASFFPACK_SEQPARTHRESHOLD 220`
- `#define __FFLASFFPACK_DIMKPENALTY 1`

### Functions

- `template<class Field , class ModeTrait , class Strat , class Param >`  
`std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElement< Tag > >::value, typename Field::Element_ptr >::type fgemm (const Field &F, const FFLAS::FFLAS_TRANSPOSE ta, const FFLAS::FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr`

C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat, Param > > &H)

## 17.145.1 Macro Definition Documentation

### 17.145.1.1 \_\_FFLASFFPACK\_fflas\_pfgemm\_INL

```
#define __FFLASFFPACK_fflas_pfgemm_INL
```

### 17.145.1.2 \_\_FFLASFFPACK\_SEQPARTHRESHOLD

```
#define __FFLASFFPACK_SEQPARTHRESHOLD 220
```

### 17.145.1.3 \_\_FFLASFFPACK\_DIMKPENALTY

```
#define __FFLASFFPACK_DIMKPENALTY 1
```

## 17.146 fflas\_pftrsm.inl File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

## Namespaces

- [FFLAS](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_pftrsm\\_INL](#)
- #define [PTRSM\\_HYBRID\\_THRESHOLD](#) 256

## Functions

- template<class [Field](#) , class [Cut](#) , class [Param](#) >  
[Field::Element\\_ptr](#) ftrsm (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_TRANSPOSE](#) TA, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, TRSMHelper< [StructureHelper::Iterative](#), [ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > > &H)
- template<class [Field](#) , class [Cut](#) , class [Param](#) >  
[Field::Element\\_ptr](#) ftrsm (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_TRANSPOSE](#) TA, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, TRSMHelper< [StructureHelper::Hybrid](#), [ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > > &H)

## 17.146.1 Macro Definition Documentation

### 17.146.1.1 \_\_FFLASFFPACK\_fflas\_pftrsm\_INL

```
#define __FFLASFFPACK_fflas_pftrsm_INL
```

### 17.146.1.2 PTRSM\_HYBRID\_THRESHOLD

```
#define PTRSM_HYBRID_THRESHOLD 256
```

## 17.147 fflas\_plevel1.h File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

### Namespaces

- [FFLAS](#)

### Functions

- `template<class Field >`  
`void pzero (const Field &F, size_t m, size_t n, typename Field::Element\_ptr C, size_t BS=0)`
- `template<class Field , class Randlter >`  
`void prand (const Field &F, Randlter &G, size_t m, size_t n, typename Field::Element\_ptr C, size_t BS=0)`
- `template<class Field , class Cut , class Param >`  
`Field::Element & fdot (const Field &F, const size_t N, typename Field::ConstElement\_ptr x, const size_t incx, typename Field::ConstElement\_ptr y, const size_t incy, typename Field::Element &d, const ParSeqHelper<  
::Parallel< Cut, Param > par)`
- `template<typename Field , class Cut , class Param >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement\_ptr X, const size_t incX, typename Field::ConstElement\_ptr Y, const size_t incY, const ParSeqHelper::Parallel< Cut, Param > par)`

## 17.148 fflas\_randommatrix.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Namespaces

- [FFPACK](#)

*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- `template<class Field , class Randlter >`  
`Field::Element\_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element\_ptr A, size_t lda, Randlter &G)`  
*Random non-zero Matrix.*
- `template<class Field , class Randlter >`  
`Field::Element\_ptr NonZeroRandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element\_ptr A, size_t lda)`  
*Random non-zero Matrix.*
- `template<class Field , class Randlter >`  
`Field::Element\_ptr RandomMatrix (const Field &F, size_t m, size_t n, typename Field::Element\_ptr A, size_t lda, Randlter &G)`

*Random Matrix.*

- template<class Field >  
Field::Element\_ptr RandomMatrix (const Field &F, size\_t m, size\_t n, typename Field::Element\_ptr A, size\_t lda)

*Random Matrix.*

- template<class Field, class Randlter >  
Field::Element\_ptr RandomTriangularMatrix (const Field &F, size\_t m, size\_t n, const FFLAS::FFLAS\_UPLO UpLo, const FFLAS::FFLAS\_DIAG Diag, bool nonsingular, typename Field::Element\_ptr A, size\_t lda, Randlter &G)

*Random Triangular Matrix.*

- template<class Field >  
Field::Element\_ptr RandomTriangularMatrix (const Field &F, size\_t m, size\_t n, const FFLAS::FFLAS\_UPLO UpLo, const FFLAS::FFLAS\_DIAG Diag, bool nonsingular, typename Field::Element\_ptr A, size\_t lda)

*Random Triangular Matrix.*

- size\_t RandInt (size\_t a, size\_t b)
- template<class Field, class Randlter >  
Field::Element\_ptr RandomSymmetricMatrix (const Field &F, size\_t n, bool nonsingular, typename Field::Element\_ptr A, size\_t lda, Randlter &G)

*Random Symmetric Matrix.*

- template<class Field, class Randlter >  
Field::Element\_ptr RandomMatrixWithRank (const Field &F, size\_t m, size\_t n, size\_t r, typename Field::Element\_ptr A, size\_t lda, Randlter &G)

*Random Matrix with prescribed rank.*

- template<class Field >  
Field::Element\_ptr RandomMatrixWithRank (const Field &F, size\_t m, size\_t n, size\_t r, typename Field::Element\_ptr A, size\_t lda)

*Random Matrix with prescribed rank.*

- size\_t \* RandomIndexSubset (size\_t N, size\_t R, size\_t \*P)  
*Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.*
- size\_t \* RandomPermutation (size\_t N, size\_t \*P)  
*Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.*
- void RandomRankProfileMatrix (size\_t M, size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.*
- void swapval (size\_t k, size\_t N, size\_t \*P, size\_t val)
- void RandomSymmetricRankProfileMatrix (size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random a symmetric  $R$ -subpermutation of dimension  $N \times N$  : a symmetric matrix with only  $R$  non-zeros, all equal to one, in a random rook placement.*

- template<class Field, class Randlter >  
Field::Element\_ptr RandomMatrixWithRankandRPM (const Field &F, size\_t M, size\_t N, size\_t R, typename Field::Element\_ptr A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, Randlter &G)

*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*

- template<class Field >  
Field::Element\_ptr RandomMatrixWithRankandRPM (const Field &F, size\_t M, size\_t N, size\_t R, typename Field::Element\_ptr A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)

*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*

- template<class Field, class Randlter >  
Field::Element\_ptr RandomSymmetricMatrixWithRankandRPM (const Field &F, size\_t N, size\_t R, typename Field::Element\_ptr A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, Randlter &G)

*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*

- template<class Field >  
Field::Element\_ptr RandomSymmetricMatrixWithRankandRPM (const Field &F, size\_t M, size\_t N, size\_t R, typename Field::Element\_ptr A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)

*Random Symmetric Matrix with prescribed rank and rank profile matrix* Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .

- `template<class Field , class RandIter >`

`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, RandIter &G)

*Random Matrix with prescribed rank, with random rank profile matrix* Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

- `template<class Field >`

`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)

*Random Matrix with prescribed rank, with random rank profile matrix* Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

- `template<class Field , class RandIter >`

`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, RandIter &G)

*Random Symmetric Matrix with prescribed rank, with random rank profile matrix* Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

- `template<class Field >`

`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)

*Random Symmetric Matrix with prescribed rank, with random rank profile matrix* Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

- `template<class Field >`

`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, size\_t n, const typename `Field::Element` d, typename `Field::Element_ptr` A, size\_t lda)

*Random Matrix with prescribed det.*

- `template<class Field , class RandIter >`

`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, size\_t n, const typename `Field::Element` d, typename `Field::Element_ptr` A, size\_t lda, RandIter &G)

*Random Matrix with prescribed det.*

## 17.149 fflas\_simd.h File Reference

```
#include "fflas-ffpack/utils/fflas_intrinsic.h"
#include <iostream>
#include <type_traits>
#include <limits>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include <fflas-ffpack/fflas/fflas_simd/simd_modular.inl>
```

### Data Structures

- struct `support_simd< T >`
- struct `is_simd< T >`
- struct `NoSimd< T >`
- struct `SimdChooser< T, bool, bool >`
- struct `SimdChooser< T, false, b >`
- struct `SimdChooser< T, true, false >`
- struct `SimdChooser< T, true, true >`

### Namespaces

- `FFLAS`



## Macros

- `#define SIMD_INT 1`
- `#define INLINE inline`
- `#define CONST`
- `#define PURE`
- `#define NORML_MOD(C, P, NEGP, MIN, MAX, Q, T)`
- `#define FLOAT_MOD(C, P, INVP, Q)`

## Typedefs

- `template<class T >`  
`using Simd = typename SimdChooser< T >::value`

## 17.149.1 Macro Definition Documentation

### 17.149.1.1 SIMD\_INT

```
#define SIMD_INT 1
```

### 17.149.1.2 INLINE

```
#define INLINE inline
```

### 17.149.1.3 CONST

```
#define CONST
```

### 17.149.1.4 PURE

```
#define PURE
```

### 17.149.1.5 NORML\_MOD

```
#define NORML_MOD(  
    C,  
    P,  
    NEGP,  
    MIN,  
    MAX,  
    Q,  
    T )
```

#### Value:

```
{  
    Q = greater(C, MAX);  
    T = lesser(C, MIN);  
    Q = vand(Q, NEGP);  
    T = vand(T, P);  
    Q = vor(Q, T);  
    C = add(C, Q);  
}
```

### 17.149.1.6 FLOAT\_MOD

```
#define FLOAT_MOD(
    C,
    P,
    INVP,
    Q )
```

**Value:**

```
{
    Q = mul(C, INVP);
    Q = floor(Q);
    C = fnmadd(C, Q, P);
}
```

## 17.149.2 Typedef Documentation

### 17.149.2.1 Simd

```
using Simd = typename SimdChooser<T>::value
```

## 17.150 fflas\_sparse.C File Reference

C functions calls for level 1.5 and 2.5 [FFLAS](#) in fflas-c.h.

### 17.150.1 Detailed Description

C functions calls for level 1.5 and 2.5 [FFLAS](#) in fflas-c.h.

**Author**

Brice Boyer

**See also**

[fflas/fflas\\_sparse.h](#)

## 17.151 fflas\_sparse.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/paladin/parallel.h"
#include <recint/recint.h>
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/fflas/fflas_memory.h"
#include <type_traits>
#include <vector>
#include <iostream>
#include "fflas-ffpack/fflas/fflas_sparse/sparse_matrix_traits.h"
#include "fflas-ffpack/fflas/fflas_sparse/fflas_sparse_utils.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr.h"
#include "fflas-ffpack/fflas/fflas_sparse/coo.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell.h"
```

```
#include "fflas-ffpack/fflas/fflas_sparse/sell.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd.h"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo.h"
#include "fflas-ffpack/fflas/fflas_sparse.inl"
#include "fflas-ffpack/fflas/fflas_sparse/read_sparse.h"
```

## Data Structures

- struct [HelperFlag](#)
- struct [CsrMat< Field >](#)
- struct [CooMat< Field >](#)
- struct [EllMat< Field >](#)
- struct [SpMat< Field, flag >](#)

## Namespaces

- [MKL\\_CONFIG](#)
- [FFLAS](#)
- [FFLAS::sparse\\_details](#)

## Macros

- [#define index\\_t uint32\\_t](#)
- [#define ROUND\\_DOWN\(x, s\) \(\(x\) & ~\(\(s\)-1\)\)](#)
- [#define \\_\\_FFLASFFPACK\\_CACHE\\_LINE\\_SIZE 64](#)
- [#define assume\\_aligned\(pout, pin, v\) decltype\(pin\) pout = pin;](#)
- [#define DENSE\\_THRESHOLD 0.5](#)

## Enumerations

- enum class [SparseMatrix\\_t](#) {  
[CSR](#) , [CSR\\_ZO](#) , [CSC](#) , [CSC\\_ZO](#) ,  
[COO](#) , [COO\\_ZO](#) , [ELL](#) , [ELL\\_ZO](#) ,  
[SELL](#) , [SELL\\_ZO](#) , [ELL\\_simd](#) , [ELL\\_simd\\_ZO](#) ,  
[CSR\\_HYB](#) , [HYB\\_ZO](#) }

## Functions

- [template<class Field >](#)  
[void init\\_y](#) (const [Field](#) &F, const [size\\_t](#) m, const [typename](#) [Field::Element](#) b, [typename](#) [Field::Element\\_ptr](#) y)
- [template<class Field >](#)  
[void init\\_y](#) (const [Field](#) &F, const [size\\_t](#) m, const [size\\_t](#) n, const [typename](#) [Field::Element](#) b, [typename](#) [Field::Element\\_ptr](#) y, const [int](#) ldy)
- [template<class Field , class SM , class FC , class MZO >](#)  
[std::enable\\_if< !\(std::is\\_same< \[typename\]\(#\) \[ElementTraits\]\(#\)< \[typename\]\(#\) \[Field::Element\]\(#\) >::value, \[ElementCategories::MachineFloatTag\]\(#\) >::value\)||std::is\\_same< \[typename\]\(#\) \[ElementTraits\]\(#\)< \[typename\]\(#\) \[Field::Element\]\(#\) >::value, \[ElementCategories::MachineIntTag\]\(#\) >::value\)>::type](#) [fspmv\\_dispatch](#) (const [Field](#) &F, const [SM](#) &A, [typename](#) [Field::ConstElement\\_ptr](#) x, [typename](#) [Field::Element\\_ptr](#) y, [FC](#) fc, [MZO](#) mzo)
- [template<class Field , class SM , class FC , class MZO >](#)  
[std::enable\\_if< std::is\\_same< \[typename\]\(#\) \[ElementTraits\]\(#\)< \[typename\]\(#\) \[Field::Element\]\(#\) >::value, \[ElementCategories::MachineFloatTag\]\(#\) >::value\)||std::is\\_same< \[typename\]\(#\) \[ElementTraits\]\(#\)< \[typename\]\(#\) \[Field::Element\]\(#\) >::value, \[ElementCategories::MachineIntTag\]\(#\) >::value >::type](#) [fspmv\\_dispatch](#) (const [Field](#) &F, const [SM](#) &A, [typename](#) [Field::ConstElement\\_ptr](#) x, [typename](#) [Field::Element\\_ptr](#) y, [FC](#) fc, [MZO](#) mzo)
- [template<class Field , class SM >](#)  
[void fspmv](#) (const [Field](#) &F, const [SM](#) &A, [typename](#) [Field::ConstElement\\_ptr](#) x, [typename](#) [Field::Element\\_ptr](#) y, [FieldCategories::GenericTag](#), [NotZOSparseMatrix](#))

- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value) >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`

- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value)>::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value||std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`

- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typename Field::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::true_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::true_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field , class SM >`  
`void fspm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)`

## 17.151.1 Macro Definition Documentation

### 17.151.1.1 index\_t

```
#define index_t uint32_t
```

### 17.151.1.2 ROUND\_DOWN

```
#define ROUND_DOWN(  
    x,  
    s ) ((x) & ~((s)-1))
```

### 17.151.1.3 \_\_FFLASFFPACK\_CACHE\_LINE\_SIZE

```
#define __FFLASFFPACK_CACHE_LINE_SIZE 64
```

### 17.151.1.4 assume\_aligned

```
#define assume_aligned(  
    pout,
```

```

    pin,
    v ) decltype(pin) pout = pin;

```

### 17.151.1.5 DENSE\_THRESHOLD

```
#define DENSE_THRESHOLD 0.5
```

## 17.152 fflas\_sparse.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details](#)

### Macros

- `#define \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL`

### Functions

- `template<class Field >`  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y)
- `template<class Field >`  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y, const int ldy)
- `template<class Field , class SM , class FC , class MZO >`  
std::enable\_if< ! (std::is\_same< typename ElementTraits< typename [Field::Element](#) >::value, ElementCategories::MachineFloatTag >::value || std::is\_same< typename ElementTraits< typename [Field::Element](#) >::value, ElementCategories::MachineIntTag >::value) >::type [fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- `template<class Field , class SM , class FC , class MZO >`  
std::enable\_if< std::is\_same< typename ElementTraits< typename [Field::Element](#) >::value, ElementCategories::MachineFloatTag >::value || std::is\_same< typename ElementTraits< typename [Field::Element](#) >::value, ElementCategories::MachineIntTag >::value >::type [fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- `template<class Field , class SM >`  
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
std::enable\_if< !isSparseMatrixSimdFormat< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
std::enable\_if< isSparseMatrixSimdFormat< [Field](#), SM >::value &&support\_simd< typename [Field::Element](#) >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
std::enable\_if< !isSparseMatrixSimdFormat< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::ModularTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
std::enable\_if< isSparseMatrixSimdFormat< [Field](#), SM >::value &&support\_simd< typename [Field::Element](#) >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::ModularTag, NotZOSparseMatrix)
- `template<class Field , class SM >`  
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag, ZOSparseMatrix)

- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value && support_simd< typename Field::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< ! (std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value) >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineFloatTag >::value || std::is_same< typename ElementTraits< typename Field::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typename Field::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`



- `template<class Field, class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field, class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)`

## 17.152.1 Macro Definition Documentation

### 17.152.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL

```
#define __FFLASFFPACK_fflas_fflas_sparse_INL
```

## 17.153 ffpack-fgesv.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <fflas-ffpack/ffpack/ffpack.h>
#include <iostream>
```

### Functions

- `int main (int argc, char **argv)`

## 17.153.1 Function Documentation

### 17.153.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.154 ffpack-solve.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <fflas-ffpack/ffpack/ffpack.h>
#include <iostream>
```

### Functions

- `int main (int argc, char **argv)`

## 17.154.1 Function Documentation

### 17.154.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

PS: the function Solve will modify the matrix A so here we used a duplicate matrix A2 otherwise  $A \cdot x$  will not be equal to b for the later verification stage

## 17.155 ffpack.C File Reference

C functions calls for [FFPACK](#) in ffpack-c.h.

```
#include "fflas-ffpack/interfaces/libs/ffpack_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

### Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)
- void [MatrixApplyS\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyS\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [MatrixApplyT\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyT\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- void [cyclic\\_shift\\_row\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [cyclic\\_shift\\_col\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [applyP\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const enum [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, double \*A, const size\_t lda, const size\_t \*P, bool positive)
- void [fgetrsin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*B, const size\_t ldb, int \*info, bool positive)
- double \* [fgetrsv\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*X, const size\_t idx, const double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesvin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesv\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, double \*A, const size\_t lda, double \*X, const size\_t idx, const double \*B, const size\_t ldb, int \*info, bool positive)
- void [ftrtri\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- void [trinv\\_left\\_modular\\_double](#) (const double p, const size\_t N, const double \*L, const size\_t ldl, double \*X, const size\_t idx, bool positive)
- void [ftrtrm\\_modular\\_double](#) (const double p, const [FFLAS::FFLAS\\_SIDE](#) side, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, double \*A, const size\_t lda, bool positive)

- [illegible]

- `size_t pReducedColumnEchelonForm_modular_int32_t` (const `int32_t` p, const `size_t` M, const `size_t` N, `int32_t` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t pReducedRowEchelonForm_modular_int32_t` (const `int32_t` p, const `size_t` M, const `size_t` N, `int32_t` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `double * Invertin_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, int \*nullity, bool positive)
- `double * Invert_modular_double` (const double p, const `size_t` M, const double \*A, const `size_t` lda, double \*X, const `size_t` idx, int \*nullity, bool positive)
- `double * Invert2_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, double \*X, const `size_t` idx, int \*nullity, bool positive)
- `size_t KrylovElim_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `size_t` deg, `size_t` \*iterates, `size_t` \*inviterates, const `size_t` maxit, `size_t` virt, bool positive)
- `size_t SpecRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, const `size_t` deg, `size_t` \*rankProfile, bool positive)
- `size_t Rank_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `bool IsSingular_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `double Det_modular_double` (const double p, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `double * Solve_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, double \*x, const int incx, const double \*b, const int incb, bool positive)
- `void solveLB_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, double \*L, const `size_t` ldl, const `size_t` \*Q, double \*B, const `size_t` ldb, bool positive)
- `void solveLB2_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, double \*L, const `size_t` ldl, const `size_t` \*Q, double \*B, const `size_t` ldb, bool positive)
- `void RandomNullSpaceVector_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*X, const `size_t` incX, bool positive)
- `size_t NullSpaceBasis_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*NS, `size_t` \*ldn, `size_t` \*NSdim, bool positive)
- `size_t RowRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ColumnRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `void RankProfileFromLU` (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag)
- `size_t LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)
- `size_t RowRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t RowRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)
- `void getTriangular_modular_double` (const double p, const enum `FFLAS::FFLAS_UPLO` Uplo, const enum `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, const `size_t` R, const double \*A, const `size_t` lda, double \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, bool positive)
- `void getTriangularin_modular_double` (const double p, const enum `FFLAS::FFLAS_UPLO` Uplo, const enum `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, const `size_t` R, double \*A, const `size_t` lda, bool positive)

- void [getEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [PLUQtoEchelonPermutation](#) (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)

### 17.155.1 Detailed Description

C functions calls for [FFPACK](#) in ffpack-c.h.

#### Author

Brice Boyer

#### See also

[ffpack/ffpack.h](#)

### 17.155.2 Function Documentation

#### 17.155.2.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N )
```

#### 17.155.2.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N )
```

#### 17.155.2.3 MatrixApplyS\_modular\_double()

```
void MatrixApplyS_modular_double (
    const double p,
    double * A,
```

```
const size_t lda,  
const size_t width,  
const size_t M2,  
const size_t R1,  
const size_t R2,  
const size_t R3,  
const size_t R4,  
bool positive )
```

#### 17.155.2.4 PermApplyS\_double()

```
void PermApplyS_double (   
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t M2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4 )
```

#### 17.155.2.5 MatrixApplyT\_modular\_double()

```
void MatrixApplyT_modular_double (   
    const double p,  
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t N2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4,  
    bool positive )
```

#### 17.155.2.6 PermApplyT\_double()

```
void PermApplyT_double (   
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t N2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4 )
```

#### 17.155.2.7 composePermutationsLLM()

```
void composePermutationsLLM (   
    size_t * MathP,  
    const size_t * P1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.155.2.8 composePermutationsLLL()

```
void composePermutationsLLL (
    size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N )
```

#### 17.155.2.9 composePermutationsMLM()

```
void composePermutationsMLM (
    size_t * MathP1,
    const size_t * P2,
    const size_t R,
    const size_t N )
```

#### 17.155.2.10 cyclic\_shift\_mathPerm()

```
void cyclic_shift_mathPerm (
    size_t * P,
    const size_t s )
```

#### 17.155.2.11 cyclic\_shift\_row\_modular\_double()

```
void cyclic_shift_row_modular_double (
    const double p,
    double * A,
    size_t m,
    size_t n,
    size_t lda,
    bool positive )
```

#### 17.155.2.12 cyclic\_shift\_col\_modular\_double()

```
void cyclic_shift_col_modular_double (
    const double p,
    double * A,
    size_t m,
    size_t n,
    size_t lda,
    bool positive )
```

#### 17.155.2.13 applyP\_modular\_double()

```
void applyP_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const enum FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    double * A,
```

```
    const size_t lda,  
    const size_t * P,  
    bool positive )
```

#### 17.155.2.14 fgetrsin\_modular\_double()

```
void fgetrsin_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t R,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    const size_t * Q,  
    double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.155.2.15 fgetrsv\_modular\_double()

```
double* fgetrsv_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t NRHS,  
    const size_t R,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    const size_t * Q,  
    double * X,  
    const size_t ldx,  
    const double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.155.2.16 fgesvin\_modular\_double()

```
size_t fgesvin_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```



**17.155.2.17 fgesv\_modular\_double()**

```
size_t fgesv_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    double * A,
    const size_t lda,
    double * X,
    const size_t ldx,
    const double * B,
    const size_t ldb,
    int * info,
    bool positive )
```

**17.155.2.18 ftrtri\_modular\_double()**

```
void ftrtri_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.155.2.19 trinv\_left\_modular\_double()**

```
void trinv_left_modular_double (
    const double p,
    const size_t N,
    const double * L,
    const size_t ldl,
    double * X,
    const size_t ldx,
    bool positive )
```

**17.155.2.20 ftrtrm\_modular\_double()**

```
void ftrtrm_modular_double (
    const double p,
    const FFLAS::FFLAS_SIDE side,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.155.2.21 PLUQ\_modular\_double()**

```
size_t PLUQ_modular_double (
    const double p,
    const enum FFLAS::FFLAS_DIAG Diag,
```

```

    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    bool positive )

```

#### 17.155.2.22 LUdivine\_modular\_double()

```

size_t LUdivine_modular_double (
    const double p,
    const enum FFLAS::FFLAS_DIAG Diag,
    const enum FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const enum FFPACK_C_LU_TAG LuTag,
    const size_t cutoff,
    bool positive )

```

#### 17.155.2.23 ColumnEchelonForm\_modular\_double()

```

size_t ColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.155.2.24 RowEchelonForm\_modular\_double()

```

size_t RowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.155.2.25 ReducedColumnEchelonForm\_modular\_double()

```

size_t ReducedColumnEchelonForm_modular_double (

```

```
const double p,  
const size_t M,  
const size_t N,  
double * A,  
const size_t lda,  
size_t * P,  
size_t * Qt,  
const bool transform,  
const enum FFPACK_C_LU_TAG LuTag,  
bool positive )
```

#### 17.155.2.26 ReducedRowEchelonForm\_modular\_double()

```
size_t ReducedRowEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.155.2.27 ColumnEchelonForm\_modular\_float()

```
size_t ColumnEchelonForm_modular_float (  
    const float p,  
    const size_t M,  
    const size_t N,  
    float * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.155.2.28 RowEchelonForm\_modular\_float()

```
size_t RowEchelonForm_modular_float (  
    const float p,  
    const size_t M,  
    const size_t N,  
    float * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

**17.155.2.29 ReducedColumnEchelonForm\_modular\_float()**

```

size_t ReducedColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.30 ReducedRowEchelonForm\_modular\_float()**

```

size_t ReducedRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.31 ColumnEchelonForm\_modular\_int32\_t()**

```

size_t ColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.32 RowEchelonForm\_modular\_int32\_t()**

```

size_t RowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.33 ReducedColumnEchelonForm\_modular\_int32\_t()**

```
size_t ReducedColumnEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

**17.155.2.34 ReducedRowEchelonForm\_modular\_int32\_t()**

```
size_t ReducedRowEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

**17.155.2.35 pColumnEchelonForm\_modular\_double()**

```
size_t pColumnEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

**17.155.2.36 pRowEchelonForm\_modular\_double()**

```
size_t pRowEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

**17.155.2.37 pReducedColumnEchelonForm\_modular\_double()**

```

size_t pReducedColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.38 pReducedRowEchelonForm\_modular\_double()**

```

size_t pReducedRowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.39 pColumnEchelonForm\_modular\_float()**

```

size_t pColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.40 pRowEchelonForm\_modular\_float()**

```

size_t pRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.41 pReducedColumnEchelonForm\_modular\_float()**

```
size_t pReducedColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.155.2.42 pReducedRowEchelonForm\_modular\_float()**

```
size_t pReducedRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.155.2.43 pColumnEchelonForm\_modular\_int32\_t()**

```
size_t pColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.155.2.44 pRowEchelonForm\_modular\_int32\_t()**

```
size_t pRowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.155.2.45 pReducedColumnEchelonForm\_modular\_int32\_t()**

```

size_t pReducedColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.46 pReducedRowEchelonForm\_modular\_int32\_t()**

```

size_t pReducedRowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.47 Invertin\_modular\_double()**

```

double* Invertin_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    int * nullity,
    bool positive )

```

**17.155.2.48 Invert\_modular\_double()**

```

double* Invert_modular_double (
    const double p,
    const size_t M,
    const double * A,
    const size_t lda,
    double * X,
    const size_t ldx,
    int * nullity,
    bool positive )

```

**17.155.2.49 Invert2\_modular\_double()**

```

double* Invert2_modular_double (
    const double p,
    const size_t M,

```



```
double * A,  
const size_t lda,  
double * X,  
const size_t ldx,  
int * nullity,  
bool positive )
```

#### 17.155.2.50 KrylovElim\_modular\_double()

```
size_t KrylovElim_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Q,  
    const size_t deg,  
    size_t * iterates,  
    size_t * inviterates,  
    const size_t maxit,  
    size_t virt,  
    bool positive )
```

#### 17.155.2.51 SpecRankProfile\_modular\_double()

```
size_t SpecRankProfile_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    const size_t deg,  
    size_t * rankProfile,  
    bool positive )
```

#### 17.155.2.52 Rank\_modular\_double()

```
size_t Rank_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.155.2.53 IsSingular\_modular\_double()

```
bool IsSingular_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

**17.155.2.54 Det\_modular\_double()**

```
double Det_modular_double (
    const double p,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.155.2.55 Solve\_modular\_double()**

```
double* Solve_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    double * x,
    const int incx,
    const double * b,
    const int incb,
    bool positive )
```

**17.155.2.56 solveLB\_modular\_double()**

```
void solveLB_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.155.2.57 solveLB2\_modular\_double()**

```
void solveLB2_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.155.2.58 RandomNullSpaceVector\_modular\_double()**

```
void RandomNullSpaceVector_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * X,
    const size_t incX,
    bool positive )
```

**17.155.2.59 NullSpaceBasis\_modular\_double()**

```
size_t NullSpaceBasis_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** NS,
    size_t * ldn,
    size_t * NSdim,
    bool positive )
```

**17.155.2.60 RowRankProfile\_modular\_double()**

```
size_t RowRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.155.2.61 ColumnRankProfile\_modular\_double()**

```
size_t ColumnRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.155.2.62 RankProfileFromLU()**

```
void RankProfileFromLU (
    const size_t * P,
```

```

    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const enum FFPACK_C_LU_TAG LuTag )

```

#### 17.155.2.63 LeadingSubmatrixRankProfiles()

```

size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP )

```

#### 17.155.2.64 RowRankProfileSubmatrixIndices\_modular\_double()

```

size_t RowRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

#### 17.155.2.65 ColRankProfileSubmatrixIndices\_modular\_double()

```

size_t ColRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

#### 17.155.2.66 RowRankProfileSubmatrix\_modular\_double()

```

size_t RowRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )

```

**17.155.2.67 ColRankProfileSubmatrix\_modular\_double()**

```
size_t ColRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )
```

**17.155.2.68 getTriangular\_modular\_double()**

```
void getTriangular_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    bool positive )
```

**17.155.2.69 getTriangularin\_modular\_double()**

```
void getTriangularin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    double * A,
    const size_t lda,
    bool positive )
```

**17.155.2.70 getEchelonForm\_modular\_double()**

```
void getEchelonForm_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const double * A,
    const size_t lda,
    double * T,
```

```

    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.155.2.71 getEchelonFormin\_modular\_double()

```

void getEchelonFormin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.155.2.72 getEchelonTransform\_modular\_double()

```

void getEchelonTransform_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.155.2.73 getReducedEchelonForm\_modular\_double()

```

void getReducedEchelonForm_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.155.2.74 getReducedEchelonFormin\_modular\_double()**

```
void getReducedEchelonFormin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.155.2.75 getReducedEchelonTransform\_modular\_double()**

```
void getReducedEchelonTransform_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.155.2.76 PLUQtoEchelonPermutation()**

```
void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm )
```

**17.156 ffpack.dox File Reference****17.157 ffpack.h File Reference**

Set of elimination based routines for dense linear algebra.

```
#include "givaro/givpoly1.h"
#include <fflas-ffpack/fflas-ffpack-config.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include <list>
#include <vector>
#include <iostream>
#include <algorithm>
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "ffpack_fgesv.inl"
#include "ffpack_fgetrs.inl"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
```

```
#include "ffpack_pluq.inl"
#include "ffpack_pluq_mp.inl"
#include "ffpack_ppluq.inl"
#include "ffpack_ludivine.inl"
#include "ffpack_ludivine_mp.inl"
#include "ffpack_echelonforms.inl"
#include "ffpack_fsytrf.inl"
#include "ffpack_invert.inl"
#include "ffpack_ftrtr.inl"
#include "ffpack_ftrstr.inl"
#include "ffpack_ftrssyr2k.inl"
#include "ffpack_charpoly_kglu.inl"
#include "ffpack_charpoly_kgfast.inl"
#include "ffpack_charpoly_kgfastgeneralized.inl"
#include "ffpack_charpoly_danilevski.inl"
#include "ffpack_charpoly.inl"
#include "ffpack_frobenius.inl"
#include "ffpack_minpoly.inl"
#include "ffpack_krylovelim.inl"
#include "ffpack_permutation.inl"
#include "ffpack_rankprofiles.inl"
#include "ffpack_det_mp.inl"
#include "ffpack.inl"
```

## Data Structures

- class [CharpolyFailed](#)

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFPACK::Protected](#)

## Macros

- `#define \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD 64`
- `#define \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD 64`

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)  
*Conversion of a permutation from LAPACK format to Math format.*
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)  
*Conversion of a permutation from Maths format to LAPACK format.*
- template<class Field >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- template<class Field >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const [FFLAS::ParSeqHelper::Sequential](#) seq)



- `template<class Field , class Cut , class Param >`  
`void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class Field >`  
`void MonotonicApplyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t R)`  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- `template<class Field >`  
`void fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Solve the system  $AX = B$  or  $XA = B$ .*
- `template<class Field >`  
`Field::Element_ptr fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Solve the system  $AX = B$  or  $XA = B$ .*
- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Square system solver.*
- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Rectangular system solver.*
- `template<class Field >`  
`void ftrtri (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FTRTRI_THRESHOLD)`  
*Compute the inverse of a triangular matrix.*
- `template<class Field >`  
`void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldl, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field >`  
`void ftrtm (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda)`  
*Compute the product of two triangular matrices of opposite shape.*
- `template<class Field >`  
`void ftrstr (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const FFLAS::FFLAS_DIAG diagB, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSTR_THRESHOLD)`  
*Solve a triangular system with a triangular right hand side of the same shape.*
- `template<class Field >`  
`void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldx, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)`  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`

*Triangular factorization of symmetric matrices.*

- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`

*Triangular factorization of symmetric matrices.*

- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Q)`

*Compute a PLUQ factorization of the given matrix.*

- `template<class Field >`  
`size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PShelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut, Param > &PShelper)`
- `template<class Field >`  
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M,`  
`const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE_THRESHOLD)`

*Compute the CUP or PLE factorization of the given matrix.*

- `template<class Field >`  
`size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::ConstElement_ptr A,`  
`const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK_MINPOLY_TAG MinTag=FpackDense,`  
`const size_t kg_mc=0, const size_t kg_mb=0, const size_t kg_j=0)`
- `template<class Field >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Compute the Column Echelon form of the input matrix in-place.*

- `template<class Field >`  
`size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FpackTileRecursive)`
- `template<class Field , class PShelper >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PShelper &psh)`
- `template<class Field >`  
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Compute the Row Echelon form of the input matrix in-place.*

- `template<class Field >`  
`size_t pRowEchelonForm` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform=false, `size_t` numthreads=0, const `FFPACK_LU_TAG` LuTag=`FfpackTileRecursive`)
- `template<class Field , class PSHelper >`  
`size_t RowEchelonForm` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag, const `PSHelper` &psH)
- `template<class Field >`  
`size_t ReducedColumnEchelonForm` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform=false, const `FFPACK_LU_TAG` LuTag=`FfpackSlabRecursive`)  
*Compute the Reduced Column Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pReducedColumnEchelonForm` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform=false, `size_t` numthreads=0, const `FFPACK_LU_TAG` LuTag=`FfpackTileRecursive`)
- `template<class Field , class PSHelper >`  
`size_t ReducedColumnEchelonForm` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag, const `PSHelper` &psH)
- `template<class Field >`  
`size_t ReducedRowEchelonForm` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform=false, const `FFPACK_LU_TAG` LuTag=`FfpackSlabRecursive`)  
*Compute the Reduced Row Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pReducedRowEchelonForm` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform=false, `size_t` numthreads=0, const `FFPACK_LU_TAG` LuTag=`FfpackTileRecursive`)
- `template<class Field , class PSHelper >`  
`size_t ReducedRowEchelonForm` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag, const `PSHelper` &psH)
- `template<class Field >`  
`size_t GaussJordan` (const `Field` &F, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, const `size_t` colbeg, const `size_t` rowbeg, const `size_t` colsize, `size_t` \*P, `size_t` \*Q, const `FFPACK::FFPACK_LU_TAG` LuTag)  
*Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.*
- `template<class Field >`  
`Field::Element_ptr Invert` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, int &nullity)  
*Invert the given matrix in place or computes its nullity if it is singular.*
- `template<class Field >`  
`Field::Element_ptr Invert` (const `Field` &F, const `size_t` M, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::Element_ptr` X, const `size_t` ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- `template<class Field >`  
`Field::Element_ptr Invert2` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::Element_ptr` X, const `size_t` ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- `template<class PolRing >`  
`std::list< typename PolRing::Element > & CharPoly` (const `PolRing` &R, `std::list< typename PolRing::Element >` &charp, const `size_t` N, typename `PolRing::Domain_t::Element_ptr` A, const `size_t` lda, typename `PolRing::Domain_t::RandIter` &G, const `FFPACK_CHARPOLY_TAG` CharpTag=`FfpackAuto`, const `size_t` degree=`_FflasFfpackArithProgThreshold`)

*Compute the characteristic polynomial of the matrix A.*

- `template<class PolRing >`  
`PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, type-`  
`name PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::Randlter &G, const`  
`FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`

*Compute the characteristic polynomial of the matrix A.*

- `template<class PolRing >`  
`PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N,`  
`typename PolRing::Domain_t::Element_ptr A, const size_t lda, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto,`  
`const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`

*Compute the characteristic polynomial of the matrix A.*

- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, type-`  
`name Field::ConstElement_ptr A, const size_t lda)`
- `template<class Field , class Polynomial >`  
`int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t`  
`N, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`  
`void fgmv_kgf (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, type-`  
`name Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, const`  
`size_t kg_mc, const size_t kg_mb, const size_t kg_j)`
- `template<class Field , class Polynomial , class Randlter >`  
`std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, typename Field::Element_ptr U, const size_t ldu, Randlter &G)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, type-`  
`name Field::Element_ptr A, const size_t lda)`
- `template<class PolRing >`  
`void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors,`  
`const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename`  
`PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::Randlter &g, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`
- `template<class PolRing >`  
`std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::Element > &frobeniusForm,`  
`const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const size_t degree)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N,`  
`typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field , class Polynomial >`  
`Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr`  
`A, const size_t lda)`

*Compute the minimal polynomial of the matrix A.*

- `template<class Field , class Polynomial , class Randlter >`  
`Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr`  
`A, const size_t lda, Randlter &G)`

*Compute the minimal polynomial of the matrix A.*

- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr`  
`A, const size_t lda, typename Field::ConstElement_ptr v, const size_t incv)`

*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*

- template<class Field , class Polynomial >  
Polynomial & [MatVecMinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) v, const size\_t incv, typename [Field::Element\\_ptr](#) K, const size\_t ldk, size\_t \*P)
- template<class Field , class Polynomial >  
Polynomial & [Hybrid\\_KGF\\_LUK\\_MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, size\_t \*P, const FFPACK\_MINPOLY\_TAG MinTag=[FFPACK::FfpackDense](#), const size\_t kg\_mc=0, const size\_t kg\_↵ mb=0, const size\_t kg\_j=0)
- template<class Field >  
size\_t [Rank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Computes the rank of the given matrix using a PLUQ factorization.*
- template<class Field >  
size\_t [pRank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t numthreads=0)
- template<class Field , class PSHelper >  
size\_t [Rank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const PSHelper &psH)
- template<class Field >  
bool [IsSingular](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Returns true if the given matrix is singular.*
- template<class Field >  
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P=NULL, size\_t \*Q=NULL)  
*Returns the determinant of the given square matrix.*
- template<class Field >  
[Field::Element](#) & [pDet](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t numthreads=0, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class Field , class PSHelper >  
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const PSHelper &psH, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class Field >  
[Field::Element\\_ptr](#) [Solve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb)  
*Solves a linear system  $AX = b$  using PLUQ factorization.*
- template<class Field , class PSHelper >  
[Field::Element\\_ptr](#) [Solve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb, PSHelper &psH)
- template<class Field >  
[Field::Element\\_ptr](#) [pSolve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb, size\_t numthreads=0)
- template<class Field >  
\*void [RandomNullSpaceVector](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
*Solve  $LX = B$  or  $XL = B$  in place.*
- template<class Field >  
size\_t [NullSpaceBasis](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &NS, size\_t &ldn, size\_t ↵ &NSdim)  
*Computes a basis of the Left/Right nullspace of the matrix A.*
- template<class Field >  
size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))

*Computes the row rank profile of A.*

- template<class Field >  
size\_t [pRowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class Field , class PSHelper >  
size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- template<class Field >  
size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))

*Computes the column rank profile of A.*

- template<class Field >  
size\_t [pColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class Field , class PSHelper >  
size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- void [RankProfileFromLU](#) (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag)

*Recovers the column/row rank profile from the permutation of an LU decomposition.*

- size\_t [LeadingSubmatrixRankProfiles](#) (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)

*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*

- template<class Field >  
size\_t [RowRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rowindices, size\_t \*colindices, size\_t &R)

*RowRankProfileSubmatrixIndices.*

- template<class Field >  
size\_t [ColRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rowindices, size\_t \*colindices, size\_t &R)

*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*

- template<class Field >  
size\_t [RowRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)

*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*

- template<class Field >  
size\_t [ColRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)

*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*

- template<class Field >  
void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false)

*Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank R.*

- template<class Field >  
void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda)

*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank R.*

- template<class Field >  
void [getEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))

*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank R obtained by RowEchelonForm or ColumnEchelonForm.*



- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .*
- `template<class Field >`  
`void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- `template<class Field >`  
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- `template<class Field >`  
`Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X, const size_t ldx)`  
*LQUPtoInverseOfFullRankMinor.*

### 17.157.1 Detailed Description

Set of elimination based routines for dense linear algebra.

Matrices are supposed over finite prime field of characteristic less than  $2^{26}$ .

### 17.157.2 Macro Definition Documentation

#### 17.157.2.1 \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD

```
#define __FFLASFFPACK_FTRSTR_THRESHOLD 64
```

#### 17.157.2.2 \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD

```
#define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64
```

## 17.158 ffpack.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_INL`

### Functions

- `template<class Field >`  
`size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida)`  
*Computes the rank of the given matrix using a PLUQ factorization.*
- `template<class Field >`  
`size_t pRank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t numthreads=0)`
- `template<class Field , class PSHelper >`  
`size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, const PSHelper &psH)`
- `template<class Field >`  
`bool IsSingular (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida)`  
*Returns true if the given matrix is singular.*
- `template<class Field >`  
`Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t *P=NULL, size_t *Q=NULL)`  
*Returns the determinant of the given square matrix.*
- `template<class Field >`  
`Field::Element & pDet (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t numthreads=0, size_t *P=NULL, size_t *Q=NULL)`
- `template<class Field , class PSHelper >`  
`Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t Ida, const PSHelper &psH, size_t *P=NULL, size_t *Q=NULL)`
- `template<class Field >`  
`Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb)`  
*Solves a linear system  $AX = b$  using PLUQ factorization.*
- `template<class Field , class PSHelper >`  
`Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, PSHelper &psH)`
- `template<class Field >`  
`Field::Element_ptr pSolve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, size_t numthreads=0)`
- `template<class Field >`  
`void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr X, const size_t incX)`  
*Solve  $LX = B$  or  $XL = B$  in place.*
- `template<class Field >`  
`size_t NullSpaceBasis (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr &NS, size_t &Idn, size_t &NSdim)`



*Computes a basis of the Left/Right nullspace of the matrix A.*

- `template<class Field >`  
`void solveLB (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field >`  
`void solveLB2 (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)`

## 17.158.1 Macro Definition Documentation

### 17.158.1.1 \_\_FFLASFFPACK\_ffpack\_INL

```
#define __FFLASFFPACK_ffpack_INL
```

## 17.159 ffpack\_c.h File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>
```

### Macros

- `#define FFPACK_COMPILED`

### Enumerations

- `enum FFLAS_C_ORDER { FflasRowMajor = 101 , FflasColMajor = 102 , FflasRowMajor = 101 , FflasColMajor = 102 }`
- `enum FFLAS_C_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 , FflasNoTrans = 111 , FflasTrans = 112 }`
- `enum FFLAS_C_UPLO { FflasUpper = 121 , FflasLower = 122 , FflasUpper = 121 , FflasLower = 122 }`
- `enum FFLAS_C_DIAG { FflasNonUnit = 131 , FflasUnit = 132 , FflasNonUnit = 131 , FflasUnit = 132 }`
- `enum FFLAS_C_SIDE { FflasLeft = 141 , FflasRight = 142 , FflasLeft = 141 , FflasRight = 142 }`
- `enum FFPACK_C_LU_TAG { FfpackSlabRecursive = 1 , FfpackTileRecursive = 2 , FfpackSingular = 3 }`
- `enum FFPACK_C_CHARPOLY_TAG { FfpackLUK = 1 , FfpackKG = 2 , FfpackHybrid = 3 , FfpackKGF = 4 , FfpackDanilevski = 5 , FfpackArithProg = 6 , FfpackKGF = 7 }`
- `enum FFPACK_C_MINPOLY_TAG { FfpackDense = 1 , FfpackKGF = 2 }`

### Functions

- `void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)`
- `void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)`
- `void MatrixApplyS_modular_double (const double p, double *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, bool positive)`
- `void PermApplyS_double (double *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `void MatrixApplyT_modular_double (const double p, double *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, bool positive)`
- `void PermApplyT_double (double *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`

- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- void [cyclic\\_shift\\_row\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [cyclic\\_shift\\_col\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [applyP\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, double \*A, const size\_t lda, const size\_t \*P, bool positive)
- void [fgetrsin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*B, const size\_t ldb, int \*info, bool positive)
- double \* [fgetrs\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*X, const size\_t ldx, const double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesvin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesv\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, double \*A, const size\_t lda, double \*X, const size\_t ldx, const double \*B, const size\_t ldb, int \*info)
- void [ftrtri\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- void [trinv\\_left\\_modular\\_double](#) (const double p, const size\_t N, const double \*L, const size\_t ldl, double \*X, const size\_t ldx, bool positive)
- void [ftrtrm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- size\_t [PLUQ\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, bool positive)
- size\_t [LUdivine\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const enum [FFLAS\\_C\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, const size\_t cutoff, bool positive)
- size\_t [LUdivine\\_small\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const enum [FFLAS\\_C\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [LUdivine\\_gauss\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_float](#) (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_float](#) (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_int32\\_t](#) (const [int32\\_t](#) p, const size\_t M, const size\_t N, [int32\\_t](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_int32\\_t](#) (const [int32\\_t](#) p, const size\_t M, const size\_t N, [int32\\_t](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedColumnEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedRowEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)

- `size_t ReducedColumnEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedColumnEchelonForm_modular_int32_t` (const `int32_t` p, const `size_t` M, const `size_t` N, `int32_t` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_int32_t` (const `int32_t` p, const `size_t` M, const `size_t` N, `int32_t` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm2_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, bool positive)
- `size_t REF_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, const `size_t` colbeg, const `size_t` rowbeg, const `size_t` colsize, `size_t` \*Qt, `size_t` \*P, bool positive)
- `double * Invertin_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, int \*nullity, bool positive)
- `double * Invert_modular_double` (const double p, const `size_t` M, const double \*A, const `size_t` lda, double \*X, const `size_t` ldx, int \*nullity, bool positive)
- `double * Invert2_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, double \*X, const `size_t` ldx, int \*nullity, bool positive)
- `size_t KrylovElim_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `size_t` deg, `size_t` \*iterates, `size_t` \*inviterates, const `size_t` maxit, `size_t` virt, bool positive)
- `size_t SpecRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, const `size_t` deg, `size_t` \*rankProfile, bool positive)
- `size_t Rank_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `bool IsSingular_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `double Det_modular_double` (const double p, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `double * Solve_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, double \*x, const int incx, const double \*b, const int incb, bool positive)
- `void solveLB_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, double \*L, const `size_t` ldl, const `size_t` \*Q, double \*B, const `size_t` ldb)
- `void solveLB2_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, double \*L, const `size_t` ldl, const `size_t` \*Q, double \*B, const `size_t` ldb, bool positive)
- `void RandomNullSpaceVector_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*X, const `size_t` incX, bool positive)
- `size_t NullSpaceBasis_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*NS, `size_t` \*ldn, `size_t` \*NSdim, bool positive)
- `size_t RowRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ColumnRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `void RankProfileFromLU` (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag)
- `size_t LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)
- `size_t RowRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t RowRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)

- void [getTriangular\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, bool positive)
- void [getTriangularin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, bool positive)
- void [getEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [PLUQtoEchelonPermutation](#) (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)

## 17.159.1 Macro Definition Documentation

### 17.159.1.1 FFPACK\_COMPILED

```
#define FFPACK_COMPILED
```

## 17.159.2 Enumeration Type Documentation

### 17.159.2.1 FFLAS\_C\_ORDER

```
enum FFLAS\_C\_ORDER
```

Enumerator

FflasRowMajor	row major
FflasColMajor	col major
FflasRowMajor	
FflasColMajor	

### 17.159.2.2 FFLAS\_C\_TRANSPOSE

```
enum FFLAS\_C\_TRANSPOSE
```

## Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.
FflasNoTrans	
FflasTrans	

**17.159.2.3 FFLAS\_C\_UPLO**enum [FFLAS\\_C\\_UPLO](#)

## Enumerator

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ )
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ )
FflasUpper	
FflasLower	

**17.159.2.4 FFLAS\_C\_DIAG**enum [FFLAS\\_C\\_DIAG](#)

## Enumerator

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ )
FflasNonUnit	
FflasUnit	

**17.159.2.5 FFLAS\_C\_SIDE**enum [FFLAS\\_C\\_SIDE](#)

## Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the righth.
FflasLeft	
FflasRight	

**17.159.2.6 FFPACK\_C\_LU\_TAG**enum [FFPACK\\_C\\_LU\\_TAG](#)

## Enumerator

FfpackSlabRecursive	
FfpackTileRecursive	

**Enumerator**

FfpackSingular	
----------------	--

**17.159.2.7 FFPACK\_C\_CHARPOLY\_TAG**

enum [FFPACK\\_C\\_CHARPOLY\\_TAG](#)

**Enumerator**

FfpackLUK	
FfpackKG	
FfpackHybrid	
FfpackKGFast	
FfpackDanilevski	
FfpackArithProg	
FfpackKGFastG	

**17.159.2.8 FFPACK\_C\_MINPOLY\_TAG**

enum [FFPACK\\_C\\_MINPOLY\\_TAG](#)

**Enumerator**

FfpackDense	
FfpackKGF	

**17.159.3 Function Documentation****17.159.3.1 LAPACKPerm2MathPerm()**

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N )
```

**17.159.3.2 MathPerm2LAPACKPerm()**

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N )
```

**17.159.3.3 MatrixApplyS\_modular\_double()**

```
void MatrixApplyS_modular_double (
    const double p,
    double * A,
```

```
const size_t lda,  
const size_t width,  
const size_t M2,  
const size_t R1,  
const size_t R2,  
const size_t R3,  
const size_t R4,  
bool positive )
```

#### 17.159.3.4 PermApplyS\_double()

```
void PermApplyS_double (  
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t M2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4 )
```

#### 17.159.3.5 MatrixApplyT\_modular\_double()

```
void MatrixApplyT_modular_double (  
    const double p,  
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t N2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4,  
    bool positive )
```

#### 17.159.3.6 PermApplyT\_double()

```
void PermApplyT_double (  
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t N2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4 )
```

#### 17.159.3.7 composePermutationsLLM()

```
void composePermutationsLLM (  
    size_t * MathP,  
    const size_t * P1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

**17.159.3.8 composePermutationsLLL()**

```
void composePermutationsLLL (
    size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N )
```

**17.159.3.9 composePermutationsMLM()**

```
void composePermutationsMLM (
    size_t * MathP1,
    const size_t * P2,
    const size_t R,
    const size_t N )
```

**17.159.3.10 cyclic\_shift\_mathPerm()**

```
void cyclic_shift_mathPerm (
    size_t * P,
    const size_t s )
```

**17.159.3.11 cyclic\_shift\_row\_modular\_double()**

```
void cyclic_shift_row_modular_double (
    const double p,
    double * A,
    size_t m,
    size_t n,
    size_t lda,
    bool positive )
```

**17.159.3.12 cyclic\_shift\_col\_modular\_double()**

```
void cyclic_shift_col_modular_double (
    const double p,
    double * A,
    size_t m,
    size_t n,
    size_t lda,
    bool positive )
```

**17.159.3.13 applyP\_modular\_double()**

```
void applyP_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const enum FFLAS_C_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    double * A,
```



```
    const size_t lda,  
    const size_t * P,  
    bool positive )
```

#### 17.159.3.14 fgetrsin\_modular\_double()

```
void fgetrsin_modular_double (  
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t R,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    const size_t * Q,  
    double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.159.3.15 fgetrs\_modular\_double()

```
double* fgetrs_modular_double (  
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t NRHS,  
    const size_t R,  
    double * A,  
    const size_t lda,  
    const size_t * P,  
    const size_t * Q,  
    double * X,  
    const size_t ldx,  
    const double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.159.3.16 fgesvin\_modular\_double()

```
size_t fgesvin_modular_double (  
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

**17.159.3.17 fgesv\_modular\_double()**

```
size_t fgesv_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    double * A,
    const size_t lda,
    double * X,
    const size_t ldx,
    const double * B,
    const size_t ldb,
    int * info )
```

**17.159.3.18 ftrtri\_modular\_double()**

```
void ftrtri_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG Diag,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.159.3.19 trinv\_left\_modular\_double()**

```
void trinv_left_modular_double (
    const double p,
    const size_t N,
    const double * L,
    const size_t ldl,
    double * X,
    const size_t ldx,
    bool positive )
```

**17.159.3.20 ftrtrm\_modular\_double()**

```
void ftrtrm_modular_double (
    const double p,
    const enum FFLAS_C_DIAG diag,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.159.3.21 PLUQ\_modular\_double()**

```
size_t PLUQ_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const size_t M,
    const size_t N,
```

```
double * A,
const size_t lda,
size_t * P,
size_t * Q,
bool positive )
```

### 17.159.3.22 LUdivine\_modular\_double()

```
size_t LUdivine_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const enum FFLAS_C_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const enum FFPACK_C_LU_TAG LuTag,
    const size_t cutoff,
    bool positive )
```

### 17.159.3.23 LUdivine\_small\_modular\_double()

```
size_t LUdivine_small_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const enum FFLAS_C_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

### 17.159.3.24 LUdivine\_gauss\_modular\_double()

```
size_t LUdivine_gauss_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

### 17.159.3.25 ColumnEchelonForm\_modular\_double()

```
size_t ColumnEchelonForm_modular_double (
    const double p,
```

```

    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.159.3.26 RowEchelonForm\_modular\_double()

```

size_t RowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.159.3.27 ColumnEchelonForm\_modular\_float()

```

size_t ColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.159.3.28 RowEchelonForm\_modular\_float()

```

size_t RowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.159.3.29 ColumnEchelonForm\_modular\_int32\_t()

```

size_t ColumnEchelonForm_modular_int32_t (

```

```
const int32_t p,  
const size_t M,  
const size_t N,  
int32_t * A,  
const size_t lda,  
size_t * P,  
size_t * Qt,  
bool transform,  
const enum FFPACK_C_LU_TAG LuTag,  
bool positive )
```

### 17.159.3.30 RowEchelonForm\_modular\_int32\_t()

```
size_t RowEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

### 17.159.3.31 ReducedColumnEchelonForm\_modular\_double()

```
size_t ReducedColumnEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

### 17.159.3.32 ReducedRowEchelonForm\_modular\_double()

```
size_t ReducedRowEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

**17.159.3.33 ReducedColumnEchelonForm\_modular\_float()**

```

size_t ReducedColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.159.3.34 ReducedRowEchelonForm\_modular\_float()**

```

size_t ReducedRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.159.3.35 ReducedColumnEchelonForm\_modular\_int32\_t()**

```

size_t ReducedColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.159.3.36 ReducedRowEchelonForm\_modular\_int32\_t()**

```

size_t ReducedRowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.159.3.37 ReducedRowEchelonForm2\_modular\_double()**

```
size_t ReducedRowEchelonForm2_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    bool positive )
```

**17.159.3.38 REF\_modular\_double()**

```
size_t REF_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    const size_t colbeg,
    const size_t rowbeg,
    const size_t colsize,
    size_t * Qt,
    size_t * P,
    bool positive )
```

**17.159.3.39 Invertin\_modular\_double()**

```
double* Invertin_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    int * nullity,
    bool positive )
```

**17.159.3.40 Invert\_modular\_double()**

```
double* Invert_modular_double (
    const double p,
    const size_t M,
    const double * A,
    const size_t lda,
    double * X,
    const size_t ldx,
    int * nullity,
    bool positive )
```

**17.159.3.41 Invert2\_modular\_double()**

```
double* Invert2_modular_double (
    const double p,
    const size_t M,
```

```
double * A,  
const size_t lda,  
double * X,  
const size_t ldX,  
int * nullity,  
bool positive )
```

#### 17.159.3.42 KrylovElim\_modular\_double()

```
size_t KrylovElim_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Q,  
    const size_t deg,  
    size_t * iterates,  
    size_t * inviterates,  
    const size_t maxit,  
    size_t virt,  
    bool positive )
```

#### 17.159.3.43 SpecRankProfile\_modular\_double()

```
size_t SpecRankProfile_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    const size_t deg,  
    size_t * rankProfile,  
    bool positive )
```

#### 17.159.3.44 Rank\_modular\_double()

```
size_t Rank_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.159.3.45 IsSingular\_modular\_double()

```
bool IsSingular_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```



**17.159.3.46 Det\_modular\_double()**

```
double Det_modular_double (
    const double p,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.159.3.47 Solve\_modular\_double()**

```
double* Solve_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    double * x,
    const int incx,
    const double * b,
    const int incb,
    bool positive )
```

**17.159.3.48 solveLB\_modular\_double()**

```
void solveLB_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb )
```

**17.159.3.49 solveLB2\_modular\_double()**

```
void solveLB2_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.159.3.50 RandomNullSpaceVector\_modular\_double()**

```
void RandomNullSpaceVector_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * X,
    const size_t incX,
    bool positive )
```

**17.159.3.51 NullSpaceBasis\_modular\_double()**

```
size_t NullSpaceBasis_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** NS,
    size_t * ldn,
    size_t * NSdim,
    bool positive )
```

**17.159.3.52 RowRankProfile\_modular\_double()**

```
size_t RowRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.159.3.53 ColumnRankProfile\_modular\_double()**

```
size_t ColumnRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.159.3.54 RankProfileFromLU()**

```
void RankProfileFromLU (
    const size_t * P,
```

```

    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const enum FFPACK_C_LU_TAG LuTag )

```

#### 17.159.3.55 LeadingSubmatrixRankProfiles()

```

size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP )

```

#### 17.159.3.56 RowRankProfileSubmatrixIndices\_modular\_double()

```

size_t RowRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

#### 17.159.3.57 ColRankProfileSubmatrixIndices\_modular\_double()

```

size_t ColRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

#### 17.159.3.58 RowRankProfileSubmatrix\_modular\_double()

```

size_t RowRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )

```

**17.159.3.59 ColRankProfileSubmatrix\_modular\_double()**

```
size_t ColRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )
```

**17.159.3.60 getTriangular\_modular\_double()**

```
void getTriangular_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    bool positive )
```

**17.159.3.61 getTriangularin\_modular\_double()**

```
void getTriangularin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    double * A,
    const size_t lda,
    bool positive )
```

**17.159.3.62 getEchelonForm\_modular\_double()**

```
void getEchelonForm_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const double * A,
    const size_t lda,
    double * T,
```

```

    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.159.3.63 getEchelonFormin\_modular\_double()

```

void getEchelonFormin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.159.3.64 getEchelonTransform\_modular\_double()

```

void getEchelonTransform_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.159.3.65 getReducedEchelonForm\_modular\_double()

```

void getReducedEchelonForm_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.159.3.66 getReducedEchelonFormin\_modular\_double()**

```
void getReducedEchelonFormin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.159.3.67 getReducedEchelonTransform\_modular\_double()**

```
void getReducedEchelonTransform_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.159.3.68 PLUQtoEchelonPermutation()**

```
void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm )
```

**17.160 ffpack\_charpoly.inl File Reference**

```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "ffpack_charpoly_mp.inl"
```

**Namespaces**

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

**Macros**

- `#define` [\\_\\_FFLASFFPACK\\_charpoly\\_INL](#)

## Functions

- `template<class PolRing >`  
`std::list< typename PolRing::Element > & CharPoly (const PolRing &R, std::list< typename PolRing::Element > &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class PolRing >`  
`PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class Field , class Polynomial , class RandIter >`  
`std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr U, const size_t ldu, RandIter &G)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)`

### 17.160.1 Macro Definition Documentation

#### 17.160.1.1 \_\_FFLASFFPACK\_charpoly\_INL

```
#define __FFLASFFPACK_charpoly_INL
```

## 17.161 ffpack\_charpoly\_danilevski.inl File Reference

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_charpoly_danilveski_INL`

### Functions

- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)`

### 17.161.1 Macro Definition Documentation

#### 17.161.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_danilveski_INL
```

## 17.162 ffpack\_charpoly\_kgfast.inl File Reference

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

### Macros

- `#define __FFLASFFPACK_ffpack_charpoly_kgfast_INL`

### Functions

- `template<class Field , class Polynomial >`  
`int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element\_ptr A,`  
`const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)`
- `template<class Field >`  
`void fgemv_kgf (const Field &F, const size_t N, typename Field::ConstElement\_ptr A, const size_t lda, type-`  
`name Field::ConstElement\_ptr X, const size_t incX, typename Field::Element\_ptr Y, const size_t incY, const`  
`size_t kg_mc, const size_t kg_mb, const size_t kg_j)`

### 17.162.1 Macro Definition Documentation

#### 17.162.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfast_INL
```

## 17.163 ffpack\_charpoly\_kgfastgeneralized.inl File Reference

```
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

### Macros

- `#define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL`

### Functions

- `template<class Field >`  
`Field::Element\_ptr buildMatrix (const Field &F, typename Field::ConstElement\_ptr E, typename Field::ConstElement\_ptr`  
`C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda,`  
`const size_t mu)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t`  
`N, typename Field::Element\_ptr A, const size_t lda)`



### 17.163.1 Macro Definition Documentation

#### 17.163.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL
```

## 17.164 ffpack\_charpoly\_kglu.inl File Reference

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kglu\\_INL](#)

### Functions

- template<class Field >  
size\_t [updateD](#) (const [Field](#) &F, size\_t \*d, size\_t k, std::vector< std::vector< typename [Field::Element](#) > > &minpt)
- template<class Field >  
size\_t [newD](#) (const [Field](#) &F, size\_t \*d, bool &KeepOn, const size\_t l, const size\_t N, typename [Field::Element\\_ptr](#) X, const size\_t \*Q, std::vector< std::vector< typename [Field::Element](#) > > &minpt)
- template<class Field, class Polynomial >  
std::list< Polynomial > & [KellerGehrig](#) (const [Field](#) &F, std::list< Polynomial > &charp, const size\_t N, type-  
name [Field::ConstElement\\_ptr](#) A, const size\_t lda)

### 17.164.1 Macro Definition Documentation

#### 17.164.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kglu_INL
```

## 17.165 ffpack\_charpoly\_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "givaro/givpoly1.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFPACK\\_charpoly\\_mp\\_INL](#)

## Functions

- [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >::Element\_ptr CharPoly (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, typename [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >::Element\_ptr charp, const size\_t N, typename [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >::Element\_ptr A, const size\_t Ida, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag, size\_t degree)
- template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & [CharPoly](#) (const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R, Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp, const size\_t N, Givaro::Integer \*A, const size\_t Ida, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag, size\_t degree)

## 17.165.1 Macro Definition Documentation

### 17.165.1.1 \_\_FFPACK\_charpoly\_mp\_INL

```
#define __FFPACK_charpoly_mp_INL
```

## 17.166 ffpack\_det\_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFPACK\\_det\\_mp\\_INL](#)

## Functions

- template<class PSHelper > [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >::Element\_ptr & [Det](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, typename [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >::Element\_ptr &det, const size\_t N, typename [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >::Element\_ptr A, const size\_t Ida, const PSHelper &psH)
- template<class PSHelper > Givaro::Integer & [Det](#) (const Givaro::ZRing< Givaro::Integer > &F, Givaro::Integer &det, const size\_t N, Givaro::Integer \*A, const size\_t Ida, const PSHelper &psH, size\_t \*P, size\_t \*Q)

## 17.166.1 Macro Definition Documentation

### 17.166.1.1 \_\_FFPACK\_det\_mp\_INL

```
#define __FFPACK_det_mp_INL
```

## 17.167 ffpack\_echelonforms.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_echelon_forms_INL`
- `#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256`

### Functions

- `template<class Field >`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false)`  
*Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank  $R$ .*
- `template<class Field >`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda)`  
*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank  $R$ .*
- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or Column↔EchelonForm.*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .*
- `template<class Field >`  
`void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelon↔Form or ColumnEchelonForm.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.*

- `template<class Field >`  
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`

*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `RowEchelonForm` or `ColumnEchelonForm`.*

## 17.167.1 Macro Definition Documentation

### 17.167.1.1 \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL

```
#define __FFLASFFPACK_ffpack_echelon_forms_INL
```

### 17.167.1.2 \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256
```

## 17.168 ffpack\_fgesv.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_fgesv_INL`

### Functions

- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Square system solver.*
- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Rectangular system solver.*

## 17.168.1 Macro Definition Documentation

### 17.168.1.1 \_\_FFLASFFPACK\_ffpack\_fgesv\_INL

```
#define __FFLASFFPACK_ffpack_fgesv_INL
```

## 17.169 ffpack\_fgetrs.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_fgetrs_INL`

### Functions

- `template<class Field >`  
`void fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Solve the system  $AX = B$  or  $XA = B$ .*
- `template<class Field >`  
`Field::Element_ptr fgetrs (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, typename Field::Element_ptr A, const size_t lda, const size_t *P, const size_t *Q, typename Field::Element_ptr X, const size_t idx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Solve the system  $AX = B$  or  $XA = B$ .*

## 17.169.1 Macro Definition Documentation

### 17.169.1.1 \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL

```
#define __FFLASFFPACK_ffpack_fgetrs_INL
```

## 17.170 ffpack\_frobenius.inl File Reference

```
#include <givaro/givranditer.h>
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFPACK::Protected](#)

### Functions

- `template<class Field >`  
`void CompressRows (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQK (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`

- `template<class Field >`  
`void DeCompressRowsQK (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQA (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRowsQA (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class PolRing >`  
`void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`
- `template<class PolRing >`  
`std::list< typename PolRing::Element > &ArithProg (const PolRing &PR, std::list< typename PolRing::Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const size_t degree)`

## 17.171 ffpack\_fsytrf.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_fsytrf_INL`

### Functions

- `template<class Field >`  
`bool fsytrf_BC_Crout (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)`
- `template<class Field >`  
`size_t fsytrf_BC_RL (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_RL (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_LOW_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM (const Field &Fi, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P, size_t BCThreshold)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename`

```
Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv,
FFLAS::ParSeqHelper::Sequential seq, size_t threshold)
```

- template<class Field , class Cut , class Param >  
bool `fsytrf_nonunit` (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr Dinv, const size\_t incDinv, FFLAS::ParSeqHelper::Parallel< Cut, Param > par, size\_t threshold)
- template<class Field >  
bool `fsytrf` (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const size\_t threshold=\_\_FFLASFFPACK\_FSYTRF\_THRESHOLD)  
*Triangular factorization of symmetric matrices.*
- template<class Field >  
bool `fsytrf` (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size\_t threshold=\_\_FFLASFFPACK\_FSYTRF\_THRESHOLD)
- template<class Field , class Cut , class Param >  
bool `fsytrf` (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size\_t threshold=\_\_FFLASFFPACK\_FSYTRF\_THRESHOLD)
- template<class Field >  
size\_t `fsytrf_RPM` (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t threshold)
- template<class Field >  
void `getTridiagonal` (const Field &F, const size\_t N, const size\_t R, typename Field::ConstElement\_ptr A, const size\_t lda, size\_t \*P, typename Field::Element\_ptr T, const size\_t ldt)

## 17.171.1 Macro Definition Documentation

### 17.171.1.1 \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL

```
#define __FFLASFFPACK_ffpack_fsytrf_INL
```

## 17.172 ffpack\_ftrssyr2k.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define `__FFLASFFPACK_ffpack_ftrssyr2k_INL`

## Functions

- template<class Field >  
void `ftrssyr2k` (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diagA, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, const size\_t threshold=\_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD)  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*

## 17.172.1 Macro Definition Documentation

### 17.172.1.1 \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL

```
#define __FFLASFFPACK_ffpack_ftrssyr2k_INL
```

## 17.173 ffpack\_ftrstr.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrstr\\_INL](#)

### Functions

- template<class Field >  
void [ftrstr](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diagA, const [FFLAS::FFLAS\\_DIAG](#) diagB, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#))

*Solve a triangular system with a triangular right hand side of the same shape.*

## 17.173.1 Macro Definition Documentation

### 17.173.1.1 \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL

```
#define __FFLASFFPACK_ffpack_ftrstr_INL
```

## 17.174 ffpack\_ftrtr.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1
- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrtr\\_INL](#)

### Functions

- template<class Field >  
void [ftrtri](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#))  
*Compute the inverse of a triangular matrix.*
- template<class Field >  
void [ftrtrm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)



*Compute the product of two triangular matrices of opposite shape.*

- `template<class Field >`  
`void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldl, typename Field::Element_ptr X, const size_t ldx)`

## 17.174.1 Macro Definition Documentation

### 17.174.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

### 17.174.1.2 \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL

```
#define __FFLASFFPACK_ffpack_ftrtr_INL
```

## 17.175 ffpack\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "ffpack_inst_implem.inl"
```

## Macros

- `#define __FFPACK_INST_C`
- `#define FFLAS_COMPILED`
- `#define INST_OR_DECL`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.175.1 Macro Definition Documentation

### 17.175.1.1 \_\_FFPACK\_INST\_C

```
#define __FFPACK_INST_C
```

### 17.175.1.2 FFLAS\_COMPILED

```
#define FFLAS_COMPILED
```

### 17.175.1.3 INST\_OR\_DECL

```
#define INST_OR_DECL
```

### 17.175.1.4 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.175.1.5 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.175.1.6 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.175.1.7 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.175.1.8 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.175.1.9 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.175.1.10 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.175.1.11 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.176 ffpack\_inst.h File Reference

```
#include "givaro/modular.h"  
#include "givaro/modular-balanced.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "ffpack_inst_implem.inl"
```

## Macros

- #define FFLAS\_COMPILED
- #define INST\_OR\_DECL <>
- #define FFLAS\_FIELD Givaro::ModularBalanced
- #define FFLAS\_ELT double
- #define FFLAS\_ELT float

- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.176.1 Macro Definition Documentation

### 17.176.1.1 FFLAS\_COMPILED

```
#define FFLAS_COMPILED
```

### 17.176.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

### 17.176.1.3 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.176.1.4 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.176.1.5 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.176.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.176.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.176.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.176.1.9 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.176.1.10 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.177 ffpack\_inst\_implem.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $\text{MathP}$  as a  $\text{MathPermutation}$  format.*
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes  $\text{MathP1} \times \text{Diag}(I_R, P2)$  where  $\text{MathP1}$  is a  $\text{MathPermutation}$  and  $P2$  a LAPACK permutation and store the result in  $\text{MathP1}$  as a  $\text{MathPermutation}$  format.*
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- template<typename Base\_t >  
void [cyclic\\_shift\\_row\\_col](#) (Base\_t \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [cyclic\\_shift\\_row](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, [FFLAS\\_ELT](#) \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [cyclic\\_shift\\_col](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, [FFLAS\\_ELT](#) \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [applyP](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t \*P)
- template [INST\\_OR\\_DECL](#) void [fgetrs](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgetrs](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, [FFLAS\\_ELT](#) \*X, const size\_t ldx, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) size\_t [fgesv](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) size\_t [fgesv](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*X, const size\_t ldx, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) void [ftrtri](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t threshold)
- template [INST\\_OR\\_DECL](#) void [trinv\\_left](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t N, const [FFLAS\\_ELT](#) \*L, const size\_t ldl, [FFLAS\\_ELT](#) \*X, const size\_t ldx)
- template [INST\\_OR\\_DECL](#) void [ftrtrm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda)
- template [INST\\_OR\\_DECL](#) size\_t [PLUQ](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template [INST\\_OR\\_DECL](#) size\_t [LUdivine](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const [FFPACK\\_LU\\_TAG](#) LuTag, const size\_t cutoff)
- template [INST\\_OR\\_DECL](#) size\_t [LUdivine\\_small](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK\\_LU\\_TAG](#) LuTag)

- template `INST_OR_DECL` `size_t` `LUdivine_gauss` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `RowEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `ReducedRowEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `ColumnEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `ReducedColumnEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, int &nullity)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` idx, int &nullity)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert2` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` idx, int &nullity)
- template `INST_OR_DECL` `std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MatVecMinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*V, const `size_t` incv)
- template `INST_OR_DECL` `size_t` `KrylovElim` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `size_t` deg, `size_t` \*iterates, `size_t` \*inviterates, const `size_t` maxit, `size_t` virt)
- template `INST_OR_DECL` `size_t` `SpecRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `size_t` deg, `size_t` \*rankProfile)
- template `INST_OR_DECL` `size_t` `Rank` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` bool `IsSingular` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` `FFLAS_ELT` & `Det` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `FFLAS_ELT` &det, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q)
- template `INST_OR_DECL` `FFLAS_ELT` & `Det` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `FFLAS_ELT` &det, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads >` &parH, `size_t` \*P, `size_t` \*Q)

- template `INST_OR_DECL FFLAS_ELT * Solve` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*x, const `int` incx, const `FFLAS_ELT` \*b, const `int` incb)
- template `INST_OR_DECL void solveLB` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*L, const `size_t` ldL, const `size_t` \*Q, `FFLAS_ELT` \*B, const `size_t` ldb)
- template `INST_OR_DECL void solveLB2` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*L, const `size_t` ldL, const `size_t` \*Q, `FFLAS_ELT` \*B, const `size_t` ldb)
- template `INST_OR_DECL void RandomNullSpaceVector` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` incX)
- template `INST_OR_DECL size_t NullSpaceBasis` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&NS, `size_t` &ldn, `size_t` &NSdim)
- template `INST_OR_DECL size_t RowRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL size_t ColumnRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- void `RankProfileFromLU` (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)

*Recovers the column/row rank profile from the permutation of an LU decomposition.*

- `size_t LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)

*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*

- template `INST_OR_DECL size_t RowRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rowindices, `size_t` \*&colindices, `size_t` &R)
- template `INST_OR_DECL size_t ColRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rowindices, `size_t` \*&colindices, `size_t` &R)
- template `INST_OR_DECL size_t RowRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&X, `size_t` &R)
- template `INST_OR_DECL size_t ColRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&X, `size_t` &R)
- template `INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldT, const `bool` OnlyNonZeroVectors)
- template `INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldT, const `bool` OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL void getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldT, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldT, const `bool` OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)

- template `INST_OR_DECL` void `getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `FFPACK_LU_TAG` LuTag)
- void `PLUQtoEchelonPermutation` (const `size_t` N, const `size_t` R, const `size_t` \*P, `size_t` \*outPerm)  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- template `INST_OR_DECL` `FFLAS_ELT` \* `LQUPtoInverseOfFullRankMinor` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` rank, `FFLAS_ELT` \*A\_factors, const `size_t` lda, const `size_t` \*QtPointer, `FFLAS_ELT` \*X, const `size_t` ldx)

## 17.178 ffpack\_invert.inl File Reference

### Namespaces

- `FFPACK`  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_invert_INL`

### Functions

- template<class Field >  
`Field::Element_ptr Invert` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, int &nullity)  
*Invert the given matrix in place or computes its nullity if it is singular.*
- template<class Field >  
`Field::Element_ptr Invert` (const `Field` &F, const `size_t` M, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::Element_ptr` X, const `size_t` ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class Field >  
`Field::Element_ptr Invert2` (const `Field` &F, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, typename `Field::Element_ptr` X, const `size_t` ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*

### 17.178.1 Macro Definition Documentation

#### 17.178.1.1 \_\_FFLASFFPACK\_ffpack\_invert\_INL

```
#define __FFLASFFPACK_ffpack_invert_INL
```

## 17.179 ffpack\_krylovelim.inl File Reference

### Macros

- `#define __FFLASFFPACK_ffpack_krylovelim_INL`

### 17.179.1 Macro Definition Documentation

### 17.179.1.1 \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL

```
#define __FFLASFFPACK_ffpack_krylovelim_INL
```

## 17.180 ffpack\_ludivine.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

### Data Structures

- class [callLUdivine\\_small< Element >](#)
- class [callLUdivine\\_small< double >](#)
- class [callLUdivine\\_small< float >](#)

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- [FFPACK::Protected](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_ffpack\\_ludivine\\_INL](#)

### Functions

- [template<class Field >](#)  
[size\\_t LUdivine\\_gauss](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::Element\\_ptr](#) A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
- [template<class Field >](#)  
[size\\_t LUdivine\\_small](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::Element\\_ptr](#) A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
- [template<class Field >](#)  
[size\\_t LUdivine](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::Element\\_ptr](#) A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag, const [size\\_t](#) cutoff)
- [template<class Field >](#)  
[size\\_t LUdivine\\_construct](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [size\\_t](#) M, const [size\\_t](#) N, typename [Field::ConstElement\\_ptr](#) A, const [size\\_t](#) lda, typename [Field::Element\\_ptr](#) X, const [size\\_t](#) ldx, typename [Field::Element\\_ptr](#) u, const [size\\_t](#) incu, [size\\_t](#) \*P, bool computeX, const [FFPACK::FFPACK\\_MINPOLY\\_TAG](#) MinTag, const [size\\_t](#) kg\_mc, const [size\\_t](#) kg\_mb, const [size\\_t](#) kg\_j)

## 17.180.1 Macro Definition Documentation

### 17.180.1.1 \_\_FFLASFFPACK\_ffpack\_ludivine\_INL

```
#define __FFLASFFPACK_ffpack_ludivine_INL
```



## 17.181 ffpack\_ludivine\_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack_ludivine.inl"
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFPACK\\_ludivine\\_mp\\_INL](#)

### Functions

- template<> size\_t [LUdivine](#) (const Givaro::Modular< Givaro::Integer > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename Givaro::Integer \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag, const size\_t cutoff)

## 17.181.1 Macro Definition Documentation

### 17.181.1.1 \_\_FFPACK\_ludivine\_mp\_INL

```
#define __FFPACK_ludivine_mp_INL
```

## 17.182 ffpack\_minpoly.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_minpoly\\_INL](#)

### Functions

- template<class Field , class Polynomial >  
Polynomial & [MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)  
*Compute the minimal polynomial of the matrix A.*
- template<class Field , class Polynomial , class RandIter >  
Polynomial & [MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, RandIter &G)  
*Compute the minimal polynomial of the matrix A.*

- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly` (const `Field` &F, `Polynomial` &minP, const `size_t` N, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` v, const `size_t` incv)  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly` (const `Field` &F, `Polynomial` &minP, const `size_t` N, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::Element_ptr` v, const `size_t` incv, typename `Field::Element_ptr` K, const `size_t` ldk, `size_t` \*P)
- `template<class Field , class Polynomial >`  
`Polynomial & Hybrid_KGF_LUK_MinPoly` (const `Field` &F, `Polynomial` &minP, const `size_t` N, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::Element_ptr` X, const `size_t` ldx, `size_t` \*P, const `FFPACK_MINPOLY_TAG` MinTag=`FFPACK::FfpackDense`, const `size_t` kg\_mc=0, const `size_t` kg\_↵ mb=0, const `size_t` kg\_j=0)

## 17.182.1 Macro Definition Documentation

### 17.182.1.1 \_\_FFLASFFPACK\_ffpack\_minpoly\_INL

```
#define __FFLASFFPACK_ffpack_minpoly_INL
```

## 17.183 ffpack\_permutation.inl File Reference

```
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

## Namespaces

- [FFPACK](#)

*Finite Field PACK Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFLASFFPACK_ffpack_permutation_INL`
- `#define FFLASFFPACK_PERM_BKSIZE 32`

## Functions

- `template<class Field >`  
`void MonotonicApplyP` (const `Field` &F, const `FFLAS::FFLAS_SIDE` Side, const `FFLAS::FFLAS_TRANSPOSE` Trans, const `size_t` M, const `size_t` ibeg, const `size_t` iend, typename `Field::Element_ptr` A, const `size_t` lda, const `size_t` \*P, const `size_t` R)  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- `template<class Field >`  
`void MonotonicCompress` (const `Field` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, const `size_t` incA, const `size_t` \*MathP, const `size_t` R, const `size_t` maxpiv, const `size_t` rowstomove, const `std::vector< bool >` &ispiv)
- `template<class Field >`  
`void MonotonicCompressMorePivots` (const `Field` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, type-name `Field::Element_ptr` A, const `size_t` lda, const `size_t` incA, const `size_t` \*MathP, const `size_t` R, const `size_t` rowstomove, const `size_t` lenP)
- `template<class Field >`  
`void MonotonicCompressCycles` (const `Field` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, typename `Field::Element_ptr` A, const `size_t` lda, const `size_t` incA, const `size_t` \*MathP, const `size_t` lenP)

- `template<class Field >`  
`void MonotonicExpand (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`
- `template<class Field >`  
`void applyP_block (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)`
- `template<class Field >`  
`void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field >`  
`void doApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)`  
*Conversion of a permutation from LAPACK format to Math format.*
- `void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)`  
*Conversion of a permutation from Maths format to LAPACK format.*
- `void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- `void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $MathP$  as a MathPermutation format.*

- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes MathP1 x Diag (I\_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.*
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- template<class Field >  
void [cyclic\\_shift\\_row\\_col](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
void [cyclic\\_shift\\_row](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<typename T >  
void [cyclic\\_shift\\_row](#) (const RNSIntegerMod< T > &F, typename T::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
void [cyclic\\_shift\\_col](#) (const Field &F, typename Field::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<typename T >  
void [cyclic\\_shift\\_col](#) (const RNSIntegerMod< T > &F, typename T::Element\_ptr A, size\_t m, size\_t n, size\_t lda)
- template<class Field >  
void [applyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P)  
*Computes P1 x Diag (I\_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.*
- template<class Field, class Cut, class Param >  
void [applyP](#) (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)

## 17.183.1 Macro Definition Documentation

### 17.183.1.1 \_\_FFLASFFPACK\_ffpack\_permutation\_INL

```
#define __FFLASFFPACK_ffpack_permutation_INL
```

### 17.183.1.2 FFLASFFPACK\_PERM\_BKSIZE

```
#define FFLASFFPACK_PERM_BKSIZE 32
```

## 17.184 fpack\_pluq.inl File Reference

### Namespaces

- [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_pluq\\_INL](#)
- #define [CROUT](#)

### Functions

- template<class Field >  
size\_t [PLUQ\\_basecaseV3](#) (const Field &Fi, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, typename Field::Element \*A, const size\_t lda, size\_t \*P, size\_t \*Q)

- `template<class Field >`  
`size_t PLUQ_basecaseV2` (const `Field` &Fi, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, typename `Field::Element` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q)
- `template<class Field >`  
`size_t PLUQ_basecaseCrout` (const `Field` &Fi, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Q)
- `template<class Field >`  
`size_t _PLUQ` (const `Field` &Fi, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, `size_t` BCThreshold)
- `template<class Field >`  
`size_t PLUQ` (const `Field` &F, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `FFLAS::ParSeqHelper::Sequential` &PSHelper, `size_t` BCThreshold=`__FFLASFFPACK_PLUQ_THRESHOLD`)
- `template<class Field >`  
`size_t PLUQ` (const `Field` &F, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, typename `Field::Element_ptr` A, const `size_t` lda, `size_t` \*P, `size_t` \*Q)

*Compute a PLUQ factorization of the given matrix.*

## 17.184.1 Macro Definition Documentation

### 17.184.1.1 \_\_FFLASFFPACK\_ffpack\_pluq\_INL

```
#define __FFLASFFPACK_ffpack_pluq_INL
```

### 17.184.1.2 CROUT

```
#define CROUT
```

## 17.185 ffpack\_pluq\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "givaro/givinteger.h"
#include "givaro/modular-integer.h"
```

## Namespaces

- `FFPACK`

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFPACK_pluq_mp_INL`

## Functions

- `template<class Cut , class Param >`  
`size_t PLUQ` (const `Givaro::Modular`< `Givaro::Integer` > &F, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, typename `Givaro::Integer` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, `size_t` BCThreshold, `FFLAS::ParSeqHelper::Parallel`< Cut, Param > &PSHelper)

## 17.185.1 Macro Definition Documentation

### 17.185.1.1 \_\_FFPACK\_pluq\_mp\_INL

```
#define __FFPACK_pluq_mp_INL
```

## 17.186 ffpack\_ppluq.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ppluq\\_INL](#)
- #define [\\_\\_FFLAS\\_\\_TRSM\\_READONLY](#)
- #define [PBASECASE\\_K](#) 256

### Functions

- template<class Field >  
void [threads\\_fgemm](#) (const size\_t m, const size\_t n, const size\_t r, int nbthreads, size\_t \*W1, size\_t \*W2, size\_t \*W3, size\_t gamma)
- template<class Field >  
void [threads\\_ftrsm](#) (const size\_t m, const size\_t n, int nbthreads, size\_t \*t1, size\_t \*t2)
- template<class Field >  
size\_t [PLUQ](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFLAS::ParSeqHelper::Parallel](#)<[FFLAS::CuttingStrategy::Recursive](#), [FFLAS::StrategyParameter::Threads](#) > &PHelper)
- template<class Field >  
size\_t [pPLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q)

## 17.186.1 Macro Definition Documentation

### 17.186.1.1 \_\_FFLASFFPACK\_ffpack\_ppluq\_INL

```
#define __FFLASFFPACK_ffpack_ppluq_INL
```

### 17.186.1.2 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

### 17.186.1.3 PBASECASE\_K

```
#define PBASECASE_K 256
```

## 17.187 ffpack\_rankprofiles.inl File Reference

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_rank_profiles_INL`

### Functions

- `template<class Field >`  
`size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Computes the row rank profile of A.*
- `template<class Field >`  
`size_t pRowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field, class PSHelper >`  
`size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)`
- `template<class Field >`  
`size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Computes the column rank profile of A.*
- `template<class Field >`  
`size_t pColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field, class PSHelper >`  
`size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)`
- `void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const FFPACK_LU_TAG LuTag)`  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- `size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)`  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*
- `template<class Field >`  
`size_t RowRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)`  
*RowRankProfileSubmatrixIndices.*
- `template<class Field >`  
`size_t ColRankProfileSubmatrixIndices (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)`  
*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*
- `template<class Field >`  
`size_t RowRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)`  
*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- `template<class Field >`  
`size_t ColRankProfileSubmatrix (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &X, size_t &R)`  
*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*

- `template<class Field >`  
`Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename`  
`Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X,`  
`const size_t ldx)`  
`LQUPtoInverseOfFullRankMinor.`

## 17.187.1 Macro Definition Documentation

### 17.187.1.1 \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL

```
#define __FFLASFFPACK_ffpack_rank_profiles_INL
```

## 17.188 fgemm\_classical.inl File Reference

```
#include <cmath>
#include "fflas-ffpack/field/field-traits.h"
```

### Macros

- `#define __FFLASFFPACK_fflas_fflas_fgemm_classical_INL`

## 17.188.1 Macro Definition Documentation

### 17.188.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_classical_INL
```

## 17.189 fgemm\_classical\_mp.inl File Reference

matrix multiplication with multiprecision input (either over  $\mathbb{Z}$  or over  $\mathbb{Z}/p\mathbb{Z}$ )

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

### Data Structures

- `struct MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >`
- `struct MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`
- `struct MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`

### Namespaces

- `FFLAS`



## Macros

- `#define __FFPACK_fgemmm_classical_INL`

## Functions

- `template<typename RNS , typename ParSeqTrait >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemmm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)`
- `template<typename RNS >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemmm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)`
- `template<typename RNS , typename ParSeqTrait >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemmm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)`
- `template<typename RNS , typename Cut , typename Param >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemmm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)`
- `template<class ParSeq >`  
`Givaro::Integer * fgemmm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<typename RNS , class ModeT >`  
`RNS::Element_ptr fgemmm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)`
- `template<typename RNS >`  
`RNS::Element_ptr fgemmm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)`

- `Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelper< Algo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)`
- `template<class ParSeq >  
Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelper< Algo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<size_t K1, size_t K2, class ParSeq >  
Reclnt::ruint< K1 > * fgemm (const Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Reclnt::ruint< K1 > alpha, const Reclnt::ruint< K1 > *A, const size_t lda, const Reclnt::ruint< K1 > *B, const size_t ldb, Reclnt::ruint< K1 > beta, Reclnt::ruint< K1 > *C, const size_t ldc, MMHelper< Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`

### 17.189.1 Detailed Description

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

### 17.189.2 Macro Definition Documentation

#### 17.189.2.1 \_\_FFPACK\_fgemm\_classical\_INL

```
#define __FFPACK_fgemm_classical_INL
```

## 17.190 fgemm\_winograd.inl File Reference

```
#include <stdint.h>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fgemm_classical.inl"
#include "schedule_winograd.inl"
#include "schedule_winograd_acc.inl"
#include "schedule_winograd_acc_ip.inl"
#include "schedule_winograd_ip.inl"
#include "fflas-ffpack/fflas-ffpack-config.h"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_fgemm\\_winograd\\_INL](#)
- `#define` [NEWWINO](#)

## Functions

- template<class Field >  
int [WinogradThreshold](#) (const [Field](#) &F)  
*Computes the number of recursive levels to perform.*
- template<> int [WinogradThreshold](#) (const Givaro::Modular< float > &F)
- template<> int [WinogradThreshold](#) (const [Givaro::ModularBalanced](#)< double > &F)
- template<> int [WinogradThreshold](#) (const [Givaro::ModularBalanced](#)< float > &F)
- template<class Field >  
int [WinogradSteps](#) (const [Field](#) &F, const size\_t &m)  
*Computes the number of recursive levels to perform.*
- template<class Field , class FieldMode >  
void [DynamicPeeling](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)
- template<class Field , class FieldMode >  
void [DynamicPeeling2](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)
- template<class Field , class FieldMode >  
void [WinogradCalc](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode > &H)
- template<class Field , class ModeT >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, ModeT > &H)
- template<class Field , class ModeT , class Cut , class Param >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut, Param > > &H)

## 17.190.1 Macro Definition Documentation

### 17.190.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
```

### 17.190.1.2 NEWWINO

```
#define NEWWINO
```

## 17.191 field-traits.h File Reference

Field Traits.

```
#include <type_traits>
#include "fflas-ffpack/field/rns-double-elt.h"
#include "recint/rmint.h"
#include "givaro/modular-general.h"
#include "givaro/zring.h"
```

### Data Structures

- struct [GenericTag](#)  
*generic ring.*
- struct [ModularTag](#)  
*This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`*
- struct [UnparametricTag](#)  
*If the field uses a representation with infix operators.*
- struct [DefaultTag](#)  
*No specific mode of action: use standard field operations.*
- struct [DefaultBoundedTag](#)  
*Use standard field operations, but keeps track of bounds on input and output.*
- struct [ConvertTo< T >](#)  
*Force conversion to appropriate element type of `ElementCategory T`.*
- struct [DelayedTag](#)  
*Performs field operations with delayed mod reductions. Ensures result is reduced.*
- struct [LazyTag](#)  
*Performs field operations with delayed mod only when necessary. Result may not be reduced.*
- struct [GenericTag](#)  
*default is generic*
- struct [MachineFloatTag](#)  
*float or double*
- struct [MachineIntTag](#)  
*short, int, long, long long, and unsigned variants*
- struct [FixedPrecIntTag](#)  
*Fixed precision integers above machine precision: `Givaro::recInt`.*
- struct [ArbitraryPrecIntTag](#)  
*Arbitrary precision integers: `GMP`.*
- struct [RNSElementTag](#)  
*Representation in a Residue Number System.*
- struct [ElementTraits< Element >](#)  
*Element Traits.*
- struct [ElementTraits< float >](#)
- struct [ElementTraits< double >](#)
- struct [ElementTraits< int8\\_t >](#)
- struct [ElementTraits< int16\\_t >](#)
- struct [ElementTraits< int32\\_t >](#)
- struct [ElementTraits< int64\\_t >](#)
- struct [ElementTraits< uint8\\_t >](#)
- struct [ElementTraits< uint16\\_t >](#)
- struct [ElementTraits< uint32\\_t >](#)
- struct [ElementTraits< uint64\\_t >](#)
- struct [ElementTraits< Givaro::Integer >](#)

- struct [ElementTraits](#)< [RecInt::rint](#)< K > >
  - struct [ElementTraits](#)< [RecInt::ruint](#)< K > >
  - struct [ElementTraits](#)< [RecInt::rmint](#)< K, MG > >
  - struct [ElementTraits](#)< [FFPACK::rns\\_double\\_elt](#) >
  - struct [ModeTraits](#)< [Field](#) >
- ModeTraits.*
- struct [ModeTraits](#)< [Givaro::Modular](#)< [Element](#), [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::Modular](#)< [int8\\_t](#), [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::Modular](#)< [int16\\_t](#), [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::Modular](#)< [int32\\_t](#), [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::Modular](#)< [uint8\\_t](#), [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::Modular](#)< [uint16\\_t](#), [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::Modular](#)< [uint32\\_t](#), [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::Modular](#)< [Givaro::Integer](#), [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::Modular](#)< [RecInt::ruint](#)< K >, [Compute](#) > >
  - struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [Element](#) > >
  - struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [int8\\_t](#) > >
  - struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [int16\\_t](#) > >
  - struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [int32\\_t](#) > >
  - struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [Givaro::Integer](#) > >
  - struct [ModeTraits](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > >
  - struct [ModeTraits](#)< [Givaro::ZRing](#)< [float](#) > >
  - struct [ModeTraits](#)< [Givaro::ZRing](#)< [double](#) > >
  - struct [ModeTraits](#)< [Givaro::Montgomery](#)< T > >
  - struct [FieldTraits](#)< [Field](#) >
- FieldTrait.*
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [RecInt::ruint](#)< K > > >
  - struct [FieldTraits](#)< [Givaro::Modular](#)< [Element](#) > >
  - struct [FieldTraits](#)< [Givaro::ModularBalanced](#)< [Element](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [double](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [float](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [int16\\_t](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint16\\_t](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [int32\\_t](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint32\\_t](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [int64\\_t](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [uint64\\_t](#) > >
  - struct [FieldTraits](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > >
  - struct [FieldTraits](#)< [FFPACK::RNSInteger](#)< T > >
  - struct [FieldTraits](#)< [FFPACK::RNSIntegerMod](#)< T > >
  - struct [associatedDelayedField](#)< [Field](#) >
  - struct [associatedDelayedField](#)< const [Givaro::Modular](#)< T, X > >
  - struct [associatedDelayedField](#)< const [Givaro::ModularBalanced](#)< T > >
  - struct [associatedDelayedField](#)< const [Givaro::ZRing](#)< T > >
  - struct [associatedDelayedField](#)< const [FFPACK::RNSIntegerMod](#)< RNS > >

## Namespaces

- [Reclnt](#)
- [Givaro](#)
- [FFPACK](#)

*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*

- [FFLAS](#)
- [FFLAS::FieldCategories](#)

*Traits and categories will need to be placed in a proper file later.*

- [FFLAS::ModeCategories](#)

*Specifies the mode of action for an algorithm w.r.t.*

- [FFLAS::ElementCategories](#)

### 17.191.1 Detailed Description

Field Traits.

## 17.192 field.doxy File Reference

## 17.193 flimits.h File Reference

```
#include <climits>
#include <limits>
#include <type_traits>
#include <givaro/givinteger.h>
```

## Data Structures

- struct [limits< unsigned char >](#)
- struct [limits< signed char >](#)
- struct [limits< char >](#)
- struct [limits< unsigned short int >](#)
- struct [limits< short int >](#)
- struct [limits< unsigned int >](#)
- struct [limits< int >](#)
- struct [limits< unsigned long >](#)
- struct [limits< long >](#)
- struct [limits< unsigned long long >](#)
- struct [limits< long long >](#)
- struct [limits< float >](#)
- struct [limits< double >](#)
- struct [limits< Givaro::Integer >](#)
- struct [limits< Reclnt::ruint< K > >](#)
- struct [limits< Reclnt::rint< K > >](#)

## Functions

- template<class T, class E >  
std::enable\_if< std::is\_signed< T >::value==std::is\_signed< E >::value, bool >::type [in\\_range](#) (E e)
- template<class T, class E >  
std::enable\_if<(std::is\_signed< T >::value) &&! (std::is\_signed< E >::value), bool >::type [in\\_range](#) (E e)
- template<class T, class E >  
std::enable\_if<! (std::is\_signed< T >::value) &&(std::is\_signed< E >::value), bool >::type [in\\_range](#) (E e)

## 17.193.1 Function Documentation

### 17.193.1.1 in\_range() [1/3]

```
std::enable_if<std::is_signed<T>::value == std::is_signed<E>::value, bool>::type in_range (
    E e )
```

### 17.193.1.2 in\_range() [2/3]

```
std::enable_if<(std::is_signed<T>::value) && !(std::is_signed<E>::value), bool>::type in_↵
range (
    E e )
```

### 17.193.1.3 in\_range() [3/3]

```
std::enable_if<!(std::is_signed<T>::value) && (std::is_signed<E>::value), bool>::type in_↵
range (
    E e )
```

## 17.194 fsyrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <ctime>
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))
- #define [GFOPS](#)(n, t) ([CUBE](#)(double(n)/1000.0)/(3.0\*t))

### Typedefs

- typedef Givaro::Timer [TTimer](#)

### Functions

- int [main](#) ()

## 17.194.1 Macro Definition Documentation

### 17.194.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

### 17.194.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t ) (CUBE(double(n)/1000.0)/(3.0*t))
```

## 17.194.2 Typedef Documentation

### 17.194.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.194.3 Function Documentation

### 17.194.3.1 main()

```
int main (  
    void )
```

## 17.195 fsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "fflas-ffpack/utils/fflas_randommatrix.h"  
#include <ctime>
```

### Macros

- #define CUBE(x) ((x)\*(x)\*(x))
- #define GFOPS(n, t) (CUBE(double(n)/1000.0)/(3.0\*t))

### Typedefs

- typedef Givaro::Timer TTimer

### Functions

- int main ()

## 17.195.1 Macro Definition Documentation

### 17.195.1.1 CUBE

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```



### 17.195.1.2 GFOPS

```
#define GFOPS(
    n,
    t ) (CUBE(double(n)/1000.0)/(3.0*t))
```

## 17.195.2 Typedef Documentation

### 17.195.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.195.3 Function Documentation

### 17.195.3.1 main()

```
int main (
    void )
```

## 17.196 ftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

### Macros

- #define CUBE(x) ((x)\*(x)\*(x))
- #define GFOPS(n, t) (CUBE(double(n)/1000.0)/(3.0\*t))

### Typedefs

- typedef Givaro::Timer TTimer

### Functions

- int main ()

## 17.196.1 Macro Definition Documentation

### 17.196.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

### 17.196.1.2 GFOPS

```
#define GFOPS(
    n,
    t ) (CUBE(double(n)/1000.0)/(3.0*t))
```

## 17.196.2 Typedef Documentation

### 17.196.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.196.3 Function Documentation

### 17.196.3.1 main()

```
int main (
    void )
```

## 17.197 gmp.C File Reference

```
#include <gmpxx.h>
```

### Functions

- int [main](#) ()

## 17.197.1 Function Documentation

### 17.197.1.1 main()

```
int main (
    void )
```

## 17.198 hyb\_zo.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmv.inl"
```

### Data Structures

- struct [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) >

### Namespaces

- [FFLAS](#)

## 17.199 hyb\_zo\_pspmm.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL`

### Functions

- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::↵`  
`UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, uint64\_t kmax)`

### 17.199.1 Macro Definition Documentation

#### 17.199.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL
```

## 17.200 hyb\_zo\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL`

### Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement\_ptr x, typename Field::Element\_ptr y, uint64\_t kmax)`

### 17.200.1 Macro Definition Documentation

### 17.200.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL
```

## 17.201 hyb\_zo\_spm.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_HYB\\_ZO\\_spm\\_INL](#)

### Functions

- `template<class Field >`  
void [fspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, size\_t blockSize, type-  
name [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, FieldCategories::GenericTag)
- `template<class Field >`  
void [fspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, size\_t blockSize,  
typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, FieldCategories::↔  
UnparametricTag)
- `template<class Field >`  
void [fspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, size\_t blockSize, type-  
name [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [uint64\\_t](#) kmax)

### 17.201.1 Macro Definition Documentation

#### 17.201.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spm\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spm_INL
```

## 17.202 hyb\_zo\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_HYB\\_ZO\\_spmv\\_INL](#)

### Functions

- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, typename [Field::ConstElement\\_ptr](#)  
x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag)
- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, typename [Field::ConstElement\\_ptr](#)  
x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag)
- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, typename [Field::ConstElement\\_ptr](#)  
x, typename [Field::Element\\_ptr](#) y, [uint64\\_t](#) kmax)

## 17.202.1 Macro Definition Documentation

### 17.202.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

## 17.203 hyb\_zo\_utils.inl File Reference

### Namespaces

- [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_HYB\\_ZO\\_utils\\_INL](#)

### Functions

- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A)
- template<class Field , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<typename \_Field >  
std::ostream & [operator<<](#) (std::ostream &os, const Sparse< \_Field, SparseMatrix\_t::HYB\_ZO > &A)

## 17.203.1 Macro Definition Documentation

### 17.203.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL
```

## 17.204 igemm.doxy File Reference

## 17.205 igemm.h File Reference

```
#include "igemm_kernels.h"
#include "igemm_tools.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm.inl"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

### Enumerations

- enum [number\\_kind](#) { [zero](#) =0 , [one](#) =1 , [mone](#) =-1 , [other](#) =2 }

## Functions

- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>`  
void `igemm_colmajor` (size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, `int64_t` \*C, size\_t ldc)
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>`  
void `igemm_colmajor` (size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, `int64_t` \*C, size\_t ldc)
- void `igemm` (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, const `int64_t` beta, `int64_t` \*C, size\_t ldc)
- void `igemm_` (const enum FFLAS\_ORDER Order, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, const size\_t M, const size\_t N, const size\_t K, const `int64_t` alpha, const `int64_t` \*A, const size\_t lda, const `int64_t` \*B, const size\_t ldb, const `int64_t` beta, `int64_t` \*C, const size\_t ldc)

## 17.206 igemm.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
```

## Namespaces

- [FFLAS](#)
- [FFLAS::Protected](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_igemm\\_igemm\\_INL](#)

## Functions

- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>`  
void `igemm_colmajor` (size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, `int64_t` \*C, size\_t ldc)
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>`  
void `igemm_colmajor` (size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, `int64_t` \*C, size\_t ldc)
- void `igemm` (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, const `int64_t` beta, `int64_t` \*C, size\_t ldc)
- void `igemm_` (const enum FFLAS\_ORDER Order, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, const size\_t M, const size\_t N, const size\_t K, const `int64_t` alpha, const `int64_t` \*A, const size\_t lda, const `int64_t` \*B, const size\_t ldb, const `int64_t` beta, `int64_t` \*C, const size\_t ldc)

### 17.206.1 Macro Definition Documentation

#### 17.206.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_igemm\\_igemm\\_INL](#)

```
#define __FFLASFFPACK_fflas_igemm_igemm_INL
```

## 17.207 igemm\_kernels.h File Reference

```
#include "igemm_kernels.inl"
```

## Namespaces

- [FFLAS](#)
- [FFLAS::details](#)

## Functions

- `template<enum number_kind K>`  
void [igebb44](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebb24](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebb14](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebb41](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebb21](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebb11](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebp](#) (size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blockA, size\_t lda, const [int64\\_t](#) \*blockB, size\_t ldb, [int64\\_t](#) \*C, size\_t ldc)

## 17.208 igemm\_kernels.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm_tools.h"
```

## Namespaces

- [FFLAS](#)
- [FFLAS::details](#)

## Macros

- `#define \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL`

## Functions

- `template<enum number_kind K>`  
void [igebb44](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebb24](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebb14](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- `template<enum number_kind K>`  
void [igebb41](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)

- `template<enum number_kind K>`  
`void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64\_t alpha, const int64\_t *blA, const int64\_t *blB, int64\_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64\_t alpha, const int64\_t *blA, const int64\_t *blB, int64\_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebp (size_t rows, size_t cols, size_t depth, const int64\_t alpha, const int64\_t *blockA, size_t lda, const int64\_t *blockB, size_t ldb, int64\_t *C, size_t ldc)`

## 17.208.1 Macro Definition Documentation

### 17.208.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_kernels_INL
```

## 17.209 igemm\_tools.h File Reference

```
#include "igemm_tools.inl"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::details](#)

### Functions

- `template<size_t k, bool transpose>`  
`void pack\_lhs (int64\_t *XX, const int64\_t *X, size_t ldx, size_t rows, size_t cols)`
- `template<size_t k, bool transpose>`  
`void pack\_rhs (int64\_t *XX, const int64\_t *X, size_t ldx, size_t rows, size_t cols)`
- `void gebp (size_t rows, size_t cols, size_t depth, int64\_t *C, size_t ldc, const int64\_t *blockA, size_t lda, const int64\_t *BlockB, size_t ldb, int64\_t *BlockW)`
- `void BlockingFactor (size_t &m, size_t &n, size_t &k)`

## 17.210 igemm\_tools.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

### Namespaces

- [FFLAS](#)
- [FFLAS::details](#)

### Macros

- `#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL`



## Functions

- template<size\_t k, bool transpose>  
void [pack\\_rhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- template<size\_t k, bool transpose>  
void [pack\\_lhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- void [BlockingFactor](#) (size\_t &m, size\_t &n, size\_t &k)

### 17.210.1 Macro Definition Documentation

#### 17.210.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL
```

## 17.211 instrset.h File Reference

```
#include <stdlib.h>
```

## Data Structures

- class [Const\\_int\\_t](#)< n >
- class [Const\\_uint\\_t](#)< n >
- class [Static\\_error\\_check](#)< bool >
- class [Static\\_error\\_check](#)< false >

## Macros

- #define [INSTRSET\\_H](#) 125
- #define [INSTRSET](#) 0
- #define [const\\_int](#)(n) ([Const\\_int\\_t](#) <n>())
- #define [const\\_uint](#)(n) ([Const\\_uint\\_t](#) <n>())

## Typedefs

- typedef signed char [int8\\_t](#)
- typedef unsigned char [uint8\\_t](#)
- typedef signed short int [int16\\_t](#)
- typedef unsigned short int [uint16\\_t](#)
- typedef signed int [int32\\_t](#)
- typedef unsigned int [uint32\\_t](#)
- typedef long long [int64\\_t](#)
- typedef unsigned long long [uint64\\_t](#)
- typedef [int32\\_t](#) [intptr\\_t](#)

## Functions

- int [instrset\\_detect](#) (void)
- bool [hasFMA3](#) (void)
- bool [hasFMA4](#) (void)
- bool [hasXOP](#) (void)
- bool [hasAVX512ER](#) (void)

## 17.211.1 Macro Definition Documentation

### 17.211.1.1 INSTRSET\_H

```
#define INSTRSET_H 125
```

### 17.211.1.2 INSTRSET

```
#define INSTRSET 0
```

### 17.211.1.3 const\_int

```
#define const_int(  
    n ) (Const_int_t <n>())
```

### 17.211.1.4 const\_uint

```
#define const_uint(  
    n ) (Const_uint_t<n>())
```

## 17.211.2 Typedef Documentation

### 17.211.2.1 int8\_t

```
typedef signed char int8_t
```

### 17.211.2.2 uint8\_t

```
typedef unsigned char uint8_t
```

### 17.211.2.3 int16\_t

```
typedef signed short int int16_t
```

### 17.211.2.4 uint16\_t

```
typedef unsigned short int uint16_t
```

### 17.211.2.5 int32\_t

```
typedef signed int int32_t
```

### 17.211.2.6 uint32\_t

```
typedef unsigned int uint32_t
```

### 17.211.2.7 int64\_t

```
typedef long long int64_t
```

### 17.211.2.8 uint64\_t

```
typedef unsigned long long uint64_t
```

### 17.211.2.9 intptr\_t

```
typedef int32_t intptr_t
```

## 17.211.3 Function Documentation

### 17.211.3.1 instrset\_detect()

```
int instrset_detect (  
    void )
```

### 17.211.3.2 hasFMA3()

```
bool hasFMA3 (  
    void )
```

### 17.211.3.3 hasFMA4()

```
bool hasFMA4 (  
    void )
```

### 17.211.3.4 hasXOP()

```
bool hasXOP (  
    void )
```

### 17.211.3.5 hasAVX512ER()

```
bool hasAVX512ER (  
    void )
```

## 17.212 instrset\_detect.cpp File Reference

```
#include "instrset.h"
```

### Functions

- int [instrset\\_detect](#) (void)
- bool [hasFMA3](#) (void)
- bool [hasFMA4](#) (void)
- bool [hasXOP](#) (void)
- bool [hasF16C](#) (void)
- bool [hasAVX512ER](#) (void)

## 17.212.1 Function Documentation

### 17.212.1.1 instrset\_detect()

```
int instrset_detect (  
    void )
```

### 17.212.1.2 hasFMA3()

```
bool hasFMA3 (  
    void )
```

### 17.212.1.3 hasFMA4()

```
bool hasFMA4 (  
    void )
```

### 17.212.1.4 hasXOP()

```
bool hasXOP (  
    void )
```

### 17.212.1.5 hasF16C()

```
bool hasF16C (  
    void )
```

### 17.212.1.6 hasAVX512ER()

```
bool hasAVX512ER (  
    void )
```

## 17.213 interfaces.doxy File Reference

## 17.214 kaapi\_routines.inl File Reference

### Macros

- `#define \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL`

### 17.214.1 Macro Definition Documentation

#### 17.214.1.1 `__FFLASFFPACK_KAAPI_ROUTINES_INL`

```
#define __FFLASFFPACK_KAAPI_ROUTINES_INL
```

## 17.215 lapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

## Macros

- `#define __FFLASFFPACK_CONFIGURATION`
- `#define __FFLASFFPACK_HAVE_LAPACK 1`

## Functions

- `int main ()`

### 17.215.1 Macro Definition Documentation

#### 17.215.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

#### 17.215.1.2 \_\_FFLASFFPACK\_HAVE\_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

### 17.215.2 Function Documentation

#### 17.215.2.1 main()

```
int main (
    void )
```

## 17.216 mainpage.doxy File Reference

### 17.217 Matio.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include "fflas_memory.h"
```

## Functions

- `template<class Field >`  
`Field::Element_ptr read_field (const Field &F, const char *mat_file, size_t *tni, size_t *tnj)`
- `template<class Field >`  
`std::ostream & write_field (const Field &F, std::ostream &c, typename Field::ConstElement_ptr E, int n, int m, int id, bool mapleFormat=false, bool column_major=false)`

#### 17.217.1 Function Documentation

### 17.217.1.1 read\_field()

```
Field::Element_ptr read_field (
    const Field & F,
    const char * mat_file,
    size_t * tni,
    size_t * tnj )
```

### 17.217.1.2 write\_field()

```
std::ostream& write_field (
    const Field & F,
    std::ostream & c,
    typename Field::ConstElement_ptr E,
    int n,
    int m,
    int id,
    bool mapleFormat = false,
    bool column_major = false )
```

## 17.218 matmul.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

*This example computes the matrix multiplication over a defined finite field.*

### 17.218.1 Function Documentation

#### 17.218.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the matrix multiplication over a defined finite field.

Outputs the product of the matrix given as input.

## 17.219 matmul.doxy File Reference

## 17.220 parallel.h File Reference

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/paladin/blockcuts.inl"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [index\\_t](#) size\_t
- #define [TASK](#)(M, l) {l;}

- `#define WAIT`
- `#define CHECK_DEPENDENCIES`
- `#define BARRIER`
- `#define PAR_BLOCK`
- `#define SYNCH_GROUP(Args...) {{Args};}`
- `#define NUM_THREADS 1`
- `#define MAX_THREADS 1`
- `#define READ(Args...)`
- `#define WRITE(Args...)`
- `#define READWRITE(Args...)`
- `#define CONSTREFERENCE(...)`
- `#define VALUE(...)`
- `#define BEGIN_PARALLEL_MAIN(Args...) int main(Args) {`
- `#define END_PARALLEL_MAIN(void) return 0; }`
- `#define FORBLOCK1D(iter, m, Helper, Args...)`
- `#define FOR1D(i, m, Helper, Args...)`
- `#define PARFORBLOCK1D(iter, m, Helper, Args...)`
- `#define PARFOR1D(iter, m, Helper, Args...)`
- `#define FORBLOCK2D(iter, m, n, Helper, Args...)`
- `#define FOR2D(i, j, m, n, Helper, Args...)`
- `#define PARFORBLOCK2D(iter, m, n, Helper, Args...) FORBLOCK2D(iter, m, n, Helper, Args)`
- `#define PARFOR2D(i, j, m, n, Helper, Args...) FOR2D(i, j, m, n, Helper, Args)`
- `#define COMMA ,`
- `#define MODE(...) __VA_ARGS__`
- `#define RETURNPARAM(f, P1, Args...) P1=f(Args)`
- `#define NUMARGS(...) PP_NARG_( __VA_ARGS__, PP_RSEQ_N())`
- `#define PP_NARG_(...) PP_ARG_N( __VA_ARGS__)`
- `#define PP_ARG_N( _1, _2, _3, _4, _5, _6, _7, _8, _9, _10, _11, _12, _13, _14, _15, _16, _17, _18, _19, _20, _21, _22, _23, _24, _25, _26, _27, _28, _29, _30, _31, _32, _33, _34, _35, _36, _37, _38, _39, _40, _41, _42, _43, _44, _45, _46, _47, _48, _49, _50, _51, _52, _53, _54, _55, _56, _57, _58, _59, _60, _61, _62, _63, N, ...) N`
- `#define PP_RSEQ_N()`
- `#define NOSPLIT() FFLAS::ParSeqHelper::Sequential()`
- `#define splitting_0() FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>()`
- `#define splitting_1(a) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>(a)`
- `#define splitting_2(a, c) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,c>(a)`
- `#define splitting_3(a, b, c) FFLAS::ParSeqHelper::Parallel<b,c>(a)`
- `#define splitt(_1, _2, _3, NAME, ...) NAME`
- `#define SPLITTER(...) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0)(__VA_ARGS__)`

## 17.220.1 Macro Definition Documentation

### 17.220.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.220.1.2 index\_t

```
#define index_t size_t
```

**17.220.1.3 TASK**

```
#define TASK(  
    M,  
    I ) {I;}
```

**17.220.1.4 WAIT**

```
#define WAIT
```

**17.220.1.5 CHECK\_DEPENDENCIES**

```
#define CHECK_DEPENDENCIES
```

**17.220.1.6 BARRIER**

```
#define BARRIER
```

**17.220.1.7 PAR\_BLOCK**

```
#define PAR_BLOCK
```

**17.220.1.8 SYNCH\_GROUP**

```
#define SYNCH_GROUP(  
    Args... ) {{Args};}
```

**17.220.1.9 NUM\_THREADS**

```
#define NUM_THREADS 1
```

**17.220.1.10 MAX\_THREADS**

```
#define MAX_THREADS 1
```

**17.220.1.11 READ**

```
#define READ(  
    Args... )
```

**17.220.1.12 WRITE**

```
#define WRITE(  
    Args... )
```

**17.220.1.13 READWRITE**

```
#define READWRITE(  
    Args... )
```



**17.220.1.14 CONSTREFERENCE**

```
#define CONSTREFERENCE(
    ... )
```

**17.220.1.15 VALUE**

```
#define VALUE(
    ... )
```

**17.220.1.16 BEGIN\_PARALLEL\_MAIN**

```
#define BEGIN_PARALLEL_MAIN(
    Args... ) int main(Args) {
```

**17.220.1.17 END\_PARALLEL\_MAIN**

```
#define END_PARALLEL_MAIN(
    void ) return 0; }
```

**17.220.1.18 FORBLOCK1D**

```
#define FORBLOCK1D(
    iter,
    m,
    Helper,
    Args... )
```

**Value:**

```
{ FFLAS::ForStrategy1D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
  decltype(Helper)::Param> iter(m, Helper); \
  for(iter.initialize(); !iter.isTerminated(); ++iter) \
  {Args; } }
```

**17.220.1.19 FOR1D**

```
#define FOR1D(
    i,
    m,
    Helper,
    Args... )
```

**Value:**

```
FORBLOCK1D(_internal_iterator, m, Helper, \
  for(auto i=_internal_iterator.begin(); i!=_internal_iterator.end(); ++i) \
  { Args; })
```

**17.220.1.20 PARFORBLOCK1D**

```
#define PARFORBLOCK1D(
    iter,
    m,
    Helper,
    Args... )
```

**Value:**

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter) \
{ Args; }
```

**17.220.1.21 PARFOR1D**

```
#define PARFOR1D(
    iter,
    m,
    Helper,
    Args... )
```

**Value:**

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter) \
{ Args; }
```

**17.220.1.22 FORBLOCK2D**

```
#define FORBLOCK2D(
    iter,
    m,
    n,
    Helper,
    Args... )
```

**Value:**

```
{ FFLAS::ForStrategy2D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
    decltype(Helper)::Param> iter(m,n,Helper); \
    for(iter.initialize(); !iter.isTerminated(); ++iter) \
    { Args; } }
```

**17.220.1.23 FOR2D**

```
#define FOR2D(
    i,
    j,
    m,
    n,
    Helper,
    Args... )
```

**Value:**

```
FORBLOCK2D(_internal_iterator, m, n, Helper, \
    for(auto i=_internal_iterator.ibegin(); i!=_internal_iterator.iend(); ++i) \
    for(auto j=_internal_iterator.jbegin(); j!=_internal_iterator.jend(); ++j) \
    { Args; })
```

**17.220.1.24 PARFORBLOCK2D**

```
#define PARFORBLOCK2D(
    iter,
    m,
    n,
    Helper,
    Args... ) FORBLOCK2D(iter, m, n, Helper, Args)
```

**17.220.1.25 PARFOR2D**

```
#define PARFOR2D(
    i,
    j,
    m,
    n,
    Helper,
    Args... ) FOR2D(i, j, m, n, Helper, Args)
```

### 17.220.1.26 COMMA

```
#define COMMA ,
```

### 17.220.1.27 MODE

```
#define MODE(  
    ... ) __VA_ARGS__
```

### 17.220.1.28 RETURNPARAM

```
#define RETURNPARAM(  
    f,  
    Pl,  
    Args... ) Pl=f(Args)
```

### 17.220.1.29 NUMARGS

```
#define NUMARGS(  
    ... ) PP_NARG_(__VA_ARGS__, PP_RSEQ_N())
```

### 17.220.1.30 PP\_NARG\_

```
#define PP_NARG_(  
    ... ) PP_ARG_N(__VA_ARGS__)
```

### 17.220.1.31 PP\_ARG\_N

```
#define PP_ARG_N(  
    _1,  
    _2,  
    _3,  
    _4,  
    _5,  
    _6,  
    _7,  
    _8,  
    _9,  
    _10,  
    _11,  
    _12,  
    _13,  
    _14,  
    _15,  
    _16,  
    _17,  
    _18,  
    _19,  
    _20,  
    _21,  
    _22,  
    _23,  
    _24,  
    _25,
```

```
_26,  
_27,  
_28,  
_29,  
_30,  
_31,  
_32,  
_33,  
_34,  
_35,  
_36,  
_37,  
_38,  
_39,  
_40,  
_41,  
_42,  
_43,  
_44,  
_45,  
_46,  
_47,  
_48,  
_49,  
_50,  
_51,  
_52,  
_53,  
_54,  
_55,  
_56,  
_57,  
_58,  
_59,  
_60,  
_61,  
_62,  
_63,  
N,  
... ) N
```

#### 17.220.1.32 PP\_RSEQ\_N

```
#define PP_RSEQ_N( )
```

**Value:**

```
63,62,61,60, \  
59,58,57,56,55,54,53,52,51,50, \  
49,48,47,46,45,44,43,42,41,40, \  
39,38,37,36,35,34,33,32,31,30, \  
29,28,27,26,25,24,23,22,21,20, \  
19,18,17,16,15,14,13,12,11,10, \  
9,8,7,6,5,4,3,2,1,0
```

#### 17.220.1.33 NOSPLIT

```
#define NOSPLIT( ) FFLAS::ParSeqHelper::Sequential()
```

**17.220.1.34 splitting\_0**

```
#define splitting_0( ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Threads>(a)
```

**17.220.1.35 splitting\_1**

```
#define splitting_1(
    a ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Threads>(a)
```

**17.220.1.36 splitting\_2**

```
#define splitting_2(
    a,
    c ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, c>(a)
```

**17.220.1.37 splitting\_3**

```
#define splitting_3(
    a,
    b,
    c ) FFLAS::ParSeqHelper::Parallel<b, c>(a)
```

**17.220.1.38 splitt**

```
#define splitt(
    _1,
    _2,
    _3,
    NAME,
    ... ) NAME
```

**17.220.1.39 SPLITTER**

```
#define SPLITTER(
    ... ) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0) (↔
__VA_ARGS__)
```

**17.221 pfgemm\_variants.inl File Reference****Namespaces**

- [FFLAS](#)

**Functions**

- `template<class Field, class AlgoT, class FieldTrait>`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > > &H)`
- `template<class Field, class AlgoT, class FieldTrait>`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`

```
Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDAdaptive > > &H)
```

- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > > &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoD > > &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace > > &H)`

## 17.222 pfgemv.inl File Reference

### Namespaces

- [FFLAS](#)

### Functions

- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)`
- `template<class Field , class AlgoT , class FieldTrait , class Cut >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row, Cut > > &H)`

## 17.223 pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
```

```
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

## Macros

- `#define CUBE(x) ((x)*(x)*(x))`
- `#define GFOPS(m, n, r, t) (2.0/3.0*CUBE(double(n)/1000.0) +2*m/1000.0*n/1000.0*double(r)/1000.0 - double(r)/1000.0*double(r)/1000.0*(m+n)/1000)/t`

## Typedefs

- `typedef Givaro::Timer TTimer`

## Functions

- `int main ()`

## 17.223.1 Macro Definition Documentation

### 17.223.1.1 CUBE

```
#define CUBE(
    x )  ((x)*(x)*(x))
```

### 17.223.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
    r,
    t ) (2.0/3.0*CUBE(double(n)/1000.0) +2*m/1000.0*n/1000.0*double(r)/1000.0 - double(r)/1000.0*double(r)/1000.0*(m+n)/1000)/t
```

## 17.223.2 Typedef Documentation

### 17.223.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.223.3 Function Documentation

### 17.223.3.1 main()

```
int main (
    void )
```

## 17.224 pluq.C File Reference

```
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.224.1 Function Documentation

#### 17.224.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.225 rank.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

*This example computes the rank of a matrix over a defined finite field.*

### 17.225.1 Function Documentation

#### 17.225.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the rank of a matrix over a defined finite field.  
Outputs the rank.

## 17.226 read\_sparse.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <fstream>
#include <string>
#include <cstdlib>
#include <iterator>
```



## Data Structures

- struct [Coo< Field >](#)
- struct [readMyMachineType< Field, T >](#)
- struct [readMyMachineType< Field, mpz\\_t >](#)

## Namespaces

- [FFLAS](#)
- [FFLAS::details\\_spmv](#)

## Macros

- `#define` [DNS\\_BIN\\_VER](#) 0
- `#define` [mask\\_t](#) [uint64\\_t](#)

## Functions

- `template<class Field , bool sorted = true, bool read_integer = false>`  
void [readSmsFormat](#) (const std::string &path, const [Field](#) &f, [index\\_t](#) \*&row, [index\\_t](#) \*&col, typename [Field::Element\\_ptr](#) &val, [index\\_t](#) &rowdim, [index\\_t](#) &coldim, [uint64\\_t](#) &n nz)
- `template<class Field >`  
void [readSprFormat](#) (const std::string &path, const [Field](#) &f, [index\\_t](#) \*&row, [index\\_t](#) \*&col, typename [Field::Element\\_ptr](#) &val, [index\\_t](#) &rowdim, [index\\_t](#) &coldim, [uint64\\_t](#) &n nz)
- `template<class T >`  
std::enable\_if< std::is\_integral< T >::value, int > [getDataType](#) ()
- `template<class T >`  
std::enable\_if< std::is\_floating\_point< T >::value, int > [getDataType](#) ()
- `template<class T >`  
std::enable\_if< std::is\_same< T, mpz\_t >::value, int > [getDataType](#) ()
- `template<class T >`  
int [getDataType](#) ()
- `template<class Field >`  
void [readMachineType](#) (const [Field](#) &F, typename [Field::Element](#) &modulo, typename [Field::Element\\_ptr](#) val, std::ifstream &file, const [uint64\\_t](#) dims, const [mask\\_t](#) data\_type, const [mask\\_t](#) field\_desc)
- `template<class Field >`  
void [readDnsFormat](#) (const std::string &path, const [Field](#) &F, [index\\_t](#) &rowdim, [index\\_t](#) &coldim, typename [Field::Element\\_ptr](#) &val)
- `template<class Field >`  
void [writeDnsFormat](#) (const std::string &path, const [Field](#) &F, const [index\\_t](#) &rowdim, const [index\\_t](#) &coldim, typename [Field::Element\\_ptr](#) A, [index\\_t](#) ldA)

## 17.226.1 Macro Definition Documentation

### 17.226.1.1 DNS\_BIN\_VER

```
#define DNS_BIN_VER 0
```

### 17.226.1.2 mask\_t

```
#define mask_t uint64_t
```

## 17.227 regression-check.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- bool [check1](#) ()
- bool [check2](#) ()
- bool [check3](#) ()
- bool [check4](#) ()
- bool [checkZeroDimCharpoly](#) ()
- bool [checkZeroDimMinPoly](#) ()
- bool [gf2ModularBalanced](#) ()
- int [main](#) ()

### 17.227.1 Function Documentation

#### 17.227.1.1 [check1\(\)](#)

```
bool check1 ( )
```

#### 17.227.1.2 [check2\(\)](#)

```
bool check2 ( )
```

#### 17.227.1.3 [check3\(\)](#)

```
bool check3 ( )
```

#### 17.227.1.4 [check4\(\)](#)

```
bool check4 ( )
```

#### 17.227.1.5 [checkZeroDimCharpoly\(\)](#)

```
bool checkZeroDimCharpoly ( )
```

#### 17.227.1.6 [checkZeroDimMinPoly\(\)](#)

```
bool checkZeroDimMinPoly ( )
```

#### 17.227.1.7 [gf2ModularBalanced\(\)](#)

```
bool gf2ModularBalanced ( )
```

### 17.227.1.8 main()

```
int main (
    void )
```

## 17.228 rns-double-elt.h File Reference

rns elt structure with double support

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/cast.h"
```

### Data Structures

- struct [rns\\_double\\_elt](#)
- struct [rns\\_double\\_elt\\_ptr](#)
- struct [rns\\_double\\_elt\\_cstptr](#)

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- template<> [rns\\_double\\_elt\\_ptr fflas\\_const\\_cast](#) (rns\_double\_elt\_cstptr x)
- template<> [rns\\_double\\_elt\\_cstptr fflas\\_const\\_cast](#) (rns\_double\_elt\_ptr x)

### 17.228.1 Detailed Description

rns elt structure with double support

## 17.229 rns-double-recint.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_field\\_rns\\_double\\_recint\\_INL](#)

### 17.229.1 Macro Definition Documentation

#### 17.229.1.1 \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL

```
#define __FFLASFFPACK_field_rns_double_recint_INL
```

## 17.230 rns-double.h File Reference

rns structure with double support

```
#include <iterator>
#include <vector>
#include <givaro/modular-floating.h>
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include "givaro/modular-extended.h"
#include <recint/ruint.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/field/rns-double-elt.h"
#include "rns-double.inl"
#include "rns-double-recint.inl"
```

### Data Structures

- struct [rns\\_double](#)
- struct [rns\\_double\\_extended](#)
- class [rnsRandIter](#)< [RNS](#) >

### Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFLAS](#)

### Macros

- #define [ROUND\\_DOWN](#)(x, s) ((x) & ~((s)-1))

### Functions

- template<> void [fflas\\_delete](#) ([FFPACK::rns\\_double\\_elt\\_ptr](#) A)
- template<> void [fflas\\_delete](#) ([FFPACK::rns\\_double\\_elt\\_cstptr](#) A)

#### 17.230.1 Detailed Description

rns structure with double support

#### 17.230.2 Macro Definition Documentation

##### 17.230.2.1 ROUND\_DOWN

```
#define ROUND_DOWN(
    x,
    s ) ((x) & ~((s)-1))
```

## 17.231 rns-double.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFLASFFPACK_field_rns_double_INL`

## 17.231.1 Macro Definition Documentation

### 17.231.1.1 \_\_FFLASFFPACK\_field\_rns\_double\_INL

```
#define __FFLASFFPACK_field_rns_double_INL
```

## 17.232 rns-integer-mod.h File Reference

representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)

```
#include <vector>
#include <cmath>
#include <recint/recint.h>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include <givaro/udl.h>
#include "givaro/modular-extended.h"
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas/fflas_level1.inl"
#include "fflas-ffpack/fflas/fflas_level2.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/fflas/fflas_fscal_mp.inl"
```

## Data Structures

- class [RNSIntegerMod< RNS >](#)
- class [RNSIntegerMod< RNS >::RandIter](#)

## Namespaces

- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- [FFLAS](#)

## Functions

- `template<> FFPACK::rns\_double\_elt\_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns\_double > &F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns\_double\_elt\_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns\_double > &F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`  
`void finit\_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element\_ptr A)`

- `template<typename RNS >`  
`void finit_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k,`  
`const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`  
`void fconvert_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha,`  
`Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`
- `template<typename RNS >`  
`void fconvert_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n,`  
`Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`

### 17.232.1 Detailed Description

representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)

## 17.233 rns-integer.h File Reference

representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)

```
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-double.h"
```

### Data Structures

- class `RNSInteger< RNS >`
- class `RNSInteger< RNS >::RandIter`

### Namespaces

- `FFPACK`  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*
- `FFLAS`

### Functions

- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double >`  
`&F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double >`  
`&F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`  
`void finit_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, size_t k, const`  
`Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::Element_ptr A)`
- `template<typename RNS >`  
`void fconvert_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, Givaro::Integer`  
`alpha, Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A)`

### 17.233.1 Detailed Description

representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)

## 17.234 rns.h File Reference

### Namespaces

- `FFPACK`  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

## 17.235 rns.inl File Reference

```
#include "rns-double.h"
#include "rns-integer.h"
#include "rns-integer-mod.h"
```

### Macros

- [#define \\_\\_FFLASFFPACK\\_field\\_rns\\_INL](#)

### 17.235.1 Macro Definition Documentation

#### 17.235.1.1 \_\_FFLASFFPACK\_field\_rns\_INL

```
#define __FFLASFFPACK_field_rns_INL
```

## 17.236 schedule\_bini.inl File Reference

Bini implementation.

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fgemm\\_bini\\_INL](#)

### Functions

- `template<class Field >`  
`void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size_t lda, const typename Field::Element\_ptr B, const size_t ldb, const typename Field::Element beta, const typename Field::Element\_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE base, const size_t rec_level)`

### 17.236.1 Detailed Description

Bini implementation.

### 17.236.2 Macro Definition Documentation

#### 17.236.2.1 \_\_FFLASFFPACK\_fgemm\_bini\_INL

```
#define __FFLASFFPACK_fgemm_bini_INL
```

## 17.237 schedule\_winograd.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

### Macros

- `#define __FFLASFFPACK_fgemm_winograd_INL`

### Functions

- `template<class Field , class FieldTrait , class Strat , class Param >`  
`Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

## 17.237.1 Macro Definition Documentation

### 17.237.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_INL

```
#define __FFLASFFPACK_fgemm_winograd_INL
```

## 17.238 schedule\_winograd\_acc.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

### Macros

- `#define __FFLASFFPACK_fgemm_winograd_acc_INL`

### Functions

- `template<class Field , class FieldTrait >`  
`void WinogradAcc_3_23 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_3_21 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`



```
const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,
MMHelperAlgo::Winograd, FieldTrait > &WH)
```

- `template<class Field , class FieldTrait >`  
`void WinogradAcc_2_24 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename`  
`Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename`  
`Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::`  
`Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_2_27 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename`  
`Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename`  
`Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::`  
`Winograd, FieldTrait > &WH)`

## 17.238.1 Macro Definition Documentation

### 17.238.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_INL
```

## 17.239 schedule\_winograd\_acc\_ip.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

### Macros

- `#define \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_ip\_INL`

### Functions

- `template<class Field , class FieldTrait >`  
`void WinogradAcc_LR (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const`  
`size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr`  
`A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta,`  
`typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, Field`  
`Trait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename`  
`Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename`  
`Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelper`  
`Algo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const`  
`size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr`  
`A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element`  
`beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd,`  
`FieldTrait > &WH)`

## 17.239.1 Macro Definition Documentation

### 17.239.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_ip\_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL
```

## 17.240 schedule\_winograd\_ip.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::BLAS3](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fgemm\\_winograd\\_ip\\_INL](#)

### Functions

- template<class Field , class FieldTrait >  
void [Winograd\\_LR\\_S](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const MMHelper< [Field](#), MMHelperAlgo::Winograd, Field↔Trait > &WH)
- template<class Field , class FieldTrait >  
void [Winograd\\_L\\_S](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class Field , class FieldTrait >  
void [Winograd\\_R\\_S](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, const typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const MMHelper< [Field](#), MMHelper↔Algo::Winograd, FieldTrait > &WH)

### 17.240.1 Macro Definition Documentation

#### 17.240.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_ip\_INL

```
#define __FFLASFFPACK_fgemm_winograd_ip_INL
```

## 17.241 sell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_spmv.inl"
```

### Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::SELL >](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::SELL\\_ZO >](#)

### Namespaces

- [FFLAS](#)

## 17.242 sell\_pspmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_sell\\_pspmv\\_INL](#)

### Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 17.242.1 Macro Definition Documentation

#### 17.242.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_pspmv_INL
```

## 17.243 sell\_spmv.inl File Reference

### Namespaces

- [FFLAS](#)
- [FFLAS::sparse\\_details\\_impl](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_sparse\\_sell\\_spmv\\_INL](#)

## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 17.243.1 Macro Definition Documentation

#### 17.243.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL
```

## 17.244 sell\_utils.inl File Reference

### Data Structures

- struct [Info](#)
- struct [Coo< ValT, IdxT >](#)

### Namespaces

- [FFLAS](#)
- [FFLAS::sell\\_details](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_sell\\_utils\\_INL](#)

## Functions

- template<class Field >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::ModularTag)
- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::SELL > &A)
- template<class Field >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A)
- template<class Field >  
void [sparse\\_print](#) (const Sparse< [Field](#), SparseMatrix\_t::SELL > &A)
- template<class Field , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::SELL > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz, [uint64\\_t](#) sigma=0)
- template<class Field , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::SELL\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

### 17.244.1 Macro Definition Documentation

#### 17.244.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_utils_INL
```

## 17.245 simd.dox File Reference

### 17.246 simd128.inl File Reference

```
#include "simd128_float.inl"
#include "simd128_double.inl"
```

## Data Structures

- struct [Simd128fp\\_base](#)
- struct [Simd128i\\_base](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_INL](#)

## Typedefs

- template<class T >  
using [Simd128](#) = [Simd128\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

### 17.246.1 Macro Definition Documentation

### 17.246.1.1 `__FFLASFFPACK_fflas_ffpack_utils_simd128_INL`

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL
```

## 17.246.2 Typedef Documentation

### 17.246.2.1 `Simd128`

```
using Simd128 = Simd128_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

## 17.247 `simd128_double.inl` File Reference

### Data Structures

- struct `Simd128_impl< true, false, true, 8 >`

### Macros

- #define `__FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL`

### 17.247.1 Macro Definition Documentation

#### 17.247.1.1 `__FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL`

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL
```

## 17.248 `simd128_float.inl` File Reference

### Data Structures

- struct `Simd128_impl< true, false, true, 4 >`

### Macros

- #define `__FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL`

### 17.248.1 Macro Definition Documentation

#### 17.248.1.1 `__FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL`

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL
```

## 17.249 `simd128_int16.inl` File Reference

### Data Structures

- struct `Simd128_impl< true, true, true, 2 >`
- union `Simd128_impl< true, true, true, 2 >::Converter`
- struct `Simd128_impl< true, true, false, 2 >`
- union `Simd128_impl< true, true, false, 2 >::Converter`

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int16\\_INL](#)

### 17.249.1 Macro Definition Documentation

#### 17.249.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL
```

## 17.250 simd128\_int32.inl File Reference

### Data Structures

- struct [Simd128\\_impl< true, true, true, 4 >](#)
- union [Simd128\\_impl< true, true, true, 4 >::Converter](#)
- struct [Simd128\\_impl< true, true, false, 4 >](#)
- union [Simd128\\_impl< true, true, false, 4 >::Converter](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int32\\_INL](#)

### 17.250.1 Macro Definition Documentation

#### 17.250.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL
```

## 17.251 simd128\_int64.inl File Reference

### Data Structures

- struct [Simd128\\_impl< true, true, true, 8 >](#)
- union [Simd128\\_impl< true, true, true, 8 >::Converter](#)
- struct [Simd128\\_impl< true, true, false, 8 >](#)
- union [Simd128\\_impl< true, true, false, 8 >::Converter](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int64\\_INL](#)
- [#define vect\\_t Simd128\\_impl<true,true,true,8>::vect\\_t](#)

### 17.251.1 Macro Definition Documentation

#### 17.251.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
```

**17.251.1.2 vect\_t**

```
#define vect_t Simd128_impl<true,true,true,8>::vect_t
```

**17.252 simd256.inl File Reference**

```
#include "simd256_float.inl"
#include "simd256_double.inl"
```

**Data Structures**

- struct [Simd256fp\\_base](#)
- struct [Simd256i\\_base](#)

**Macros**

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_INL](#)

**Typedefs**

- template<class T >  
using [Simd256](#) = [Simd256\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

**17.252.1 Macro Definition Documentation****17.252.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL**

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL
```

**17.252.2 Typedef Documentation****17.252.2.1 Simd256**

```
using Simd256 = Simd256\_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↵  
::is_signed<T>::value, sizeof(T)>
```

**17.253 simd256\_double.inl File Reference****Data Structures**

- struct [Simd256\\_impl](#)< true, false, true, 8 >

**Macros**

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_double\\_INL](#)

**17.253.1 Macro Definition Documentation**



**17.253.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL**

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL
```

**17.254 simd256\_float.inl File Reference****Data Structures**

- struct [Simd256\\_impl](#)< true, false, true, 4 >

**Macros**

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_float\\_INL](#)

**17.254.1 Macro Definition Documentation****17.254.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL**

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL
```

**17.255 simd256\_int16.inl File Reference****Data Structures**

- struct [Simd256\\_impl](#)< true, true, true, 2 >
- union [Simd256\\_impl](#)< true, true, true, 2 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 2 >
- union [Simd256\\_impl](#)< true, true, false, 2 >::Converter

**Macros**

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int16\\_INL](#)

**17.255.1 Macro Definition Documentation****17.255.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL**

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL
```

**17.256 simd256\_int32.inl File Reference****Data Structures**

- struct [Simd256\\_impl](#)< true, true, true, 4 >
- union [Simd256\\_impl](#)< true, true, true, 4 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 4 >
- union [Simd256\\_impl](#)< true, true, false, 4 >::Converter

**Macros**

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int32\\_INL](#)

## 17.256.1 Macro Definition Documentation

### 17.256.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL
```

## 17.257 simd256\_int64.inl File Reference

### Data Structures

- struct [Simd256\\_impl< true, true, true, 8 >](#)
- union [Simd256\\_impl< true, true, true, 8 >::Converter](#)
- struct [Simd256\\_impl< true, true, false, 8 >](#)
- union [Simd256\\_impl< true, true, false, 8 >::Converter](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int64\\_INL](#)
- #define [vect\\_t Simd256\\_impl<true, true, true, 8>::vect\\_t](#)

## 17.257.1 Macro Definition Documentation

### 17.257.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
```

### 17.257.1.2 vect\_t

```
#define vect_t Simd256\_impl<true, true, true, 8>::vect\_t
```

## 17.258 simd512.inl File Reference

```
#include "simd512_float.inl"
#include "simd512_double.inl"
#include "simd512_int64.inl"
```

### Data Structures

- struct [Simd512fp\\_base](#)
- struct [Simd512i\\_base](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_INL](#)

### Typedefs

- template<class T >  
using [Simd512](#) = [Simd512\\_impl< std::is\\_arithmetic< T >::value, std::is\\_integral< T >::value, std::is\\_↵  
signed< T >::value, sizeof\(T\)>](#)

## 17.258.1 Macro Definition Documentation

### 17.258.1.1 \_\_FFLASFFPACK\_simd512\_INL

```
#define __FFLASFFPACK_simd512_INL
```

## 17.258.2 Typedef Documentation

### 17.258.2.1 Simd512

```
using Simd512 = Simd512_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

## 17.259 simd512\_double.inl File Reference

### Data Structures

- struct [Simd512\\_impl< true, false, true, 8 >](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_double\\_INL](#)

## 17.259.1 Macro Definition Documentation

### 17.259.1.1 \_\_FFLASFFPACK\_simd512\_double\_INL

```
#define __FFLASFFPACK_simd512_double_INL
```

## 17.260 simd512\_float.inl File Reference

### Data Structures

- struct [Simd512\\_impl< true, false, true, 4 >](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_float\\_INL](#)

## 17.260.1 Macro Definition Documentation

### 17.260.1.1 \_\_FFLASFFPACK\_simd512\_float\_INL

```
#define __FFLASFFPACK_simd512_float_INL
```

## 17.261 simd512\_int32.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd/simd512_int64.inl"
```

## Data Structures

- struct [Simd256\\_impl](#)< true, true, true, 4 >
- union [Simd256\\_impl](#)< true, true, true, 4 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 4 >
- union [Simd256\\_impl](#)< true, true, false, 4 >::Converter

## Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_int32\\_INL](#)

### 17.261.1 Macro Definition Documentation

#### 17.261.1.1 \_\_FFLASFFPACK\_simd512\_int32\_INL

```
#define __FFLASFFPACK_simd512_int32_INL
```

## 17.262 simd512\_int64.inl File Reference

### Data Structures

- struct [Simd512\\_impl](#)< true, true, true, 8 >
- union [Simd512\\_impl](#)< true, true, true, 8 >::Converter
- struct [Simd512\\_impl](#)< true, true, false, 8 >
- union [Simd512\\_impl](#)< true, true, false, 8 >::Converter

### Macros

- #define [\\_simd512\\_int64\\_INL](#)
- #define [vect\\_t Simd512\\_impl](#)<true, true, true, 8>::vect\_t

### 17.262.1 Macro Definition Documentation

#### 17.262.1.1 \_simd512\_int64\_INL

```
#define _simd512_int64_INL
```

#### 17.262.1.2 vect\_t

```
#define vect_t Simd512\_impl<true, true, true, 8>::vect_t
```

## 17.263 simd\_modular.inl File Reference

### Data Structures

- class [FieldSimd](#)< [\\_Field](#) >

## 17.264 solve.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

*This example solve the quare system defined by the input over a defined finite field.*

### 17.264.1 Function Documentation

#### 17.264.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example solve the quare system defined by the input over a defined finite field.

## 17.265 sparse\_matrix\_traits.h File Reference

```
#include <type_traits>
```

## Data Structures

- struct [isSparseMatrix](#)< Field, M >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::COO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >
- struct [isZOSparseMatrix](#)< F, M >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >
- struct [isSparseMatrixSimdFormat](#)< F, M >
- struct [isSparseMatrixMKLFormat](#)< F, M >
- struct [tfn\\_plus](#)
- struct [tfn\\_mul](#)
- struct [tfn\\_mul\\_eq](#)
- struct [tfn\\_minus](#)
- struct [tfn\\_plus\\_eq](#)
- struct [tfn\\_minus\\_eq](#)
- struct [has\\_plus\\_impl](#)< C >
- struct [has\\_mul\\_impl](#)< C >
- struct [has\\_mul\\_eq\\_impl](#)< C >
- struct [has\\_plus\\_eq\\_impl](#)< C >
- struct [has\\_minus\\_eq\\_impl](#)< C >
- struct [has\\_minus\\_impl](#)< C >
- struct [has\\_operation](#)< T >

## Namespaces

- [FFLAS](#)

## Typedefs

- using [ZOSparseMatrix](#) = std::true\_type
- using [NotZOSparseMatrix](#) = std::false\_type
- using [SimdSparseMatrix](#) = std::true\_type
- using [NoSimdSparseMatrix](#) = std::false\_type
- using [MKLSparseMatrixFormat](#) = std::true\_type
- using [NotMKLSparseMatrixFormat](#) = std::false\_type
- template<class T >  
using [has\\_plus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_plus\_impl< T > >::type
- template<class T >  
using [has\\_minus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_minus\_↵\_impl< T > >::type
- template<class T >  
using [has\\_equal](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, std::is\_copy\_↵\_assignable< T > >::type
- template<class T >  
using [has\\_plus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_plus\_↵\_eq\_impl< T > >::type
- template<class T >  
using [has\\_minus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_↵\_minus\_eq\_impl< T > >::type
- template<class T >  
using [has\\_mul](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_mul\_impl< T > >::type
- template<class T >  
using [has\\_mul\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_mul\_↵\_eq\_impl< T > >::type

## 17.266 test-charpoly-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

## Macros

- #define [ENABLE\\_CHECKER\\_charpoly](#) 1
- #define [TIME\\_CHECKER\\_CHARPOLY](#) 1

## Functions

- template<class Field , class Polynomial >  
void [printPolynomial](#) (const [Field](#) &F, Polynomial &v)
- int [main](#) (int argc, char \*\*argv)

## 17.266.1 Macro Definition Documentation

### 17.266.1.1 ENABLE\_CHECKER\_charpoly

```
#define ENABLE_CHECKER_charpoly 1
```

### 17.266.1.2 TIME\_CHECKER\_CHARPOLY

```
#define TIME_CHECKER_CHARPOLY 1
```

## 17.266.2 Function Documentation

### 17.266.2.1 printPolynomial()

```
void printPolynomial (
    const Field & F,
    Polynomial & v )
```

### 17.266.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.267 test-charpoly.C File Reference

```
#include <iostream>
#include <iomanip>
#include "givaro/modular.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
#include <chrono>
```

## Functions

- `template<class Field, class RandIter>`  
`bool launch_test (const Field &F, size_t n, typename Field::Element *A, size_t lda, size_t nbit, RandIter &G, FFPACK::FFPACK_CHARPOLY_TAG CT)`
- `template<class Field>`  
`bool run_with_field (const Givaro::Integer p, uint64_t bits, size_t n, std::string file, int variant, size_t iter, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.267.1 Function Documentation

### 17.267.1.1 launch\_test()

```
bool launch_test (
    const Field & F,
    size_t n,
    typename Field::Element * A,
    size_t lda,
    size_t nbit,
    RandIter & G,
    FFPACK::FFPACK_CHARPOLY_TAG CT )
```

### 17.267.1.2 run\_with\_field()

```
bool run_with_field (
    const Givaro::Integer p,
    uint64_t bits,
    size_t n,
    std::string file,
    int variant,
    size_t iter,
    uint64_t seed )
```

### 17.267.1.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.268 test-compressQ.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <list>
#include <vector>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Typedefs

- typedef Givaro::Modular< double > [Field](#)

## Functions

- template<class T >  
std::ostream & [printvect](#) (std::ostream &o, vector< T > &vect)
- int [main](#) (int argc, char \*\*argv)

## 17.268.1 Typedef Documentation

### 17.268.1.1 Field

```
typedef Givaro::Modular<double> Field
```



## 17.268.2 Function Documentation

### 17.268.2.1 printvect()

```
std::ostream& printvect (
    std::ostream & o,
    vector< T > & vect )
```

**Bug** does not belong here

### 17.268.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.269 test-det-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
```

### Macros

- `#define` [ENABLE\\_CHECKER\\_Det](#) 1
- `#define` [TIME\\_CHECKER\\_Det](#) 1

### Functions

- `int` [main](#) (int argc, char \*\*argv)

## 17.269.1 Macro Definition Documentation

### 17.269.1.1 ENABLE\_CHECKER\_Det

```
#define ENABLE_CHECKER_Det 1
```

### 17.269.1.2 TIME\_CHECKER\_Det

```
#define TIME_CHECKER_Det 1
```

## 17.269.2 Function Documentation

**17.269.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.270 test-det.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

**Functions**

- `template<class Field , class RandIter >`  
`bool test_det (Field &F, size_t n, int iter, RandIter &G)`
- `int main (int argc, char **argv)`

**17.270.1 Function Documentation****17.270.1.1 test\_det()**

```
bool test_det (
    Field & F,
    size_t n,
    int iter,
    RandIter & G )
```

**Todo** test with stride

**17.270.1.2 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.271 test-echelon.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <iomanip>
#include <givaro/modular-balanced.h>
#include <givaro/udl.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

```
#include <random>
#include <chrono>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25`
- `#define __FFLASFFPACK_PLUQ_THRESHOLD 25`

## Functions

- `template<class Field , class Randlter >`  
`bool test_colechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, Randlter &G, bool par)`
- `template<class Field , class Randlter >`  
`bool test_rowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, Randlter &G, bool par)`
- `template<class Field , class Randlter >`  
`bool test_redcolechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, Randlter &G, bool par)`
- `template<class Field , class Randlter >`  
`bool test_redrowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, Randlter &G, bool par)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.271.1 Macro Definition Documentation

### 17.271.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.271.1.2 \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25
```

### 17.271.1.3 \_\_FFLASFFPACK\_PLUQ\_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 25
```

## 17.271.2 Function Documentation

### 17.271.2.1 test\_colechelon()

```
bool test_colechelon (
    Field & F,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
```

```
FFPACK::FFPACK_LU_TAG LuTag,  
RandIter & G,  
bool par )
```

**Todo** check Ida

#### 17.271.2.2 test\_rowechelon()

```
bool test_rowechelon (  
    Field & F,  
    size_t m,  
    size_t n,  
    size_t r,  
    size_t iters,  
    FFPACK::FFPACK_LU_TAG LuTag,  
    RandIter & G,  
    bool par )
```

**Todo** check Ida

#### 17.271.2.3 test\_redcolechelon()

```
bool test_redcolechelon (  
    Field & F,  
    size_t m,  
    size_t n,  
    size_t r,  
    size_t iters,  
    FFPACK::FFPACK_LU_TAG LuTag,  
    RandIter & G,  
    bool par )
```

**Todo** check Ida

#### 17.271.2.4 test\_redrowechelon()

```
bool test_redrowechelon (  
    Field & F,  
    size_t m,  
    size_t n,  
    size_t r,  
    size_t iters,  
    FFPACK::FFPACK_LU_TAG LuTag,  
    RandIter & G,  
    bool par )
```

**Todo** check Ida

#### 17.271.2.5 run\_with\_field()

```
bool run_with_field (  
    Givaro::Integer q,  
    uint64_t b,
```

```

size_t m,
size_t n,
size_t r,
size_t iters,
uint64_t seed )

```

### 17.271.2.6 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.272 test-fadd.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"

```

## Functions

- template<class Field >  
bool [test\\_fadd](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class Field >  
bool [test\\_faddin](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class Field >  
bool [test\\_fsub](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class Field >  
bool [test\\_fsubin](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- int [main](#) (int ac, char \*\*av)

## 17.272.1 Function Documentation

### 17.272.1.1 test\_fadd()

```

bool test_fadd (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64\_t seed )

```

### 17.272.1.2 test\_faddin()

```

bool test_faddin (
    const Field & F,
    size_t m,
    size_t k,

```

```

    size_t n,
    bool timing,
    uint64_t seed )

```

### 17.272.1.3 test\_fsub()

```

bool test_fsub (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

### 17.272.1.4 test\_fsubin()

```

bool test_fsubin (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

### 17.272.1.5 main()

```

int main (
    int ac,
    char ** av )

```

## 17.273 test-fdot.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include <givaro/zring.h>
#include <givaro/modular.h>
#include <random>
#include <chrono>

```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- template<typename Field >  
 bool [check\\_fdot](#) (const Field &F, size\_t n, typename Field::ConstElement\_ptr a, size\_t inca, typename Field::ConstElement\_ptr b, size\_t incb)

- template<class Field >  
  bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t BS, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- bool [run\\_with\\_Integer](#) (size\_t BS, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.273.1 Macro Definition Documentation

### 17.273.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.273.2 Function Documentation

### 17.273.2.1 check\_fdot()

```
bool check_fdot (
    const Field & F,
    size_t n,
    typename Field::ConstElement\_ptr a,
    size_t inca,
    typename Field::ConstElement\_ptr b,
    size_t incb )
```

### 17.273.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t BS,
    size_t n,
    size_t iters,
    uint64\_t seed )
```

### 17.273.2.3 run\_with\_Integer()

```
bool run_with_Integer (
    size_t BS,
    size_t n,
    size_t iters,
    uint64\_t seed )
```

### 17.273.2.4 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.274 test-fgemm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
```

```
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

## Macros

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- `template<class Field , class RandIter >`  
`bool` [launch\\_MM\\_dispatch](#) (const [Field](#) &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t iters, RandIter &G)
- `template<class Field >`  
`bool` [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, int m, int n, int k, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.274.1 Macro Definition Documentation

### 17.274.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.274.2 Function Documentation

### 17.274.2.1 launch\_MM\_dispatch()

```
bool launch_MM_dispatch (
    const Field & F,
    const int mm,
    const int nn,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    RandIter & G )
```

**Bug** test for ldX equal

**Bug** test for transpo

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 17.274.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64\_t b,
    int m,
    int n,
    int k,
    size_t iters,
    uint64\_t seed )
```



### 17.274.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.275 test-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
```

### Macros

- #define [ENABLE\\_CHECKER\\_fgemm](#) 1

### Functions

- template<class Field >  
bool [check\\_MM](#) (const [Field](#) &F, const typename [Field::Element\\_ptr](#) Cd, enum [FFLAS\\_TRANSPOSE](#) &ta, enum [FFLAS\\_TRANSPOSE](#) &tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element\\_ptr](#) A, size\_t lda, const typename [Field::Element\\_ptr](#) B, size\_t ldb, const typename [Field::Element](#) &beta, const typename [Field::Element\\_ptr](#) C, size\_t ldc)
- template<class Field , class RandIter >  
bool [launch\\_MM](#) (const [Field](#) &F, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t ldc, const size\_t lda, enum [FFLAS\\_TRANSPOSE](#) ta, const size\_t ldb, enum [FFLAS\\_TRANSPOSE](#) tb, size\_t iters, int nbw, bool par, RandIter &G)
- template<class Field , class RandIter >  
bool [launch\\_MM\\_dispatch](#) (const [Field](#) &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t iters, const int nbw, const bool par, RandIter &G)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, int m, int n, int k, int nbw, size\_t iters, bool par, size\_t seed)
- int [main](#) (int argc, char \*\*argv)

## 17.275.1 Macro Definition Documentation

### 17.275.1.1 ENABLE\_CHECKER\_fgemm

```
#define ENABLE_CHECKER_fgemm 1
```

## 17.275.2 Function Documentation

**17.275.2.1 check\_MM()**

```

bool check_MM (
    const Field & F,
    const typename Field::Element_ptr Cd,
    enum FFLAS_TRANSPOSE & ta,
    enum FFLAS_TRANSPOSE & tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element_ptr A,
    size_t lda,
    const typename Field::Element_ptr B,
    size_t ldb,
    const typename Field::Element & beta,
    const typename Field::Element_ptr C,
    size_t ldc )

```

**17.275.2.2 launch\_MM()**

```

bool launch_MM (
    const Field & F,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t ldc,
    const size_t lda,
    enum FFLAS_TRANSPOSE ta,
    const size_t ldb,
    enum FFLAS_TRANSPOSE tb,
    size_t iters,
    int nbw,
    bool par,
    RandIter & G )

```

**17.275.2.3 launch\_MM\_dispatch()**

```

bool launch_MM_dispatch (
    const Field & F,
    const int mm,
    const int nn,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    const int nbw,
    const bool par,
    RandIter & G )

```

**Bug** test for ldX equal

**Bug** test for transpo

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

#### 17.275.2.4 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    int m,
    int n,
    int k,
    int nbw,
    size_t iters,
    bool par,
    size_t seed )
```

#### 17.275.2.5 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.276 test-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

## Functions

- `template<class Field >`  
`bool check_MV (const Field &F, const typename Field::Element_ptr Cd, enum FFLAS_TRANSPOSE &ta, const size_t m, const size_t k, const typename Field::Element &alpha, const typename Field::Element_ptr A, size_t lda, const typename Field::Element_ptr X, size_t incX, const typename Field::Element &beta, const typename Field::Element_ptr Y, size_t incY)`
- `template<class Field , class RandIter >`  
`bool launch_MV (const Field &F, const size_t m, const size_t k, const typename Field::Element alpha, const typename Field::Element beta, const size_t lda, enum FFLAS_TRANSPOSE ta, const size_t incX, const size_t incY, size_t iters, bool par, RandIter &G)`
- `template<class Field , class RandIter >`  
`bool launch_MV_dispatch (const Field &F, const int mm, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, const bool par, RandIter &G)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, int m, int k, size_t iters, bool par, uint64_t seed)`
- `int main (int argc, char **argv)`

### 17.276.1 Function Documentation

**17.276.1.1 check\_MV()**

```

bool check_MV (
    const Field & F,
    const typename Field::Element_ptr Cd,
    enum FFLAS_TRANSPOSE & ta,
    const size_t m,
    const size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element_ptr A,
    size_t lda,
    const typename Field::Element_ptr X,
    size_t incX,
    const typename Field::Element & beta,
    const typename Field::Element_ptr Y,
    size_t incY )

```

**17.276.1.2 launch\_MV()**

```

bool launch_MV (
    const Field & F,
    const size_t m,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t lda,
    enum FFLAS_TRANSPOSE ta,
    const size_t incX,
    const size_t incY,
    size_t iters,
    bool par,
    RandIter & G )

```

**17.276.1.3 launch\_MV\_dispatch()**

```

bool launch_MV_dispatch (
    const Field & F,
    const int mm,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    const bool par,
    RandIter & G )

```

**17.276.1.4 run\_with\_field()**

```

bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    int m,
    int k,
    size_t iters,
    bool par,
    uint64_t seed )

```

### 17.276.1.5 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.277 test-fger.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular-integral.h>
#include <givaro/modular-balanced.h>
#include <givaro/givintprime.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

### Macros

- `#define TIME 1`

### Functions

- `template<class Field >`  
`bool check_fger (const Field &F, const typename Field::Element_ptr Cd, const size_t m, const size_t n, const`  
`typename Field::Element &alpha, const typename Field::Element_ptr x, const size_t incx, const typename`  
`Field::Element_ptr y, const size_t incy, const typename Field::Element_ptr C, const size_t ldc)`
- `template<class Field , class RandIter >`  
`bool launch_fger (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, const`  
`size_t ldc, const size_t inca, const size_t incb, size_t iters, RandIter &G)`
- `template<class Field , class RandIter >`  
`bool launch_fger_dispatch (const Field &F, const size_t nn, const typename Field::Element alpha, const`  
`size_t iters, RandIter &G)`
- `template<class Field >`  
`bool run_with_field (int64_t q, uint64_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.277.1 Macro Definition Documentation

### 17.277.1.1 TIME

```
#define TIME 1
```

## 17.277.2 Function Documentation

### 17.277.2.1 check\_fger()

```
bool check_fger (
    const Field & F,
    const typename Field::Element_ptr Cd,
```

```

    const size_t m,
    const size_t n,
    const typename Field::Element & alpha,
    const typename Field::Element_ptr x,
    const size_t incx,
    const typename Field::Element_ptr y,
    const size_t incy,
    const typename Field::Element_ptr C,
    const size_t ldc )

```

### 17.277.2.2 launch\_fger()

```

bool launch_fger (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const size_t ldc,
    const size_t inca,
    const size_t incb,
    size_t iters,
    RandIter & G )

```

### 17.277.2.3 launch\_fger\_dispatch()

```

bool launch_fger_dispatch (
    const Field & F,
    const size_t nn,
    const typename Field::Element alpha,
    const size_t iters,
    RandIter & G )

```

**Bug** test for incx equal

**Bug** test for transpo

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 17.277.2.4 run\_with\_field()

```

bool run_with_field (
    int64_t q,
    uint64_t b,
    size_t n,
    size_t iters,
    uint64_t seed )

```

### 17.277.2.5 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.278 test-fgesv.C File Reference

```
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

### Functions

- template<class Field, class RandIter >  
bool [test\\_square\\_fgesv](#) (Field &F, [FFLAS\\_SIDE](#) side, string fileA, string fileB, size\_t m, size\_t k, size\_t r, RandIter &G)
- template<class Field, class RandIter >  
bool [test\\_rect\\_fgesv](#) (Field &F, [FFLAS\\_SIDE](#) side, string fileA, string fileB, size\_t m, size\_t n, size\_t k, size\_t r, RandIter &G)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, size\_t m, size\_t n, size\_t k, size\_t r, size\_t iters, string fileA, string fileB, [uint64\\_t](#) &seed)
- int [main](#) (int argc, char \*\*argv)

### 17.278.1 Function Documentation

#### 17.278.1.1 test\_square\_fgesv()

```
bool test_square_fgesv (
    Field & F,
    FFLAS_SIDE side,
    string fileA,
    string fileB,
    size_t m,
    size_t k,
    size_t r,
    RandIter & G )
```

#### 17.278.1.2 test\_rect\_fgesv()

```
bool test_rect_fgesv (
    Field & F,
    FFLAS_SIDE side,
    string fileA,
    string fileB,
    size_t m,
    size_t n,
    size_t k,
    size_t r,
    RandIter & G )
```

#### 17.278.1.3 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
```

```

uint64_t b,
size_t m,
size_t n,
size_t k,
size_t r,
size_t iters,
string fileA,
string fileB,
uint64_t & seed )

```

#### 17.278.1.4 main()

```

int main (
    int argc,
    char ** argv )

```

### 17.279 test-finit.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
#include <random>
#include <chrono>

```

### Functions

- template<class Field >  
bool [test\\_freduce](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t m, size\_t k, size\_t n, size\_t iters, bool timing, [uint64\\_t](#) seed)
- int [main](#) (int ac, char \*\*av)

#### 17.279.1 Function Documentation

##### 17.279.1.1 test\_freduce()

```

bool test_freduce (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64\_t seed )

```



### 17.279.1.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t k,
    size_t n,
    size_t iters,
    bool timing,
    uint64_t seed )
```

### 17.279.1.3 main()

```
int main (
    int ac,
    char ** av )
```

## 17.280 test-fscal.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
```

## Functions

- template<class Field , class Randlter >  
bool [test\\_fscal](#) (const [Field](#) &F, const typename [Field::Element](#) &alpha, size\_t m, size\_t k, size\_t n, bool timing, Randlter &G)
- template<class Field >  
bool [test\\_fscal](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class Field , class Randlter >  
bool [test\\_fscaln](#) (const [Field](#) &F, const typename [Field::Element](#) &alpha, size\_t m, size\_t k, size\_t n, bool timing, Randlter &G)
- template<class Field >  
bool [test\\_fscaln](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- int [main](#) (int ac, char \*\*av)

## 17.280.1 Function Documentation

### 17.280.1.1 test\_fscal() [1/2]

```
bool test_fscal (
    const Field & F,
    const typename Field::Element & alpha,
    size_t m,
    size_t k,
    size_t n,
```

```

    bool timing,
    RandIter & G )

```

#### 17.280.1.2 test\_fscal() [2/2]

```

bool test_fscal (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

#### 17.280.1.3 test\_fscalin() [1/2]

```

bool test_fscalin (
    const Field & F,
    const typename Field::Element & alpha,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    RandIter & G )

```

#### 17.280.1.4 test\_fscalin() [2/2]

```

bool test_fscalin (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

#### 17.280.1.5 main()

```

int main (
    int ac,
    char ** av )

```

## 17.281 test-fsyr2k.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>

```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- template<typename Field , class RandIter >  
bool [check\\_fsyr2k](#) (const [Field](#) &F, size\_t n, size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element](#) &beta, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, RandIter &Rand)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t k, int a, int c, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.281.1 Macro Definition Documentation

### 17.281.1.1 [ENABLE\\_ALL\\_CHECKINGS](#)

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.281.2 Function Documentation

### 17.281.2.1 [check\\_fsyr2k\(\)](#)

```
bool check_fsyr2k (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS\_UPLO uplo,
    FFLAS::FFLAS\_TRANSPOSE trans,
    RandIter & Rand )
```

### 17.281.2.2 [run\\_with\\_field\(\)](#)

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t k,
    int a,
    int c,
    size_t iters,
    uint64\_t seed )
```

### 17.281.2.3 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

## 17.282 test-fsyrrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- template<typename Field , class RandIter >  
bool [check\\_fsyrrk](#) (const [Field](#) &F, size\_t n, size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element](#) &beta, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, RandIter &Rand)
- template<typename Field , class RandIter >  
bool [check\\_fsyrrk\\_diag](#) (const [Field](#) &F, size\_t n, size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element](#) &beta, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, RandIter &Rand)
- template<typename Field , class RandIter >  
bool [check\\_fsyrrk\\_bkdiag](#) (const [Field](#) &F, size\_t n, size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element](#) &beta, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, RandIter &Rand)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t k, int a, int c, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.282.1 Macro Definition Documentation

### 17.282.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.282.2 Function Documentation

### 17.282.2.1 check\_fsyrrk()

```
bool check_fsyrrk (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS\_UPLO uplo,
    FFLAS::FFLAS\_TRANSPOSE trans,
    RandIter & Rand )
```

**17.282.2.2 check\_fsyrrk\_diag()**

```
bool check_fsyrrk_diag (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    RandIter & Rand )
```

**17.282.2.3 check\_fsyrrk\_bkdiag()**

```
bool check_fsyrrk_bkdiag (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    RandIter & Rand )
```

**17.282.2.4 run\_with\_field()**

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t k,
    int a,
    int c,
    size_t iters,
    uint64_t seed )
```

**17.282.2.5 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.283 test-fsytrf.C File Reference**

```
#include <iostream>
#include <iterator>
#include <vector>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <iomanip>
#include <random>
#include <chrono>
#include <givaro/modular.h>
#include "fflas-ffpack/utils/test-utils.h"
```

## Functions

- `template<typename T >`  
`std::ostream & operator<< (std::ostream &os, const std::vector< T > &x)`
- `template<class Field , class RandIter >`  
`bool test_RPM_fsytrf (Field &F, FFLAS_UPLO uplo, string file, size_t n, size_t r, RandIter &G, size_t threshold)`
- `template<class Field , class RandIter >`  
`bool test_generic_fsytrf (Field &F, FFLAS_UPLO uplo, string file, size_t n, RandIter &G, size_t threshold)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t n, size_t r, size_t iters, string file, size_t threshold, uint64_t &seed)`
- `int main (int argc, char **argv)`

### 17.283.1 Function Documentation

#### 17.283.1.1 operator<<()

```
std::ostream& operator<< (
    std::ostream & os,
    const std::vector< T > & x )
```

#### 17.283.1.2 test\_RPM\_fsytrf()

```
bool test_RPM_fsytrf (
    Field & F,
    FFLAS_UPLO uplo,
    string file,
    size_t n,
    size_t r,
    RandIter & G,
    size_t threshold )
```

#### 17.283.1.3 test\_generic\_fsytrf()

```
bool test_generic_fsytrf (
    Field & F,
    FFLAS_UPLO uplo,
    string file,
    size_t n,
    RandIter & G,
    size_t threshold )
```

#### 17.283.1.4 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t n,
    size_t r,
    size_t iters,
    string file,
    size_t threshold,
    uint64_t & seed )
```

### 17.283.1.5 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.284 test-fftrmm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

### Macros

- `#define __FFLASFFPACK_SEQUENTIAL`

### Functions

- `template<typename Field , class RandIter >`  
`bool check_fftrmm (const Field &F, size_t m, size_t n, const typename Field::Element &alpha,`  
`FFLAS::FFLAS_SIDE side, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, FFLAS::FFLAS_DIAG`  
`diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t n, uint64_t a, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.284.1 Macro Definition Documentation

### 17.284.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

## 17.284.2 Function Documentation

### 17.284.2.1 check\_fftrmm()

```
bool check_fftrmm (
    const Field & F,
    size_t m,
    size_t n,
    const typename Field::Element & alpha,
    FFLAS::FFLAS_SIDE side,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
```

```

    FFLAS::FFLAS_DIAG diag,
    RandIter & Rand )

```

### 17.284.2.2 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t n,
    uint64_t a,
    size_t iters,
    uint64_t seed )

```

### 17.284.2.3 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.285 test-ffrmv.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <chrono>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>

```

## Macros

- #define `__FFLASFFPACK_SEQUENTIAL`
- #define `ENABLE_ALL_CHECKINGS` 1

## Functions

- template<typename Field , class RandIter >  
bool `check_ffrmv` (const Field &F, size\_t n, FFLAS\_UPLO uplo, FFLAS\_TRANSPOSE trans, FFLAS\_DIAG diag, RandIter &Rand)
- template<class Field >  
bool `run_with_field` (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, uint64\_t seed)
- int `main` (int argc, char \*\*argv)

## 17.285.1 Macro Definition Documentation

### 17.285.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```

#define __FFLASFFPACK_SEQUENTIAL

```



### 17.285.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.285.2 Function Documentation

### 17.285.2.1 check\_ftrmv()

```
bool check_ftrmv (
    const Field & F,
    size_t n,
    FFLAS_UPLO uplo,
    FFLAS_TRANSPOSE trans,
    FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.285.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.285.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.286 test-ftrsm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.286.1 Macro Definition Documentation

### 17.286.1.1 `ENABLE_ALL_CHECKINGS`

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.286.2 Function Documentation

### 17.286.2.1 `main()`

```
int main (
    int argc,
    char ** argv )
```

## 17.287 `test-ftsrm.C` File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- `template<typename Field , class RandIter >`  
`bool` [check\\_ftsrm](#) (const [Field](#) &F, `size_t` m, `size_t` n, const `typename` [Field::Element](#) &alpha, [FFLAS::FFLAS\\_SIDE](#) side, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, [FFLAS::FFLAS\\_DIAG](#) diag, [RandIter](#) &Rand)
- `template<class Field >`  
`bool` [run\\_with\\_field](#) ([Givaro::Integer](#) q, `size_t` b, `size_t` m, `size_t` n, [uint64\\_t](#) a, `size_t` iters, [uint64\\_t](#) seed)
- `int` [main](#) (`int` argc, `char **`argv)

### 17.287.1 Macro Definition Documentation

#### 17.287.1.1 `__FFLASFFPACK_SEQUENTIAL`

```
#define __FFLASFFPACK_SEQUENTIAL
```

#### 17.287.1.2 `ENABLE_ALL_CHECKINGS`

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.287.2 Function Documentation

### 17.287.2.1 check\_ffrsm()

```
bool check_ffrsm (
    const Field & F,
    size_t m,
    size_t n,
    const typename Field::Element & alpha,
    FFLAS::FFLAS_SIDE side,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    FFLAS::FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.287.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t n,
    uint64_t a,
    size_t iters,
    uint64_t seed )
```

### 17.287.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.288 test-ffrssyr2k.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

## Macros

- #define `ENABLE_ALL_CHECKINGS` 1

## Functions

- `template<typename Field , class RandIter >`  
`bool check_ftrssyr2k` (const `Field` &`F`, `size_t` `n`, `FFLAS::FFLAS_UPLO` `uplo`, `FFLAS::FFLAS_DIAG` `diagA`, `RandIter` &`Rand`)
- `template<class Field >`  
`bool run_with_field` (`Givaro::Integer` `q`, `size_t` `b`, `size_t` `n`, `size_t` `iters`, `uint64_t` `seed`)
- `int main` (`int` `argc`, `char **argv`)

## 17.288.1 Macro Definition Documentation

### 17.288.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.288.2 Function Documentation

### 17.288.2.1 check\_ftrssyr2k()

```
bool check_ftrssyr2k (
    const Field & F,
    size_t n,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_DIAG diagA,
    RandIter & Rand )
```

### 17.288.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.288.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.289 test-ftrstr.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

```
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- template<typename Field , class RandIter >  
bool [check\\_ftrstr](#) (const [Field](#) &F, size\_t n, [FFLAS::FFLAS\\_SIDE](#) side, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_DIAG](#) diagA, [FFLAS::FFLAS\\_DIAG](#) diagB, RandIter &Rand)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.289.1 Macro Definition Documentation

### 17.289.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.289.2 Function Documentation

### 17.289.2.1 check\_ftrstr()

```
bool check_ftrstr (
    const Field & F,
    size_t n,
    FFLAS::FFLAS\_SIDE side,
    FFLAS::FFLAS\_UPLO uplo,
    FFLAS::FFLAS\_DIAG diagA,
    FFLAS::FFLAS\_DIAG diagB,
    RandIter & Rand )
```

### 17.289.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64\_t seed )
```

### 17.289.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.290 test-fftrsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

### Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define ENABLE_ALL_CHECKINGS 1`

### Functions

- `template<typename Field , class RandIter >`  
`bool check_fftrsv (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_TRANSPOSE trans, FFLAS_DIAG diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.290.1 Macro Definition Documentation

### 17.290.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.290.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.290.2 Function Documentation

### 17.290.2.1 check\_fftrsv()

```
bool check_fftrsv (
    const Field & F,
    size_t n,
    FFLAS_UPLO uplo,
    FFLAS_TRANSPOSE trans,
    FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.290.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.290.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.291 test-fftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- template<typename Field , class RandIter >  
bool [check\\_fftrtri](#) (const [Field](#) &F, size\_t n, [FFLAS\\_UPLO](#) uplo, [FFLAS\\_DIAG](#) diag, RandIter &Rand)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.291.1 Macro Definition Documentation

### 17.291.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.291.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.291.2 Function Documentation

### 17.291.2.1 check\_ftrtri()

```
bool check_ftrtri (
    const Field & F,
    size_t n,
    FFLAS_UPLO uplo,
    FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.291.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.291.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.292 test-interfaces-c.c File Reference

```
#include <fflas-ffpack/interfaces/libs/fflas_c.h>
#include <fflas-ffpack/interfaces/libs/ffpack_c.h>
#include <stdlib.h>
#include <stdio.h>
```

## Functions

- int [main](#) ()

## 17.292.1 Function Documentation

### 17.292.1.1 main()

```
int main (
    void )
```

## 17.293 test-invert-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```



```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.293.1 Macro Definition Documentation

#### 17.293.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

### 17.293.2 Function Documentation

#### 17.293.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.294 test-io.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <random>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Data Structures

- struct [CompactElement](#)< [Element](#) >
- struct [CompactElement](#)< [double](#) >
- struct [CompactElement](#)< [float](#) >
- struct [CompactElement](#)< [int64\\_t](#) >
- struct [CompactElement](#)< [int32\\_t](#) >
- struct [CompactElement](#)< [int16\\_t](#) >

## Functions

- template<class [Field](#) >  
bool [run\\_with\\_field](#) ([Givaro::Integer](#) q, [uint64\\_t](#) b, [size\\_t](#) m, [size\\_t](#) n, [size\\_t](#) iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.294.1 Function Documentation

### 17.294.1.1 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.294.1.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.295 test-lu.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-balanced.h>
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <random>
```

## Macros

- #define `BASECASE_K` 37
- #define `__FFLASFFPACK_SEQUENTIAL`
- #define `__LUDIVINE_CUTOFF` 1

## Functions

- template<class Field , FFLAS::FFLAS\_DIAG diag, FFLAS\_TRANSPOSE trans>  
bool `test_LUdivine` (const Field &F, typename Field::ConstElement\_ptr A, size\_t lda, size\_t r, size\_t m, size\_t n)  
*Tests the LUdivine routine.*
- template<class Field , FFLAS\_DIAG diag>  
bool `verifPLUQ` (const Field &F, typename Field::ConstElement\_ptr A, size\_t lda, typename Field::Element\_ptr PLUQ, size\_t ldpluq, size\_t \*P, size\_t \*Q, size\_t m, size\_t n, size\_t R)  
*Verifies that  $B = PLUQ$  where A stores  $[L|U]$ .*
- template<class Field , FFLAS\_DIAG diag, class RandIter >  
bool `test_pluq` (const Field &F, typename Field::ConstElement\_ptr A, size\_t r, size\_t m, size\_t n, size\_t lda, RandIter &G)  
*Tests the LUdivine routine.*
- template<class Field , FFLAS\_DIAG diag, FFLAS\_TRANSPOSE trans, class RandIter >  
bool `launch_test` (const Field &F, size\_t r, size\_t m, size\_t n, RandIter &G)

- template<class Field >  
bool `run_with_field` (Givaro::Integer q, `uint64_t` b, `size_t` m, `size_t` n, `size_t` r, `size_t` iters, `uint64_t` seed)
- int `main` (int argc, char \*\*argv)

## Variables

- Givaro::Timer `tperm`
- Givaro::Timer `tgemm`
- Givaro::Timer `tBC`
- Givaro::Timer `ttrsm`
- Givaro::Timer `trest`
- Givaro::Timer `timtot`
- `size_t` `mvcnt` = 0

## 17.295.1 Macro Definition Documentation

### 17.295.1.1 BASECASE\_K

```
#define BASECASE_K 37
```

### 17.295.1.2 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.295.1.3 \_\_LUDIVINE\_CUTOFF

```
#define __LUDIVINE_CUTOFF 1
```

## 17.295.2 Function Documentation

### 17.295.2.1 test\_LUdivine()

```
bool test_LUdivine (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t lda,
    size_t r,
    size_t m,
    size_t n )
```

Tests the LUdivine routine.

#### Template Parameters

<i>Field</i>	Field
<i>Diag</i>	Unit diagonal in U
<i>Trans</i>	

#### Parameters

<i>F</i>	field
<i>A</i>	Matrix (preallocated)

**Parameters**

<i>r</i>	rank of A
<i>m</i>	rows
<i>n</i>	cols
<i>lda</i>	leading dim of A

**Returns**

0 iff correct, 1 otherwise

**17.295.2.2   verifPLUQ()**

```
bool verificPLUQ (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t lda,
    typename Field::Element_ptr PLUQ,
    size_t ldpluq,
    size_t * P,
    size_t * Q,
    size_t m,
    size_t n,
    size_t R )
```

Verifies that  $B = PLUQ$  where A stores  $[L\backslash U]$ .

**Template Parameters**

<i>Field</i>	Field
<i>Diag</i>	Unit diagonal in U

**Parameters**

<i>F</i>	field
<i>A</i>	Matrix (preallocated)
<i>r</i>	rank of A
<i>m</i>	rows
<i>n</i>	cols
<i>lda</i>	leading dim of A

**Returns**

0 iff correct, 1 otherwise

**17.295.2.3   test\_pluq()**

```
bool test_pluq (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t r,
    size_t m,
    size_t n,
```

```

    size_t lda,
    RandIter & G )

```

Tests the LUdivine routine.

#### Template Parameters

<i>Field</i>	Field
<i>Diag</i>	Unit diagonal in U
<i>Trans</i>	

#### Parameters

<i>F</i>	field
<i>A</i>	Matrix (preallocated)
<i>r</i>	rank of A
<i>m</i>	rows
<i>n</i>	cols
<i>lda</i>	leading dim of A

#### Returns

0 iff correct, 1 otherwise

#### 17.295.2.4 launch\_test()

```

bool launch_test (
    const Field & F,
    size_t r,
    size_t m,
    size_t n,
    RandIter & G )

```

#### 17.295.2.5 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    uint64_t seed )

```

#### 17.295.2.6 main()

```

int main (
    int argc,
    char ** argv )

```

### 17.295.3 Variable Documentation

**17.295.3.1 tperm**

```
Givaro::Timer tperm
```

**17.295.3.2 tgemm**

```
Givaro::Timer tgemm
```

**17.295.3.3 tBC**

```
Givaro::Timer tBC
```

**17.295.3.4 ttrsm**

```
Givaro::Timer ttrsm
```

**17.295.3.5 trest**

```
Givaro::Timer trest
```

**17.295.3.6 timtot**

```
Givaro::Timer timtot
```

**17.295.3.7 mvcnt**

```
size_t mvcnt = 0
```

**17.296 test-maxdelayeddim.C File Reference**

```
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include <stdlib.h>
#include <stdio.h>
```

**Macros**

- #define [MAX\\_WITH\\_SIZE\\_T](#)(x) ( (static\_cast<[uint64\\_t](#)>(std::numeric\_limits<size\_t>::max()) < x)? std::numeric\_limits<size\_t>::max() : x )

**Functions**

- template<class Field >  
bool [test](#) (Givaro::Integer p, size\_t kmax)
- int [main](#) ()

**17.296.1 Macro Definition Documentation**

### 17.296.1.1 MAX\_WITH\_SIZE\_T

```
#define MAX_WITH_SIZE_T(
    x ) ( (static_cast<uint64_t>(std::numeric_limits<size_t>::max()) < x)? std::
::numeric_limits<size_t>::max() : x )
```

## 17.296.2 Function Documentation

### 17.296.2.1 test()

```
bool test (
    Givaro::Integer p,
    size_t kmax )
```

### 17.296.2.2 main()

```
int main (
    void )
```

## 17.297 test-minpoly.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include <chrono>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/modular-integer.h>
#include <givaro/givpoly1factor.h>
#include <givaro/givpoly1.h>
```

## Functions

- template<typename Field , class RandIter >  
bool [check\\_minpoly](#) (const [Field](#) &F, size\_t n, RandIter &G)
- template<class Field >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.297.1 Function Documentation

### 17.297.1.1 check\_minpoly()

```
bool check_minpoly (
    const Field & F,
```

```
    size_t n,
    RandIter & G )
```

### 17.297.1.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.297.1.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.298 test-multifile1.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

## 17.299 test-multifile2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

## Functions

- int [main](#) (void)

### 17.299.1 Function Documentation

#### 17.299.1.1 main()

```
int main (
    void )
```

## 17.300 test-nullspace.C File Reference

```
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/timer.h"
```



## Functions

- `template<class Field >`  
`std::string checkingMessage (const Field &F)`
- `template<class Field >`  
`Field::Element_ptr readOrRandomMatrixWithRankAndRandomRPM (const Field &F, std::string file, size_t m, size_t n, size_t lda, size_t r, uint64_t seed)`  
*If file is not empty, read it and set m, n, lda and r.*
- `template<class Field >`  
`bool test_nullspace (Field &F, FFLAS::FFLAS_SIDE side, size_t m, size_t n, size_t r, typename Field::Element_ptr A, size_t lda)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, std::string file, uint64_t &seed)`
- `int main (int argc, char **argv)`

## 17.300.1 Function Documentation

### 17.300.1.1 checkingMessage()

```
std::string checkingMessage (
    const Field & F )
```

### 17.300.1.2 readOrRandomMatrixWithRankAndRandomRPM()

```
Field::Element_ptr readOrRandomMatrixWithRankAndRandomRPM (
    const Field & F,
    std::string file,
    size_t m,
    size_t n,
    size_t lda,
    size_t r,
    uint64_t seed )
```

If file is not empty, read it and set m, n, lda and r.

Otherwise, generate a random matrix of size m x n with random lda.

### 17.300.1.3 test\_nullspace()

```
bool test_nullspace (
    Field & F,
    FFLAS::FFLAS_SIDE side,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda )
```

### 17.300.1.4 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
```

```

    size_t iters,
    std::string file,
    uint64_t & seed )

```

### 17.300.1.5 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.301 test-permutations.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/ffpack/ffpack.h"

```

### Functions

- bool [checkMonotonicApplyP](#) ([FFLAS\\_SIDE](#) Side, [FFLAS\\_TRANSPOSE](#) trans, size\_t \*P, size\_t N, size\_t R)
- int [main](#) ()

### Variables

- Givaro::Timer [tperm](#)
- Givaro::Timer [tgemm](#)
- Givaro::Timer [tBC](#)
- Givaro::Timer [ttrsm](#)
- Givaro::Timer [trest](#)
- Givaro::Timer [timtot](#)

## 17.301.1 Function Documentation

### 17.301.1.1 checkMonotonicApplyP()

```

bool checkMonotonicApplyP (
    FFLAS\_SIDE Side,
    FFLAS\_TRANSPOSE trans,
    size_t * P,
    size_t N,
    size_t R )

```

### 17.301.1.2 main()

```

int main (
    void )

```

## 17.301.2 Variable Documentation

### 17.301.2.1 tperm

Givaro::Timer tperm

### 17.301.2.2 tgemm

Givaro::Timer tgemm

### 17.301.2.3 tBC

Givaro::Timer tBC

### 17.301.2.4 ttrsm

Givaro::Timer ttrsm

### 17.301.2.5 trest

Givaro::Timer trest

### 17.301.2.6 timtot

Givaro::Timer timtot

## 17.302 test-pluq-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- int [main](#) (int argc, char \*\*argv)

## 17.302.1 Macro Definition Documentation

### 17.302.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.302.2 Function Documentation

**17.302.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.303 test-rankprofiles.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <iostream>
#include <iomanip>
#include <random>
#include <chrono>
```

**Macros**

- `#define __FFLASFFPACK_SEQUENTIAL`

**Functions**

- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed, bool par)`
- `int main (int argc, char **argv)`

**17.303.1 Macro Definition Documentation****17.303.1.1 \_\_FFLASFFPACK\_SEQUENTIAL**

```
#define __FFLASFFPACK_SEQUENTIAL
```

**17.303.2 Function Documentation****17.303.2.1 run\_with\_field()**

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    uint64_t seed,
    bool par )
```

### 17.303.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.304 test-rpm.C File Reference

```
#include <iostream>
#include "givaro/modular.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Functions

- bool [checkRPM](#) (size\_t M, size\_t N, size\_t R)
- bool [checkSymmetricRPM](#) (size\_t N, size\_t R)
- int [main](#) (int argc, char \*\*argv)

### 17.304.1 Function Documentation

#### 17.304.1.1 checkRPM()

```
bool checkRPM (
    size_t M,
    size_t N,
    size_t R )
```

#### 17.304.1.2 checkSymmetricRPM()

```
bool checkSymmetricRPM (
    size_t N,
    size_t R )
```

#### 17.304.1.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.305 test-simd.C File Reference

```
#include "givaro/givinteger.h"
#include "givaro/givprint.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <array>
```

```
#include <vector>
#include <random>
#include <string>
#include <functional>
#include <limits>
#include <type_traits>
#include <algorithm>
```

## Data Structures

- struct [ScalFunctions](#)< [Element](#), typename [enable\\_if](#)< [is\\_floating\\_point](#)< [Element](#) >::value >::type >
- struct [ScalFunctions](#)< [Element](#), typename [enable\\_if](#)< [is\\_integral](#)< [Element](#) >::value >::type >

## Macros

- #define [REGISTER\\_TYPE\\_NAME](#)(type) template<> const char \*[TypeName](#)<type>(){return #type;}
- #define [TEST\\_ONE\\_OP](#)(name) btest &= [test\\_op](#)<simd> (simd::name, Scal::name, #name);

## Typedefs

- typedef [Givaro::Integer](#) [integer](#)

## Functions

- template<typename... >  
const char \* [TypeName](#) ()
- [REGISTER\\_TYPE\\_NAME](#) (float)
- [REGISTER\\_TYPE\\_NAME](#) (double)
- [REGISTER\\_TYPE\\_NAME](#) (int16\_t)
- [REGISTER\\_TYPE\\_NAME](#) (int32\_t)
- [REGISTER\\_TYPE\\_NAME](#) (int64\_t)
- [REGISTER\\_TYPE\\_NAME](#) (uint16\_t)
- [REGISTER\\_TYPE\\_NAME](#) (uint32\_t)
- [REGISTER\\_TYPE\\_NAME](#) (uint64\_t)
- template<class [Element](#) , class [Alloc](#) >  
[enable\\_if](#)< [is\\_integral](#)< [Element](#) >::value >::type [generate\\_random\\_vector](#) (vector< [Element](#), [Alloc](#) > &a)
- template<class [Element](#) , class [Alloc](#) >  
[enable\\_if](#)< [is\\_floating\\_point](#)< [Element](#) >::value >::type [generate\\_random\\_vector](#) (vector< [Element](#), [Alloc](#) > &a)
- template<class [Element](#) >  
[enable\\_if](#)< [is\\_integral](#)< [Element](#) >::value, bool >::type [check\\_eq](#) ([Element](#) x, [Element](#) y)
- template<class [Element](#) >  
[enable\\_if](#)< [is\\_floating\\_point](#)< [Element](#) >::value, bool >::type [check\\_eq](#) ([Element](#) x, [Element](#) y)
- template<class [Ret](#) , class [T](#) >  
[Ret](#) [eval\\_func\\_on\\_array](#) (function< [Ret](#)()> f, array< [T](#), 0 > arr)
- template<class [Ret](#) , class [T](#) , class... [TArgs](#)>  
[Ret](#) [eval\\_func\\_on\\_array](#) (function< [Ret](#)([T](#), [TArgs](#)...)> f, array< typename [remove\\_reference](#)< [T](#) >::type, sizeof...(TArgs)+1 > &arr)
- template<class [Simd](#) , class [RScal](#) , class... [AScal](#), class [RSimd](#) , class... [ASimd](#)>  
[enable\\_if](#)< sizeof...(AScal)==sizeof...(ASimd), bool >::type [test\\_op](#) ([RSimd](#)(&FSimd)([ASimd](#)...), [RScal](#)(&FScal)([AScal](#)...), string frame)
- template<class [simd](#) , class [Element](#) >  
[enable\\_if](#)< [is\\_floating\\_point](#)< [Element](#) >::value, bool >::type [test\\_impl](#) ()
- template<class [simd](#) , class [Element](#) >  
[enable\\_if](#)< [is\\_integral](#)< [Element](#) >::value, bool >::type [test\\_impl](#) ()

- `template<class Element >`  
`enable_if< is_integral< Element >::value, bool >::type test ()`
- `template<class Element >`  
`enable_if< is_floating_point< Element >::value, bool >::type test ()`
- `int main (int argc, char *argv[])`

## 17.305.1 Macro Definition Documentation

### 17.305.1.1 REGISTER\_TYPE\_NAME

```
#define REGISTER_TYPE_NAME(  
    type )    template<> const char *TypeName<type>(){return #type;}
```

### 17.305.1.2 TEST\_ONE\_OP

```
#define TEST_ONE_OP(  
    name )    btest &= test_op<simd> (simd::name, Scal::name, #name);
```

## 17.305.2 Typedef Documentation

### 17.305.2.1 integer

```
typedef Givaro::Integer integer
```

## 17.305.3 Function Documentation

### 17.305.3.1 TypeName()

```
const char* TypeName ( )
```

### 17.305.3.2 REGISTER\_TYPE\_NAME() [1/8]

```
REGISTER_TYPE_NAME (  
    float )
```

### 17.305.3.3 REGISTER\_TYPE\_NAME() [2/8]

```
REGISTER_TYPE_NAME (  
    double )
```

### 17.305.3.4 REGISTER\_TYPE\_NAME() [3/8]

```
REGISTER_TYPE_NAME (  
    int16_t )
```

**17.305.3.5 REGISTER\_TYPE\_NAME() [4/8]**

```
REGISTER_TYPE_NAME (
    int32_t )
```

**17.305.3.6 REGISTER\_TYPE\_NAME() [5/8]**

```
REGISTER_TYPE_NAME (
    int64_t )
```

**17.305.3.7 REGISTER\_TYPE\_NAME() [6/8]**

```
REGISTER_TYPE_NAME (
    uint16_t )
```

**17.305.3.8 REGISTER\_TYPE\_NAME() [7/8]**

```
REGISTER_TYPE_NAME (
    uint32_t )
```

**17.305.3.9 REGISTER\_TYPE\_NAME() [8/8]**

```
REGISTER_TYPE_NAME (
    uint64_t )
```

**17.305.3.10 generate\_random\_vector() [1/2]**

```
enable_if<is_integral<Element>::value>::type generate_random_vector (
    vector< Element, Alloc > & a )
```

**17.305.3.11 generate\_random\_vector() [2/2]**

```
enable_if<is_floating_point<Element>::value>::type generate_random_vector (
    vector< Element, Alloc > & a )
```

**17.305.3.12 check\_eq() [1/2]**

```
enable_if<is_integral<Element>::value, bool>::type check_eq (
    Element x,
    Element y )
```

**17.305.3.13 check\_eq() [2/2]**

```
enable_if<is_floating_point<Element>::value, bool>::type check_eq (
    Element x,
    Element y )
```



**17.305.3.14 eval\_func\_on\_array() [1/2]**

```
Ret eval_func_on_array (
    function< Ret ()> f,
    array< T, 0 > arr )
```

**17.305.3.15 eval\_func\_on\_array() [2/2]**

```
Ret eval_func_on_array (
    function< Ret (T, TArgs...)> f,
    array< typename remove_reference< T >::type, sizeof...(TArgs)+1 > & arr )
```

**17.305.3.16 test\_op()**

```
enable_if<sizeof...(AScal) == sizeof...(ASimd), bool>::type test_op (
    RSimd(&) (ASimd...)   FSimd,
    RScal(&) (AScal...)   FScal,
    string fname )
```

**17.305.3.17 test\_impl() [1/2]**

```
enable_if<is_floating_point<Element>::value, bool>::type test_impl ( )
```

**17.305.3.18 test\_impl() [2/2]**

```
enable_if<is_integral<Element>::value, bool>::type test_impl ( )
```

**17.305.3.19 test() [1/2]**

```
enable_if<is_integral<Element>::value, bool>::type test ( )
```

**17.305.3.20 test() [2/2]**

```
enable_if<is_floating_point<Element>::value, bool>::type test ( )
```

**17.305.3.21 main()**

```
int main (
    int argc,
    char * argv[] )
```

**17.306 test-solve.C File Reference**

```
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

```
#include <givaro/modular-balanced.h>
```

## Functions

- `template<typename Field , class RandIter >`  
`bool check_solve (const Field &F, size_t m, RandIter &Rand, bool isParallel)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.306.1 Function Documentation

### 17.306.1.1 check\_solve()

```
bool check_solve (
    const Field & F,
    size_t m,
    RandIter & Rand,
    bool isParallel )
```

### 17.306.1.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t iters,
    uint64_t seed )
```

### 17.306.1.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.307 test-utils.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include <random>
#include <functional>
```

## Namespaces

- [FFLAS](#)
- [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Functions

- [uint64\\_t](#) [getSeed](#) ()
- `template<typename Field >`  
[Givaro::Integer](#) [maxFieldElt](#) ()
- `template<>` [Givaro::Integer](#) [maxFieldElt](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > > ()
- `template<typename Field >`  
[Field](#) \* [chooseField](#) ([Givaro::Integer](#) q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- `template<>` [Givaro::ZRing](#)< [int32\\_t](#) > \* [chooseField](#)< [Givaro::ZRing](#)< [int32\\_t](#) > > ([Givaro::Integer](#) q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- `template<>` [Givaro::ZRing](#)< [int64\\_t](#) > \* [chooseField](#)< [Givaro::ZRing](#)< [int64\\_t](#) > > ([Givaro::Integer](#) q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- `template<>` [Givaro::ZRing](#)< [float](#) > \* [chooseField](#)< [Givaro::ZRing](#)< [float](#) > > ([Givaro::Integer](#) q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- `template<>` [Givaro::ZRing](#)< [double](#) > \* [chooseField](#)< [Givaro::ZRing](#)< [double](#) > > ([Givaro::Integer](#) q, [uint64\\_t](#) b, [uint64\\_t](#) seed)

## 17.308 timer.h File Reference

```
#include <time.h>
#include <givaro/givtimer.h>
```

## Namespaces

- [FFLAS](#)

## Typedefs

- `typedef Givaro::Timer` [Timer](#)
- `typedef Givaro::BaseTimer` [BaseTimer](#)
- `typedef Givaro::UserTimer` [UserTimer](#)
- `typedef Givaro::SysTimer` [SysTimer](#)

## 17.309 utils.h File Reference

```
#include <algorithm>
#include <numeric>
#include <vector>
```

## Data Structures

- `struct` [StatsMatrix](#)

## Namespaces

- [FFLAS](#)

## Functions

- `template<class It >`  
[double](#) [computeDeviation](#) (It begin, It end)
- `template<class Field >`  
[StatsMatrix](#) [getStat](#) (const [Field](#) &F, const [index\\_t](#) \*row, const [index\\_t](#) \*col, typename [Field::ConstElement\\_ptr](#) val, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

## 17.310 winograd.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <ctime>
```

### Macros

- #define [DOUBLE\\_TO\\_FLOAT\\_CROSSOVER](#) 0
- #define [GFOPS](#)(n, t) (2.0/t\*(double)n/1000.0\*(double)n/1000.0\*(double)n/1000.0)

### Typedefs

- typedef Givaro::Timer [TTimer](#)

### Functions

- template<class Field >  
bool [balanced](#) (const [Field](#) &)
- template<class T >  
bool [balanced](#) (const [Givaro::ModularBalanced](#)< T > &)
- int [main](#) ()

## 17.310.1 Macro Definition Documentation

### 17.310.1.1 [DOUBLE\\_TO\\_FLOAT\\_CROSSOVER](#)

```
#define DOUBLE_TO_FLOAT_CROSSOVER 0
```

### 17.310.1.2 [GFOPS](#)

```
#define GFOPS(
    n,
    t ) (2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)
```

## 17.310.2 Typedef Documentation

### 17.310.2.1 [TTimer](#)

```
typedef Givaro::Timer TTimer
```

## 17.310.3 Function Documentation

**17.310.3.1 balanced()** [1/2]

```
bool balanced (
    const Field & )
```

**17.310.3.2 balanced()** [2/2]

```
bool balanced (
    const Givaro::ModularBalanced< T > & )
```

**17.310.3.3 main()**

```
int main (
    void )
```



# Index

- [\\_F](#)
  - [RNSIntegerMod< RNS >, 552](#)
- [\\_M](#)
  - [rns\\_double, 530](#)
  - [rns\\_double\\_extended, 542](#)
- [\\_MAX\\_SIZE\\_MATRICES](#)
  - [benchmark-checkers.C, 766](#)
- [\\_MMi](#)
  - [rns\\_double, 530](#)
  - [rns\\_double\\_extended, 542](#)
- [\\_Mi](#)
  - [rns\\_double, 530](#)
  - [rns\\_double\\_extended, 542](#)
- [\\_Mi\\_modp\\_rns](#)
  - [RNSIntegerMod< RNS >, 552](#)
- [\\_NR\\_TESTS](#)
  - [benchmark-checkers.C, 766](#)
- [\\_PLUQ](#)
  - [FFPACK, 363](#)
- [\\_RNSdelayed](#)
  - [RNSIntegerMod< RNS >, 553](#)
- [\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 838](#)
- [\\_\\_FFLASFFPACK\\_CACHE\\_LINE\\_SIZE](#)
  - [fflas\\_sparse.h, 920](#)
- [\\_\\_FFLASFFPACK\\_CHARPOLY\\_Danilevskii\\_LUKrylov\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 838](#)
- [\\_\\_FFLASFFPACK\\_CHARPOLY\\_LUKrylov\\_ArithProg\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 838](#)
- [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)
  - [cblas.C, 792](#)
  - [clapack.C, 799](#)
  - [fblas.C, 835](#)
  - [lapack.C, 1023](#)
- [\\_\\_FFLASFFPACK\\_DIMKPENALTY](#)
  - [fflas\\_pfgemm.inl, 911](#)
- [\\_\\_FFLASFFPACK\\_FORCE\\_SEQ](#)
  - [benchmark-charpoly-mp.C, 764](#)
- [\\_\\_FFLASFFPACK\\_FSYRK\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 838](#)
- [\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 838](#)
- [\\_\\_FFLASFFPACK\\_FTRSSYR2K\\_THRESHOLD](#)
  - [ffpack.h, 953](#)
- [\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#)
  - [ffpack.h, 953](#)
- [\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 838](#)
- [\\_\\_FFLASFFPACK\\_GAUSSJORDAN\\_BASECASE](#)
  - [ffpack\\_echelonforms.inl, 982](#)
  - [test-echelon.C, 1061](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_BLAS](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CBLAS](#)
  - [cblas.C, 792](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CLAPACK](#)
  - [clapack.C, 799](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CXX11](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DGETRF](#)
  - [benchmark-dgetrf.C, 768](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DLFCN\\_H](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DTRTRI](#)
  - [benchmark-dtrtri.C, 771](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_FLOAT\\_H](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_INT128](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_INTTYPES\\_H](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LAPACK](#)
  - [clapack.C, 799](#)
  - [fflas-ffpack/config.h, 812](#)
  - [lapack.C, 1023](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LIMITS\\_H](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LITTLE\\_ENDIAN](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_PTHREAD\\_H](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDDEF\\_H](#)
  - [fflas-ffpack/config.h, 812](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDINT\\_H](#)
  - [fflas-ffpack/config.h, 813](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDIO\\_H](#)
  - [fflas-ffpack/config.h, 813](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDLIB\\_H](#)
  - [fflas-ffpack/config.h, 813](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STRINGS\\_H](#)
  - [fflas-ffpack/config.h, 813](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STRING\\_H](#)
  - [fflas-ffpack/config.h, 813](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_STAT\\_H](#)
  - [fflas-ffpack/config.h, 813](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_TIME\\_H](#)
  - [fflas-ffpack/config.h, 813](#)

- \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H  
fflas-ffpack/config.h, [813](#)
- \_\_FFLASFFPACK\_HAVE\_UNISTD\_H  
fflas-ffpack/config.h, [813](#)
- \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL  
kaapi\_routines.inl, [1022](#)
- \_\_FFLASFFPACK\_LT\_OBJDIR  
fflas-ffpack/config.h, [813](#)
- \_\_FFLASFFPACK\_MINBLOCKCUTS  
blockcuts.inl, [792](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET  
benchmark-charpoly.C, [765](#)  
benchmark-fadd-lvl2.C, [772](#)  
benchmark-fdot.C, [772](#)  
benchmark-fgemm-mp.C, [773](#)  
benchmark-fgemm-rns.C, [774](#)  
benchmark-fgemv-mp.C, [777](#)  
benchmark-fgemv.C, [778](#)  
benchmark-fgesv.C, [781](#)  
benchmark-fsyrc.C, [781](#)  
benchmark-fsytrf.C, [782](#)  
benchmark-ftrsm-mp.C, [783](#)  
benchmark-ftrsm.C, [784](#)  
benchmark-ftrsv.C, [784](#)  
benchmark-ftrtri.C, [785](#)  
benchmark-pluq.C, [788](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS  
fflas-ffpack/config.h, [813](#)
- \_\_FFLASFFPACK\_PACKAGE  
fflas-ffpack/config.h, [813](#)
- \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_PACKAGE\_NAME  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_PACKAGE\_STRING  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_PACKAGE\_TARNAME  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_PACKAGE\_URL  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_PACKAGE\_VERSION  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_PLUQ\_THRESHOLD  
fflas-ffpack-default-thresholds.h, [838](#)  
test-echelon.C, [1061](#)
- \_\_FFLASFFPACK\_SEQPARTHRESHOLD  
fflas\_ptgemm.inl, [911](#)
- \_\_FFLASFFPACK\_SEQUENTIAL  
parallel.h, [1025](#)  
test-echelon.C, [1061](#)  
test-ftrmm.C, [1081](#)  
test-ftrmv.C, [1082](#)  
test-ftrsm.C, [1084](#)  
test-ftrsv.C, [1088](#)  
test-ftrtri.C, [1089](#)  
test-lu.C, [1093](#)  
test-rankprofiles.C, [1102](#)
- \_\_FFLASFFPACK\_SIZEOF\_CHAR  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_SIZEOF\_INT  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_SIZEOF\_SHORT  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_SIZEOF\_\_INT64  
fflas-ffpack/config.h, [814](#)
- \_\_FFLASFFPACK\_STDC\_HEADERS  
fflas-ffpack/config.h, [815](#)
- \_\_FFLASFFPACK\_USE\_OPENMP  
fflas-ffpack/config.h, [815](#)
- \_\_FFLASFFPACK\_VERSION  
fflas-ffpack/config.h, [815](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD  
fflas-ffpack-default-thresholds.h, [838](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL  
fflas-ffpack-default-thresholds.h, [838](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT  
fflas-ffpack-default-thresholds.h, [838](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT  
fflas-ffpack-default-thresholds.h, [838](#)
- \_\_FFLASFFPACK\_charpoly\_INL  
ffpack\_charpoly.inl, [977](#)
- \_\_FFLASFFPACK\_checker\_charpoly\_INL  
checker\_charpoly.inl, [794](#)
- \_\_FFLASFFPACK\_checker\_det\_INL  
checker\_det.inl, [795](#)
- \_\_FFLASFFPACK\_checker\_fgemm\_INL  
checker\_fgemm.inl, [796](#)
- \_\_FFLASFFPACK\_checker\_ftrsm\_INL  
checker\_ftrsm.inl, [796](#)
- \_\_FFLASFFPACK\_checker\_invert\_INL  
checker\_invert.inl, [796](#)
- \_\_FFLASFFPACK\_checker\_pluq\_INL  
checker\_pluq.inl, [797](#)
- \_\_FFLASFFPACK\_fadd\_INL  
fflas\_fadd.inl, [858](#)
- \_\_FFLASFFPACK\_fassign\_INL  
fflas\_fassign.inl, [859](#)
- \_\_FFLASFFPACK\_faxpy\_INL  
fflas\_faxpy.inl, [860](#)
- \_\_FFLASFFPACK\_fdot\_INL  
fflas\_fdot.inl, [860](#)
- \_\_FFLASFFPACK\_fflas\_blockcuts\_INL  
blockcuts.inl, [792](#)
- \_\_FFLASFFPACK\_fflas\_bounds\_INL  
fflas\_bounds.inl, [842](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL  
fgemm\_classical.inl, [1002](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL  
fgemm\_winograd.inl, [1005](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL  
fflas\_level1.inl, [892](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL



- fflas\_level2.inl, [895](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL
- fflas\_level3.inl, [897](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL
- fflas\_helpers.inl, [878](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL
- fflas\_sparse.inl, [923](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL
- simd128.inl, [1047](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL
- simd128\_double.inl, [1048](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL
- simd128\_float.inl, [1048](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL
- simd128\_int16.inl, [1049](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL
- simd128\_int32.inl, [1049](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL
- simd128\_int64.inl, [1049](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL
- simd256.inl, [1050](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL
- simd256\_double.inl, [1050](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL
- simd256\_float.inl, [1051](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL
- simd256\_int16.inl, [1051](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL
- simd256\_int32.inl, [1052](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL
- simd256\_int64.inl, [1052](#)
- \_\_FFLASFFPACK\_fflas\_freduce\_INL
- fflas\_freduce.inl, [869](#)
- \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL
- fflas\_freduce\_mp.inl, [870](#)
- \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL
- fflas\_fsyr2k.inl, [873](#)
- \_\_FFLASFFPACK\_fflas\_fsyrrk\_INL
- fflas\_fsyrrk.inl, [874](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL
- igemm.inl, [1016](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL
- igemm\_kernels.inl, [1018](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL
- igemm\_tools.inl, [1019](#)
- \_\_FFLASFFPACK\_fflas\_pfgemm\_INL
- fflas\_pfgemm.inl, [911](#)
- \_\_FFLASFFPACK\_fflas\_pftsm\_INL
- fflas\_pftsm.inl, [911](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL
- csr\_hyb\_pspmm.inl, [819](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL
- csr\_hyb\_pspmv.inl, [820](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spm INL
- csr\_hyb\_spm.inl, [820](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL
- csr\_hyb\_spmv.inl, [821](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL
- csr\_hyb\_utils.inl, [821](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL
- csr\_pspmm.inl, [822](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL
- csr\_pspmv.inl, [823](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spm INL
- csr\_spm.inl, [824](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL
- csr\_spmv.inl, [825](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL
- ell\_pspmm.inl, [829](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL
- ell\_pspmv.inl, [830](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL
- ell\_simd\_pspmv.inl, [831](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL
- ell\_simd\_spmv.inl, [832](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL
- ell\_simd\_utils.inl, [833](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spm INL
- ell\_spm.inl, [834](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL
- ell\_spmv.inl, [834](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL
- ell\_utils.inl, [835](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL
- hyb\_zo\_pspmm.inl, [1013](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL
- hyb\_zo\_pspmv.inl, [1013](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spm INL
- hyb\_zo\_spm.inl, [1014](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL
- hyb\_zo\_spmv.inl, [1015](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL
- hyb\_zo\_utils.inl, [1015](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spm INL
- coo\_spm.inl, [816](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL
- coo\_spmv.inl, [817](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL
- coo\_utils.inl, [818](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL
- sell\_pspmv.inl, [1045](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL
- sell\_spmv.inl, [1046](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL
- sell\_utils.inl, [1047](#)
- \_\_FFLASFFPACK\_ffpack\_INL
- ffpack.inl, [955](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL
- ffpack\_charpoly\_danilevski.inl, [977](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL
- ffpack\_charpoly\_kgfast.inl, [978](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL
- ffpack\_charpoly\_kgfastgeneralized.inl, [979](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL
- ffpack\_charpoly\_kglu.inl, [979](#)
- \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL

- ffpack\_echelonforms.inl, [982](#)
- \_\_FFLASFFPACK\_ffpack\_fgesv\_INL
  - ffpack\_fgesv.inl, [982](#)
- \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL
  - ffpack\_fgetrs.inl, [983](#)
- \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL
  - ffpack\_fsytrf.inl, [985](#)
- \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL
  - ffpack\_ftrssyr2k.inl, [985](#)
- \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL
  - ffpack\_ftrstr.inl, [986](#)
- \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL
  - ffpack\_ftrtr.inl, [987](#)
- \_\_FFLASFFPACK\_ffpack\_invert\_INL
  - ffpack\_invert.inl, [993](#)
- \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL
  - ffpack\_krylovelim.inl, [993](#)
- \_\_FFLASFFPACK\_ffpack\_ludivine\_INL
  - ffpack\_ludivine.inl, [994](#)
- \_\_FFLASFFPACK\_ffpack\_minpoly\_INL
  - ffpack\_minpoly.inl, [996](#)
- \_\_FFLASFFPACK\_ffpack\_permutation\_INL
  - ffpack\_permutation.inl, [998](#)
- \_\_FFLASFFPACK\_ffpack\_pluq\_INL
  - ffpack\_pluq.inl, [999](#)
- \_\_FFLASFFPACK\_ffpack\_ppluq\_INL
  - ffpack\_ppluq.inl, [1000](#)
- \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL
  - ffpack\_rankprofiles.inl, [1002](#)
- \_\_FFLASFFPACK\_ffgemm\_INL
  - fflas\_fgemm.inl, [863](#)
- \_\_FFLASFFPACK\_ffgemm\_bini\_INL
  - schedule\_bini.inl, [1041](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_INL
  - schedule\_winograd.inl, [1042](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_acc\_INL
  - schedule\_winograd\_acc.inl, [1043](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_acc\_ip\_INL
  - schedule\_winograd\_acc\_ip.inl, [1043](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_ip\_INL
  - schedule\_winograd\_ip.inl, [1044](#)
- \_\_FFLASFFPACK\_ffgemv\_INL
  - fflas\_ffgemv.inl, [864](#)
- \_\_FFLASFFPACK\_ffgemv\_mp\_INL
  - fflas\_ffgemv\_mp.inl, [865](#)
- \_\_FFLASFFPACK\_ffger\_INL
  - fflas\_ffger.inl, [866](#)
- \_\_FFLASFFPACK\_field\_rns\_INL
  - rns.inl, [1041](#)
- \_\_FFLASFFPACK\_field\_rns\_double\_INL
  - rns-double.inl, [1039](#)
- \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL
  - rns-double-recint.inl, [1037](#)
- \_\_FFLASFFPACK\_freivalds\_INL
  - fflas\_freivalds.inl, [870](#)
- \_\_FFLASFFPACK\_fscal\_INL
  - fflas\_fscal.inl, [872](#)
- \_\_FFLASFFPACK\_fscal\_mp\_INL
  - fflas\_fscal\_mp.inl, [873](#)
- \_\_FFLASFFPACK\_ftrmm\_INL
  - fflas\_ftrmm.inl, [875](#)
- \_\_FFLASFFPACK\_ftrsm\_INL
  - fflas\_ftrsm.inl, [875](#)
- \_\_FFLASFFPACK\_ftrsv\_INL
  - fflas\_ftrsv.inl, [877](#)
- \_\_FFLASFFPACK\_simd512\_INL
  - simd512.inl, [1053](#)
- \_\_FFLASFFPACK\_simd512\_double\_INL
  - simd512\_double.inl, [1053](#)
- \_\_FFLASFFPACK\_simd512\_float\_INL
  - simd512\_float.inl, [1053](#)
- \_\_FFLASFFPACK\_simd512\_int32\_INL
  - simd512\_int32.inl, [1054](#)
- \_\_FFLAS\_L1\_INST\_C
  - fflas\_L1\_inst.C, [879](#)
- \_\_FFLAS\_L2\_INST\_C
  - fflas\_L2\_inst.C, [883](#)
- \_\_FFLAS\_L3\_INST\_C
  - fflas\_L3\_inst.C, [887](#)
- \_\_FFLAS\_TRSM\_READONLY
  - fflas\_L3\_inst\_implement.inl, [890](#)
  - fflas\_level3.inl, [897](#)
  - ffpack\_ppluq.inl, [1000](#)
- \_\_FFPACK\_FSYTRF\_BC\_CROUT
  - benchmark-fsytrf.C, [782](#)
- \_\_FFPACK\_INST\_C
  - ffpack\_inst.C, [987](#)
- \_\_FFPACK\_charpoly\_mp\_INL
  - ffpack\_charpoly\_mp.inl, [980](#)
- \_\_FFPACK\_det\_mp\_INL
  - ffpack\_det\_mp.inl, [980](#)
- \_\_FFPACK\_ffgemm\_classical\_INL
  - ffgemm\_classical\_mp.inl, [1004](#)
- \_\_FFPACK\_fger\_mp\_INL
  - fflas\_fger\_mp.inl, [867](#)
- \_\_FFPACK\_ftrsm\_mp\_INL
  - fflas\_ftrsm\_mp.inl, [876](#)
- \_\_FFPACK\_ludivine\_mp\_INL
  - ffpack\_ludivine\_mp.inl, [995](#)
- \_\_FFPACK\_pluq\_mp\_INL
  - ffpack\_pluq\_mp.inl, [1000](#)
- \_\_LUDIVINE\_CUTOFF
  - test-lu.C, [1093](#)
- \_\_has\_builtin
  - bit\_manipulation.h, [790](#)
- \_alloc
  - rns\_double\_elt, [532](#)
  - rns\_double\_elt\_cstptr, [535](#)
  - rns\_double\_elt\_ptr, [538](#)
- \_basis
  - rns\_double, [529](#)
  - rns\_double\_extended, [541](#)
- \_basisMax
  - rns\_double, [529](#)
  - rns\_double\_extended, [541](#)
- \_coo

- SpMat< Field, flag >, 747
- \_coo16
  - CooMat< Field >, 431
- \_coo16\_zo
  - CooMat< Field >, 431
- \_coo32
  - CooMat< Field >, 431
- \_coo32\_zo
  - CooMat< Field >, 431
- \_coo64
  - CooMat< Field >, 431
- \_coo64\_zo
  - CooMat< Field >, 431
- \_crt\_in
  - rns\_double, 530
  - rns\_double\_extended, 542
- \_crt\_out
  - rns\_double, 530
  - rns\_double\_extended, 542
- \_csr
  - SpMat< Field, flag >, 747
- \_csr16
  - CsrMat< Field >, 432
- \_csr16\_zo
  - CsrMat< Field >, 432
- \_csr32
  - CsrMat< Field >, 432
- \_csr32\_zo
  - CsrMat< Field >, 432
- \_csr64
  - CsrMat< Field >, 432
- \_csr64\_zo
  - CsrMat< Field >, 432
- \_ell
  - SpMat< Field, flag >, 748
- \_ell16
  - EllMat< Field >, 439
- \_ell16\_zo
  - EllMat< Field >, 439
- \_ell32
  - EllMat< Field >, 439
- \_ell32\_zo
  - EllMat< Field >, 439
- \_ell64
  - EllMat< Field >, 439
- \_ell64\_zo
  - EllMat< Field >, 439
- \_errorStream
  - Failure, 441
- \_field\_rns
  - rns\_double, 529
  - rns\_double\_extended, 542
- \_iM\_modp\_rns
  - RNSIntegerMod< RNS >, 552
- \_ibeg
  - ForStrategy2D< blocksize\_t, Cut, Param >, 461
- \_iend
  - ForStrategy2D< blocksize\_t, Cut, Param >, 461
- \_invbasis
  - rns\_double, 529
  - rns\_double\_extended, 541
- \_jbeg
  - ForStrategy2D< blocksize\_t, Cut, Param >, 461
- \_jend
  - ForStrategy2D< blocksize\_t, Cut, Param >, 461
- \_ldm
  - rns\_double, 530
  - rns\_double\_extended, 542
- \_mi\_sum
  - rns\_double, 530
- \_negbasis
  - rns\_double, 529
  - rns\_double\_extended, 541
- \_p
  - RNSIntegerMod< RNS >, 552
- \_pbits
  - rns\_double, 530
  - rns\_double\_extended, 542
- \_ptr
  - rns\_double\_elt, 532
  - rns\_double\_elt\_cstptr, 535
  - rns\_double\_elt\_ptr, 538
- \_rns
  - RNSInteger< RNS >, 546
  - RNSIntegerMod< RNS >, 552
- \_simd512\_int64\_INL
  - simd512\_int64.inl, 1054
- \_size
  - rns\_double, 530
  - rns\_double\_extended, 542
- \_stride
  - rns\_double\_elt, 532
  - rns\_double\_elt\_cstptr, 535
  - rns\_double\_elt\_ptr, 538
- ~CheckerImplem\_Det
  - CheckerImplem\_Det< Field >, 411
- ~CheckerImplem\_PLUQ
  - CheckerImplem\_PLUQ< Field >, 415
- ~CheckerImplem\_charpoly
  - CheckerImplem\_charpoly< Field, Polynomial >, 410
- ~CheckerImplem\_fgemm
  - CheckerImplem\_fgemm< Field >, 412
- ~CheckerImplem\_ftsm
  - CheckerImplem\_ftsm< Field >, 413
- ~CheckerImplem\_invert
  - CheckerImplem\_invert< Field >, 414
- ~rns\_double\_elt
  - rns\_double\_elt, 531
- 101-fgemm.C, 759
  - main, 759
- 2x2-fgemm.C, 759
  - main, 759
- 2x2-ftsv.C, 760
  - main, 760
- 2x2-pluq.C, 760
  - main, 760

- main, 760
- add
  - FFLAS::vectorised, 287
  - FieldSimd< \_Field >, 444
  - RNSIntegerMod< RNS >, 550
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 556
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 560
  - Simd128\_impl< true, true, false, 2 >, 571
  - Simd128\_impl< true, true, false, 4 >, 579
  - Simd128\_impl< true, true, false, 8 >, 588
  - Simd128\_impl< true, true, true, 2 >, 596
  - Simd128\_impl< true, true, true, 4 >, 604
  - Simd128\_impl< true, true, true, 8 >, 613
  - Simd256\_impl< true, false, true, 8 >, 624
  - Simd256\_impl< true, true, false, 2 >, 634
  - Simd256\_impl< true, true, false, 4 >, 648
  - Simd256\_impl< true, true, false, 8 >, 659
  - Simd256\_impl< true, true, true, 2 >, 666
  - Simd256\_impl< true, true, true, 4 >, 676, 682
  - Simd256\_impl< true, true, true, 8 >, 691
  - Simd512\_impl< true, false, true, 8 >, 701
  - Simd512\_impl< true, true, false, 8 >, 710
  - Simd512\_impl< true, true, true, 8 >, 718
- add\_r
  - FieldSimd< \_Field >, 444
- addin
  - FieldSimd< \_Field >, 444
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 556
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 560
  - Simd128\_impl< true, true, false, 2 >, 571
  - Simd128\_impl< true, true, false, 4 >, 580
  - Simd128\_impl< true, true, false, 8 >, 588
  - Simd128\_impl< true, true, true, 2 >, 596
  - Simd128\_impl< true, true, true, 4 >, 604
  - Simd128\_impl< true, true, true, 8 >, 613
  - Simd256\_impl< true, false, true, 8 >, 624
  - Simd256\_impl< true, true, false, 2 >, 634
  - Simd256\_impl< true, true, false, 4 >, 648
  - Simd256\_impl< true, true, false, 8 >, 659
  - Simd256\_impl< true, true, true, 2 >, 666
  - Simd256\_impl< true, true, true, 4 >, 676, 682
  - Simd256\_impl< true, true, true, 8 >, 691
  - Simd512\_impl< true, false, true, 8 >, 701
  - Simd512\_impl< true, true, false, 8 >, 710
  - Simd512\_impl< true, true, true, 8 >, 718
- addin\_r
  - FieldSimd< \_Field >, 445
- addp
  - FFLAS::vectorised, 287
- AlgoChooser< ModeCategories::ConvertTo< Element-Categories::RNSElementTag >, ParSeq >, 403
  - value, 403
- AlgoChooser< ModeT, ParSeq >, 403
  - value, 403
- align-allocator.h, 761
- alignable
  - FFLAS, 193
- alignable< Givaro::Integer \* >
  - FFLAS, 193
- alignment
  - FieldSimd< \_Field >, 448
  - Simd128\_impl< true, true, false, 2 >, 574
  - Simd128\_impl< true, true, false, 4 >, 582
  - Simd128\_impl< true, true, false, 8 >, 592
  - Simd128\_impl< true, true, true, 2 >, 600
  - Simd128\_impl< true, true, true, 4 >, 609
  - Simd128\_impl< true, true, true, 8 >, 618
  - Simd256\_impl< true, false, true, 8 >, 627
  - Simd256\_impl< true, true, false, 2 >, 636
  - Simd256\_impl< true, true, false, 4 >, 653
  - Simd256\_impl< true, true, false, 8 >, 661
  - Simd256\_impl< true, true, true, 2 >, 670
  - Simd256\_impl< true, true, true, 4 >, 686
  - Simd256\_impl< true, true, true, 8 >, 695
  - Simd512\_impl< true, false, true, 8 >, 703
  - Simd512\_impl< true, true, false, 8 >, 713
  - Simd512\_impl< true, true, true, 8 >, 723
- Amax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 500
- Amin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 500
- applyP
  - FFPACK, 309, 310, 365
- applyP\_block
  - FFPACK, 356
- applyP\_modular\_double
  - ffpack.C, 929
  - ffpack\_c.h, 962
- ArbitraryPreclntTag, 403
- areEqual
  - RNSIntegerMod< RNS >, 551
- AreEqual< X, X >, 404
  - value, 404
- AreEqual< X, Y >, 404
  - value, 404
- args-parser.h, 761
  - ArgumentType, 762
  - END\_OF\_ARGUMENTS, 761
  - findArgument, 762
  - getListArgs, 762
  - printHelpMessage, 762
  - TYPE\_BOOL, 761
  - TYPE\_DOUBLE, 762
  - TYPE\_INT, 762
  - TYPE\_INTEGER, 762
  - type\_integer, 762
  - TYPE\_INTLIST, 762

- TYPE\_LONGLONG, 762
- TYPE\_NONE, 762
- TYPE\_STR, 762
- TYPE\_UINT64, 762
- Argument, 404
  - c, 405
  - data, 405
  - example, 405
  - helpString, 405
  - type, 405
- ArgumentType
  - args-parser.h, 762
- ArithProg
  - FFPACK::Protected, 398
- arithprog.C, 763
  - CUBE, 763
  - GFOPS, 763
  - main, 764
  - TTimer, 763
- assign
  - RNSInteger< RNS >, 546
  - RNSIntegerMod< RNS >, 550
- associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, 405
  - field, 406
  - type, 406
- associatedDelayedField< const Givaro::Modular< T, X > >, 406
  - field, 406
  - type, 406
- associatedDelayedField< const Givaro::ModularBalanced< T > >, 406
  - field, 407
  - type, 407
- associatedDelayedField< const Givaro::ZRing< T > >, 407
  - field, 407
  - type, 407
- associatedDelayedField< Field >, 405
  - field, 405
  - type, 405
- assume\_aligned
  - fflas\_sparse.h, 920
- AtlasConj
  - config-blas.h, 801
- Aunfit
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 499
- aut
  - HelperFlag, 470
- Auto, 407
- autotune/charpoly.C
  - CUBE, 793
  - GFOPS, 793
  - main, 794
  - TTimer, 793
- autotune/pluq.C
  - CUBE, 1033
- GFOPS, 1033
  - main, 1033
  - TTimer, 1033
- averageCol
  - StatsMatrix, 750
- averageColDifference
  - StatsMatrix, 750
- averageRow
  - StatsMatrix, 750
- averageRowDifference
  - StatsMatrix, 751
- axpy
  - FieldSimd< \_Field >, 446, 447
- axpy\_r
  - FieldSimd< \_Field >, 447
- axpyin
  - FieldSimd< \_Field >, 447
  - RNSIntegerMod< RNS >, 551
- axpyin\_r
  - FieldSimd< \_Field >, 447
- balanced
  - FieldTraits< FFPACK::RNSInteger< T > >, 449
  - FieldTraits< FFPACK::RNSIntegerMod< T > >, 450
  - FieldTraits< Field >, 448
  - FieldTraits< Givaro::Modular< Element > >, 450
  - FieldTraits< Givaro::ModularBalanced< Element > >, 451
  - FieldTraits< Givaro::ZRing< double > >, 451
  - FieldTraits< Givaro::ZRing< float > >, 452
  - FieldTraits< Givaro::ZRing< Givaro::Integer > >, 452
  - FieldTraits< Givaro::ZRing< int16\_t > >, 453
  - FieldTraits< Givaro::ZRing< int32\_t > >, 453
  - FieldTraits< Givaro::ZRing< int64\_t > >, 454
  - FieldTraits< Givaro::ZRing< Reclnt::ruint< K > >, 455
  - FieldTraits< Givaro::ZRing< uint16\_t > >, 455
  - FieldTraits< Givaro::ZRing< uint32\_t > >, 456
  - FieldTraits< Givaro::ZRing< uint64\_t > >, 456
  - winograd.C, 1110, 1111
- BARRIER
  - parallel.h, 1026
- BASECASE\_K
  - test-lu.C, 1093
- BaseTimer
  - FFLAS, 76
- BasisElement
  - rns\_double, 526
  - rns\_double\_extended, 539
  - RNSInteger< RNS >, 544
  - RNSIntegerMod< RNS >, 548
- begin
  - ForStrategy1D< blocksize\_t, Cut, Param >, 458
  - Info, 475, 476
- BEGIN\_PARALLEL\_MAIN
  - parallel.h, 1027
- benchmark-charpoly-mp.C, 764

- \_\_FFLASFFPACK\_FORCE\_SEQ, 764
- main, 764
- benchmark-charpoly.C, 764
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 765
  - main, 765
  - run\_with\_field, 765
- benchmark-checkers.C, 765
  - \_MAX\_SIZE\_MATRICES, 766
  - \_NR\_TESTS, 766
  - CUBE, 766
  - ENABLE\_ALL\_CHECKINGS, 766
  - main, 766
- benchmark-dgemm.C, 766
  - CBLAS\_GEMM, 767
  - Floats, 767
  - main, 767
  - TTimer, 767
- benchmark-dgetrf.C, 767
  - \_\_FFLASFFPACK\_HAVE\_DGETRF, 768
  - main, 768
  - TTimer, 768
- benchmark-dgetri.C, 768
  - main, 769
  - TTimer, 768
- benchmark-dsytrf.C, 769
  - EFFGFF, 769
  - main, 770
  - TTimer, 769
- benchmark-dtrsm.C, 770
  - main, 770
  - TTimer, 770
- benchmark-dtrtri.C, 770
  - \_\_FFLASFFPACK\_HAVE\_DTRTRI, 771
  - main, 771
  - TTimer, 771
- benchmark-fadd-lvl2.C, 771
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 772
  - main, 772
- benchmark-fdot.C, 772
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 772
  - main, 773
  - run\_with\_field, 772
- benchmark-fgemm-mp.C, 773
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 773
  - main, 773
  - tmain, 773
- benchmark-fgemm-rns.C, 774
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 774
  - ConstElement\_ptr, 775
  - Element\_ptr, 774
  - Field, 774
  - GRAIN, 775
  - main, 775
- PSeq, 775
- RNS, 774
- THREADS, 775
- THREED, 775
- THREEDA, 775
- THREEDIP, 775
- TWOD, 775
- TWODA, 775
- benchmark-fgemm.C, 776
  - CLASSIC\_HYBRID, 776
  - main, 776
- benchmark-fgemv-mp.C, 776
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 777
  - write\_matrix, 777
- benchmark-fgemv.C, 777
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 778
  - benchmark\_disp, 779
  - benchmark\_in\_Field, 779
  - benchmark\_with\_field, 780
  - benchmark\_with\_timer, 779
  - check\_result, 779
  - fill\_value, 778
  - genData, 778
  - main, 780
- benchmark-fgesv.C, 780
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 781
  - main, 781
- benchmark-fsyrk.C, 781
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 781
  - CUBE, 782
  - main, 782
- benchmark-fsytrf.C, 782
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 782
  - \_\_FFPACK\_FSYTRF\_BC\_CROUT, 782
  - CUBE, 782
  - main, 783
- benchmark-ftsrm-mp.C, 783
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 783
  - main, 783
- benchmark-ftsrm.C, 783
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 784
  - main, 784
- benchmark-ftsv.C, 784
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 784
  - main, 784
- benchmark-fttrtri.C, 785
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 785
  - CUBE, 785
  - main, 785



- benchmark-inverse.C, [785](#)
  - CUBE, [786](#)
  - main, [786](#)
- benchmark-lqup-mp.C, [786](#)
  - main, [786](#)
- benchmark-lqup.C, [787](#)
  - CUBE, [787](#)
  - main, [787](#)
- benchmark-pluq.C, [787](#)
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, [788](#)
  - CUBE, [788](#)
  - Field, [788](#)
  - main, [788](#)
  - Rec\_Initialize, [788](#)
  - verification\_PLUQ, [788](#)
- benchmark-wino.C, [789](#)
  - CUBE, [789](#)
  - launch\_wino, [789](#)
  - main, [789](#)
- benchmark\_disp
  - benchmark-fgemv.C, [779](#)
- benchmark\_in\_Field
  - benchmark-fgemv.C, [779](#)
- benchmark\_with\_field
  - benchmark-fgemv.C, [780](#)
- benchmark\_with\_timer
  - benchmark-fgemv.C, [779](#)
- Bini, [407](#)
  - FFLAS::BLAS3, [197](#)
- bit\_manipulation.h, [790](#)
  - \_\_has\_builtin, [790](#)
  - clz, [790](#)
  - ctz, [790](#)
- bitsize
  - FFLAS, [142](#)
- bitsize< Givaro::ZRing< Givaro::Integer > >
  - FFLAS, [142](#)
- blas\_enum
  - config-blas.h, [801](#)
- blend
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [588](#)
  - Simd128\_impl< true, true, true, 2 >, [596](#)
  - Simd128\_impl< true, true, true, 4 >, [604](#)
  - Simd128\_impl< true, true, true, 8 >, [613](#)
  - Simd256\_impl< true, false, true, 8 >, [624](#)
  - Simd256\_impl< true, true, false, 4 >, [648](#)
  - Simd256\_impl< true, true, false, 8 >, [659](#)
  - Simd256\_impl< true, true, true, 4 >, [676](#)
  - Simd256\_impl< true, true, true, 8 >, [691](#)
  - Simd512\_impl< true, false, true, 8 >, [700](#)
  - Simd512\_impl< true, true, false, 8 >, [710](#)
  - Simd512\_impl< true, true, true, 8 >, [718](#)
- blend\_twice
  - Simd256\_impl< true, true, false, 2 >, [634](#)
  - Simd256\_impl< true, true, true, 2 >, [666](#)
- blendv
  - Simd256\_impl< true, false, true, 8 >, [624](#)
  - Simd512\_impl< true, false, true, 8 >, [700](#)
- Block, [407](#)
- BlockCuts
  - FFLAS, [184](#), [186](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >
  - FFLAS, [186](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >
  - FFLAS, [185](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >
  - FFLAS, [186](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >
  - FFLAS, [185](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >
  - FFLAS, [185](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >
  - FFLAS, [186](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >
  - FFLAS, [185](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >
  - FFLAS, [185](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >
  - FFLAS, [186](#)
- BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >
  - FFLAS, [184](#)
- blockcuts.inl, [790](#)
  - \_\_FFLASFFPACK\_MINBLOCKCUTS, [792](#)
  - \_\_FFLASFFPACK\_fflas\_blockcuts\_INL, [792](#)
- blockindex
  - ForStrategy1D< blocksize\_t, Cut, Param >, [458](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [461](#)
- BlockingFactor
  - FFLAS::details, [211](#)
- BLOCKS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- Bmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [500](#)
- Bmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [500](#)
- build
  - ForStrategy1D< blocksize\_t, Cut, Param >, [457](#)
- buildMatrix
  - FFPACK, [350](#)
- Bunfit

- MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 499
- c
  - Argument, 405
  - callLUdivine\_small< double >, 408
    - operator(), 408
  - callLUdivine\_small< Element >, 408
    - operator(), 408
  - callLUdivine\_small< float >, 409
    - operator(), 409
  - cardinality
    - RNSInteger< RNS >, 545
    - RNSIntegerMod< RNS >, 549
  - cast.h, 792
  - category
    - FieldTraits< FFPACK::RNSInteger< T > >, 449
    - FieldTraits< FFPACK::RNSIntegerMod< T > >, 449
    - FieldTraits< Field >, 448
    - FieldTraits< Givaro::Modular< Element > >, 450
    - FieldTraits< Givaro::ModularBalanced< Element > >, 451
    - FieldTraits< Givaro::ZRing< double > >, 451
    - FieldTraits< Givaro::ZRing< float > >, 452
    - FieldTraits< Givaro::ZRing< Givaro::Integer > >, 452
    - FieldTraits< Givaro::ZRing< int16\_t > >, 453
    - FieldTraits< Givaro::ZRing< int32\_t > >, 453
    - FieldTraits< Givaro::ZRing< int64\_t > >, 454
    - FieldTraits< Givaro::ZRing< Reclnt::ruint< K > >, 454
    - FieldTraits< Givaro::ZRing< uint16\_t > >, 455
    - FieldTraits< Givaro::ZRing< uint32\_t > >, 455
    - FieldTraits< Givaro::ZRing< uint64\_t > >, 456
  - cblas.C, 792
    - \_\_FFLASFFPACK\_CONFIGURATION, 792
    - \_\_FFLASFFPACK\_HAVE\_CBLAS, 792
    - main, 793
  - CBLAS\_DIAG
    - config-blas.h, 802
  - CBLAS\_ENUM\_DEFINED\_H
    - config-blas.h, 801
  - CBLAS\_EXTERNALS
    - config-blas.h, 801
  - CBLAS\_GEMM
    - benchmark-dgemm.C, 767
  - cblas\_imptrsm
    - FFLAS, 130
  - CBLAS\_INT
    - config-blas.h, 801
  - CBLAS\_ORDER
    - config-blas.h, 801
  - CBLAS\_SIDE
    - config-blas.h, 802
  - CBLAS\_TRANSPOSE
    - config-blas.h, 801
  - CBLAS\_UPLO
    - config-blas.h, 802
  - CblasColMajor
    - config-blas.h, 801
  - CblasConjTrans
    - config-blas.h, 801
  - CblasLeft
    - config-blas.h, 802
  - CblasLower
    - config-blas.h, 802
  - CblasNonUnit
    - config-blas.h, 802
  - CblasNoTrans
    - config-blas.h, 801
  - CblasRight
    - config-blas.h, 802
  - CblasRowMajor
    - config-blas.h, 801
  - CblasTrans
    - config-blas.h, 801
  - CblasUnit
    - config-blas.h, 802
  - CblasUpper
    - config-blas.h, 802
  - ceil
    - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 556
    - Simd256\_impl< true, false, true, 8 >, 627
    - Simd512\_impl< true, false, true, 8 >, 703
  - changeBS
    - ForStrategy1D< blocksize\_t, Cut, Param >, 459
  - changeCBS
    - ForStrategy2D< blocksize\_t, Cut, Param >, 462
  - changeRBS
    - ForStrategy2D< blocksize\_t, Cut, Param >, 462
  - characteristic
    - RNSInteger< RNS >, 545
    - RNSIntegerMod< RNS >, 549
  - CharPoly
    - FFPACK, 328, 329, 350, 370, 371
  - charpoly.C, 793, 794
  - CharpolyFailed, 409
  - check
    - Checker\_Empty< Field >, 410
    - CheckerImplem\_charpoly< Field, Polynomial >, 410
    - CheckerImplem\_Det< Field >, 411
    - CheckerImplem\_fgemm< Field >, 412
    - CheckerImplem\_ftsm< Field >, 414
    - CheckerImplem\_invert< Field >, 415
    - CheckerImplem\_PLUQ< Field >, 415
  - check1
    - regression-check.C, 1036
  - check2
    - regression-check.C, 1036
  - check3
    - regression-check.C, 1036
  - check4
    - regression-check.C, 1036



- CHECK\_DEPENDENCIES
  - parallel.h, [1026](#)
- check\_eq
  - test-simd.C, [1106](#)
- check\_fdot
  - test-fdot.C, [1065](#)
- check\_fger
  - test-fger.C, [1071](#)
- check\_fsyr2k
  - test-fsyr2k.C, [1077](#)
- check\_fsyrk
  - test-fsyrk.C, [1078](#)
- check\_fsyrk\_bkdiag
  - test-fsyrk.C, [1079](#)
- check\_fsyrk\_diag
  - test-fsyrk.C, [1078](#)
- check\_ftrmm
  - test-ftrmm.C, [1081](#)
- check\_ftrmv
  - test-ftrmv.C, [1083](#)
- check\_ftrsm
  - test-ftrsm.C, [1085](#)
- check\_ftrssyr2k
  - test-ftrssyr2k.C, [1086](#)
- check\_ftrstr
  - test-ftrstr.C, [1087](#)
- check\_ftrsv
  - test-ftrsv.C, [1088](#)
- check\_ftrtri
  - test-ftrtri.C, [1090](#)
- check\_minpoly
  - test-minpoly.C, [1097](#)
- check\_MM
  - test-fgemm.C, [1067](#)
- check\_MV
  - test-fgemv.C, [1069](#)
- check\_result
  - benchmark-fgemv.C, [779](#)
- check\_solve
  - test-solve.C, [1108](#)
- checkA
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [499](#)
- checkB
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [499](#)
- CHECKER, [45](#)
- Checker\_charpoly
  - FFPACK, [308](#)
- checker\_charpoly.inl, [794](#)
  - \_\_FFLASFFPACK\_checker\_charpoly\_INL, [794](#)
- Checker\_Det
  - FFPACK, [307](#)
- checker\_det.inl, [795](#)
  - \_\_FFLASFFPACK\_checker\_det\_INL, [795](#)
- Checker\_Empty
  - Checker\_Empty< Field >, [409](#)
- Checker\_Empty< Field >, [409](#)
  - check, [410](#)
  - Checker\_Empty, [409](#)
- checker\_empty.h, [795](#)
- Checker\_fgemm
  - FFLAS, [74](#)
- checker\_fgemm.inl, [795](#)
  - \_\_FFLASFFPACK\_checker\_fgemm\_INL, [796](#)
- Checker\_ftrsm
  - FFLAS, [74](#)
- checker\_ftrsm.inl, [796](#)
  - \_\_FFLASFFPACK\_checker\_ftrsm\_INL, [796](#)
- Checker\_invert
  - FFPACK, [307](#)
- checker\_invert.inl, [796](#)
  - \_\_FFLASFFPACK\_checker\_invert\_INL, [796](#)
- Checker\_PLUQ
  - FFPACK, [307](#)
- checker\_pluq.inl, [796](#)
  - \_\_FFLASFFPACK\_checker\_pluq\_INL, [797](#)
- CheckerImplem\_charpoly
  - CheckerImplem\_charpoly< Field, Polynomial >, [410](#)
- CheckerImplem\_charpoly< Field, Polynomial >, [410](#)
  - ~CheckerImplem\_charpoly, [410](#)
  - check, [410](#)
  - CheckerImplem\_charpoly, [410](#)
- CheckerImplem\_Det
  - CheckerImplem\_Det< Field >, [411](#)
- CheckerImplem\_Det< Field >, [411](#)
  - ~CheckerImplem\_Det, [411](#)
  - check, [411](#)
  - CheckerImplem\_Det, [411](#)
- CheckerImplem\_fgemm
  - CheckerImplem\_fgemm< Field >, [412](#)
- CheckerImplem\_fgemm< Field >, [412](#)
  - ~CheckerImplem\_fgemm, [412](#)
  - check, [412](#)
  - CheckerImplem\_fgemm, [412](#)
- CheckerImplem\_ftrsm
  - CheckerImplem\_ftrsm< Field >, [413](#)
- CheckerImplem\_ftrsm< Field >, [413](#)
  - ~CheckerImplem\_ftrsm, [413](#)
  - check, [414](#)
  - CheckerImplem\_ftrsm, [413](#)
- CheckerImplem\_invert
  - CheckerImplem\_invert< Field >, [414](#)
- CheckerImplem\_invert< Field >, [414](#)
  - ~CheckerImplem\_invert, [414](#)
  - check, [415](#)
  - CheckerImplem\_invert, [414](#)
- CheckerImplem\_PLUQ
  - CheckerImplem\_PLUQ< Field >, [415](#)
- CheckerImplem\_PLUQ< Field >, [415](#)
  - ~CheckerImplem\_PLUQ, [415](#)
  - check, [415](#)
  - CheckerImplem\_PLUQ, [415](#)
- checkers.doxy, [797](#)
- checkers\_fflas.h, [797](#)

- checkers\_fflas.inl, [797](#)
  - FFLASFFPACK\_checkers\_fflas\_inl\_H, [798](#)
- checkers\_ffpack.h, [798](#)
- checkers\_ffpack.inl, [798](#)
  - FFLASFFPACK\_checkers\_ffpack\_inl\_H, [799](#)
- checkingMessage
  - test-nullspace.C, [1099](#)
- checkMonotonicApplyP
  - test-permutations.C, [1100](#)
- checkOut
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [499](#)
- checkRPM
  - test-rpm.C, [1103](#)
- checkSymmetricRPM
  - test-rpm.C, [1103](#)
- checkZeroDimCharpoly
  - regression-check.C, [1036](#)
- checkZeroDimMinPoly
  - regression-check.C, [1036](#)
- chooseField
  - FFPACK, [393](#)
- chooseField< Givaro::ZRing< double > >
  - FFPACK, [394](#)
- chooseField< Givaro::ZRing< float > >
  - FFPACK, [393](#)
- chooseField< Givaro::ZRing< int32\_t > >
  - FFPACK, [393](#)
- chooseField< Givaro::ZRing< int64\_t > >
  - FFPACK, [393](#)
- chunk
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [746](#)
- chunkSize
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [745](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [747](#)
- clapack.C, [799](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [799](#)
  - \_\_FFLASFFPACK\_HAVE\_CLAPACK, [799](#)
  - \_\_FFLASFFPACK\_HAVE\_LAPACK, [799](#)
  - main, [800](#)
- Classic, [416](#)
- CLASSIC\_HYBRID
  - benchmark-fgemm.C, [776](#)
- clz
  - bit\_manipulation.h, [790](#)
- Cmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- Cmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- col
  - Coo< Field >, [429](#)
  - Coo< ValT, IdxT >, [427](#), [431](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [727](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [728](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [731](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [736](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [738](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [745](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [747](#)
- colblockindex
  - ForStrategy2D< blocksize\_t, Cut, Param >, [461](#)
- colBlockSize
  - ForStrategy2D< blocksize\_t, Cut, Param >, [461](#)
- coldim
  - StatsMatrix, [749](#)
- colnumblocks
  - ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- ColRankProfileSubmatrix
  - FFPACK, [341](#), [376](#)
- ColRankProfileSubmatrix\_modular\_double
  - ffpack.C, [943](#)
  - ffpack\_c.h, [974](#)
- ColRankProfileSubmatrixIndices
  - FFPACK, [340](#), [375](#)
- ColRankProfileSubmatrixIndices\_modular\_double
  - ffpack.C, [942](#)
  - ffpack\_c.h, [973](#)
- Column, [416](#)
- ColumnEchelonForm
  - FFPACK, [321](#), [322](#), [369](#)
- ColumnEchelonForm\_modular\_double
  - ffpack.C, [932](#)
  - ffpack\_c.h, [965](#)
- ColumnEchelonForm\_modular\_float
  - ffpack.C, [933](#)
  - ffpack\_c.h, [966](#)
- ColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [934](#)
  - ffpack\_c.h, [966](#)
- ColumnRankProfile
  - FFPACK, [337](#), [338](#), [375](#)
- ColumnRankProfile\_modular\_double
  - ffpack.C, [941](#)
  - ffpack\_c.h, [972](#)
- COMMA
  - parallel.h, [1028](#)
- CompactElement< double >, [416](#)
  - type, [417](#)
- CompactElement< Element >, [416](#)
  - type, [416](#)
- CompactElement< float >, [417](#)
  - type, [417](#)
- CompactElement< int16\_t >, [417](#)
  - type, [417](#)

- CompactElement< int32\_t >, [417](#)
  - type, [417](#)
- CompactElement< int64\_t >, [418](#)
  - type, [418](#)
- compatible\_data\_type< Field >, [418](#)
  - value, [418](#)
- compatible\_data\_type< Givaro::ZRing< double > >, [418](#)
  - value, [418](#)
- compatible\_data\_type< Givaro::ZRing< float > >, [418](#)
  - value, [419](#)
- compliant
  - NoSimd< T >, [519](#)
  - Simd128\_impl< true, true, false, 2 >, [570](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [587](#)
  - Simd128\_impl< true, true, true, 2 >, [594](#)
  - Simd128\_impl< true, true, true, 4 >, [603](#)
  - Simd128\_impl< true, true, true, 8 >, [611](#)
  - Simd256\_impl< true, false, true, 8 >, [622](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [646](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [664](#)
  - Simd256\_impl< true, true, true, 4 >, [674](#), [680](#)
  - Simd256\_impl< true, true, true, 8 >, [689](#)
  - Simd512\_impl< true, false, true, 8 >, [699](#)
  - Simd512\_impl< true, true, false, 8 >, [709](#)
  - Simd512\_impl< true, true, true, 8 >, [716](#)
- Compose
  - Compose< H1, H2 >, [419](#)
- Compose< H1, H2 >, [419](#)
  - Compose, [419](#)
  - first\_component, [420](#)
  - operator<<, [420](#)
  - second\_component, [420](#)
- composePermutationsLLL
  - FFPACK, [360](#)
  - ffpack.C, [929](#)
  - ffpack\_c.h, [962](#)
- composePermutationsLLM
  - FFPACK, [360](#)
  - ffpack.C, [928](#)
  - ffpack\_c.h, [961](#)
- composePermutationsMLM
  - FFPACK, [361](#)
  - ffpack.C, [929](#)
  - ffpack\_c.h, [962](#)
- CompressRows
  - FFPACK::Protected, [399](#)
- CompressRowsQA
  - FFPACK::Protected, [400](#)
- CompressRowsQK
  - FFPACK::Protected, [400](#)
- computeDeviation
  - FFLAS, [156](#)
- computeFactorClassic
  - FFLAS::Protected, [216](#)
- config-blas.h, [800](#)
  - AtlasConj, [801](#)
  - blas\_enum, [801](#)
  - CBLAS\_DIAG, [802](#)
  - CBLAS\_ENUM\_DEFINED\_H, [801](#)
  - CBLAS\_EXTERNALS, [801](#)
  - CBLAS\_INT, [801](#)
  - CBLAS\_ORDER, [801](#)
  - CBLAS\_SIDE, [802](#)
  - CBLAS\_TRANSPOSE, [801](#)
  - CBLAS\_UPLO, [802](#)
  - CblasColMajor, [801](#)
  - CblasConjTrans, [801](#)
  - CblasLeft, [802](#)
  - CblasLower, [802](#)
  - CblasNonUnit, [802](#)
  - CblasNoTrans, [801](#)
  - CblasRight, [802](#)
  - CblasRowMajor, [801](#)
  - CblasTrans, [801](#)
  - CblasUnit, [802](#)
  - CblasUpper, [802](#)
  - dasum\_, [803](#)
  - daxpy\_, [802](#)
  - dcopy\_, [804](#)
  - ddot\_, [803](#)
  - dgemm\_, [806](#)
  - dgemv\_, [803](#)
  - dger\_, [804](#)
  - dnrm2\_, [803](#)
  - dscal\_, [805](#)
  - dtrmm\_, [805](#)
  - dtrsm\_, [805](#)
  - idamax\_, [803](#)
  - saxpy\_, [802](#)
  - scopy\_, [804](#)
  - sdot\_, [803](#)
  - sgemm\_, [806](#)
  - sgemv\_, [804](#)
  - sger\_, [804](#)
  - sscal\_, [805](#)
  - strmm\_, [806](#)
  - strsm\_, [805](#)
- config.h, [807](#), [811](#)
  - HAVE\_BLAS, [807](#)
  - HAVE\_CBLAS, [808](#)
  - HAVE\_CXX11, [808](#)
  - HAVE\_DLFCN\_H, [808](#)
  - HAVE\_FLOAT\_H, [808](#)
  - HAVE\_INT128, [808](#)
  - HAVE\_INTTYPES\_H, [808](#)
  - HAVE\_LAPACK, [808](#)
  - HAVE\_LIMITS\_H, [808](#)
  - HAVE\_LITTLE\_ENDIAN, [808](#)
  - HAVE\_PTHREAD\_H, [808](#)
  - HAVE\_STDDEF\_H, [808](#)
  - HAVE\_STDINT\_H, [808](#)
  - HAVE\_STDIO\_H, [809](#)

- HAVE\_STDLIB\_H, [809](#)
- HAVE\_STRING\_H, [809](#)
- HAVE\_STRINGS\_H, [809](#)
- HAVE\_SYS\_STAT\_H, [809](#)
- HAVE\_SYS\_TIME\_H, [809](#)
- HAVE\_SYS\_TYPES\_H, [809](#)
- HAVE\_UNISTD\_H, [809](#)
- LT\_OBJDIR, [809](#)
- OPENBLAS\_NUM\_THREADS, [809](#)
- PACKAGE, [809](#)
- PACKAGE\_BUGREPORT, [809](#)
- PACKAGE\_NAME, [810](#)
- PACKAGE\_STRING, [810](#)
- PACKAGE\_TARNAME, [810](#)
- PACKAGE\_URL, [810](#)
- PACKAGE\_VERSION, [810](#)
- SIZEOF\_\_\_INT64, [810](#)
- SIZEOF\_CHAR, [810](#)
- SIZEOF\_INT, [810](#)
- SIZEOF\_LONG, [810](#)
- SIZEOF\_LONG\_LONG, [810](#)
- SIZEOF\_SHORT, [810](#)
- STDC\_HEADERS, [810](#)
- USE\_OPENMP, [811](#)
- VERSION, [811](#)
- CONST
  - fflas\_simd.h, [915](#)
- const\_int
  - instrset.h, [1020](#)
- Const\_int\_t< n >, [420](#)
- const\_uint
  - instrset.h, [1020](#)
- Const\_uint\_t< n >, [420](#)
- ConstElement\_ptr
  - benchmark-fgemm-rns.C, [775](#)
  - rns\_double, [527](#)
  - rns\_double\_extended, [539](#)
  - RNSInteger< RNS >, [544](#)
  - RNSIntegerMod< RNS >, [548](#)
- CONSTREFERENCE
  - parallel.h, [1026](#)
- convert
  - rns\_double, [528](#), [529](#)
  - rns\_double\_extended, [540](#), [541](#)
  - RNSInteger< RNS >, [545](#)
  - RNSIntegerMod< RNS >, [550](#)
- convert\_transpose
  - rns\_double, [528](#)
- ConvertTo< T >, [426](#)
- COO
  - FFLAS, [79](#)
- Coo
  - Coo< Field >, [428](#)
  - Coo< ValT, IdxT >, [427](#), [430](#)
- coo
  - HelperFlag, [470](#)
- Coo< Field >, [428](#)
- col, [429](#)
- Coo, [428](#)
- deleted, [429](#)
- operator=, [428](#), [429](#)
- row, [429](#)
- val, [429](#)
- Coo< ValT, IdxT >, [426](#), [429](#)
- col, [427](#), [431](#)
- Coo, [427](#), [430](#)
- operator=, [427](#), [430](#)
- row, [427](#), [430](#)
- Self, [426](#), [430](#)
- val, [427](#), [430](#)
- coo.h, [815](#)
- coo\_spmv.inl, [815](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL, [816](#)
- coo\_spmv.inl, [816](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL, [817](#)
- coo\_utils.inl, [817](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL, [818](#)
- COO\_ZO
  - FFLAS, [79](#)
- CooMat< Field >, [431](#)
- \_coo16, [431](#)
- \_coo16\_zo, [431](#)
- \_coo32, [431](#)
- \_coo32\_zo, [431](#)
- \_coo64, [431](#)
- \_coo64\_zo, [431](#)
- CROUT
  - ffpack\_pluq.inl, [999](#)
- CSC
  - FFLAS, [79](#)
- CSC\_ZO
  - FFLAS, [79](#)
- CSR
  - FFLAS, [79](#)
- csr
  - HelperFlag, [470](#)
- csr.h, [818](#)
- CSR\_HYB
  - FFLAS, [79](#)
- csr\_hyb.h, [818](#)
- csr\_hyb\_pspmm.inl, [819](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL, [819](#)
- csr\_hyb\_pspmv.inl, [819](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL, [820](#)
- csr\_hyb\_spmv.inl, [820](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL, [820](#)
- csr\_hyb\_spmv.inl, [821](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL, [821](#)
- csr\_hyb\_utils.inl, [821](#)

- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL, 821
- csr\_pspmm.inl, 822
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL, 822
- csr\_pspmv.inl, 822
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL, 823
- csr\_spm.inl, 823
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spm\_INL, 824
- csr\_spmv.inl, 824
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL, 825
- csr\_utils.inl, 825
- CSR\_ZO
  - FFLAS, 79
- CsrMat< Field >, 432
  - \_csr16, 432
  - \_csr16\_zo, 432
  - \_csr32, 432
  - \_csr32\_zo, 432
  - \_csr64, 432
  - \_csr64\_zo, 432
- cst
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 728
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 734
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 738
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 740
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 746
- ctz
  - bit\_manipulation.h, 790
- CUBE
  - arithprog.C, 763
  - autotune/charpoly.C, 793
  - autotune/pluq.C, 1033
  - benchmark-checkers.C, 766
  - benchmark-fsyk.C, 782
  - benchmark-fsytrf.C, 782
  - benchmark-fttri.C, 785
  - benchmark-inverse.C, 786
  - benchmark-lqp.C, 787
  - benchmark-pluq.C, 788
  - benchmark-wino.C, 789
  - fsyk.C, 1009
  - fsytrf.C, 1010
  - fttri.C, 1011
- cuda.C, 826
  - main, 826
- current
  - ForStrategy1D< blocksize\_t, Cut, Param >, 458
  - ForStrategy2D< blocksize\_t, Cut, Param >, 462
- Cut
  - Parallel< C, P >, 520
- cyclic\_shift\_col
  - FFPACK, 362, 365
- cyclic\_shift\_col\_modular\_double
  - ffpack.C, 929
  - ffpack\_c.h, 962
- cyclic\_shift\_mathPerm
  - FFPACK, 361
  - ffpack.C, 929
  - ffpack\_c.h, 962
- cyclic\_shift\_row
  - FFPACK, 361, 362, 365
- cyclic\_shift\_row\_col
  - FFPACK, 361, 364
- cyclic\_shift\_row\_modular\_double
  - ffpack.C, 929
  - ffpack\_c.h, 962
- Danilevski
  - FFPACK, 350
  - FFPACK::Protected, 398
- dasum\_
  - config-blas.h, 803
- dat
  - Sparse< \_Field, SparseMatrix\_t::COO >, 727
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 729
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 731
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 732
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 735
  - Sparse< \_Field, SparseMatrix\_t::ELL >, 736
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 738
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 739
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 741
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 742
  - Sparse< \_Field, SparseMatrix\_t::SELL >, 745
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 747
- data
  - Argument, 405
- daxpy\_
  - config-blas.h, 802
- dcopy\_
  - config-blas.h, 804
- ddot\_
  - config-blas.h, 803
- debug.h, 826
  - FFLASFFPACK\_abort, 827
  - FFLASFFPACK\_check, 827
- DeCompressRows
  - FFPACK::Protected, 400
- DeCompressRowsQA
  - FFPACK::Protected, 401
- DeCompressRowsQK
  - FFPACK::Protected, 400
- DefaultBoundedTag, 432
- DefaultTag, 433
- delayed
  - RNSIntegerMod< RNS >, 548
  - Sparse< \_Field, SparseMatrix\_t::COO >, 727
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 729
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 730

- Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 732
- Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 734
- Sparse< \_Field, SparseMatrix\_t::ELL >, 735
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 737
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 738
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 740
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 742
- Sparse< \_Field, SparseMatrix\_t::SELL >, 744
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 746
- DelayedField
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 497
- delayedField
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 501
- DelayedField\_t
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 497
- DelayedTag, 433
- deleted
  - Coo< Field >, 429
- DENSE\_THRESHOLD
  - fflas\_sparse.h, 921
- denseCols
  - StatsMatrix, 751
- denseRows
  - StatsMatrix, 751
- Det
  - FFPACK, 332, 333, 351, 373
- det.C, 827
  - main, 828
- Det\_modular\_double
  - ffpack.C, 940
  - ffpack\_c.h, 971
- deviationCol
  - StatsMatrix, 750
- deviationColDifference
  - StatsMatrix, 750
- deviationRow
  - StatsMatrix, 750
- deviationRowDifference
  - StatsMatrix, 751
- DfElt
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 497
- dgemm\_
  - config-blas.h, 806
  - fblas.C, 836
- dgemv\_
  - config-blas.h, 803
- dger\_
  - config-blas.h, 804
- digits
  - limits< char >, 484
  - limits< double >, 485
  - limits< float >, 486
  - limits< int >, 487
  - limits< long >, 488
  - limits< long long >, 489
  - limits< short int >, 491
  - limits< signed char >, 491
  - limits< unsigned char >, 492
  - limits< unsigned int >, 493
  - limits< unsigned long >, 494
  - limits< unsigned long long >, 494
  - limits< unsigned short int >, 495
- div
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 557
  - Simd256\_impl< true, false, true, 8 >, 625
  - Simd512\_impl< true, false, true, 8 >, 701
- dnrm2\_
  - config-blas.h, 803
- DNS\_BIN\_VER
  - read\_sparse.h, 1035
- doApplyS
  - FFPACK, 357
- doApplyT
  - FFPACK, 358
- DotProdBoundClassic
  - FFLAS::Protected, 217
- DOUBLE\_TO\_FLOAT\_CROSSOVER
  - fflas.h, 840
  - winograd.C, 1110
- dscal\_
  - config-blas.h, 805
- dtrmm\_
  - config-blas.h, 805
- dtrsm\_
  - config-blas.h, 805
- DynamicPeeling
  - FFLAS::Protected, 218
- DynamicPeeling2
  - FFLAS::Protected, 219
- EFFGFF
  - benchmark-dsytrf.C, 769
- Element
  - FieldSimd< \_Field >, 443
  - readMyMachineType< Field, mpz\_t >, 524
  - readMyMachineType< Field, T >, 523
  - rns\_double, 526
  - rns\_double\_extended, 539
  - RNSInteger< RNS >, 544
  - RNSIntegerMod< RNS >, 548
- Element\_ptr
  - benchmark-fgemm-rns.C, 774
  - readMyMachineType< Field, mpz\_t >, 524
  - readMyMachineType< Field, T >, 524
  - rns\_double, 526
  - rns\_double\_extended, 539
  - RNSInteger< RNS >, 544
  - RNSIntegerMod< RNS >, 548
- ElementTraits< double >, 433

- value, [434](#)
- ElementTraits< Element >, [433](#)
  - value, [433](#)
- ElementTraits< FFPACK::rns\_double\_elt >, [434](#)
  - value, [434](#)
- ElementTraits< float >, [434](#)
  - value, [434](#)
- ElementTraits< Givaro::Integer >, [434](#)
  - value, [435](#)
- ElementTraits< int16\_t >, [435](#)
  - value, [435](#)
- ElementTraits< int32\_t >, [435](#)
  - value, [435](#)
- ElementTraits< int64\_t >, [435](#)
  - value, [436](#)
- ElementTraits< int8\_t >, [436](#)
  - value, [436](#)
- ElementTraits< Reclnt::rint< K > >, [436](#)
  - value, [436](#)
- ElementTraits< Reclnt::rmint< K, MG > >, [436](#)
  - value, [437](#)
- ElementTraits< Reclnt::ruint< K > >, [437](#)
  - value, [437](#)
- ElementTraits< uint16\_t >, [437](#)
  - value, [437](#)
- ElementTraits< uint32\_t >, [437](#)
  - value, [438](#)
- ElementTraits< uint64\_t >, [438](#)
  - value, [438](#)
- ElementTraits< uint8\_t >, [438](#)
  - value, [438](#)
- ELL
  - FFLAS, [79](#)
- ell
  - HelperFlag, [470](#)
- ell.h, [828](#)
- ell\_pspmm.inl, [828](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL, [829](#)
- ell\_pspmv.inl, [829](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL, [830](#)
- ELL\_simd
  - FFLAS, [79](#)
- ell\_simd.h, [830](#)
- ell\_simd\_pspmv.inl, [830](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL, [831](#)
- ell\_simd\_spmv.inl, [831](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL, [832](#)
- ell\_simd\_utils.inl, [832](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL, [833](#)
- ELL\_simd\_ZO
  - FFLAS, [79](#)
- ell\_spmv.inl, [833](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL, [834](#)
- ell\_spmv.inl, [834](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL, [834](#)
- ell\_utils.inl, [835](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL, [835](#)
- ELL\_ZO
  - FFLAS, [79](#)
- ElIMat< Field >, [438](#)
  - \_ell16, [439](#)
  - \_ell16\_zo, [439](#)
  - \_ell32, [439](#)
  - \_ell32\_zo, [439](#)
  - \_ell64, [439](#)
  - \_ell64\_zo, [439](#)
- ENABLE\_ALL\_CHECKINGS
  - benchmark-checkers.C, [766](#)
  - ffpack\_ftrtr.inl, [987](#)
  - test-fdot.C, [1065](#)
  - test-fgemm-check.C, [1066](#)
  - test-fsyr2k.C, [1077](#)
  - test-fsyrk.C, [1078](#)
  - test-ftrmv.C, [1082](#)
  - test-ftrsm-check.C, [1083](#)
  - test-ftrsm.C, [1084](#)
  - test-ftrssyr2k.C, [1086](#)
  - test-ftrstr.C, [1087](#)
  - test-ftrsv.C, [1088](#)
  - test-ftrtri.C, [1089](#)
  - test-invert-check.C, [1091](#)
  - test-pluq-check.C, [1101](#)
- ENABLE\_CHECKER\_charpoly
  - test-charpoly-check.C, [1057](#)
- ENABLE\_CHECKER\_Det
  - test-det-check.C, [1059](#)
- ENABLE\_CHECKER\_fgemm
  - test-fgemm.C, [1067](#)
- end
  - ForStrategy1D< blocksize\_t, Cut, Param >, [458](#)
- END\_OF\_ARGUMENTS
  - args-parser.h, [761](#)
- END\_PARALLEL\_MAIN
  - parallel.h, [1027](#)
- eq
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [563](#)
- Simd128\_impl< true, true, false, 2 >, [573](#)
- Simd128\_impl< true, true, false, 4 >, [581](#)
- Simd128\_impl< true, true, false, 8 >, [590](#)
- Simd128\_impl< true, true, true, 2 >, [598](#)
- Simd128\_impl< true, true, true, 4 >, [607](#)
- Simd128\_impl< true, true, true, 8 >, [616](#)
- Simd256\_impl< true, false, true, 8 >, [626](#)



- Simd256\_impl< true, true, false, 2 >, [635](#)
- Simd256\_impl< true, true, false, 4 >, [651](#)
- Simd256\_impl< true, true, false, 8 >, [660](#)
- Simd256\_impl< true, true, true, 2 >, [669](#)
- Simd256\_impl< true, true, true, 4 >, [679](#), [684](#)
- Simd256\_impl< true, true, true, 8 >, [694](#)
- Simd512\_impl< true, false, true, 8 >, [702](#)
- Simd512\_impl< true, true, false, 8 >, [712](#)
- Simd512\_impl< true, true, true, 8 >, [721](#)
- eval\_func\_on\_array
  - test-simd.C, [1106](#), [1107](#)
- example
  - Argument, [405](#)
- examples/charpoly.C
  - main, [794](#)
- examples/pluq.C
  - main, [1034](#)
- fadd
  - FFLAS, [81–83](#), [86](#), [167](#), [168](#), [175](#), [176](#)
  - FFLAS::details, [204](#), [205](#)
- fadd\_1\_modular\_double
  - fflas\_c.h, [848](#)
  - fflas\_lvl1.C, [901](#)
- fadd\_2\_modular\_double
  - fflas\_c.h, [852](#)
  - fflas\_lvl2.C, [906](#)
- faddin
  - FFLAS, [81](#), [85](#), [167](#), [177](#)
- faddin\_1\_modular\_double
  - fflas\_c.h, [849](#)
  - fflas\_lvl1.C, [901](#)
- faddin\_2\_modular\_double
  - fflas\_c.h, [853](#)
  - fflas\_lvl2.C, [907](#)
- Failure, [439](#)
  - \_errorStream, [441](#)
  - Failure, [440](#)
  - operator(), [440](#)
  - print, [441](#)
  - setErrorStream, [440](#)
- failure
  - FFPACK, [379](#)
- FailureCharpolyCheck, [441](#)
- FailureDetCheck, [441](#)
- FailureFgemmCheck, [441](#)
- FailureInvertCheck, [441](#)
- FailurePLUQCheck, [442](#)
- FailureTrsmCheck, [442](#)
- fassign
  - FFLAS, [87](#), [88](#), [163](#), [168](#)
- fassign\_1\_modular\_double
  - fflas\_c.h, [847](#)
  - fflas\_lvl1.C, [900](#)
- fassign\_2\_modular\_double
  - fflas\_c.h, [849](#)
  - fflas\_lvl2.C, [903](#)
- faxpby
  - FFLAS, [135](#), [141](#)
- faxpy
  - FFLAS, [89](#), [90](#), [165](#), [173](#)
- faxpy\_1\_modular\_double
  - fflas\_c.h, [848](#)
  - fflas\_lvl1.C, [900](#)
- faxpy\_2\_modular\_double
  - fflas\_c.h, [851](#)
  - fflas\_lvl2.C, [905](#)
- fblas.C, [835](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [835](#)
  - dgemm\_, [836](#)
  - main, [836](#)
- fconvert
  - FFLAS, [132](#), [139](#), [161](#)
- fconvert\_rns
  - FFLAS, [158](#), [159](#)
- fconvert\_trans\_rns
  - FFLAS, [158](#)
- fdot
  - FFLAS, [90–92](#), [136](#), [166](#), [187](#)
- fdot\_1\_modular\_double
  - fflas\_c.h, [848](#)
  - fflas\_lvl1.C, [901](#)
- fequal
  - FFLAS, [135](#), [137](#), [163](#), [169](#)
- fequal\_1\_modular\_double
  - fflas\_c.h, [847](#)
  - fflas\_lvl1.C, [900](#)
- fequal\_2\_modular\_double
  - fflas\_c.h, [850](#)
  - fflas\_lvl2.C, [903](#)
- FFLAS, [45](#), [49](#)
  - alignable, [193](#)
  - alignable< Givaro::Integer \* >, [193](#)
  - BaseTimer, [76](#)
  - bitsize, [142](#)
  - bitsize< Givaro::ZRing< Givaro::Integer > >, [142](#)
  - BlockCuts, [184](#), [186](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >, [186](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >, [185](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >, [186](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >, [185](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >, [185](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >, [186](#)
  - BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >, [185](#)
  - BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >, [185](#)
  - BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >, [186](#)
  - BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >, [184](#)



cblas\_imptrsm, 130  
Checker\_fgemm, 74  
Checker\_ftrsm, 74  
computeDeviation, 156  
COO, 79  
COO\_ZO, 79  
CSC, 79  
CSC\_ZO, 79  
CSR, 79  
CSR\_HYB, 79  
CSR\_ZO, 79  
ELL, 79  
ELL\_simd, 79  
ELL\_simd\_ZO, 79  
ELL\_ZO, 79  
fadd, 81–83, 86, 167, 168, 175, 176  
faddin, 81, 85, 167, 177  
fassign, 87, 88, 163, 168  
faxpby, 135, 141  
faxpy, 89, 90, 165, 173  
fconvert, 132, 139, 161  
fconvert\_rns, 158, 159  
fconvert\_trans\_rns, 158  
fdot, 90–92, 136, 166, 187  
fequal, 135, 137, 163, 169  
FFLAS\_BASE, 78  
fflas\_delete, 157, 194  
FFLAS\_DIAG, 78  
FFLAS\_FORMAT, 79  
fflas\_new, 157–159, 194  
FFLAS\_ORDER, 77  
FFLAS\_SIDE, 78  
FFLAS\_TRANSPOSE, 77  
FFLAS\_UPLO, 77  
FflasAuto, 80  
FflasBinary, 80  
FflasColMajor, 77  
FflasDense, 80  
FflasDouble, 79  
FflasFloat, 79  
FflasGeneric, 79  
FflasLeft, 78  
FflasLower, 78  
FflasMaple, 80  
FflasMath, 80  
FflasNonUnit, 78  
FflasNoTrans, 77  
FflasRight, 78  
FflasRowMajor, 77  
FflasSageMath, 80  
FflasSMS, 80  
FflasTrans, 77  
FflasUnit, 78  
FflasUpper, 78  
fgemm, 93–100, 146, 182, 183  
fgemv, 102–108, 177, 190  
fger, 108–112, 179  
fidentity, 138, 170  
finit, 114, 116, 131, 139, 160, 171  
finit\_rns, 157, 159  
finit\_trans\_rns, 158  
fiszero, 134, 138, 163, 169  
fmove, 141, 174  
fneg, 133, 140, 162, 172  
fnegin, 132, 140, 161, 172  
ForceCheck\_fgemm, 74  
ForceCheck\_ftrsm, 74  
frand, 134, 137  
freduce, 112–115, 117, 159, 160, 170, 171  
freduce\_constoverride, 113, 115  
freivalds, 117  
fscal, 118–122, 164, 173  
fscaln, 118–122, 164, 173  
fspmm, 156  
fspmv, 154, 156  
fsquare, 100–102, 184  
fsub, 81, 85, 167, 175  
fsubin, 81, 86, 176  
fswap, 136, 166  
fsyr2k, 123  
fsyrk, 124–126  
ftrmm, 127, 128, 181  
ftrmv, 143  
ftrsm, 129, 130, 143, 146, 147, 180  
ftrsv, 130, 179  
fzero, 133, 136, 162, 168  
getDataTypes, 153, 154  
getSeed, 195  
getStat, 156  
getTLBSize, 194  
has\_equal, 76  
has\_minus, 75  
has\_minus\_eq, 76  
has\_mul, 76  
has\_mul\_eq, 76  
has\_plus, 75  
has\_plus\_eq, 76  
HYB\_ZO, 79  
igemm\_, 131  
InfNorm, 80  
max3, 80  
max4, 80  
min3, 80  
min4, 80  
MKLSparseMatrixFormat, 75  
mone, 79  
NoSimdSparseMatrix, 75  
NotMKLSparseMatrixFormat, 75  
NotZOSparseMatrix, 75  
number\_kind, 79  
one, 79  
operator<<, 152  
other, 79  
parseArguments, 191  
pfadd, 82  
pfaddin, 83

- pfgemm, 144, 187–189
- pfgemm\_1D\_rec, 144
- pfgemm\_2D\_rec, 145
- pfgemm\_3D\_rec, 145
- pfgemm\_3D\_rec2, 145
- pfrand, 187
- pfreduce, 115
- pfsub, 82
- pfsubin, 83
- pfzero, 187
- preamble, 192
- prefetch, 194
- queryCacheSizes, 195
- queryL1CacheSize, 195
- queryTopLevelCacheSize, 195
- readDnsFormat, 154
- readMachineType, 154
- ReadMatrix, 192
- readSmsFormat, 153
- readSprFormat, 153
- SELL, 79
- SELL\_ZO, 79
- SimdSparseMatrix, 75
- sparse\_delete, 147, 148, 150–152, 155
- sparse\_init, 147–152, 155
- sparse\_print, 148, 151, 155
- SparseMatrix\_t, 79
- SysTimer, 77
- Timer, 76
- UserTimer, 77
- writeCommandString, 191
- writeDnsFormat, 154
- WriteMatrix, 191, 193
- WritePermutation, 193
- zero, 79
- ZOSparseMatrix, 74
- fflas-101\_1.C, 836
  - main, 836
- fflas-101\_3.C, 836
  - main, 837
- FFLAS-FFPACK, 46
- FFLAS-FFPACK fields, 46
- fflas-ffpack-config.h, 837
  - GCC\_VERSION, 837
- fflas-ffpack-default-thresholds.h, 837
  - \_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD, 838
  - \_\_FFLASFFPACK\_CHARPOLY\_Danilevskii\_LUKrylov\_THRESHOLD, 838
  - \_\_FFLASFFPACK\_CHARPOLY\_LUKrylov\_ArithProg\_THRESHOLD, 838
  - \_\_FFLASFFPACK\_FSYRK\_THRESHOLD, 838
  - \_\_FFLASFFPACK\_FSYTRF\_THRESHOLD, 838
  - \_\_FFLASFFPACK\_FTRTRI\_THRESHOLD, 838
  - \_\_FFLASFFPACK\_PLUQ\_THRESHOLD, 838
  - \_\_FFLASFFPACK\_WINOTHRESHOLD, 838
  - \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL, 838
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT, 838
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT, 838
- fflas-ffpack-thresholds.h, 839
- fflas-ffpack.doxy, 839
- fflas-ffpack.h, 839
- fflas-ffpack/config.h
  - \_\_FFLASFFPACK\_HAVE\_BLAS, 812
  - \_\_FFLASFFPACK\_HAVE\_CBLAS, 812
  - \_\_FFLASFFPACK\_HAVE\_CXX11, 812
  - \_\_FFLASFFPACK\_HAVE\_DLFCN\_H, 812
  - \_\_FFLASFFPACK\_HAVE\_FLOAT\_H, 812
  - \_\_FFLASFFPACK\_HAVE\_INT128, 812
  - \_\_FFLASFFPACK\_HAVE\_INTPYPES\_H, 812
  - \_\_FFLASFFPACK\_HAVE\_LAPACK, 812
  - \_\_FFLASFFPACK\_HAVE\_LIMITS\_H, 812
  - \_\_FFLASFFPACK\_HAVE\_LITTLE\_ENDIAN, 812
  - \_\_FFLASFFPACK\_HAVE\_PTHREAD\_H, 812
  - \_\_FFLASFFPACK\_HAVE\_STDDEF\_H, 812
  - \_\_FFLASFFPACK\_HAVE\_STDINT\_H, 813
  - \_\_FFLASFFPACK\_HAVE\_STDIO\_H, 813
  - \_\_FFLASFFPACK\_HAVE\_STDLIB\_H, 813
  - \_\_FFLASFFPACK\_HAVE\_STRINGS\_H, 813
  - \_\_FFLASFFPACK\_HAVE\_STRING\_H, 813
  - \_\_FFLASFFPACK\_HAVE\_SYS\_STAT\_H, 813
  - \_\_FFLASFFPACK\_HAVE\_SYS\_TIME\_H, 813
  - \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H, 813
  - \_\_FFLASFFPACK\_HAVE\_UNISTD\_H, 813
  - \_\_FFLASFFPACK\_LT\_OBJDIR, 813
  - \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS, 813
  - \_\_FFLASFFPACK\_PACKAGE, 813
  - \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT, 814
  - \_\_FFLASFFPACK\_PACKAGE\_NAME, 814
  - \_\_FFLASFFPACK\_PACKAGE\_STRING, 814
  - \_\_FFLASFFPACK\_PACKAGE\_TARNAME, 814
  - \_\_FFLASFFPACK\_PACKAGE\_URL, 814
  - \_\_FFLASFFPACK\_PACKAGE\_VERSION, 814
  - \_\_FFLASFFPACK\_SIZEOF\_CHAR, 814
  - \_\_FFLASFFPACK\_SIZEOF\_INT, 814
  - \_\_FFLASFFPACK\_SIZEOF\_LONG, 814
  - \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG, 814
  - \_\_FFLASFFPACK\_SIZEOF\_SHORT, 814
  - \_\_FFLASFFPACK\_SIZEOF\_\_INT64, 814
  - \_\_FFLASFFPACK\_STDC\_HEADERS, 815
  - \_\_FFLASFFPACK\_USE\_OPENMP, 815
  - \_\_FFLASFFPACK\_VERSION, 815
- fflas.h, 839
  - FFLAS::BLAS3, 196
  - Bini, 197
  - Winograd, 198
  - Winograd\_L\_S, 201
  - Winograd\_LR\_S, 201
  - Winograd\_R\_S, 202
  - WinogradAcc\_2\_24, 199

- WinogradAcc\_2\_27, 199
- WinogradAcc\_3\_21, 199
- WinogradAcc\_3\_23, 198
- WinogradAcc\_L\_S, 201
- WinogradAcc\_LR, 200
- WinogradAcc\_R\_S, 200
- WinoPar, 197
- FFLAS::csr\_hyb\_details, 202
- FFLAS::CuttingStrategy, 202
  - RNSModulus, 203
- FFLAS::details, 203
  - BlockingFactor, 211
  - fadd, 204, 205
  - freduce, 206
  - fscal, 207
  - fscalin, 207
  - gebp, 210
  - igebb11, 209
  - igebb14, 208
  - igebb21, 209
  - igebb24, 208
  - igebb41, 208
  - igebb44, 208
  - igebp, 209
  - pack\_lhs, 210
  - pack\_rhs, 210
- FFLAS::details\_spmv, 211
- FFLAS::ElementCategories, 211
- FFLAS::FieldCategories, 212
- FFLAS::MMHelperAlgo, 212
- FFLAS::ModeCategories, 212
- FFLAS::ParSeqHelper, 213
- FFLAS::Protected, 213
  - computeFactorClassic, 216
  - DotProdBoundClassic, 217
  - DynamicPeeling, 218
  - DynamicPeeling2, 219
  - fgemm\_convert, 220
  - fgemv\_convert, 222
  - fger\_convert, 223
  - fsquareCommon, 222
  - igemm, 225
  - igemm\_colmajor, 225
  - MatF2MatD\_Triangular, 226
  - MatF2MatFI\_Triangular, 226
  - min\_types, 223, 224
  - NeedDoublePreAddReduction, 221
  - NeedPreAddReduction, 220
  - NeedPreSubReduction, 221
  - ScalAndReduce, 222
  - TRSMBound, 217
  - unfit, 224
  - WinogradCalc, 219
  - WinogradSteps, 218
  - WinogradThreshold, 217, 218
- FFLAS::sell\_details, 226
- FFLAS::sparse\_details, 227
  - fspmm, 233–236
  - fspmm\_dispatch, 233
  - fspmv, 230–232, 241, 242
  - fspmv\_dispatch, 230
  - init\_y, 229, 230
  - pfspmm, 237–240
  - pfspmm\_dispatch, 236, 237
  - pfspmv, 240, 241
- FFLAS::sparse\_details\_impl, 242
  - fspmm, 251, 258, 259, 265, 270, 279, 280
  - fspmm\_mone, 252, 260, 271
  - fspmm\_mone\_simd\_aligned, 253, 261, 272
  - fspmm\_mone\_simd\_unaligned, 253, 261, 272
  - fspmm\_one, 252, 260, 271
  - fspmm\_one\_simd\_aligned, 252, 260, 272
  - fspmm\_one\_simd\_unaligned, 253, 260, 272
  - fspmm\_simd\_aligned, 251, 259
  - fspmm\_simd\_unaligned, 252, 259
  - fspmv, 253, 254, 261, 262, 266, 273, 276, 280–283
  - fspmv\_mone, 254, 255, 262, 274, 277, 283, 284
  - fspmv\_mone\_simd, 278, 284
  - fspmv\_one, 254, 262, 273, 274, 277, 283, 284
  - fspmv\_one\_simd, 277, 284
  - fspmv\_simd, 276, 282, 283
  - pfspmm, 255, 263, 264, 266–268, 278
  - pfspmm\_mone, 256
  - pfspmm\_one, 256
  - pfspmm\_zo, 268
  - pfspmv, 257, 264, 265, 268, 269, 274, 275, 279, 281
  - pfspmv\_mone, 258, 269, 270, 275, 282
  - pfspmv\_one, 258, 269, 275, 281, 282
  - pfspmv\_task, 257
- FFLAS::StrategyParameter, 285
- FFLAS::StructureHelper, 285
- FFLAS::vectorised, 285
  - add, 287
  - addp, 287
  - modp, 289, 290
  - reduce, 288, 289
  - scalp, 290
  - sub, 288
  - subp, 287
  - VEC\_ADD, 286
  - VEC\_SUB, 287
- FFLAS::vectorised::unswitch, 291
  - modp, 291
  - scalp, 291, 292
- fflas\_101.C, 840
  - main, 840
- fflas\_101\_lvl1.C, 841
  - main, 841
- FFLAS\_BASE
  - FFLAS, 78
- fflas\_bounds.inl, 841
  - \_\_FFLASFFPACK\_fflas\_bounds\_INL, 842
  - FFLAS\_INT\_TYPE, 842
- fflas\_c.h, 842
  - fadd\_1\_modular\_double, 848

fadd\_2\_modular\_double, 852  
 faddin\_1\_modular\_double, 849  
 faddin\_2\_modular\_double, 853  
 fassign\_1\_modular\_double, 847  
 fassign\_2\_modular\_double, 849  
 faxpy\_1\_modular\_double, 848  
 faxpy\_2\_modular\_double, 851  
 fdot\_1\_modular\_double, 848  
 fequal\_1\_modular\_double, 847  
 fequal\_2\_modular\_double, 850  
 FFLAS\_C\_BASE, 846  
 FFLAS\_C\_DIAG, 845  
 FFLAS\_C\_ORDER, 844  
 FFLAS\_C\_SIDE, 845  
 FFLAS\_C\_TRANSPOSE, 845  
 FFLAS\_C\_UPLO, 845  
 FFLAS\_COMPILED, 844  
 FflasColMajor, 844  
 FflasDouble, 846  
 FflasFloat, 846  
 FflasGeneric, 846  
 FflasLeft, 845  
 FflasLower, 845  
 FflasNonUnit, 845  
 FflasNoTrans, 845  
 FflasRight, 845  
 FflasRowMajor, 844  
 FflasTrans, 845  
 FflasUnit, 845  
 FflasUpper, 845  
 fgemm\_3\_modular\_double, 854  
 fgemv\_2\_modular\_double, 853  
 fger\_2\_modular\_double, 853  
 fidentity\_2\_modular\_double, 850  
 fiszero\_1\_modular\_double, 847  
 fiszero\_2\_modular\_double, 850  
 fmove\_2\_modular\_double, 852  
 fneg\_1\_modular\_double, 846  
 fneg\_2\_modular\_double, 851  
 fnegin\_1\_modular\_double, 846  
 fnegin\_2\_modular\_double, 851  
 freduce\_1\_modular\_double, 846  
 freduce\_2\_modular\_double, 850  
 freducein\_1\_modular\_double, 846  
 freducein\_2\_modular\_double, 850  
 fscale\_1\_modular\_double, 847  
 fscale\_2\_modular\_double, 851  
 fscalein\_1\_modular\_double, 847  
 fscalein\_2\_modular\_double, 851  
 fsquare\_3\_modular\_double, 855  
 fsub\_1\_modular\_double, 849  
 fsub\_2\_modular\_double, 852  
 fsubin\_1\_modular\_double, 849  
 fsubin\_2\_modular\_double, 852  
 fswap\_1\_modular\_double, 848  
 ftrmm\_3\_modular\_double, 854  
 ftrsm\_3\_modular\_double, 854  
 ftrsv\_2\_modular\_double, 853  
 fzero\_1\_modular\_double, 847  
 fzero\_2\_modular\_double, 849  
 FFLAS\_C\_BASE  
     fflas\_c.h, 846  
 FFLAS\_C\_DIAG  
     fflas\_c.h, 845  
     ffpack\_c.h, 959  
 FFLAS\_C\_ORDER  
     fflas\_c.h, 844  
     ffpack\_c.h, 958  
 FFLAS\_C\_SIDE  
     fflas\_c.h, 845  
     ffpack\_c.h, 959  
 FFLAS\_C\_TRANSPOSE  
     fflas\_c.h, 845  
     ffpack\_c.h, 958  
 FFLAS\_C\_UPLO  
     fflas\_c.h, 845  
     ffpack\_c.h, 959  
 FFLAS\_COMPILED  
     fflas\_c.h, 844  
     ffpack\_inst.C, 987  
     ffpack\_inst.h, 989  
 fflas\_const\_cast  
     FFPACK, 364, 379  
 fflas\_delete  
     FFLAS, 157, 194  
 FFLAS\_DIAG  
     FFLAS, 78  
 FFLAS\_ELT  
     fflas\_L1\_inst.C, 879, 880  
     fflas\_L1\_inst.h, 881  
     fflas\_L2\_inst.C, 883, 884  
     fflas\_L2\_inst.h, 884, 885  
     fflas\_L3\_inst.C, 887, 888  
     fflas\_L3\_inst.h, 888, 889  
     ffpack\_inst.C, 988  
     ffpack\_inst.h, 989  
 fflas\_enum.h, 855  
 fflas\_fadd.h, 856  
 fflas\_fadd.inl, 857  
     \_\_FFLASFFPACK\_fadd\_INL, 858  
 fflas\_fassign.h, 858  
 fflas\_fassign.inl, 858  
     \_\_FFLASFFPACK\_fassign\_INL, 859  
 fflas\_faxpy.inl, 859  
     \_\_FFLASFFPACK\_faxpy\_INL, 860  
 fflas\_fdot.inl, 860  
     \_\_FFLASFFPACK\_fdot\_INL, 860  
 fflas\_fgemm.inl, 861  
     \_\_FFLASFFPACK\_fgemm\_INL, 863  
 fflas\_fgemv.inl, 863  
     \_\_FFLASFFPACK\_fgemv\_INL, 864  
 fflas\_fgemv\_mp.inl, 864  
     \_\_FFLASFFPACK\_fgemv\_mp\_INL, 865  
 fflas\_fger.inl, 865  
     \_\_FFLASFFPACK\_fger\_INL, 866  
 fflas\_fger\_mp.inl, 866

- \_\_FFPACK\_fger\_mp\_INL, 867
- FFLAS\_FIELD
  - fflas\_L1\_inst.C, 879, 880
  - fflas\_L1\_inst.h, 881
  - fflas\_L2\_inst.C, 883
  - fflas\_L2\_inst.h, 884, 885
  - fflas\_L3\_inst.C, 887, 888
  - fflas\_L3\_inst.h, 888, 889
  - ffpack\_inst.C, 988
  - ffpack\_inst.h, 989
- FFLAS\_FORMAT
  - FFLAS, 79
- fflas\_freduces.h, 867
- fflas\_freduces.inl, 868
  - \_\_FFLASFFPACK\_fflas\_freduces\_INL, 869
  - FFLASFFPACK\_COPY\_REDUCE, 869
- fflas\_freduces\_mp.inl, 870
  - \_\_FFLASFFPACK\_fflas\_freduces\_mp\_INL, 870
- fflas\_freivalds.inl, 870
  - \_\_FFLASFFPACK\_freivalds\_INL, 870
- fflas\_fscal.h, 871
- fflas\_fscal.inl, 871
  - \_\_FFLASFFPACK\_fscal\_INL, 872
- fflas\_fscal\_mp.inl, 872
  - \_\_FFLASFFPACK\_fscal\_mp\_INL, 873
- fflas\_fsyr2k.inl, 873
  - \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL, 873
- fflas\_fsyrk.inl, 873
  - \_\_FFLASFFPACK\_fflas\_fsyrk\_INL, 874
- fflas\_ftrmm.inl, 874
  - \_\_FFLASFFPACK\_ftrmm\_INL, 875
- fflas\_ftrsm.inl, 875
  - \_\_FFLASFFPACK\_ftrsm\_INL, 875
- fflas\_ftrsm\_mp.inl, 876
  - \_\_FFPACK\_ftrsm\_mp\_INL, 876
- fflas\_ftrsv.inl, 876
  - \_\_FFLASFFPACK\_ftrsv\_INL, 877
- fflas\_helpers.inl, 877
  - \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL, 878
- FFLAS\_INT\_TYPE
  - fflas\_bounds.inl, 842
- fflas\_intrinsic.h, 878
- fflas\_io.h, 878
- fflas\_L1\_inst.C, 879
  - \_\_FFLAS\_L1\_INST\_C, 879
  - FFLAS\_ELT, 879, 880
  - FFLAS\_FIELD, 879, 880
  - INST\_OR\_DECL, 879
- fflas\_L1\_inst.h, 880
  - FFLAS\_ELT, 881
  - FFLAS\_FIELD, 881
  - INST\_OR\_DECL, 880
- fflas\_L1\_inst\_implement.inl, 881
- fflas\_L2\_inst.C, 883
  - \_\_FFLAS\_L2\_INST\_C, 883
  - FFLAS\_ELT, 883, 884
  - FFLAS\_FIELD, 883
  - INST\_OR\_DECL, 883
- fflas\_L2\_inst.h, 884
  - FFLAS\_ELT, 884, 885
  - FFLAS\_FIELD, 884, 885
  - INST\_OR\_DECL, 884
- fflas\_L2\_inst\_implement.inl, 885
- fflas\_L3\_inst.C, 887
  - \_\_FFLAS\_L3\_INST\_C, 887
  - FFLAS\_ELT, 887, 888
  - FFLAS\_FIELD, 887, 888
  - INST\_OR\_DECL, 887
- fflas\_L3\_inst.h, 888
  - FFLAS\_ELT, 888, 889
  - FFLAS\_FIELD, 888, 889
  - INST\_OR\_DECL, 888
- fflas\_L3\_inst\_implement.inl, 889
  - \_\_FFLAS\_\_TRSM\_READONLY, 890
- fflas\_level1.inl, 890
  - \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL, 892
- fflas\_level2.inl, 892
  - \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL, 895
- fflas\_level3.inl, 895
  - \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL, 897
  - \_\_FFLAS\_\_TRSM\_READONLY, 897
- fflas\_lvl1.C, 897
  - fadd\_1\_modular\_double, 901
  - faddin\_1\_modular\_double, 901
  - fassign\_1\_modular\_double, 900
  - faxpy\_1\_modular\_double, 900
  - fdot\_1\_modular\_double, 901
  - fequal\_1\_modular\_double, 900
  - fiszero\_1\_modular\_double, 899
  - fneg\_1\_modular\_double, 899
  - fnegin\_1\_modular\_double, 899
  - freduce\_1\_modular\_double, 899
  - freducein\_1\_modular\_double, 898
  - fscal\_1\_modular\_double, 900
  - fscalin\_1\_modular\_double, 900
  - fsub\_1\_modular\_double, 901
  - fsubin\_1\_modular\_double, 902
  - fswap\_1\_modular\_double, 901
  - fzero\_1\_modular\_double, 899
- fflas\_lvl2.C, 902
  - fadd\_2\_modular\_double, 906
  - faddin\_2\_modular\_double, 907
  - fassign\_2\_modular\_double, 903
  - faxpy\_2\_modular\_double, 905
  - fequal\_2\_modular\_double, 903
  - fgemv\_2\_modular\_double, 907
  - fger\_2\_modular\_double, 907
  - fidentity\_2\_modular\_double, 904
  - fiszero\_2\_modular\_double, 904
  - fmove\_2\_modular\_double, 906
  - fneg\_2\_modular\_double, 905
  - fnegin\_2\_modular\_double, 904
  - freduce\_2\_modular\_double, 904
  - freducein\_2\_modular\_double, 904
  - fscal\_2\_modular\_double, 905
  - fscalin\_2\_modular\_double, 905

- fsub\_2\_modular\_double, 906
  - fsubin\_2\_modular\_double, 906
  - ftsv\_2\_modular\_double, 907
  - fzero\_2\_modular\_double, 903
- fflas\_lvl3.C, 908
  - fgemm\_3\_modular\_double, 909
  - fsquare\_3\_modular\_double, 909
  - ftmm\_3\_modular\_double, 909
  - ftsm\_3\_modular\_double, 908
- fflas\_memory.h, 910
- fflas\_new
  - FFLAS, 157–159, 194
- FFLAS\_ORDER
  - FFLAS, 77
- fflas\_pfgemm.inl, 910
  - \_\_FFLASFFPACK\_DIMKPENALTY, 911
  - \_\_FFLASFFPACK\_SEQPARTHRESHOLD, 911
  - \_\_FFLASFFPACK\_fflas\_pfgemm\_INL, 911
- fflas\_pftrsm.inl, 911
  - \_\_FFLASFFPACK\_fflas\_pftrsm\_INL, 911
  - PTRSM\_HYBRID\_THRESHOLD, 911
- fflas\_plevel1.h, 912
- fflas\_randommatrix.h, 912
- FFLAS\_SIDE
  - FFLAS, 78
- fflas\_simd.h, 914
  - CONST, 915
  - FLOAT\_MOD, 915
  - INLINE, 915
  - NORML\_MOD, 915
  - PURE, 915
  - Simd, 916
  - SIMD\_INT, 915
- fflas\_sparse.C, 916
- fflas\_sparse.h, 916
  - \_\_FFLASFFPACK\_CACHE\_LINE\_SIZE, 920
  - assume\_aligned, 920
  - DENSE\_THRESHOLD, 921
  - index\_t, 920
  - ROUND\_DOWN, 920
- fflas\_sparse.inl, 921
  - \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL, 923
- FFLAS\_TRANSPOSE
  - FFLAS, 77
- FFLAS\_UPLO
  - FFLAS, 77
- FflasAuto
  - FFLAS, 80
- FflasBinary
  - FFLAS, 80
- FflasColMajor
  - FFLAS, 77
  - fflas\_c.h, 844
  - ffpack\_c.h, 958
- FflasDense
  - FFLAS, 80
- FflasDouble
  - FFLAS, 79
- fflas\_c.h, 846
- FFLASFFPACK\_abort
  - debug.h, 827
- FFLASFFPACK\_check
  - debug.h, 827
- FFLASFFPACK\_checkers\_fflas\_inl\_H
  - checkers\_fflas.inl, 798
- FFLASFFPACK\_checkers\_ffpack\_inl\_H
  - checkers\_ffpack.inl, 799
- FFLASFFPACK\_COPY\_REDUCE
  - fflas\_freduce.inl, 869
- FFLASFFPACK\_PERM\_BKSIZE
  - ffpack\_permutation.inl, 998
- FflasFloat
  - FFLAS, 79
  - fflas\_c.h, 846
- FflasGeneric
  - FFLAS, 79
  - fflas\_c.h, 846
- FflasLeft
  - FFLAS, 78
  - fflas\_c.h, 845
  - ffpack\_c.h, 959
- FflasLower
  - FFLAS, 78
  - fflas\_c.h, 845
  - ffpack\_c.h, 959
- FflasMaple
  - FFLAS, 80
- FflasMath
  - FFLAS, 80
- FflasNonUnit
  - FFLAS, 78
  - fflas\_c.h, 845
  - ffpack\_c.h, 959
- FflasNoTrans
  - FFLAS, 77
  - fflas\_c.h, 845
  - ffpack\_c.h, 959
- FflasRight
  - FFLAS, 78
  - fflas\_c.h, 845
  - ffpack\_c.h, 959
- FflasRowMajor
  - FFLAS, 77
  - fflas\_c.h, 844
  - ffpack\_c.h, 958
- FflasSageMath
  - FFLAS, 80
- FflasSMS
  - FFLAS, 80
- FflasTrans
  - FFLAS, 77
  - fflas\_c.h, 845
  - ffpack\_c.h, 959
- FflasUnit
  - FFLAS, 78
  - fflas\_c.h, 845

- ffpack\_c.h, [959](#)
- FflasUpper
  - FFLAS, [78](#)
  - fflas\_c.h, [845](#)
  - ffpack\_c.h, [959](#)
- FFPACK, [46](#), [292](#)
  - \_PLUQ, [363](#)
  - applyP, [309](#), [310](#), [365](#)
  - applyP\_block, [356](#)
  - buildMatrix, [350](#)
  - CharPoly, [328](#), [329](#), [350](#), [370](#), [371](#)
  - Checker\_charpoly, [308](#)
  - Checker\_Det, [307](#)
  - Checker\_invert, [307](#)
  - Checker\_PLUQ, [307](#)
  - chooseField, [393](#)
  - chooseField< Givaro::ZRing< double > >, [394](#)
  - chooseField< Givaro::ZRing< float > >, [393](#)
  - chooseField< Givaro::ZRing< int32\_t > >, [393](#)
  - chooseField< Givaro::ZRing< int64\_t > >, [393](#)
  - ColRankProfileSubmatrix, [341](#), [376](#)
  - ColRankProfileSubmatrixIndices, [340](#), [375](#)
  - ColumnEchelonForm, [321](#), [322](#), [369](#)
  - ColumnRankProfile, [337](#), [338](#), [375](#)
  - composePermutationsLLL, [360](#)
  - composePermutationsLLM, [360](#)
  - composePermutationsMLM, [361](#)
  - cyclic\_shift\_col, [362](#), [365](#)
  - cyclic\_shift\_mathPerm, [361](#)
  - cyclic\_shift\_row, [361](#), [362](#), [365](#)
  - cyclic\_shift\_row\_col, [361](#), [364](#)
  - Danilevski, [350](#)
  - Det, [332](#), [333](#), [351](#), [373](#)
  - doApplyS, [357](#)
  - doApplyT, [358](#)
  - failure, [379](#)
  - fflas\_const\_cast, [364](#), [379](#)
  - fgesv, [313](#), [366](#)
  - fgets, [311](#), [312](#), [365](#), [366](#)
  - ForceCheck\_charpoly, [308](#)
  - ForceCheck\_Det, [308](#)
  - ForceCheck\_invert, [308](#)
  - ForceCheck\_PLUQ, [308](#)
  - fsytrf, [316](#), [317](#)
  - fsytrf\_BC\_Crout, [351](#)
  - fsytrf\_BC\_RL, [351](#)
  - fsytrf\_LOW\_RPM\_BC\_Crout, [352](#)
  - fsytrf\_nonunit, [317](#), [353](#)
  - fsytrf\_RPM, [353](#)
  - fsytrf\_UP\_RPM, [352](#)
  - fsytrf\_UP\_RPM\_BC\_Crout, [352](#)
  - fsytrf\_UP\_RPM\_BC\_RL, [352](#)
  - ftssyr2k, [316](#)
  - ftstr, [315](#)
  - fttri, [314](#), [367](#)
  - fttrm, [315](#), [367](#)
  - getEchelonForm, [343](#), [344](#)
  - getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >, [377](#)
  - getEchelonTransform, [345](#)
  - getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >, [377](#)
  - getReducedEchelonForm, [345](#), [346](#)
  - getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >, [378](#)
  - getReducedEchelonTransform, [347](#)
  - getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >, [378](#)
  - getTriangular, [342](#), [343](#)
  - getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >, [376](#)
  - getTridiagonal, [354](#)
  - Invert, [326](#), [370](#)
  - Invert2, [327](#), [370](#)
  - isOdd, [379](#), [380](#)
  - IsSingular, [332](#), [373](#)
  - KrylovElim, [372](#)
  - LAPACKPerm2MathPerm, [308](#)
  - LeadingSubmatrixRankProfiles, [338](#)
  - LQUPtoInverseOfFullRankMinor, [348](#), [379](#)
  - LUdivine, [320](#), [354](#), [355](#), [368](#)
  - LUdivine\_gauss, [354](#), [368](#)
  - LUdivine\_small, [354](#), [368](#)
  - MathPerm2LAPACKPerm, [308](#)
  - MatrixApplyS, [357](#), [358](#)
  - MatrixApplyT, [359](#)
  - MatVecMinPoly, [330](#), [372](#)
  - maxFieldElt, [393](#)
  - maxFieldElt< Givaro::ZRing< Givaro::Integer > >, [393](#)
  - MinPoly, [329](#), [330](#), [371](#)
  - MonotonicApplyP, [310](#)
  - MonotonicCompress, [355](#)
  - MonotonicCompressCycles, [356](#)
  - MonotonicCompressMorePivots, [355](#)
  - MonotonicExpand, [356](#)
  - NonZeroRandomMatrix, [380](#)
  - NullSpaceBasis, [335](#), [374](#)
  - pColumnEchelonForm, [321](#)
  - pColumnRankProfile, [337](#)
  - pDet, [333](#)
  - PermApplyS, [358](#)
  - PermApplyT, [360](#)
  - PLUQ, [318](#), [319](#), [363](#), [364](#), [367](#)
  - PLUQ\_basecaseCrout, [363](#)
  - PLUQ\_basecaseV2, [362](#)
  - PLUQ\_basecaseV3, [362](#)
  - PLUQtoEchelonPermutation, [348](#)
  - pPLUQ, [319](#)
  - pRank, [331](#)
  - pReducedColumnEchelonForm, [324](#)
  - pReducedRowEchelonForm, [325](#)
  - pRowEchelonForm, [323](#)
  - pRowRankProfile, [336](#)
  - pSolve, [334](#)



- RandInt, 383
- RandomIndexSubset, 385
- RandomMatrix, 381
- RandomMatrixWithDet, 392
- RandomMatrixWithRank, 384, 385
- RandomMatrixWithRankandRandomRPM, 389, 390
- RandomMatrixWithRankandRPM, 387
- RandomNullSpaceVector, 334, 349, 374
- RandomPermutation, 386
- RandomRankProfileMatrix, 386
- RandomSymmetricMatrix, 383
- RandomSymmetricMatrixWithRankandRandomRPM, 390, 391
- RandomSymmetricMatrixWithRankandRPM, 388, 389
- RandomSymmetricRankProfileMatrix, 386
- RandomTriangularMatrix, 382, 383
- Rank, 331, 372
- RankProfileFromLU, 338
- ReducedColumnEchelonForm, 323, 324, 369
- ReducedRowEchelonForm, 324, 325, 369
- RowEchelonForm, 322, 323, 369
- RowRankProfile, 336, 337, 375
- RowRankProfileSubmatrix, 341, 376
- RowRankProfileSubmatrixIndices, 339, 375
- Solve, 333, 334, 373
- solveLB, 349, 374
- solveLB2, 349, 374
- SpecRankProfile, 372
- swapval, 386
- threads\_fgemm, 363
- threads\_ftsm, 364
- trinv\_left, 314, 367
- ffpack-fgesv.C, 923
  - main, 923
- ffpack-solve.C, 923
  - main, 923
- ffpack.C, 924
  - applyP\_modular\_double, 929
  - ColRankProfileSubmatrix\_modular\_double, 943
  - ColRankProfileSubmatrixIndices\_modular\_double, 942
  - ColumnEchelonForm\_modular\_double, 932
  - ColumnEchelonForm\_modular\_float, 933
  - ColumnEchelonForm\_modular\_int32\_t, 934
  - ColumnRankProfile\_modular\_double, 941
  - composePermutationsLLL, 929
  - composePermutationsLLM, 928
  - composePermutationsMLM, 929
  - cyclic\_shift\_col\_modular\_double, 929
  - cyclic\_shift\_mathPerm, 929
  - cyclic\_shift\_row\_modular\_double, 929
  - Det\_modular\_double, 940
  - fgesv\_modular\_double, 930
  - fgesvin\_modular\_double, 930
  - fgetrsin\_modular\_double, 930
  - fgetrsv\_modular\_double, 930
  - fttrtri\_modular\_double, 931
  - fttrrm\_modular\_double, 931
  - getEchelonForm\_modular\_double, 943
  - getEchelonFormin\_modular\_double, 944
  - getEchelonTransform\_modular\_double, 944
  - getReducedEchelonForm\_modular\_double, 944
  - getReducedEchelonFormin\_modular\_double, 944
  - getReducedEchelonTransform\_modular\_double, 945
  - getTriangular\_modular\_double, 943
  - getTriangularin\_modular\_double, 943
  - Invert2\_modular\_double, 938
  - Invert\_modular\_double, 938
  - Invertin\_modular\_double, 938
  - IsSingular\_modular\_double, 939
  - KrylovElim\_modular\_double, 939
  - LAPACKPerm2MathPerm, 927
  - LeadingSubmatrixRankProfiles, 942
  - LUdivine\_modular\_double, 932
  - MathPerm2LAPACKPerm, 927
  - MatrixApplyS\_modular\_double, 927
  - MatrixApplyT\_modular\_double, 928
  - NullSpaceBasis\_modular\_double, 941
  - pColumnEchelonForm\_modular\_double, 935
  - pColumnEchelonForm\_modular\_float, 936
  - pColumnEchelonForm\_modular\_int32\_t, 937
  - PermApplyS\_double, 928
  - PermApplyT\_double, 928
  - PLUQ\_modular\_double, 931
  - PLUQtoEchelonPermutation, 945
  - pReducedColumnEchelonForm\_modular\_double, 935
  - pReducedColumnEchelonForm\_modular\_float, 936
  - pReducedColumnEchelonForm\_modular\_int32\_t, 937
  - pReducedRowEchelonForm\_modular\_double, 936
  - pReducedRowEchelonForm\_modular\_float, 937
  - pReducedRowEchelonForm\_modular\_int32\_t, 938
  - pRowEchelonForm\_modular\_double, 935
  - pRowEchelonForm\_modular\_float, 936
  - pRowEchelonForm\_modular\_int32\_t, 937
  - RandomNullSpaceVector\_modular\_double, 940
  - Rank\_modular\_double, 939
  - RankProfileFromLU, 941
  - ReducedColumnEchelonForm\_modular\_double, 932
  - ReducedColumnEchelonForm\_modular\_float, 933
  - ReducedColumnEchelonForm\_modular\_int32\_t, 934
  - ReducedRowEchelonForm\_modular\_double, 933
  - ReducedRowEchelonForm\_modular\_float, 934
  - ReducedRowEchelonForm\_modular\_int32\_t, 935
  - RowEchelonForm\_modular\_double, 932
  - RowEchelonForm\_modular\_float, 933
  - RowEchelonForm\_modular\_int32\_t, 934
  - RowRankProfile\_modular\_double, 941
  - RowRankProfileSubmatrix\_modular\_double, 942



- RowRankProfileSubmatrixIndices\_modular\_double, 942
- Solve\_modular\_double, 940
- solveLB2\_modular\_double, 940
- solveLB\_modular\_double, 940
- SpecRankProfile\_modular\_double, 939
- trinv\_left\_modular\_double, 931
- ffpack.dox, 945
- ffpack.h, 945
  - \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD, 953
  - \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD, 953
- ffpack.inl, 954
  - \_\_FFLASFFPACK\_ffpack\_INL, 955
- FFPACK::Protected, 394
  - ArithProg, 398
  - CompressRows, 399
  - CompressRowsQA, 400
  - CompressRowsQK, 400
  - Danilevski, 398
  - DeCompressRows, 400
  - DeCompressRowsQA, 401
  - DeCompressRowsQK, 400
  - fgemv\_kgf, 397
  - GaussJordan, 396
  - Hybrid\_KGF\_LUK\_MinPoly, 399
  - KellerGehrig, 397
  - KGFast, 397
  - KGFast\_generalized, 397
  - LUdivine\_construct, 395, 401
  - LUKrylov, 397
  - LUKrylov\_KGFast, 398
  - MatVecMinPoly, 399
  - newD, 399
  - RandomKrylovPrecond, 398
  - updatedD, 399
- ffpack\_c.h, 955
  - applyP\_modular\_double, 962
  - ColRankProfileSubmatrix\_modular\_double, 974
  - ColRankProfileSubmatrixIndices\_modular\_double, 973
  - ColumnEchelonForm\_modular\_double, 965
  - ColumnEchelonForm\_modular\_float, 966
  - ColumnEchelonForm\_modular\_int32\_t, 966
  - ColumnRankProfile\_modular\_double, 972
  - composePermutationsLLL, 962
  - composePermutationsLLM, 961
  - composePermutationsMLM, 962
  - cyclic\_shift\_col\_modular\_double, 962
  - cyclic\_shift\_mathPerm, 962
  - cyclic\_shift\_row\_modular\_double, 962
  - Det\_modular\_double, 971
  - FFLAS\_C\_DIAG, 959
  - FFLAS\_C\_ORDER, 958
  - FFLAS\_C\_SIDE, 959
  - FFLAS\_C\_TRANSPOSE, 958
  - FFLAS\_C\_UPLO, 959
  - FflasColMajor, 958
  - FflasLeft, 959
  - FflasLower, 959
  - FflasNonUnit, 959
  - FflasNoTrans, 959
  - FflasRight, 959
  - FflasRowMajor, 958
  - FflasTrans, 959
  - FflasUnit, 959
  - FflasUpper, 959
  - FFPACK\_C\_CHARPOLY\_TAG, 960
  - FFPACK\_C\_LU\_TAG, 959
  - FFPACK\_C\_MINPOLY\_TAG, 960
  - FFPACK\_COMPILED, 958
  - FfpackArithProg, 960
  - FfpackDanilevski, 960
  - FfpackDense, 960
  - FfpackHybrid, 960
  - FfpackKG, 960
  - FfpackKGF, 960
  - FfpackKGFast, 960
  - FfpackKGFastG, 960
  - FfpackLUK, 960
  - FfpackSingular, 960
  - FfpackSlabRecursive, 959
  - FfpackTileRecursive, 959
  - fgesv\_modular\_double, 963
  - fgesvin\_modular\_double, 963
  - fgetrs\_modular\_double, 963
  - fgetrsin\_modular\_double, 963
  - fttrtri\_modular\_double, 964
  - fttrrm\_modular\_double, 964
  - getEchelonForm\_modular\_double, 974
  - getEchelonFormin\_modular\_double, 975
  - getEchelonTransform\_modular\_double, 975
  - getReducedEchelonForm\_modular\_double, 975
  - getReducedEchelonFormin\_modular\_double, 975
  - getReducedEchelonTransform\_modular\_double, 976
  - getTriangular\_modular\_double, 974
  - getTriangularin\_modular\_double, 974
  - Invert2\_modular\_double, 969
  - Invert\_modular\_double, 969
  - Invertin\_modular\_double, 969
  - IsSingular\_modular\_double, 970
  - KrylovElim\_modular\_double, 970
  - LAPACKPerm2MathPerm, 960
  - LeadingSubmatrixRankProfiles, 973
  - LUdivine\_gauss\_modular\_double, 965
  - LUdivine\_modular\_double, 965
  - LUdivine\_small\_modular\_double, 965
  - MathPerm2LAPACKPerm, 960
  - MatrixApplyS\_modular\_double, 960
  - MatrixApplyT\_modular\_double, 961
  - NullSpaceBasis\_modular\_double, 972
  - PermApplyS\_double, 961
  - PermApplyT\_double, 961
  - PLUQ\_modular\_double, 964
  - PLUQtoEchelonPermutation, 976

- RandomNullSpaceVector\_modular\_double, 971
- Rank\_modular\_double, 970
- RankProfileFromLU, 972
- ReducedColumnEchelonForm\_modular\_double, 967
- ReducedColumnEchelonForm\_modular\_float, 967
- ReducedColumnEchelonForm\_modular\_int32\_t, 968
- ReducedRowEchelonForm2\_modular\_double, 968
- ReducedRowEchelonForm\_modular\_double, 967
- ReducedRowEchelonForm\_modular\_float, 968
- ReducedRowEchelonForm\_modular\_int32\_t, 968
- REF\_modular\_double, 969
- RowEchelonForm\_modular\_double, 966
- RowEchelonForm\_modular\_float, 966
- RowEchelonForm\_modular\_int32\_t, 967
- RowRankProfile\_modular\_double, 972
- RowRankProfileSubmatrix\_modular\_double, 973
- RowRankProfileSubmatrixIndices\_modular\_double, 973
- Solve\_modular\_double, 971
- solveLB2\_modular\_double, 971
- solveLB\_modular\_double, 971
- SpecRankProfile\_modular\_double, 970
- trinv\_left\_modular\_double, 964
- FFPACK\_C\_CHARPOLY\_TAG
  - ffpack\_c.h, 960
- FFPACK\_C\_LU\_TAG
  - ffpack\_c.h, 959
- FFPACK\_C\_MINPOLY\_TAG
  - ffpack\_c.h, 960
- ffpack\_charpoly.inl, 976
  - \_\_FFLASFFPACK\_charpoly\_INL, 977
- ffpack\_charpoly\_danilevski.inl, 977
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL, 977
- ffpack\_charpoly\_kgfast.inl, 978
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL, 978
- ffpack\_charpoly\_kgfastgeneralized.inl, 978
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL, 979
- ffpack\_charpoly\_kglu.inl, 979
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL, 979
- ffpack\_charpoly\_mp.inl, 979
  - \_\_FFPACK\_charpoly\_mp\_INL, 980
- FFPACK\_COMPILED
  - ffpack\_c.h, 958
- ffpack\_det\_mp.inl, 980
  - \_\_FFPACK\_det\_mp\_INL, 980
- ffpack\_echelonforms.inl, 981
  - \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE, 982
  - \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL, 982
- ffpack\_fgesv.inl, 982
  - \_\_FFLASFFPACK\_ffpack\_fgesv\_INL, 982
- ffpack\_fgetrs.inl, 983
  - \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL, 983
- ffpack\_frobenius.inl, 983
- ffpack\_fsytrf.inl, 984
  - \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL, 985
- ffpack\_ftrssyr2k.inl, 985
  - \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL, 985
- ffpack\_ftrstr.inl, 986
  - \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL, 986
- ffpack\_ftrtr.inl, 986
  - \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL, 987
- ENABLE\_ALL\_CHECKINGS, 987
- ffpack\_inst.C, 987
  - \_\_FFPACK\_INST\_C, 987
  - FFLAS\_COMPILED, 987
  - FFLAS\_ELT, 988
  - FFLAS\_FIELD, 988
  - INST\_OR\_DECL, 987
- ffpack\_inst.h, 988
  - FFLAS\_COMPILED, 989
  - FFLAS\_ELT, 989
  - FFLAS\_FIELD, 989
  - INST\_OR\_DECL, 989
- ffpack\_inst\_implem.inl, 990
- ffpack\_invert.inl, 993
  - \_\_FFLASFFPACK\_ffpack\_invert\_INL, 993
- ffpack\_krylovelim.inl, 993
  - \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL, 993
- ffpack\_ludivine.inl, 994
  - \_\_FFLASFFPACK\_ffpack\_ludivine\_INL, 994
- ffpack\_ludivine\_mp.inl, 995
  - \_\_FFPACK\_ludivine\_mp\_INL, 995
- ffpack\_minpoly.inl, 995
  - \_\_FFLASFFPACK\_ffpack\_minpoly\_INL, 996
- ffpack\_permutation.inl, 996
  - \_\_FFLASFFPACK\_ffpack\_permutation\_INL, 998
  - FFLASFFPACK\_PERM\_BKSIZE, 998
- ffpack\_pluq.inl, 998
  - \_\_FFLASFFPACK\_ffpack\_pluq\_INL, 999
  - CROUT, 999
- ffpack\_pluq\_mp.inl, 999
  - \_\_FFPACK\_pluq\_mp\_INL, 1000
- ffpack\_ppluq.inl, 1000
  - \_\_FFLASFFPACK\_ffpack\_ppluq\_INL, 1000
  - \_\_FFLAS\_TRSM\_READONLY, 1000
  - PBASECASE\_K, 1000
- ffpack\_rankprofiles.inl, 1001
  - \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL, 1002
- FfpackArithProg
  - ffpack\_c.h, 960
- FfpackDanilevski
  - ffpack\_c.h, 960
- FfpackDense
  - ffpack\_c.h, 960
- FfpackHybrid
  - ffpack\_c.h, 960
- FfpackKG
  - ffpack\_c.h, 960
- FfpackKGF

- ffpack\_c.h, [960](#)
- FpackKGFast
  - ffpack\_c.h, [960](#)
- FpackKGFastG
  - ffpack\_c.h, [960](#)
- FpackLUK
  - ffpack\_c.h, [960](#)
- FpackSingular
  - ffpack\_c.h, [960](#)
- FpackSlabRecursive
  - ffpack\_c.h, [959](#)
- FpackTileRecursive
  - ffpack\_c.h, [959](#)
- fgemm
  - FFLAS, [93–100](#), [146](#), [182](#), [183](#)
- fgemm\_3\_modular\_double
  - fflas\_c.h, [854](#)
  - fflas\_lvl3.C, [909](#)
- fgemm\_classical.inl, [1002](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL, [1002](#)
- fgemm\_classical\_mp.inl, [1002](#)
  - \_\_FFPACK\_fgemm\_classical\_INL, [1004](#)
- fgemm\_convert
  - FFLAS::Protected, [220](#)
- fgemm\_winograd.inl, [1004](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL, [1005](#)
  - NEWWINO, [1005](#)
- fgemv
  - FFLAS, [102–108](#), [177](#), [190](#)
- fgemv\_2\_modular\_double
  - fflas\_c.h, [853](#)
  - fflas\_lvl2.C, [907](#)
- fgemv\_convert
  - FFLAS::Protected, [222](#)
- fgemv\_kgf
  - FFPACK::Protected, [397](#)
- fger
  - FFLAS, [108–112](#), [179](#)
- fger\_2\_modular\_double
  - fflas\_c.h, [853](#)
  - fflas\_lvl2.C, [907](#)
- fger\_convert
  - FFLAS::Protected, [223](#)
- fgesv
  - FFPACK, [313](#), [366](#)
- fgesv\_modular\_double
  - ffpack.C, [930](#)
  - ffpack\_c.h, [963](#)
- fgesvin\_modular\_double
  - ffpack.C, [930](#)
  - ffpack\_c.h, [963](#)
- fgetrs
  - FFPACK, [311](#), [312](#), [365](#), [366](#)
- fgetrs\_modular\_double
  - ffpack\_c.h, [963](#)
- fgetrsin\_modular\_double
  - ffpack.C, [930](#)
  - ffpack\_c.h, [963](#)
- fgetrsv\_modular\_double
  - ffpack.C, [930](#)
- fidentity
  - FFLAS, [138](#), [170](#)
- fidentity\_2\_modular\_double
  - fflas\_c.h, [850](#)
  - fflas\_lvl2.C, [904](#)
- Field
  - benchmark-fgemm-rns.C, [774](#)
  - benchmark-pluq.C, [788](#)
  - FieldSimd< \_Field >, [443](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [727](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [728](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [730](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [733](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [735](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [740](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [743](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [746](#)
  - test-compressQ.C, [1058](#)
- field
  - associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, [406](#)
  - associatedDelayedField< const Givaro::Modular< T, X > >, [406](#)
  - associatedDelayedField< const Givaro::ModularBalanced< T > >, [407](#)
  - associatedDelayedField< const Givaro::ZRing< T > >, [407](#)
  - associatedDelayedField< Field >, [405](#)
- field-traits.h, [1006](#)
- field.doxy, [1008](#)
- FieldMax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- FieldMin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- FieldSimd
  - FieldSimd< \_Field >, [443](#)
- FieldSimd< \_Field >, [442](#)
  - add, [444](#)
  - add\_r, [444](#)
  - addin, [444](#)
  - addin\_r, [445](#)
  - alignment, [448](#)
  - axpy, [446](#), [447](#)
  - axpy\_r, [447](#)
  - axpyin, [447](#)
  - axpyin\_r, [447](#)
  - Element, [443](#)
  - Field, [443](#)
  - FieldSimd, [443](#)

- init, [444](#)
- maxpy, [447](#)
- maxpyin, [448](#)
- mod, [446](#)
- mul, [446](#)
- mul\_r, [446](#)
- mulin, [446](#)
- operator=, [444](#)
- scalar\_t, [443](#)
- simd, [443](#)
- sub, [445](#)
- sub\_r, [445](#)
- subin, [445](#)
- subin\_r, [445](#)
- vect\_size, [448](#)
- vect\_t, [443](#)
- zero, [445](#), [446](#)
- FieldTraits< FFPACK::RNSInteger< T > >, [449](#)
  - balanced, [449](#)
  - category, [449](#)
- FieldTraits< FFPACK::RNSIntegerMod< T > >, [449](#)
  - balanced, [450](#)
  - category, [449](#)
- FieldTraits< Field >, [448](#)
  - balanced, [448](#)
  - category, [448](#)
- FieldTraits< Givaro::Modular< Element > >, [450](#)
  - balanced, [450](#)
  - category, [450](#)
- FieldTraits< Givaro::ModularBalanced< Element > >, [450](#)
  - balanced, [451](#)
  - category, [451](#)
- FieldTraits< Givaro::ZRing< double > >, [451](#)
  - balanced, [451](#)
  - category, [451](#)
- FieldTraits< Givaro::ZRing< float > >, [451](#)
  - balanced, [452](#)
  - category, [452](#)
- FieldTraits< Givaro::ZRing< Givaro::Integer > >, [452](#)
  - balanced, [452](#)
  - category, [452](#)
- FieldTraits< Givaro::ZRing< int16\_t > >, [453](#)
  - balanced, [453](#)
  - category, [453](#)
- FieldTraits< Givaro::ZRing< int32\_t > >, [453](#)
  - balanced, [453](#)
  - category, [453](#)
- FieldTraits< Givaro::ZRing< int64\_t > >, [454](#)
  - balanced, [454](#)
  - category, [454](#)
- FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > >, [454](#)
  - balanced, [455](#)
  - category, [454](#)
- FieldTraits< Givaro::ZRing< uint16\_t > >, [455](#)
  - balanced, [455](#)
  - category, [455](#)
- FieldTraits< Givaro::ZRing< uint32\_t > >, [455](#)
  - balanced, [456](#)
  - category, [455](#)
- FieldTraits< Givaro::ZRing< uint64\_t > >, [456](#)
  - balanced, [456](#)
  - category, [456](#)
- fill\_value
  - benchmark-fgemv.C, [778](#)
- findArgument
  - args-parser.h, [762](#)
- finit
  - FFLAS, [114](#), [116](#), [131](#), [139](#), [160](#), [171](#)
- finit\_rns
  - FFLAS, [157](#), [159](#)
- finit\_trans\_rns
  - FFLAS, [158](#)
- first\_component
  - Compose< H1, H2 >, [420](#)
- firstBlockSize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [458](#)
- fiszero
  - FFLAS, [134](#), [138](#), [163](#), [169](#)
- fiszero\_1\_modular\_double
  - fflas\_c.h, [847](#)
  - fflas\_lvl1.C, [899](#)
- fiszero\_2\_modular\_double
  - fflas\_c.h, [850](#)
  - fflas\_lvl2.C, [904](#)
- Fixed, [456](#)
- FixedPreclntTag, [456](#)
- flimits.h, [1008](#)
  - in\_range, [1009](#)
- FLOAT\_MOD
  - fflas\_simd.h, [915](#)
- Floats
  - benchmark-dgemm.C, [767](#)
- floor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [556](#)
  - Simd256\_impl< true, false, true, 8 >, [627](#)
  - Simd512\_impl< true, false, true, 8 >, [703](#)
- fmadd
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [557](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [561](#)
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [580](#)
  - Simd128\_impl< true, true, false, 8 >, [589](#)
  - Simd128\_impl< true, true, true, 2 >, [597](#)
  - Simd128\_impl< true, true, true, 4 >, [605](#)
  - Simd128\_impl< true, true, true, 8 >, [614](#)
  - Simd256\_impl< true, false, true, 8 >, [625](#)
  - Simd256\_impl< true, true, false, 2 >, [635](#)
  - Simd256\_impl< true, true, false, 4 >, [649](#)
  - Simd256\_impl< true, true, false, 8 >, [660](#)

[illegible]

- Simd128\_impl< true, true, false, 8 >, [587](#)
- Simd128\_impl< true, true, true, 2 >, [598](#)
- Simd128\_impl< true, true, true, 4 >, [607](#)
- Simd128\_impl< true, true, true, 8 >, [615](#)
- Simd256\_impl< true, true, false, 2 >, [632](#)
- Simd256\_impl< true, true, false, 4 >, [642](#), [645](#)
- Simd256\_impl< true, true, false, 8 >, [657](#)
- Simd256\_impl< true, true, true, 2 >, [669](#)
- Simd256\_impl< true, true, true, 4 >, [679](#), [684](#)
- Simd256\_impl< true, true, true, 8 >, [693](#)
- Simd512\_impl< true, true, false, 8 >, [708](#)
- Simd512\_impl< true, true, true, 8 >, [720](#)
- fmsubxin
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [570](#)
  - Simd128\_impl< true, true, false, 4 >, [578](#)
  - Simd128\_impl< true, true, false, 8 >, [587](#), [590](#)
  - Simd128\_impl< true, true, true, 2 >, [598](#)
  - Simd128\_impl< true, true, true, 4 >, [607](#)
  - Simd128\_impl< true, true, true, 8 >, [615](#)
  - Simd256\_impl< true, true, false, 2 >, [632](#)
  - Simd256\_impl< true, true, false, 4 >, [642](#), [645](#)
  - Simd256\_impl< true, true, false, 8 >, [657](#)
  - Simd256\_impl< true, true, true, 2 >, [669](#)
  - Simd256\_impl< true, true, true, 4 >, [679](#), [684](#)
  - Simd256\_impl< true, true, true, 8 >, [693](#)
  - Simd512\_impl< true, true, false, 8 >, [708](#)
  - Simd512\_impl< true, true, true, 8 >, [720](#)
- fneg
  - FFLAS, [133](#), [140](#), [162](#), [172](#)
- fneg\_1\_modular\_double
  - fflas\_c.h, [846](#)
  - fflas\_lvl1.C, [899](#)
- fneg\_2\_modular\_double
  - fflas\_c.h, [851](#)
  - fflas\_lvl2.C, [905](#)
- fnegin
  - FFLAS, [132](#), [140](#), [161](#), [172](#)
- fnegin\_1\_modular\_double
  - fflas\_c.h, [846](#)
  - fflas\_lvl1.C, [899](#)
- fnegin\_2\_modular\_double
  - fflas\_c.h, [851](#)
  - fflas\_lvl2.C, [904](#)
- fnmadd
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [557](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [580](#)
  - Simd128\_impl< true, true, false, 8 >, [589](#)
  - Simd128\_impl< true, true, true, 2 >, [597](#)
  - Simd128\_impl< true, true, true, 4 >, [606](#)
  - Simd128\_impl< true, true, true, 8 >, [614](#)
  - Simd256\_impl< true, false, true, 8 >, [625](#)
- Simd256\_impl< true, true, false, 2 >, [635](#)
- Simd256\_impl< true, true, false, 4 >, [650](#)
- Simd256\_impl< true, true, false, 8 >, [660](#)
- Simd256\_impl< true, true, true, 2 >, [668](#)
- Simd256\_impl< true, true, true, 4 >, [678](#), [683](#)
- Simd256\_impl< true, true, true, 8 >, [692](#)
- Simd512\_impl< true, false, true, 8 >, [702](#)
- Simd512\_impl< true, true, false, 8 >, [711](#)
- Simd512\_impl< true, true, true, 8 >, [719](#)
- fnmaddin
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [557](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [581](#)
  - Simd128\_impl< true, true, false, 8 >, [589](#)
  - Simd128\_impl< true, true, true, 2 >, [597](#)
  - Simd128\_impl< true, true, true, 4 >, [606](#)
  - Simd128\_impl< true, true, true, 8 >, [615](#)
  - Simd256\_impl< true, false, true, 8 >, [625](#)
  - Simd256\_impl< true, true, false, 2 >, [635](#)
  - Simd256\_impl< true, true, false, 4 >, [650](#)
  - Simd256\_impl< true, true, false, 8 >, [660](#)
  - Simd256\_impl< true, true, true, 2 >, [668](#)
  - Simd256\_impl< true, true, true, 4 >, [678](#), [683](#)
  - Simd256\_impl< true, true, true, 8 >, [693](#)
  - Simd512\_impl< true, false, true, 8 >, [702](#)
  - Simd512\_impl< true, true, false, 8 >, [711](#)
  - Simd512\_impl< true, true, true, 8 >, [720](#)
- fnmaddx
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [570](#)
  - Simd128\_impl< true, true, false, 4 >, [578](#)
  - Simd128\_impl< true, true, false, 8 >, [587](#)
  - Simd128\_impl< true, true, true, 2 >, [598](#)
  - Simd128\_impl< true, true, true, 4 >, [606](#)
  - Simd128\_impl< true, true, true, 8 >, [615](#)
  - Simd256\_impl< true, true, false, 2 >, [632](#)
  - Simd256\_impl< true, true, false, 4 >, [642](#), [645](#)
  - Simd256\_impl< true, true, false, 8 >, [657](#)
  - Simd256\_impl< true, true, true, 2 >, [668](#)
  - Simd256\_impl< true, true, true, 4 >, [678](#), [684](#)
  - Simd256\_impl< true, true, true, 8 >, [693](#)
  - Simd512\_impl< true, true, false, 8 >, [708](#)
  - Simd512\_impl< true, true, true, 8 >, [720](#)
- fnmaddxin
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [570](#)
  - Simd128\_impl< true, true, false, 4 >, [578](#)
  - Simd128\_impl< true, true, false, 8 >, [587](#)
  - Simd128\_impl< true, true, true, 2 >, [598](#)
  - Simd128\_impl< true, true, true, 4 >, [606](#)
  - Simd128\_impl< true, true, true, 8 >, [615](#)
  - Simd256\_impl< true, true, false, 2 >, [632](#)



- Simd256\_impl< true, true, false, 4 >, [642](#), [645](#)
- Simd256\_impl< true, true, false, 8 >, [657](#)
- Simd256\_impl< true, true, true, 2 >, [668](#)
- Simd256\_impl< true, true, true, 4 >, [678](#), [684](#)
- Simd256\_impl< true, true, true, 8 >, [693](#)
- Simd512\_impl< true, true, false, 8 >, [708](#)
- Simd512\_impl< true, true, true, 8 >, [720](#)
- FOR1D
  - parallel.h, [1027](#)
- FOR2D
  - parallel.h, [1028](#)
- FORBLOCK1D
  - parallel.h, [1027](#)
- FORBLOCK2D
  - parallel.h, [1028](#)
- ForceCheck\_charpoly
  - FFPACK, [308](#)
- ForceCheck\_Det
  - FFPACK, [308](#)
- ForceCheck\_fgemm
  - FFLAS, [74](#)
- ForceCheck\_ftsrn
  - FFLAS, [74](#)
- ForceCheck\_invert
  - FFPACK, [308](#)
- ForceCheck\_PLUQ
  - FFPACK, [308](#)
- ForStrategy1D
  - ForStrategy1D< blocksize\_t, Cut, Param >, [457](#)
- ForStrategy1D< blocksize\_t, Cut, Param >, [457](#)
  - begin, [458](#)
  - blockindex, [458](#)
  - build, [457](#)
  - changeBS, [459](#)
  - current, [458](#)
  - end, [458](#)
  - firstBlockSize, [458](#)
  - ForStrategy1D, [457](#)
  - ibeg, [458](#)
  - iend, [458](#)
  - initialize, [458](#)
  - isTerminated, [458](#)
  - lastBlockSize, [459](#)
  - numBlock, [459](#)
  - numblocks, [458](#)
  - operator++, [458](#)
- ForStrategy2D
  - ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- ForStrategy2D< blocksize\_t, Cut, Param >, [459](#)
  - \_ibeg, [461](#)
  - \_iend, [461](#)
  - \_jbeg, [461](#)
  - \_jend, [461](#)
  - blockindex, [461](#)
  - BLOCKS, [462](#)
  - changeCBS, [462](#)
  - changeRBS, [462](#)
  - colblockindex, [461](#)
  - colBlockSize, [461](#)
  - colnumblocks, [460](#)
  - current, [462](#)
  - ForStrategy2D, [460](#)
  - ibegin, [460](#)
  - iend, [460](#)
  - initialize, [460](#)
  - isTerminated, [460](#)
  - jbegin, [460](#)
  - jend, [460](#)
  - lastCBS, [462](#)
  - lastRBS, [462](#)
  - numColBlock, [462](#)
  - numRowBlock, [462](#)
  - operator<<, [461](#)
  - operator++, [460](#)
  - rowblockindex, [461](#)
  - rowBlockSize, [461](#)
  - rownumblocks, [460](#)
- frand
  - FFLAS, [134](#), [137](#)
- freduce
  - FFLAS, [112–115](#), [117](#), [159](#), [160](#), [170](#), [171](#)
  - FFLAS::details, [206](#)
- freduce\_1\_modular\_double
  - fflas\_c.h, [846](#)
  - fflas\_lvl1.C, [899](#)
- freduce\_2\_modular\_double
  - fflas\_c.h, [850](#)
  - fflas\_lvl2.C, [904](#)
- freduce\_constoverride
  - FFLAS, [113](#), [115](#)
- freducein\_1\_modular\_double
  - fflas\_c.h, [846](#)
  - fflas\_lvl1.C, [898](#)
- freducein\_2\_modular\_double
  - fflas\_c.h, [850](#)
  - fflas\_lvl2.C, [904](#)
- freivalds
  - FFLAS, [117](#)
- fscal
  - FFLAS, [118–122](#), [164](#), [173](#)
  - FFLAS::details, [207](#)
- fscal\_1\_modular\_double
  - fflas\_c.h, [847](#)
  - fflas\_lvl1.C, [900](#)
- fscal\_2\_modular\_double
  - fflas\_c.h, [851](#)
  - fflas\_lvl2.C, [905](#)
- fscalin
  - FFLAS, [118–122](#), [164](#), [173](#)
  - FFLAS::details, [207](#)
- fscalin\_1\_modular\_double
  - fflas\_c.h, [847](#)
  - fflas\_lvl1.C, [900](#)
- fscalin\_2\_modular\_double
  - fflas\_c.h, [851](#)
  - fflas\_lvl2.C, [905](#)

- fspmm
  - FFLAS, [156](#)
  - FFLAS::sparse\_details, [233–236](#)
  - FFLAS::sparse\_details\_impl, [251](#), [258](#), [259](#), [265](#), [270](#), [279](#), [280](#)
- fspmm\_dispatch
  - FFLAS::sparse\_details, [233](#)
- fspmm\_mone
  - FFLAS::sparse\_details\_impl, [252](#), [260](#), [271](#)
- fspmm\_mone\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [253](#), [261](#), [272](#)
- fspmm\_mone\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [253](#), [261](#), [272](#)
- fspmm\_one
  - FFLAS::sparse\_details\_impl, [252](#), [260](#), [271](#)
- fspmm\_one\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [252](#), [260](#), [272](#)
- fspmm\_one\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [253](#), [260](#), [272](#)
- fspmm\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [251](#), [259](#)
- fspmm\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [252](#), [259](#)
- fspmv
  - FFLAS, [154](#), [156](#)
  - FFLAS::sparse\_details, [230–232](#), [241](#), [242](#)
  - FFLAS::sparse\_details\_impl, [253](#), [254](#), [261](#), [262](#), [266](#), [273](#), [276](#), [280–283](#)
- fspmv\_dispatch
  - FFLAS::sparse\_details, [230](#)
- fspmv\_mone
  - FFLAS::sparse\_details\_impl, [254](#), [255](#), [262](#), [274](#), [277](#), [283](#), [284](#)
- fspmv\_mone\_simd
  - FFLAS::sparse\_details\_impl, [278](#), [284](#)
- fspmv\_one
  - FFLAS::sparse\_details\_impl, [254](#), [262](#), [273](#), [274](#), [277](#), [283](#), [284](#)
- fspmv\_one\_simd
  - FFLAS::sparse\_details\_impl, [277](#), [284](#)
- fspmv\_simd
  - FFLAS::sparse\_details\_impl, [276](#), [282](#), [283](#)
- fsquare
  - FFLAS, [100–102](#), [184](#)
- fsquare\_3\_modular\_double
  - fflas\_c.h, [855](#)
  - fflas\_lvl3.C, [909](#)
- fsquareCommon
  - FFLAS::Protected, [222](#)
- fsub
  - FFLAS, [81](#), [85](#), [167](#), [175](#)
- fsub\_1\_modular\_double
  - fflas\_c.h, [849](#)
  - fflas\_lvl1.C, [901](#)
- fsub\_2\_modular\_double
  - fflas\_c.h, [852](#)
  - fflas\_lvl2.C, [906](#)
- fsubin
  - FFLAS, [81](#), [86](#), [176](#)
- fsubin\_1\_modular\_double
  - fflas\_c.h, [849](#)
  - fflas\_lvl1.C, [902](#)
- fsubin\_2\_modular\_double
  - fflas\_c.h, [852](#)
  - fflas\_lvl2.C, [906](#)
- fswap
  - FFLAS, [136](#), [166](#)
- fswap\_1\_modular\_double
  - fflas\_c.h, [848](#)
  - fflas\_lvl1.C, [901](#)
- fsyr2k
  - FFLAS, [123](#)
- fsyrk
  - FFLAS, [124–126](#)
- fsyrk.C, [1009](#)
  - CUBE, [1009](#)
  - GFOPS, [1009](#)
  - main, [1010](#)
  - TTimer, [1010](#)
- fsytrf
  - FFPACK, [316](#), [317](#)
- fsytrf.C, [1010](#)
  - CUBE, [1010](#)
  - GFOPS, [1010](#)
  - main, [1011](#)
  - TTimer, [1011](#)
- fsytrf\_BC\_Crout
  - FFPACK, [351](#)
- fsytrf\_BC\_RL
  - FFPACK, [351](#)
- fsytrf\_LOW\_RPM\_BC\_Crout
  - FFPACK, [352](#)
- fsytrf\_nonunit
  - FFPACK, [317](#), [353](#)
- fsytrf\_RPM
  - FFPACK, [353](#)
- fsytrf\_UP\_RPM
  - FFPACK, [352](#)
- fsytrf\_UP\_RPM\_BC\_Crout
  - FFPACK, [352](#)
- fsytrf\_UP\_RPM\_BC\_RL
  - FFPACK, [352](#)
- ftmm
  - FFLAS, [127](#), [128](#), [181](#)
- ftmm\_3\_modular\_double
  - fflas\_c.h, [854](#)
  - fflas\_lvl3.C, [909](#)
- ftmmLeftLowerNoTransNonUnit< Element >, [462](#)
- ftmmLeftLowerNoTransUnit< Element >, [462](#)
- ftmmLeftLowerTransNonUnit< Element >, [463](#)
- ftmmLeftLowerTransUnit< Element >, [463](#)
- ftmmLeftUpperNoTransNonUnit< Element >, [463](#)
- ftmmLeftUpperNoTransUnit< Element >, [463](#)
- ftmmLeftUpperTransNonUnit< Element >, [463](#)
- ftmmLeftUpperTransUnit< Element >, [463](#)
- ftmmRightLowerNoTransNonUnit< Element >, [463](#)



- ftmmRightLowerNoTransUnit< Element >, [463](#)
- ftmmRightLowerTransNonUnit< Element >, [464](#)
- ftmmRightLowerTransUnit< Element >, [464](#)
- ftmmRightUpperNoTransNonUnit< Element >, [464](#)
- ftmmRightUpperNoTransUnit< Element >, [464](#)
- ftmmRightUpperTransNonUnit< Element >, [464](#)
- ftmmRightUpperTransUnit< Element >, [464](#)
- ftmv
  - FFLAS, [143](#)
- ftsm
  - FFLAS, [129](#), [130](#), [143](#), [146](#), [147](#), [180](#)
- ftsm\_3\_modular\_double
  - fflas\_c.h, [854](#)
  - fflas\_lvl3.C, [908](#)
- ftsmLeftLowerNoTransNonUnit< Element >, [464](#)
- ftsmLeftLowerNoTransUnit< Element >, [464](#)
- ftsmLeftLowerTransNonUnit< Element >, [465](#)
- ftsmLeftLowerTransUnit< Element >, [465](#)
- ftsmLeftUpperNoTransNonUnit< Element >, [465](#)
- ftsmLeftUpperNoTransUnit< Element >, [465](#)
- ftsmLeftUpperTransNonUnit< Element >, [465](#)
- ftsmLeftUpperTransUnit< Element >, [465](#)
- ftsmRightLowerNoTransNonUnit< Element >, [466](#)
- ftsmRightLowerNoTransUnit< Element >, [466](#)
- ftsmRightLowerTransNonUnit< Element >, [466](#)
- ftsmRightLowerTransUnit< Element >, [466](#)
- ftsmRightUpperNoTransNonUnit< Element >, [466](#)
- ftsmRightUpperNoTransUnit< Element >, [466](#)
- ftsmRightUpperTransNonUnit< Element >, [466](#)
- ftsmRightUpperTransUnit< Element >, [466](#)
- ftssyr2k
  - FFPACK, [316](#)
- ftstr
  - FFPACK, [315](#)
- ftsv
  - FFLAS, [130](#), [179](#)
- ftsv\_2\_modular\_double
  - fflas\_c.h, [853](#)
  - fflas\_lvl2.C, [907](#)
- fttri
  - FFPACK, [314](#), [367](#)
- fttri.C, [1011](#)
  - CUBE, [1011](#)
  - GFOPS, [1011](#)
  - main, [1012](#)
  - TTimer, [1012](#)
- fttri\_modular\_double
  - ffpack.C, [931](#)
  - ffpack\_c.h, [964](#)
- fttrm
  - FFPACK, [315](#), [367](#)
- fttrm\_modular\_double
  - ffpack.C, [931](#)
  - ffpack\_c.h, [964](#)
- fzero
  - FFLAS, [133](#), [136](#), [162](#), [168](#)
- fzero\_1\_modular\_double
  - fflas\_c.h, [847](#)
- fflas\_lvl1.C, [899](#)
- fzero\_2\_modular\_double
  - fflas\_c.h, [849](#)
  - fflas\_lvl2.C, [903](#)
- gather
  - Simd128\_impl< true, true, false, 2 >, [568](#)
  - Simd128\_impl< true, true, false, 4 >, [576](#)
  - Simd128\_impl< true, true, false, 8 >, [585](#)
  - Simd128\_impl< true, true, true, 2 >, [594](#)
  - Simd128\_impl< true, true, true, 4 >, [603](#)
  - Simd128\_impl< true, true, true, 8 >, [612](#)
  - Simd256\_impl< true, false, true, 8 >, [623](#)
  - Simd256\_impl< true, true, false, 2 >, [630](#)
  - Simd256\_impl< true, true, false, 4 >, [640](#), [643](#)
  - Simd256\_impl< true, true, false, 8 >, [655](#)
  - Simd256\_impl< true, true, true, 2 >, [664](#)
  - Simd256\_impl< true, true, true, 4 >, [674](#), [681](#)
  - Simd256\_impl< true, true, true, 8 >, [689](#)
  - Simd512\_impl< true, false, true, 8 >, [699](#)
  - Simd512\_impl< true, true, false, 8 >, [706](#)
  - Simd512\_impl< true, true, true, 8 >, [716](#)
- GaussJordan
  - FFPACK::Protected, [396](#)
- GCC\_VERSION
  - fflas-ffpack-config.h, [837](#)
- gebp
  - FFLAS::details, [210](#)
- genData
  - benchmark-fgemv.C, [778](#)
- generate\_random\_vector
  - test-simd.C, [1106](#)
- GenericTag, [467](#)
- get
  - Simd128\_impl< true, true, false, 8 >, [588](#)
  - Simd128\_impl< true, true, true, 8 >, [612](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 8 >, [689](#)
- getDataType
  - FFLAS, [153](#), [154](#)
- getEchelonForm
  - FFPACK, [343](#), [344](#)
- getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [377](#)
- getEchelonForm\_modular\_double
  - ffpack.C, [943](#)
  - ffpack\_c.h, [974](#)
- getEchelonFormin\_modular\_double
  - ffpack.C, [944](#)
  - ffpack\_c.h, [975](#)
- getEchelonTransform
  - FFPACK, [345](#)
- getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [377](#)
- getEchelonTransform\_modular\_double
  - ffpack.C, [944](#)
  - ffpack\_c.h, [975](#)
- getListArgs

- args-parser.h, [762](#)
- getReducedEchelonForm
  - FFPACK, [345](#), [346](#)
- getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [378](#)
- getReducedEchelonForm\_modular\_double
  - ffpack.C, [944](#)
  - ffpack\_c.h, [975](#)
- getReducedEchelonFormin\_modular\_double
  - ffpack.C, [944](#)
  - ffpack\_c.h, [975](#)
- getReducedEchelonTransform
  - FFPACK, [347](#)
- getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [378](#)
- getReducedEchelonTransform\_modular\_double
  - ffpack.C, [945](#)
  - ffpack\_c.h, [976](#)
- getSeed
  - FFLAS, [195](#)
- getStat
  - FFLAS, [156](#)
- getTLBSize
  - FFLAS, [194](#)
- getTriangular
  - FFPACK, [342](#), [343](#)
- getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [376](#)
- getTriangular\_modular\_double
  - ffpack.C, [943](#)
  - ffpack\_c.h, [974](#)
- getTriangularin\_modular\_double
  - ffpack.C, [943](#)
  - ffpack\_c.h, [974](#)
- getTridiagonal
  - FFPACK, [354](#)
- gf2ModularBalanced
  - regression-check.C, [1036](#)
- GFOPS
  - arithprog.C, [763](#)
  - autotune/charpoly.C, [793](#)
  - autotune/pluq.C, [1033](#)
  - fsyrk.C, [1009](#)
  - fsytrf.C, [1010](#)
  - fttrtri.C, [1011](#)
  - winograd.C, [1110](#)
- Givaro, [401](#)
- gmp.C, [1012](#)
  - main, [1012](#)
- GRAIN
  - benchmark-fgemm-rns.C, [775](#)
- Grain, [467](#)
- greater
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
- ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [563](#)
- Simd128\_impl< true, true, false, 2 >, [569](#)
- Simd128\_impl< true, true, false, 4 >, [577](#)
- Simd128\_impl< true, true, false, 8 >, [586](#)
- Simd128\_impl< true, true, true, 2 >, [598](#)
- Simd128\_impl< true, true, true, 4 >, [607](#)
- Simd128\_impl< true, true, true, 8 >, [616](#)
- Simd256\_impl< true, false, true, 8 >, [626](#)
- Simd256\_impl< true, true, false, 2 >, [631](#)
- Simd256\_impl< true, true, false, 4 >, [641](#), [644](#)
- Simd256\_impl< true, true, false, 8 >, [656](#)
- Simd256\_impl< true, true, true, 2 >, [669](#)
- Simd256\_impl< true, true, true, 4 >, [679](#), [685](#)
- Simd256\_impl< true, true, true, 8 >, [694](#)
- Simd512\_impl< true, false, true, 8 >, [702](#)
- Simd512\_impl< true, true, false, 8 >, [707](#)
- Simd512\_impl< true, true, true, 8 >, [721](#)
- greater\_eq
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
- ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [563](#)
- Simd128\_impl< true, true, false, 2 >, [569](#)
- Simd128\_impl< true, true, false, 4 >, [577](#)
- Simd128\_impl< true, true, false, 8 >, [586](#)
- Simd128\_impl< true, true, true, 2 >, [599](#)
- Simd128\_impl< true, true, true, 4 >, [607](#)
- Simd128\_impl< true, true, true, 8 >, [616](#)
- Simd256\_impl< true, false, true, 8 >, [626](#)
- Simd256\_impl< true, true, false, 2 >, [631](#)
- Simd256\_impl< true, true, false, 4 >, [641](#), [644](#)
- Simd256\_impl< true, true, false, 8 >, [656](#)
- Simd256\_impl< true, true, true, 2 >, [669](#)
- Simd256\_impl< true, true, true, 4 >, [679](#), [685](#)
- Simd256\_impl< true, true, true, 8 >, [694](#)
- Simd512\_impl< true, false, true, 8 >, [703](#)
- Simd512\_impl< true, true, false, 8 >, [707](#)
- Simd512\_impl< true, true, true, 8 >, [721](#)
- hadd
  - Simd256\_impl< true, false, true, 8 >, [627](#)
  - Simd512\_impl< true, false, true, 8 >, [703](#)
- hadd\_to\_scal
  - Simd128\_impl< true, true, false, 2 >, [570](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [587](#)
  - Simd128\_impl< true, true, true, 2 >, [599](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [616](#)
  - Simd256\_impl< true, false, true, 8 >, [627](#)
  - Simd256\_impl< true, true, false, 2 >, [632](#)
  - Simd256\_impl< true, true, false, 4 >, [642](#), [645](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [669](#)
  - Simd256\_impl< true, true, true, 4 >, [680](#), [685](#)
  - Simd256\_impl< true, true, true, 8 >, [694](#)
  - Simd512\_impl< true, false, true, 8 >, [703](#)

- Simd512\_impl< true, true, false, 8 >, [709](#)
- Simd512\_impl< true, true, true, 8 >, [721](#)
- half\_t
  - Simd256\_impl< true, true, false, 2 >, [630](#)
  - Simd256\_impl< true, true, false, 4 >, [640](#)
  - Simd256\_impl< true, true, false, 8 >, [655](#)
  - Simd256\_impl< true, true, true, 2 >, [663](#)
  - Simd256\_impl< true, true, true, 4 >, [673](#), [674](#)
  - Simd256\_impl< true, true, true, 8 >, [688](#)
  - Simd512\_impl< true, true, false, 8 >, [706](#)
  - Simd512\_impl< true, true, true, 8 >, [715](#)
- has\_equal
  - FFLAS, [76](#)
- has\_minus
  - FFLAS, [75](#)
- has\_minus\_eq
  - FFLAS, [76](#)
- has\_minus\_eq\_impl< C >, [467](#)
  - value, [467](#)
- has\_minus\_impl< C >, [467](#)
  - value, [468](#)
- has\_mul
  - FFLAS, [76](#)
- has\_mul\_eq
  - FFLAS, [76](#)
- has\_mul\_eq\_impl< C >, [468](#)
  - value, [468](#)
- has\_mul\_impl< C >, [468](#)
  - value, [468](#)
- has\_operation< T >, [468](#)
  - value, [469](#)
- has\_plus
  - FFLAS, [75](#)
- has\_plus\_eq
  - FFLAS, [76](#)
- has\_plus\_eq\_impl< C >, [469](#)
  - value, [469](#)
- has\_plus\_impl< C >, [469](#)
  - value, [469](#)
- hasAVX512ER
  - instrset.h, [1021](#)
  - instrset\_detect.cpp, [1022](#)
- hasF16C
  - instrset\_detect.cpp, [1022](#)
- hasFMA3
  - instrset.h, [1021](#)
  - instrset\_detect.cpp, [1022](#)
- hasFMA4
  - instrset.h, [1021](#)
  - instrset\_detect.cpp, [1022](#)
- hasXOP
  - instrset.h, [1021](#)
  - instrset\_detect.cpp, [1022](#)
- HAVE\_BLAS
  - config.h, [807](#)
- HAVE\_CBLAS
  - config.h, [808](#)
- HAVE\_CXX11
  - config.h, [808](#)
- HAVE\_DLFCN\_H
  - config.h, [808](#)
- HAVE\_FLOAT\_H
  - config.h, [808](#)
- HAVE\_INT128
  - config.h, [808](#)
- HAVE\_INTPTR\_T
  - config.h, [808](#)
- HAVE\_INTTYPES\_H
  - config.h, [808](#)
- HAVE\_LAPACK
  - config.h, [808](#)
- HAVE\_LIMITS\_H
  - config.h, [808](#)
- HAVE\_LITTLE\_ENDIAN
  - config.h, [808](#)
- HAVE\_PTHREAD\_H
  - config.h, [808](#)
- HAVE\_STDDEF\_H
  - config.h, [808](#)
- HAVE\_STDINT\_H
  - config.h, [808](#)
- HAVE\_STDIO\_H
  - config.h, [809](#)
- HAVE\_STDLIB\_H
  - config.h, [809](#)
- HAVE\_STRING\_H
  - config.h, [809](#)
- HAVE\_STRINGS\_H
  - config.h, [809](#)
- HAVE\_SYS\_STAT\_H
  - config.h, [809](#)
- HAVE\_SYS\_TIME\_H
  - config.h, [809](#)
- HAVE\_SYS\_TYPES\_H
  - config.h, [809](#)
- HAVE\_UNISTD\_H
  - config.h, [809](#)
- HelperFlag, [469](#)
  - aut, [470](#)
  - coo, [470](#)
  - csr, [470](#)
  - ell, [470](#)
  - none, [470](#)
  - pm1, [470](#)
- HelperMod
  - HelperMod< Field, ElementCategories::MachineIntTag >, [471](#)
  - HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, [472](#)
  - HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, [472](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [473](#)
  - HelperMod< Field, ElementCategories::MachineIntTag >, [470](#)
  - HelperMod, [471](#)
  - invp, [471](#)
  - max, [471](#)

- min, [471](#)
- p, [471](#)
- pow50rem, [471](#)
- HelperMod< Field, ElementTraits >, [470](#)
- HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecisionFlag >, [472](#)
  - HelperMod, [472](#)
  - p, [472](#)
- HelperMod< Field, FFLAS::ElementCategories::FixedPrecisionFlag >, [472](#)
  - HelperMod, [472](#)
  - p, [473](#)
- HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [473](#)
  - HelperMod, [473](#)
  - invp, [473](#)
  - max, [474](#)
  - min, [473](#)
  - p, [473](#)
- helpString
  - Argument, [405](#)
- HYB\_ZO
  - FFLAS, [79](#)
- hyb\_zo.h, [1012](#)
- hyb\_zo\_pspmm.inl, [1013](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL, [1013](#)
- hyb\_zo\_pspmv.inl, [1013](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL, [1013](#)
- hyb\_zo\_spm.inl, [1014](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spm\_INL, [1014](#)
- hyb\_zo\_spmv.inl, [1014](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL, [1015](#)
- hyb\_zo\_utils.inl, [1015](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL, [1015](#)
- Hybrid, [474](#)
- Hybrid\_KGF\_LUK\_MinPoly
  - FFPACK::Protected, [399](#)
- ibeg
  - ForStrategy1D< blocksize\_t, Cut, Param >, [458](#)
- ibegin
  - ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- idamax\_
  - config-blas.h, [803](#)
- iend
  - ForStrategy1D< blocksize\_t, Cut, Param >, [458](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- igebb11
  - FFLAS::details, [209](#)
- igebb14
  - FFLAS::details, [208](#)
- igebb21
  - FFLAS::details, [209](#)
- igebb24
  - FFLAS::details, [208](#)
- igebb41
  - FFLAS::details, [208](#)
- igebb44
  - FFLAS::details, [208](#)
- igebp
  - FFLAS::details, [209](#)
- igemm
  - FFLAS::Protected, [225](#)
  - igemm.doxy, [1015](#)
  - igemm.h, [1015](#)
  - igemm.inl, [1016](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL, [1016](#)
- igemm\_
  - FFLAS, [131](#)
- igemm\_colmajor
  - FFLAS::Protected, [225](#)
- igemm\_kernels.h, [1016](#)
- igemm\_kernels.inl, [1017](#)
  - \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL, [1018](#)
- igemm\_tools.h, [1018](#)
- igemm\_tools.inl, [1018](#)
  - \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL, [1019](#)
- inl\_range
  - flimits.h, [1009](#)
- index\_t
  - fflas\_sparse.h, [920](#)
  - parallel.h, [1025](#)
- InfNorm
  - FFLAS, [80](#)
- Info, [474](#), [475](#)
  - begin, [475](#), [476](#)
  - Info, [474](#), [476](#)
  - operator=, [475](#), [476](#)
  - perm, [475](#), [476](#)
  - size, [475](#), [476](#)
- init
  - FieldSimd< \_Field >, [444](#)
  - rns\_double, [527](#)–[529](#)
  - rns\_double\_extended, [540](#), [541](#)
  - RNSInteger< RNS >, [545](#)
  - RNSIntegerMod< RNS >, [549](#), [550](#)
- init\_transpose
  - rns\_double, [528](#)
- init\_y
  - FFLAS::sparse\_details, [229](#), [230](#)
- initA
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [498](#)
- initB
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [498](#)
- initC
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [498](#)
- initialize

- ForStrategy1D< blocksize\_t, Cut, Param >, [458](#)
- ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- initOut
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [498](#)
- INLINE
  - fflas\_simd.h, [915](#)
- INST\_OR\_DECL
  - fflas\_L1\_inst.C, [879](#)
  - fflas\_L1\_inst.h, [880](#)
  - fflas\_L2\_inst.C, [883](#)
  - fflas\_L2\_inst.h, [884](#)
  - fflas\_L3\_inst.C, [887](#)
  - fflas\_L3\_inst.h, [888](#)
  - ffpack\_inst.C, [987](#)
  - ffpack\_inst.h, [989](#)
- INSTRSET
  - instrset.h, [1020](#)
- instrset.h, [1019](#)
  - const\_int, [1020](#)
  - const\_uint, [1020](#)
  - hasAVX512ER, [1021](#)
  - hasFMA3, [1021](#)
  - hasFMA4, [1021](#)
  - hasXOP, [1021](#)
  - INSTRSET, [1020](#)
  - instrset\_detect, [1021](#)
  - INSTRSET\_H, [1020](#)
  - int16\_t, [1020](#)
  - int32\_t, [1020](#)
  - int64\_t, [1020](#)
  - int8\_t, [1020](#)
  - intptr\_t, [1021](#)
  - uint16\_t, [1020](#)
  - uint32\_t, [1020](#)
  - uint64\_t, [1021](#)
  - uint8\_t, [1020](#)
- instrset\_detect
  - instrset.h, [1021](#)
  - instrset\_detect.cpp, [1022](#)
- instrset\_detect.cpp, [1021](#)
  - hasAVX512ER, [1022](#)
  - hasF16C, [1022](#)
  - hasFMA3, [1022](#)
  - hasFMA4, [1022](#)
  - hasXOP, [1022](#)
  - instrset\_detect, [1022](#)
- INSTRSET\_H
  - instrset.h, [1020](#)
- int16\_t
  - instrset.h, [1020](#)
- int32\_t
  - instrset.h, [1020](#)
- int64\_t
  - instrset.h, [1020](#)
- int8\_t
  - instrset.h, [1020](#)
- integer
  - rns\_double, [526](#)
  - rns\_double\_extended, [539](#)
  - RNSInteger< RNS >, [544](#)
  - RNSIntegerMod< RNS >, [548](#)
  - test-simd.C, [1105](#)
- Interfaces, [47](#)
- interfaces.doxy, [1022](#)
- intptr\_t
  - instrset.h, [1021](#)
- inv
  - RNSIntegerMod< RNS >, [551](#)
- Invert
  - FFPACK, [326](#), [370](#)
- Invert2
  - FFPACK, [327](#), [370](#)
- Invert2\_modular\_double
  - ffpack.C, [938](#)
  - ffpack\_c.h, [969](#)
- Invert\_modular\_double
  - ffpack.C, [938](#)
  - ffpack\_c.h, [969](#)
- Invertin\_modular\_double
  - ffpack.C, [938](#)
  - ffpack\_c.h, [969](#)
- invp
  - HelperMod< Field, ElementCategories::MachineIntTag >, [471](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [473](#)
- is\_simd< T >, [477](#)
  - type, [477](#)
  - value, [477](#)
- isMOne
  - RNSInteger< RNS >, [545](#)
  - RNSIntegerMod< RNS >, [549](#)
- isOdd
  - FFPACK, [379](#), [380](#)
- isOne
  - RNSInteger< RNS >, [544](#)
  - RNSIntegerMod< RNS >, [549](#)
- IsSingular
  - FFPACK, [332](#), [373](#)
- IsSingular\_modular\_double
  - ffpack.C, [939](#)
  - ffpack\_c.h, [970](#)
- isSparseMatrix< Field, M >, [477](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO > >, [478](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >, [478](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR > >, [478](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > >, [478](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >, [479](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > >, [479](#)

- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd >, >, [479](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO >, >, [480](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO >, >, [480](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO >, >, [480](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL >, >, [481](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO >, >, [481](#)
- isSparseMatrixMKLFormat< F, M >, [481](#)
- isSparseMatrixSimdFormat< F, M >, [481](#)
- isTerminated
  - ForStrategy1D< blocksize\_t, Cut, Param >, [458](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- isZero
  - RNSInteger< RNS >, [545](#)
  - RNSIntegerMod< RNS >, [549](#)
- isZOSparseMatrix< F, M >, [482](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO >, >, [482](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO >, >, [482](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO >, >, [483](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO >, >, [483](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO >, >, [483](#)
- Iterative, [483](#)
- jbegin
  - ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- jend
  - ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- kaapi\_routines.inl, [1022](#)
- \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL, [1022](#)
- KellerGehrig
  - FFPACK::Protected, [397](#)
- KGFast
  - FFPACK::Protected, [397](#)
- KGFast\_generalized
  - FFPACK::Protected, [397](#)
- kmax
  - Sparse< \_Field, SparseMatrix\_t::COO >, [727](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [729](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [730](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [735](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [740](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [746](#)
- KrylovElim
  - FFPACK, [372](#)
- KrylovElim\_modular\_double
  - ffpack.C, [939](#)
  - ffpack\_c.h, [970](#)
- lapack.C, [1022](#)
- \_\_FFLASFFPACK\_CONFIGURATION, [1023](#)
- \_\_FFLASFFPACK\_HAVE\_LAPACK, [1023](#)
- main, [1023](#)
- LAPACKPerm2MathPerm
  - FFPACK, [308](#)
  - ffpack.C, [927](#)
  - ffpack\_c.h, [960](#)
- lastBlockSize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [459](#)
- lastCBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- lastRBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- launch\_fger
  - test-fger.C, [1072](#)
- launch\_fger\_dispatch
  - test-fger.C, [1072](#)
- launch\_MM
  - test-fgemm.C, [1068](#)
- launch\_MM\_dispatch
  - test-fgemm-check.C, [1066](#)
  - test-fgemm.C, [1068](#)
- launch\_MV
  - test-fgemv.C, [1070](#)
- launch\_MV\_dispatch
  - test-fgemv.C, [1070](#)
- launch\_test
  - test-charpoly.C, [1057](#)
  - test-lu.C, [1095](#)
- launch\_wino
  - benchmark-wino.C, [789](#)
- LazyTag, [484](#)
- ld
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [736](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [741](#)
- LeadingSubmatrixRankProfiles
  - FFPACK, [338](#)
  - ffpack.C, [942](#)
  - ffpack\_c.h, [973](#)
- lesser
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [563](#)
  - Simd128\_impl< true, true, false, 2 >, [569](#)
  - Simd128\_impl< true, true, false, 4 >, [577](#)



Simd128\_impl< true, true, false, 8 >, 586  
 Simd128\_impl< true, true, true, 2 >, 599  
 Simd128\_impl< true, true, true, 4 >, 607  
 Simd128\_impl< true, true, true, 8 >, 616  
 Simd256\_impl< true, false, true, 8 >, 626  
 Simd256\_impl< true, true, false, 2 >, 631  
 Simd256\_impl< true, true, false, 4 >, 641, 644  
 Simd256\_impl< true, true, false, 8 >, 656  
 Simd256\_impl< true, true, true, 2 >, 669  
 Simd256\_impl< true, true, true, 4 >, 679, 685  
 Simd256\_impl< true, true, true, 8 >, 694  
 Simd512\_impl< true, false, true, 8 >, 702  
 Simd512\_impl< true, true, false, 8 >, 707  
 Simd512\_impl< true, true, true, 8 >, 721  
 lesser\_eq  
   ScalFunctions< Element, typename enable\_if<  
     is\_floating\_point< Element >::value >::type  
     >, 558  
   ScalFunctions< Element, typename enable\_if<  
     is\_integral< Element >::value >::type >, 563  
   Simd128\_impl< true, true, false, 2 >, 569  
   Simd128\_impl< true, true, false, 4 >, 577  
   Simd128\_impl< true, true, false, 8 >, 586  
   Simd128\_impl< true, true, true, 2 >, 599  
   Simd128\_impl< true, true, true, 4 >, 607  
   Simd128\_impl< true, true, true, 8 >, 616  
   Simd256\_impl< true, false, true, 8 >, 626  
   Simd256\_impl< true, true, false, 2 >, 631  
   Simd256\_impl< true, true, false, 4 >, 641, 644  
   Simd256\_impl< true, true, false, 8 >, 656  
   Simd256\_impl< true, true, true, 2 >, 669  
   Simd256\_impl< true, true, true, 4 >, 679, 685  
   Simd256\_impl< true, true, true, 8 >, 694  
   Simd512\_impl< true, false, true, 8 >, 702  
   Simd512\_impl< true, true, false, 8 >, 707  
   Simd512\_impl< true, true, true, 8 >, 721  
 limits< char >, 484  
   digits, 484  
   max, 484  
   min, 484  
   T, 484  
 limits< double >, 485  
   digits, 485  
   max, 485  
   min, 485  
   T, 485  
 limits< float >, 485  
   digits, 486  
   max, 486  
   min, 486  
   T, 486  
 limits< Givaro::Integer >, 486  
   max, 486  
   min, 486  
   T, 486  
 limits< int >, 487  
   digits, 487  
   max, 487  
   min, 487  
   T, 487  
 limits< long >, 487  
   digits, 488  
   max, 488  
   min, 488  
   T, 488  
 limits< long long >, 488  
   digits, 489  
   max, 489  
   min, 489  
   T, 488  
 limits< Reclnt::rint< K > >, 489  
   max, 489  
   min, 489  
   T, 489  
 limits< Reclnt::ruint< K > >, 490  
   max, 490  
   min, 490  
   T, 490  
 limits< short int >, 490  
   digits, 491  
   max, 491  
   min, 491  
   T, 490  
 limits< signed char >, 491  
   digits, 491  
   max, 491  
   min, 491  
   T, 491  
 limits< T >, 484  
 limits< unsigned char >, 492  
   digits, 492  
   max, 492  
   min, 492  
   T, 492  
 limits< unsigned int >, 492  
   digits, 493  
   max, 493  
   min, 493  
   T, 493  
 limits< unsigned long >, 493  
   digits, 494  
   max, 494  
   min, 494  
   T, 493  
 limits< unsigned long long >, 494  
   digits, 494  
   max, 494  
   min, 494  
   T, 494  
 limits< unsigned short int >, 495  
   digits, 495  
   max, 495  
   min, 495  
   T, 495  
 load  
   Simd128\_impl< true, true, false, 2 >, 568

- Simd128\_impl< true, true, false, 4 >, [576](#)
- Simd128\_impl< true, true, false, 8 >, [585](#)
- Simd128\_impl< true, true, true, 2 >, [594](#)
- Simd128\_impl< true, true, true, 4 >, [603](#)
- Simd128\_impl< true, true, true, 8 >, [612](#)
- Simd256\_impl< true, false, true, 8 >, [623](#)
- Simd256\_impl< true, true, false, 2 >, [630](#)
- Simd256\_impl< true, true, false, 4 >, [640](#), [643](#)
- Simd256\_impl< true, true, false, 8 >, [655](#)
- Simd256\_impl< true, true, true, 2 >, [664](#)
- Simd256\_impl< true, true, true, 4 >, [674](#), [681](#)
- Simd256\_impl< true, true, true, 8 >, [689](#)
- Simd512\_impl< true, false, true, 8 >, [699](#)
- Simd512\_impl< true, true, false, 8 >, [706](#)
- Simd512\_impl< true, true, true, 8 >, [716](#)
- loadu
  - Simd128\_impl< true, true, false, 2 >, [568](#)
  - Simd128\_impl< true, true, false, 4 >, [576](#)
  - Simd128\_impl< true, true, false, 8 >, [585](#)
  - Simd128\_impl< true, true, true, 2 >, [594](#)
  - Simd128\_impl< true, true, true, 4 >, [603](#)
  - Simd128\_impl< true, true, true, 8 >, [612](#)
  - Simd256\_impl< true, false, true, 8 >, [623](#)
  - Simd256\_impl< true, true, false, 2 >, [630](#)
  - Simd256\_impl< true, true, false, 4 >, [640](#), [643](#)
  - Simd256\_impl< true, true, false, 8 >, [655](#)
  - Simd256\_impl< true, true, true, 2 >, [665](#)
  - Simd256\_impl< true, true, true, 4 >, [675](#), [681](#)
  - Simd256\_impl< true, true, true, 8 >, [689](#)
  - Simd512\_impl< true, false, true, 8 >, [699](#)
  - Simd512\_impl< true, true, false, 8 >, [706](#)
  - Simd512\_impl< true, true, true, 8 >, [716](#)
- LQUPtoInverseOfFullRankMinor
  - FFPACK, [348](#), [379](#)
- LT\_OBJDIR
  - config.h, [809](#)
- LUdivine
  - FFPACK, [320](#), [354](#), [355](#), [368](#)
- LUdivine\_construct
  - FFPACK::Protected, [395](#), [401](#)
- LUdivine\_gauss
  - FFPACK, [354](#), [368](#)
- LUdivine\_gauss\_modular\_double
  - ffpack\_c.h, [965](#)
- LUdivine\_modular\_double
  - ffpack.C, [932](#)
  - ffpack\_c.h, [965](#)
- LUdivine\_small
  - FFPACK, [354](#), [368](#)
- LUdivine\_small\_modular\_double
  - ffpack\_c.h, [965](#)
- LUKrylov
  - FFPACK::Protected, [397](#)
- LUKrylov\_KGFast
  - FFPACK::Protected, [398](#)
- m
  - Sparse< \_Field, SparseMatrix\_t::COO >, [727](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [729](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [730](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [736](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [740](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [746](#)
  - MachineFloatTag, [495](#)
  - MachineIntTag, [496](#)
  - main
    - 101-fgemv.C, [759](#)
    - 2x2-fgemv.C, [759](#)
    - 2x2-ftsrv.C, [760](#)
    - 2x2-pluq.C, [760](#)
    - arithprog.C, [764](#)
    - autotune/charpoly.C, [794](#)
    - autotune/pluq.C, [1033](#)
    - benchmark-charpoly-mp.C, [764](#)
    - benchmark-charpoly.C, [765](#)
    - benchmark-checkers.C, [766](#)
    - benchmark-dgemm.C, [767](#)
    - benchmark-dgetrf.C, [768](#)
    - benchmark-dgetri.C, [769](#)
    - benchmark-dsytrf.C, [770](#)
    - benchmark-dtrsm.C, [770](#)
    - benchmark-dtrtri.C, [771](#)
    - benchmark-fadd-lvl2.C, [772](#)
    - benchmark-fdot.C, [773](#)
    - benchmark-fgemv-mp.C, [773](#)
    - benchmark-fgemv-rns.C, [775](#)
    - benchmark-fgemv.C, [776](#)
    - benchmark-fgemv.C, [780](#)
    - benchmark-fgesv.C, [781](#)
    - benchmark-fsyrk.C, [782](#)
    - benchmark-fsytrf.C, [783](#)
    - benchmark-ftsm-mp.C, [783](#)
    - benchmark-ftsm.C, [784](#)
    - benchmark-ftsrv.C, [784](#)
    - benchmark-fttri.C, [785](#)
    - benchmark-inverse.C, [786](#)
    - benchmark-lqmp-mp.C, [786](#)
    - benchmark-lqmp.C, [787](#)
    - benchmark-pluq.C, [788](#)
    - benchmark-wino.C, [789](#)
    - cblas.C, [793](#)
    - clapack.C, [800](#)
    - cuda.C, [826](#)
    - det.C, [828](#)
    - examples/charpoly.C, [794](#)
    - examples/pluq.C, [1034](#)
    - fblas.C, [836](#)
    - fflas-101\_1.C, [836](#)
    - fflas-101\_3.C, [837](#)



- fflas\_101.C, [840](#)
- fflas\_101\_lvl1.C, [841](#)
- ffpack-fgesv.C, [923](#)
- ffpack-solve.C, [923](#)
- fsyrk.C, [1010](#)
- fsytrf.C, [1011](#)
- fttrtri.C, [1012](#)
- gmp.C, [1012](#)
- lapack.C, [1023](#)
- matmul.C, [1024](#)
- rank.C, [1034](#)
- regression-check.C, [1036](#)
- solve.C, [1055](#)
- test-charpoly-check.C, [1057](#)
- test-charpoly.C, [1058](#)
- test-compressQ.C, [1059](#)
- test-det-check.C, [1059](#)
- test-det.C, [1060](#)
- test-echelon.C, [1063](#)
- test-fadd.C, [1064](#)
- test-fdot.C, [1065](#)
- test-fgemm-check.C, [1066](#)
- test-fgemm.C, [1069](#)
- test-fgemv.C, [1070](#)
- test-fger.C, [1072](#)
- test-fgesv.C, [1074](#)
- test-finit.C, [1075](#)
- test-fscal.C, [1076](#)
- test-fsyr2k.C, [1077](#)
- test-fsyrk.C, [1079](#)
- test-fsytrf.C, [1080](#)
- test-ftmm.C, [1082](#)
- test-ftmmv.C, [1083](#)
- test-ftsm-check.C, [1084](#)
- test-ftsm.C, [1085](#)
- test-ftssyr2k.C, [1086](#)
- test-ftstr.C, [1087](#)
- test-ftsv.C, [1089](#)
- test-fttrtri.C, [1090](#)
- test-interfaces-c.c, [1090](#)
- test-invert-check.C, [1091](#)
- test-io.C, [1092](#)
- test-lu.C, [1095](#)
- test-maxdelayeddim.C, [1097](#)
- test-minpoly.C, [1098](#)
- test-multifile2.C, [1098](#)
- test-nullspace.C, [1100](#)
- test-permutations.C, [1100](#)
- test-pluq-check.C, [1101](#)
- test-rankprofiles.C, [1102](#)
- test-rpm.C, [1103](#)
- test-simd.C, [1107](#)
- test-solve.C, [1108](#)
- winograd.C, [1111](#)
- mainpage.doxy, [1023](#)
- mask\_high
  - Simd128\_impl< true, true, false, 8 >, [590](#)
  - Simd128\_impl< true, true, true, 8 >, [616](#)
- Simd256\_impl< true, true, false, 8 >, [661](#)
- Simd256\_impl< true, true, true, 8 >, [694](#)
- Simd512\_impl< true, true, false, 8 >, [712](#)
- Simd512\_impl< true, true, true, 8 >, [721](#)
- mask\_t
  - read\_sparse.h, [1035](#)
- maskstore
  - Simd512\_impl< true, true, false, 8 >, [707](#)
  - Simd512\_impl< true, true, true, 8 >, [717](#)
- MatF2MatD\_Triangular
  - FFLAS::Protected, [226](#)
- MatF2MatFI\_Triangular
  - FFLAS::Protected, [226](#)
- MathPerm2LAPACKPerm
  - FFPACK, [308](#)
  - ffpack.C, [927](#)
  - ffpack\_c.h, [960](#)
- Matio.h, [1023](#)
  - read\_field, [1023](#)
  - write\_field, [1024](#)
- matmul.C, [1024](#)
  - main, [1024](#)
- matmul.doxy, [1024](#)
- Matrix Multiplication Algorithms, [45](#)
- MatrixApplyS
  - FFPACK, [357](#), [358](#)
- MatrixApplyS\_modular\_double
  - ffpack.C, [927](#)
  - ffpack\_c.h, [960](#)
- MatrixApplyT
  - FFPACK, [359](#)
- MatrixApplyT\_modular\_double
  - ffpack.C, [928](#)
  - ffpack\_c.h, [961](#)
- MatVecMinPoly
  - FFPACK, [330](#), [372](#)
  - FFPACK::Protected, [399](#)
- max
  - HelperMod< Field, ElementCategories::MachineIntTag >, [471](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [474](#)
  - limits< char >, [484](#)
  - limits< double >, [485](#)
  - limits< float >, [486](#)
  - limits< Givaro::Integer >, [486](#)
  - limits< int >, [487](#)
  - limits< long >, [488](#)
  - limits< long long >, [489](#)
  - limits< RecInt::rint< K > >, [489](#)
  - limits< RecInt::ruint< K > >, [490](#)
  - limits< short int >, [491](#)
  - limits< signed char >, [491](#)
  - limits< unsigned char >, [492](#)
  - limits< unsigned int >, [493](#)
  - limits< unsigned long >, [494](#)
  - limits< unsigned long long >, [494](#)
  - limits< unsigned short int >, [495](#)

- max3
  - FFLAS, [80](#)
- max4
  - FFLAS, [80](#)
- MAX\_THREADS
  - parallel.h, [1026](#)
- MAX\_WITH\_SIZE\_T
  - test-maxdelayeddim.C, [1096](#)
- maxCol
  - StatsMatrix, [750](#)
- maxColDifference
  - StatsMatrix, [750](#)
- MaxDelayedDim
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [498](#)
- maxElement
  - RNSIntegerMod< RNS >, [549](#)
- maxFieldElt
  - FFPACK, [393](#)
- maxFieldElt< Givaro::ZRing< Givaro::Integer > >
  - FFPACK, [393](#)
- maxpy
  - FieldSimd< \_Field >, [447](#)
- maxpyin
  - FieldSimd< \_Field >, [448](#)
- maxRow
  - StatsMatrix, [749](#)
- maxrow
  - Sparse< \_Field, SparseMatrix\_t::COO >, [728](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [729](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [731](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [736](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [746](#)
- maxRowDifference
  - StatsMatrix, [750](#)
- MaxStorableValue
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [501](#)
- min
  - HelperMod< Field, ElementCategories::MachineIntTag >, [471](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFieldTag >, [473](#)
  - limits< char >, [484](#)
  - limits< double >, [485](#)
  - limits< float >, [486](#)
  - limits< Givaro::Integer >, [486](#)
  - limits< int >, [487](#)
  - limits< long >, [488](#)
  - limits< long long >, [489](#)
  - limits< RecInt::rint< K > >, [489](#)
  - limits< RecInt::ruint< K > >, [490](#)
  - limits< short int >, [491](#)
  - limits< signed char >, [491](#)
  - limits< unsigned char >, [492](#)
  - limits< unsigned int >, [493](#)
  - limits< unsigned long >, [494](#)
  - limits< unsigned long long >, [494](#)
  - limits< unsigned short int >, [495](#)
- min3
  - FFLAS, [80](#)
- min4
  - FFLAS, [80](#)
- min\_types
  - FFLAS::Protected, [223](#), [224](#)
- minCol
  - StatsMatrix, [750](#)
- minColDifference
  - StatsMatrix, [750](#)
- minElement
  - RNSIntegerMod< RNS >, [549](#)
- MinPoly
  - FFPACK, [329](#), [330](#), [371](#)
- minRow
  - StatsMatrix, [749](#)
- minRowDifference
  - StatsMatrix, [750](#)
- MKL\_CONFIG, [401](#)
- MKLSparseMatrixFormat
  - FFLAS, [75](#)
- MMHelper
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [502](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [504](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [506](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [507](#), [508](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [510](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [497](#), [498](#)
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [501](#)
  - MMHelper, [502](#)
  - normA, [503](#)
  - normB, [503](#)
  - operator<<, [502](#)
  - parseq, [503](#)
  - recLevel, [503](#)
  - Self\_t, [501](#)
  - setNorm, [502](#)

- MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait,  
ModeCategories::DefaultTag, ParSeqTrait >,  
503
  - MMHelper, 504
  - normA, 505
  - normB, 505
  - operator<<, 505
  - parseq, 505
  - recLevel, 505
  - Self\_t, 504
  - setNorm, 504
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<mod  
Dest >, ParSeqTrait >, 505
  - MMHelper, 506
  - operator<<, 506
  - parseq, 506
  - recLevel, 506
  - Self\_t, 506
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
ElementCategories::RNSElementTag >,  
ParSeqTrait >, 507
  - MMHelper, 507, 508
  - normA, 508
  - normB, 508
  - operator<<, 508
  - parseq, 509
  - recLevel, 509
  - Self\_t, 507
  - setNorm, 508
- MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag,  
ParSeqTrait >, 509
  - MMHelper, 510
  - operator<<, 510
  - parseq, 510
  - recLevel, 510
  - Self\_t, 509
- MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >,  
496
  - Amax, 500
  - Amin, 500
  - Aunfit, 499
  - Bmax, 500
  - Bmin, 500
  - Bunfit, 499
  - checkA, 499
  - checkB, 499
  - checkOut, 499
  - Cmax, 500
  - Cmin, 500
  - DelayedField, 497
  - delayedField, 501
  - DelayedField\_t, 497
  - DFElt, 497
  - FieldMax, 500
  - FieldMin, 500
  - initA, 498
  - initB, 498
  - initC, 498
  - initOut, 498
  - MaxDelayedDim, 498
  - MaxStorableValue, 501
  - MMHelper, 497, 498
  - operator<<, 499
  - Outmax, 500
  - Outmin, 500
  - parseq, 501
  - recLevel, 500
  - Self\_t, 497
  - setOutBounds, 499
  - FieldSimd< \_Field >, 446
  - Simd128\_impl< true, true, false, 2 >, 573
  - Simd128\_impl< true, true, false, 4 >, 581
  - Simd128\_impl< true, true, false, 8 >, 590
  - Simd128\_impl< true, true, true, 2 >, 599
  - Simd128\_impl< true, true, true, 4 >, 608
  - Simd128\_impl< true, true, true, 8 >, 617
  - Simd256\_impl< true, false, true, 8 >, 627
  - Simd256\_impl< true, true, false, 2 >, 636
  - Simd256\_impl< true, true, false, 4 >, 651
  - Simd256\_impl< true, true, false, 8 >, 661
  - Simd256\_impl< true, true, true, 2 >, 670
  - Simd256\_impl< true, true, true, 4 >, 680, 685
  - Simd256\_impl< true, true, true, 8 >, 695
  - Simd512\_impl< true, true, false, 8 >, 712
  - Simd512\_impl< true, true, true, 8 >, 722
- MODE
  - parallel.h, 1029
- ModeTraits< Field >, 511
  - value, 511
- ModeTraits< Givaro::Modular< Element, Compute >  
>, 511
  - value, 511
- ModeTraits< Givaro::Modular< Givaro::Integer, Com-  
pute > >, 511
  - value, 512
- ModeTraits< Givaro::Modular< int16\_t, Compute > >,  
512
  - value, 512
- ModeTraits< Givaro::Modular< int32\_t, Compute > >,  
512
  - value, 512
- ModeTraits< Givaro::Modular< int8\_t, Compute > >,  
512
  - value, 513
- ModeTraits< Givaro::Modular< Reclnt::ruint< K >,  
Compute > >, 513
  - value, 513
- ModeTraits< Givaro::Modular< uint16\_t, Compute > >,  
513
  - value, 513
- ModeTraits< Givaro::Modular< uint32\_t, Compute > >,  
514
  - value, 514
- ModeTraits< Givaro::Modular< uint8\_t, Compute > >,  
514

- value, [514](#)
- ModeTraits< Givaro::ModularBalanced< Element > >, [514](#)
  - value, [514](#)
- ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, [515](#)
  - value, [515](#)
- ModeTraits< Givaro::ModularBalanced< int16\_t > >, [515](#)
  - value, [515](#)
- ModeTraits< Givaro::ModularBalanced< int32\_t > >, [515](#)
  - value, [516](#)
- ModeTraits< Givaro::ModularBalanced< int8\_t > >, [516](#)
  - value, [516](#)
- ModeTraits< Givaro::Montgomery< T > >, [516](#)
  - value, [516](#)
- ModeTraits< Givaro::ZRing< double > >, [516](#)
  - value, [517](#)
- ModeTraits< Givaro::ZRing< float > >, [517](#)
  - value, [517](#)
- ModeTraits< Givaro::ZRing< Givaro::Integer > >, [517](#)
  - value, [517](#)
- ModField
  - rns\_double, [526](#)
  - rns\_double\_extended, [539](#)
  - RNSIntegerMod< RNS >, [548](#)
- modp
  - FFLAS::vectorised, [289](#), [290](#)
  - FFLAS::vectorised::unswitch, [291](#)
- ModularBalanced< T >, [517](#)
- ModularTag, [518](#)
- mOne
  - RNSInteger< RNS >, [546](#)
  - RNSIntegerMod< RNS >, [553](#)
- mone
  - FFLAS, [79](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [743](#)
- MonotonicApplyP
  - FFPACK, [310](#)
- MonotonicCompress
  - FFPACK, [355](#)
- MonotonicCompressCycles
  - FFPACK, [356](#)
- MonotonicCompressMorePivots
  - FFPACK, [355](#)
- MonotonicExpand
  - FFPACK, [356](#)
- Montgomery< T >, [518](#)
- mul
  - FieldSimd< \_Field >, [446](#)
  - RNSIntegerMod< RNS >, [551](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [556](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [560](#)
- Simd128\_impl< true, true, false, 2 >, [572](#)
- Simd128\_impl< true, true, false, 4 >, [580](#)
- Simd128\_impl< true, true, false, 8 >, [589](#)
- Simd128\_impl< true, true, true, 2 >, [596](#)
- Simd128\_impl< true, true, true, 4 >, [605](#)
- Simd128\_impl< true, true, true, 8 >, [614](#)
- Simd256\_impl< true, false, true, 8 >, [624](#)
- Simd256\_impl< true, true, false, 2 >, [634](#)
- Simd256\_impl< true, true, false, 4 >, [649](#)
- Simd256\_impl< true, true, false, 8 >, [659](#)
- Simd256\_impl< true, true, true, 2 >, [667](#)
- Simd256\_impl< true, true, true, 4 >, [677](#), [682](#)
- Simd256\_impl< true, true, true, 8 >, [692](#)
- Simd512\_impl< true, false, true, 8 >, [701](#)
- Simd512\_impl< true, true, false, 8 >, [711](#)
- Simd512\_impl< true, true, true, 8 >, [719](#)
- mul\_r
  - FieldSimd< \_Field >, [446](#)
- mulhi
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [561](#)
  - Simd128\_impl< true, true, false, 2 >, [569](#)
  - Simd128\_impl< true, true, false, 4 >, [577](#)
  - Simd128\_impl< true, true, true, 2 >, [596](#)
  - Simd128\_impl< true, true, true, 4 >, [605](#)
  - Simd256\_impl< true, true, false, 2 >, [631](#)
  - Simd256\_impl< true, true, false, 4 >, [641](#), [644](#)
  - Simd256\_impl< true, true, true, 2 >, [667](#)
  - Simd256\_impl< true, true, true, 4 >, [677](#), [683](#)
- mulhi\_fast
  - Simd128\_impl< true, true, false, 8 >, [590](#)
  - Simd128\_impl< true, true, true, 8 >, [616](#)
  - Simd256\_impl< true, true, false, 8 >, [661](#)
  - Simd256\_impl< true, true, true, 8 >, [694](#)
  - Simd512\_impl< true, true, false, 8 >, [712](#)
  - Simd512\_impl< true, true, true, 8 >, [721](#)
- mulin
  - FieldSimd< \_Field >, [446](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [557](#)
  - Simd256\_impl< true, false, true, 8 >, [624](#)
  - Simd512\_impl< true, false, true, 8 >, [701](#)
- mullo
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [560](#)
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [580](#)
  - Simd128\_impl< true, true, false, 8 >, [586](#)
  - Simd128\_impl< true, true, true, 2 >, [596](#)
  - Simd128\_impl< true, true, true, 4 >, [605](#)
  - Simd128\_impl< true, true, true, 8 >, [614](#)
  - Simd256\_impl< true, true, false, 2 >, [634](#)
  - Simd256\_impl< true, true, false, 4 >, [649](#)
  - Simd256\_impl< true, true, false, 8 >, [656](#)
  - Simd256\_impl< true, true, true, 2 >, [667](#)
  - Simd256\_impl< true, true, true, 4 >, [677](#), [682](#)
  - Simd256\_impl< true, true, true, 8 >, [692](#)

- Simd512\_impl< true, true, false, 8 >, [708](#)
- Simd512\_impl< true, true, true, 8 >, [719](#)
- mulx
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [561](#)
  - Simd128\_impl< true, true, false, 2 >, [569](#)
  - Simd128\_impl< true, true, false, 4 >, [578](#)
  - Simd128\_impl< true, true, false, 8 >, [586](#)
  - Simd128\_impl< true, true, true, 2 >, [597](#)
  - Simd128\_impl< true, true, true, 4 >, [605](#)
  - Simd128\_impl< true, true, true, 8 >, [614](#)
  - Simd256\_impl< true, true, false, 2 >, [632](#)
  - Simd256\_impl< true, true, false, 4 >, [642](#), [644](#)
  - Simd256\_impl< true, true, false, 8 >, [657](#)
  - Simd256\_impl< true, true, true, 2 >, [667](#)
  - Simd256\_impl< true, true, true, 4 >, [677](#), [683](#)
  - Simd256\_impl< true, true, true, 8 >, [692](#)
  - Simd512\_impl< true, true, false, 8 >, [708](#)
  - Simd512\_impl< true, true, true, 8 >, [719](#)
- mvcnt
  - test-lu.C, [1096](#)
- n
  - Sparse< \_Field, SparseMatrix\_t::COO >, [727](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [729](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [730](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [736](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [746](#)
- nChunks
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [738](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [746](#)
- nDenseCols
  - StatsMatrix, [751](#)
- nDenseRows
  - StatsMatrix, [751](#)
- need\_field\_characteristic< Field >, [518](#)
  - value, [518](#)
- need\_field\_characteristic< Givaro::Modular< Field > >, [518](#)
  - value, [518](#)
- need\_field\_characteristic< Givaro::ModularBalanced< Field > >, [518](#)
  - value, [519](#)
- NeedDoublePreAddReduction
  - FFLAS::Protected, [221](#)
- NeedPreAddReduction
  - FFLAS::Protected, [220](#)
- NeedPreSubReduction
  - FFLAS::Protected, [221](#)
- neg
  - RNSIntegerMod< RNS >, [551](#)
- nElements
  - Sparse< \_Field, SparseMatrix\_t::COO >, [727](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [729](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [731](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [736](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [747](#)
- nEmptyCols
  - StatsMatrix, [751](#)
- nEmptyColsEnd
  - StatsMatrix, [751](#)
- nEmptyRows
  - StatsMatrix, [751](#)
- newD
  - FFPACK::Protected, [399](#)
- NEWWINO
  - fgemm\_winograd.inl, [1005](#)
- nMOnes
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [733](#)
  - StatsMatrix, [749](#)
- nnz
  - Sparse< \_Field, SparseMatrix\_t::COO >, [727](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [729](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [730](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [736](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [747](#)
  - StatsMatrix, [749](#)
- none
  - HelperFlag, [470](#)
- nOnes
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [733](#)
  - StatsMatrix, [749](#)
- NonZeroRandomMatrix
  - FFPACK, [380](#)
- normA

- MMHelper< FFPACK::RNSInteger< E >, Algo-  
Trait, ModeCategories::DefaultTag, ParSeq-  
Trait >, 503
- MMHelper< FFPACK::RNSIntegerMod< E >, Al-  
goTrait, ModeCategories::DefaultTag, ParSeq-  
Trait >, 505
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
ElementCategories::RNSElementTag >,  
ParSeqTrait >, 508
- normB
  - MMHelper< FFPACK::RNSInteger< E >, Algo-  
Trait, ModeCategories::DefaultTag, ParSeq-  
Trait >, 503
  - MMHelper< FFPACK::RNSIntegerMod< E >, Al-  
goTrait, ModeCategories::DefaultTag, ParSeq-  
Trait >, 505
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
ElementCategories::RNSElementTag >,  
ParSeqTrait >, 508
- NORML\_MOD
  - fflas\_simd.h, 915
- NoSimd< T >, 519
  - compliant, 519
  - scalar\_t, 519
  - type\_string, 519
  - valid, 519
  - vect\_size, 520
  - vect\_t, 519
- NoSimdSparseMatrix
  - FFLAS, 75
- NOSPLIT
  - parallel.h, 1030
- nOthers
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >,  
733
  - StatsMatrix, 749
- NotMKLSparseMatrixFormat
  - FFLAS, 75
- NotZOSparseMatrix
  - FFLAS, 75
- NullSpaceBasis
  - FFPACK, 335, 374
- NullSpaceBasis\_modular\_double
  - ffpack.C, 941
  - ffpack\_c.h, 972
- NUM\_THREADS
  - parallel.h, 1026
- NUMARGS
  - parallel.h, 1029
- number\_kind
  - FFLAS, 79
- numBlock
  - ForStrategy1D< blocksize\_t, Cut, Param >, 459
- numblocks
  - ForStrategy1D< blocksize\_t, Cut, Param >, 458
- numColBlock
  - ForStrategy2D< blocksize\_t, Cut, Param >, 462
- numRowBlock
  - ForStrategy2D< blocksize\_t, Cut, Param >, 462
- numthreads
  - Parallel< C, P >, 520
  - Sequential, 564
- one
  - FFLAS, 79
  - RNSInteger< RNS >, 546
  - RNSIntegerMod< RNS >, 553
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 743
- OPENBLAS\_NUM\_THREADS
  - config.h, 809
- operator!=
  - rns\_double\_elt\_cstptr, 534
  - rns\_double\_elt\_ptr, 537
- operator<
  - rns\_double\_elt\_cstptr, 534
  - rns\_double\_elt\_ptr, 537
- operator<<
  - Compose< H1, H2 >, 420
  - FFLAS, 152
  - ForStrategy2D< blocksize\_t, Cut, Param >, 461
  - MMHelper< FFPACK::RNSInteger< E >, Algo-  
Trait, ModeCategories::DefaultTag, ParSeq-  
Trait >, 502
  - MMHelper< FFPACK::RNSIntegerMod< E >, Al-  
goTrait, ModeCategories::DefaultTag, ParSeq-  
Trait >, 505
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
Dest >, ParSeqTrait >, 506
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
ElementCategories::RNSElementTag >,  
ParSeqTrait >, 508
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag,  
ParSeqTrait >, 510
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
Trait >, 499
  - Parallel< C, P >, 521
  - Sequential, 564
  - test-fsytrf.C, 1080
- operator\*
  - rns\_double\_elt\_cstptr, 533
  - rns\_double\_elt\_ptr, 536
- operator()
  - callUdivine\_small< double >, 408
  - callUdivine\_small< Element >, 408
  - callUdivine\_small< float >, 409
  - Failure, 440
  - readMyMachineType< Field, mpz\_t >, 524
  - readMyMachineType< Field, T >, 524
  - RNSInteger< RNS >::RandIter, 522
  - RNSIntegerMod< RNS >::RandIter, 523
  - rnsRandIter< RNS >, 554
  - tfn\_minus, 754
  - tfn\_minus\_eq, 754
  - tfn\_mul, 754
  - tfn\_mul\_eq, 755
  - tfn\_plus, 755
  - tfn\_plus\_eq, 755



operator+  
     rns\_double\_elt\_cstptr, 534  
     rns\_double\_elt\_ptr, 537  
 operator++  
     ForStrategy1D< blocksize\_t, Cut, Param >, 458  
     ForStrategy2D< blocksize\_t, Cut, Param >, 460  
     rns\_double\_elt\_cstptr, 534  
     rns\_double\_elt\_ptr, 537  
 operator+=  
     rns\_double\_elt\_cstptr, 534  
     rns\_double\_elt\_ptr, 537  
 operator-  
     rns\_double\_elt\_cstptr, 534  
     rns\_double\_elt\_ptr, 537  
 operator--  
     rns\_double\_elt\_cstptr, 534  
     rns\_double\_elt\_ptr, 537  
 operator-=  
     rns\_double\_elt\_cstptr, 534  
     rns\_double\_elt\_ptr, 537  
 operator=  
     Coo< Field >, 428, 429  
     Coo< ValT, IdxT >, 427, 430  
     FieldSimd< \_Field >, 444  
     Info, 475, 476  
     rns\_double\_elt\_cstptr, 534  
     rns\_double\_elt\_ptr, 537  
 operator&  
     rns\_double\_elt, 531  
     rns\_double\_elt\_cstptr, 533, 534  
     rns\_double\_elt\_ptr, 536, 537  
 operator[]  
     rns\_double\_elt\_cstptr, 533  
     rns\_double\_elt\_ptr, 536  
 other  
     FFLAS, 79  
     rns\_double\_elt\_cstptr, 535  
     rns\_double\_elt\_ptr, 538  
 Outmax  
     MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
         Trait >, 500  
 Outmin  
     MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
         Trait >, 500  
 p  
     HelperMod< Field, ElementCategories::MachineIntTag  
         >, 471  
     HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag  
         >, 472  
     HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag  
         >, 473  
     HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag  
         >, 473  
 pack\_lhs  
     FFLAS::details, 210  
 pack\_rhs  
     FFLAS::details, 210  
 PACKAGE  
     config.h, 809  
 PACKAGE\_BUGREPORT  
     config.h, 809  
 PACKAGE\_NAME  
     config.h, 810  
 PACKAGE\_STRING  
     config.h, 810  
 PACKAGE\_TARNAME  
     config.h, 810  
 PACKAGE\_URL  
     config.h, 810  
 PACKAGE\_VERSION  
     config.h, 810  
 PAR\_BLOCK  
     parallel.h, 1026  
 Parallel  
     Parallel< C, P >, 520  
 Parallel< C, P >, 520  
     Cut, 520  
     numthreads, 520  
     operator<<, 521  
     Parallel, 520  
     Param, 520  
     set\_numthreads, 521  
 parallel.h, 1024  
     \_\_FFLASFFPACK\_SEQUENTIAL, 1025  
     BARRIER, 1026  
     BEGIN\_PARALLEL\_MAIN, 1027  
     CHECK\_DEPENDENCIES, 1026  
     COMMA, 1028  
     CONSTREFERENCE, 1026  
     END\_PARALLEL\_MAIN, 1027  
     FOR1D, 1027  
     FOR2D, 1028  
     FORBLOCK1D, 1027  
     FORBLOCK2D, 1028  
     index\_t, 1025  
     MAX\_THREADS, 1026  
     MODE, 1029  
     NOSPLIT, 1030  
     NUM\_THREADS, 1026  
     NUMARGS, 1029  
     PAR\_BLOCK, 1026  
     PARFOR1D, 1027  
     PARFOR2D, 1028  
     PARFORBLOCK1D, 1027  
     PARFORBLOCK2D, 1028  
     PP\_ARG\_N, 1029  
     PP\_ARG\_, 1029  
     PP\_RSEQ\_N, 1030  
     READ, 1026  
     READWRITE, 1026  
     RETURNPARAM, 1029  
     split, 1031  
     SPLITTER, 1031  
     splitting\_0, 1030  
     splitting\_1, 1031  
     splitting\_2, 1031

- splitting\_3, [1031](#)
- SYNCH\_GROUP, [1026](#)
- TASK, [1025](#)
- VALUE, [1027](#)
- WAIT, [1026](#)
- WRITE, [1026](#)
- Param
  - Parallel< C, P >, [520](#)
- PARFOR1D
  - parallel.h, [1027](#)
- PARFOR2D
  - parallel.h, [1028](#)
- PARFORBLOCK1D
  - parallel.h, [1027](#)
- PARFORBLOCK2D
  - parallel.h, [1028](#)
- parseArguments
  - FFLAS, [191](#)
- parseq
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [503](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [505](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [506](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [509](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [510](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [501](#)
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [757](#)
- PBASECASE\_K
  - ffpack\_ppluq.inl, [1000](#)
- pColumnEchelonForm
  - FFPACK, [321](#)
- pColumnEchelonForm\_modular\_double
  - ffpack.C, [935](#)
- pColumnEchelonForm\_modular\_float
  - ffpack.C, [936](#)
- pColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [937](#)
- pColumnRankProfile
  - FFPACK, [337](#)
- pDet
  - FFPACK, [333](#)
- perm
  - Info, [475](#), [476](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [747](#)
- PermApplyS
  - FFPACK, [358](#)
- PermApplyS\_double
  - ffpack.C, [928](#)
  - ffpack\_c.h, [961](#)
- PermApplyT
  - FFPACK, [360](#)
- PermApplyT\_double
  - ffpack.C, [928](#)
  - ffpack\_c.h, [961](#)
- pfadd
  - FFLAS, [82](#)
- pfaddin
  - FFLAS, [83](#)
- pfgemm
  - FFLAS, [144](#), [187–189](#)
- pfgemm\_1D\_rec
  - FFLAS, [144](#)
- pfgemm\_2D\_rec
  - FFLAS, [145](#)
- pfgemm\_3D\_rec
  - FFLAS, [145](#)
- pfgemm\_3D\_rec2
  - FFLAS, [145](#)
- pfgemm\_variants.inl, [1031](#)
- pfgemv.inl, [1032](#)
- pfrend
  - FFLAS, [187](#)
- pfreduce
  - FFLAS, [115](#)
- pfspmm
  - FFLAS::sparse\_details, [237–240](#)
  - FFLAS::sparse\_details\_impl, [255](#), [263](#), [264](#), [266–268](#), [278](#)
- pfspmm\_dispatch
  - FFLAS::sparse\_details, [236](#), [237](#)
- pfspmm\_mone
  - FFLAS::sparse\_details\_impl, [256](#)
- pfspmm\_one
  - FFLAS::sparse\_details\_impl, [256](#)
- pfspmm\_zo
  - FFLAS::sparse\_details\_impl, [268](#)
- pfspmv
  - FFLAS::sparse\_details, [240](#), [241](#)
  - FFLAS::sparse\_details\_impl, [257](#), [264](#), [265](#), [268](#), [269](#), [274](#), [275](#), [279](#), [281](#)
- pfspmv\_mone
  - FFLAS::sparse\_details\_impl, [258](#), [269](#), [270](#), [275](#), [282](#)
- pfspmv\_one
  - FFLAS::sparse\_details\_impl, [258](#), [269](#), [275](#), [281](#), [282](#)
- pfspmv\_task
  - FFLAS::sparse\_details\_impl, [257](#)
- pfsub
  - FFLAS, [82](#)
- pfsubin
  - FFLAS, [83](#)
- pfzero
  - FFLAS, [187](#)
- PLUQ
  - FFPACK, [318](#), [319](#), [363](#), [364](#), [367](#)
- pluq.C, [1032](#), [1034](#)



- PLUQ\_basecaseCrout
  - FFPACK, [363](#)
- PLUQ\_basecaseV2
  - FFPACK, [362](#)
- PLUQ\_basecaseV3
  - FFPACK, [362](#)
- PLUQ\_modular\_double
  - ffpack.C, [931](#)
  - ffpack\_c.h, [964](#)
- PLUQtoEchelonPermutation
  - FFPACK, [348](#)
  - ffpack.C, [945](#)
  - ffpack\_c.h, [976](#)
- pm1
  - HelperFlag, [470](#)
- pMMH
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [757](#)
- pow50rem
  - HelperMod< Field, ElementCategories::MachineIntTag >, [471](#)
- PP\_ARG\_N
  - parallel.h, [1029](#)
- PP\_NARG\_
  - parallel.h, [1029](#)
- PP\_RSEQ\_N
  - parallel.h, [1030](#)
- pPLUQ
  - FFPACK, [319](#)
- pRank
  - FFPACK, [331](#)
- preamble
  - FFLAS, [192](#)
- precompute\_cst
  - rns\_double, [527](#)
  - rns\_double\_extended, [540](#)
- pReducedColumnEchelonForm
  - FFPACK, [324](#)
- pReducedColumnEchelonForm\_modular\_double
  - ffpack.C, [935](#)
- pReducedColumnEchelonForm\_modular\_float
  - ffpack.C, [936](#)
- pReducedColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [937](#)
- pReducedRowEchelonForm
  - FFPACK, [325](#)
- pReducedRowEchelonForm\_modular\_double
  - ffpack.C, [936](#)
- pReducedRowEchelonForm\_modular\_float
  - ffpack.C, [937](#)
- pReducedRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [938](#)
- prefetch
  - FFLAS, [194](#)
- print
  - Failure, [441](#)
- printHelpMessage
  - args-parser.h, [762](#)
- printPolynomial
  - test-charpoly-check.C, [1057](#)
- printvect
  - test-compressQ.C, [1059](#)
- pRowEchelonForm
  - FFPACK, [323](#)
- pRowEchelonForm\_modular\_double
  - ffpack.C, [935](#)
- pRowEchelonForm\_modular\_float
  - ffpack.C, [936](#)
- pRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [937](#)
- pRowRankProfile
  - FFPACK, [336](#)
- PSeq
  - benchmark-fgemm-rns.C, [775](#)
- pSolve
  - FFPACK, [334](#)
- PTRSM\_HYBRID\_THRESHOLD
  - fflas\_ptrsm.inl, [911](#)
- PURE
  - fflas\_simd.h, [915](#)
- queryCacheSizes
  - FFLAS, [195](#)
- queryL1CacheSize
  - FFLAS, [195](#)
- queryTopLevelCacheSize
  - FFLAS, [195](#)
- RandInt
  - FFPACK, [383](#)
- RandIter
  - RNSInteger< RNS >::RandIter, [521](#)
  - RNSIntegerMod< RNS >::RandIter, [522](#)
- random
  - RNSInteger< RNS >::RandIter, [521](#), [522](#)
  - RNSIntegerMod< RNS >::RandIter, [523](#)
  - rnsRandIter< RNS >, [554](#)
- RandomIndexSubset
  - FFPACK, [385](#)
- RandomKrylovPrecond
  - FFPACK::Protected, [398](#)
- RandomMatrix
  - FFPACK, [381](#)
- RandomMatrixWithDet
  - FFPACK, [392](#)
- RandomMatrixWithRank
  - FFPACK, [384](#), [385](#)
- RandomMatrixWithRankandRandomRPM
  - FFPACK, [389](#), [390](#)
- RandomMatrixWithRankandRPM
  - FFPACK, [387](#)
- RandomNullSpaceVector
  - FFPACK, [334](#), [349](#), [374](#)
- RandomNullSpaceVector\_modular\_double
  - ffpack.C, [940](#)
  - ffpack\_c.h, [971](#)
- RandomPermutation
  - FFPACK, [386](#)

- RandomRankProfileMatrix
  - FFPACK, [386](#)
- RandomSymmetricMatrix
  - FFPACK, [383](#)
- RandomSymmetricMatrixWithRankandRandomRPM
  - FFPACK, [390](#), [391](#)
- RandomSymmetricMatrixWithRankandRPM
  - FFPACK, [388](#), [389](#)
- RandomSymmetricRankProfileMatrix
  - FFPACK, [386](#)
- RandomTriangularMatrix
  - FFPACK, [382](#), [383](#)
- Rank
  - FFPACK, [331](#), [372](#)
- rank.C, [1034](#)
  - main, [1034](#)
- Rank\_modular\_double
  - ffpack.C, [939](#)
  - ffpack\_c.h, [970](#)
- RankProfileFromLU
  - FFPACK, [338](#)
  - ffpack.C, [941](#)
  - ffpack\_c.h, [972](#)
- READ
  - parallel.h, [1026](#)
- read\_field
  - Matio.h, [1023](#)
- read\_sparse.h, [1034](#)
  - DNS\_BIN\_VER, [1035](#)
  - mask\_t, [1035](#)
- readDnsFormat
  - FFLAS, [154](#)
- readMachineType
  - FFLAS, [154](#)
- ReadMatrix
  - FFLAS, [192](#)
- readMyMachineType< Field, mpz\_t >, [524](#)
  - Element, [524](#)
  - Element\_ptr, [524](#)
  - operator(), [524](#)
- readMyMachineType< Field, T >, [523](#)
  - Element, [523](#)
  - Element\_ptr, [524](#)
  - operator(), [524](#)
- readOrRandomMatrixWithRankAndRandomRPM
  - test-nullspace.C, [1099](#)
- readSmsFormat
  - FFLAS, [153](#)
- readSprFormat
  - FFLAS, [153](#)
- READWRITE
  - parallel.h, [1026](#)
- Rec\_Initialize
  - benchmark-pluq.C, [788](#)
- RecInt, [401](#)
- recLevel
  - MMHelper< FFPACK::RNSInteger< E >, Algo-Trait, ModeCategories::DefaultTag, ParSeq-Trait >, [503](#)
- MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeq-Trait >, [505](#)
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [506](#)
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [509](#)
- MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [510](#)
- MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- Recursive, [525](#)
- reduce
  - FFLAS::vectorised, [288](#), [289](#)
  - rns\_double, [528](#)
  - rns\_double\_extended, [541](#)
  - RNSInteger< RNS >, [545](#)
  - RNSIntegerMod< RNS >, [550](#)
- reduce\_modp
  - RNSIntegerMod< RNS >, [551](#), [552](#)
- reduce\_modp\_rnsmajor
  - RNSIntegerMod< RNS >, [552](#)
- ReducedColumnEchelonForm
  - FFPACK, [323](#), [324](#), [369](#)
- ReducedColumnEchelonForm\_modular\_double
  - ffpack.C, [932](#)
  - ffpack\_c.h, [967](#)
- ReducedColumnEchelonForm\_modular\_float
  - ffpack.C, [933](#)
  - ffpack\_c.h, [967](#)
- ReducedColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [934](#)
  - ffpack\_c.h, [968](#)
- ReducedRowEchelonForm
  - FFPACK, [324](#), [325](#), [369](#)
- ReducedRowEchelonForm2\_modular\_double
  - ffpack\_c.h, [968](#)
- ReducedRowEchelonForm\_modular\_double
  - ffpack.C, [933](#)
  - ffpack\_c.h, [967](#)
- ReducedRowEchelonForm\_modular\_float
  - ffpack.C, [934](#)
  - ffpack\_c.h, [968](#)
- ReducedRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [935](#)
  - ffpack\_c.h, [968](#)
- REF\_modular\_double
  - ffpack\_c.h, [969](#)
- REGISTER\_TYPE\_NAME
  - test-simd.C, [1105](#), [1106](#)
- regression-check.C, [1036](#)
  - check1, [1036](#)
  - check2, [1036](#)
  - check3, [1036](#)
  - check4, [1036](#)
  - checkZeroDimCharpoly, [1036](#)

- checkZeroDimMinPoly, 1036
- gf2ModularBalanced, 1036
- main, 1036
- RETURNPARAM
  - parallel.h, 1029
- ring
  - RNSInteger< RNS >::RandIter, 522
  - RNSIntegerMod< RNS >::RandIter, 523
  - rnsRandIter< RNS >, 554
- rint< K >, 525
- RNS, 47
  - benchmark-fgemm-rns.C, 774
- rns
  - RNSInteger< RNS >, 544
  - RNSIntegerMod< RNS >, 548
- rns-double-elt.h, 1037
- rns-double-recint.inl, 1037
  - \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL, 1037
- rns-double.h, 1038
  - ROUND\_DOWN, 1038
- rns-double.inl, 1038
  - \_\_FFLASFFPACK\_field\_rns\_double\_INL, 1039
- rns-integer-mod.h, 1039
- rns-integer.h, 1040
- rns.h, 1040
- rns.inl, 1041
  - \_\_FFLASFFPACK\_field\_rns\_INL, 1041
- rns\_double, 525
  - \_M, 530
  - \_MMi, 530
  - \_Mi, 530
  - \_basis, 529
  - \_basisMax, 529
  - \_crt\_in, 530
  - \_crt\_out, 530
  - \_field\_rns, 529
  - \_invbasis, 529
  - \_ldm, 530
  - \_mi\_sum, 530
  - \_negbasis, 529
  - \_pbits, 530
  - \_size, 530
  - BasisElement, 526
  - ConstElement\_ptr, 527
  - convert, 528, 529
  - convert\_transpose, 528
  - Element, 526
  - Element\_ptr, 526
  - init, 527–529
  - init\_transpose, 528
  - integer, 526
  - ModField, 526
  - precompute\_cst, 527
  - reduce, 528
  - rns\_double, 527
- rns\_double\_elt, 530
  - \_alloc, 532
  - \_ptr, 532
  - \_stride, 532
  - ~rns\_double\_elt, 531
  - operator&, 531
  - rns\_double\_elt, 531
- rns\_double\_elt\_cstptr, 532
  - \_alloc, 535
  - \_ptr, 535
  - \_stride, 535
  - operator!=, 534
  - operator<, 534
  - operator\*, 533
  - operator+, 534
  - operator++, 534
  - operator+==, 534
  - operator-, 534
  - operator--, 534
  - operator-=, 534
  - operator=, 534
  - operator&, 533, 534
  - operator[], 533
  - other, 535
  - rns\_double\_elt\_cstptr, 533
- rns\_double\_elt\_ptr, 535
  - \_alloc, 538
  - \_ptr, 538
  - \_stride, 538
  - operator!=, 537
  - operator<, 537
  - operator\*, 536
  - operator+, 537
  - operator++, 537
  - operator+==, 537
  - operator-, 537
  - operator--, 537
  - operator-=, 537
  - operator=, 537
  - operator&, 536, 537
  - operator[], 536
  - other, 538
  - rns\_double\_elt\_ptr, 536
- rns\_double\_extended, 538
  - \_M, 542
  - \_MMi, 542
  - \_Mi, 542
  - \_basis, 541
  - \_basisMax, 541
  - \_crt\_in, 542
  - \_crt\_out, 542
  - \_field\_rns, 542
  - \_invbasis, 541
  - \_ldm, 542
  - \_negbasis, 541
  - \_pbits, 542
  - \_size, 542
  - BasisElement, 539
  - ConstElement\_ptr, 539
  - convert, 540, 541

- Element, 539
- Element\_ptr, 539
- init, 540, 541
- integer, 539
- ModField, 539
- precompute\_cst, 540
- reduce, 541
- rns\_double\_extended, 539, 540
- RNSElementTag, 542
- RNSInteger
  - RNSInteger< RNS >, 544
- RNSInteger< RNS >, 543
  - \_rns, 546
  - assign, 546
  - BasisElement, 544
  - cardinality, 545
  - characteristic, 545
  - ConstElement\_ptr, 544
  - convert, 545
  - Element, 544
  - Element\_ptr, 544
  - init, 545
  - integer, 544
  - isMOne, 545
  - isOne, 544
  - isZero, 545
  - mOne, 546
  - one, 546
  - reduce, 545
  - rns, 544
  - RNSInteger, 544
  - size, 544
  - write, 546
  - zero, 546
- RNSInteger< RNS >::RandIter, 521
  - operator(), 522
  - RandIter, 521
  - random, 521, 522
  - ring, 522
- RNSIntegerMod
  - RNSIntegerMod< RNS >, 548
- RNSIntegerMod< RNS >, 546
  - \_F, 552
  - \_Mi\_modp\_rns, 552
  - \_RNSdelayed, 553
  - \_iM\_modp\_rns, 552
  - \_p, 552
  - \_rns, 552
  - add, 550
  - areEqual, 551
  - assign, 550
  - axpyin, 551
  - BasisElement, 548
  - cardinality, 549
  - characteristic, 549
  - ConstElement\_ptr, 548
  - convert, 550
  - delayed, 548
  - Element, 548
  - Element\_ptr, 548
  - init, 549, 550
  - integer, 548
  - inv, 551
  - isMOne, 549
  - isOne, 549
  - isZero, 549
  - maxElement, 549
  - minElement, 549
  - ModField, 548
  - mOne, 553
  - mul, 551
  - neg, 551
  - one, 553
  - reduce, 550
  - reduce\_modp, 551, 552
  - reduce\_modp\_rnsmajor, 552
  - rns, 548
  - RNSIntegerMod, 548
  - size, 549
  - sub, 550
  - write, 551
  - write\_matrix, 551
  - write\_matrix\_long, 552
  - zero, 553
- RNSIntegerMod< RNS >::RandIter, 522
  - operator(), 523
  - RandIter, 522
  - random, 523
  - ring, 523
- RNSModulus
  - FFLAS::CuttingStrategy, 203
- rnsRandIter
  - rnsRandIter< RNS >, 553
- rnsRandIter< RNS >, 553
  - operator(), 554
  - random, 554
  - ring, 554
  - rnsRandIter, 553
- round
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 556
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 559
  - Simd128\_impl< true, true, false, 2 >, 573
  - Simd128\_impl< true, true, false, 4 >, 581
  - Simd128\_impl< true, true, false, 8 >, 590
  - Simd128\_impl< true, true, true, 2 >, 599
  - Simd128\_impl< true, true, true, 4 >, 608
  - Simd128\_impl< true, true, true, 8 >, 616
  - Simd256\_impl< true, false, true, 8 >, 627
  - Simd256\_impl< true, true, false, 2 >, 636
  - Simd256\_impl< true, true, false, 4 >, 651
  - Simd256\_impl< true, true, false, 8 >, 661
  - Simd256\_impl< true, true, true, 2 >, 669
  - Simd256\_impl< true, true, true, 4 >, 680, 685

- Simd256\_impl< true, true, true, 8 >, [694](#)
- Simd512\_impl< true, false, true, 8 >, [703](#)
- Simd512\_impl< true, true, false, 8 >, [712](#)
- Simd512\_impl< true, true, true, 8 >, [721](#)
- ROUND\_DOWN
  - fflas\_sparse.h, [920](#)
  - rns-double.h, [1038](#)
- Row, [554](#)
- row
  - Coo< Field >, [429](#)
  - Coo< ValT, IdxT >, [427](#), [430](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [727](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [729](#)
- rowblockindex
  - ForStrategy2D< blocksize\_t, Cut, Param >, [461](#)
- rowBlockSize
  - ForStrategy2D< blocksize\_t, Cut, Param >, [461](#)
- rowdim
  - StatsMatrix, [749](#)
- RowEchelonForm
  - FFPACK, [322](#), [323](#), [369](#)
- RowEchelonForm\_modular\_double
  - ffpack.C, [932](#)
  - ffpack\_c.h, [966](#)
- RowEchelonForm\_modular\_float
  - ffpack.C, [933](#)
  - ffpack\_c.h, [966](#)
- RowEchelonForm\_modular\_int32\_t
  - ffpack.C, [934](#)
  - ffpack\_c.h, [967](#)
- rownumblocks
  - ForStrategy2D< blocksize\_t, Cut, Param >, [460](#)
- RowRankProfile
  - FFPACK, [336](#), [337](#), [375](#)
- RowRankProfile\_modular\_double
  - ffpack.C, [941](#)
  - ffpack\_c.h, [972](#)
- RowRankProfileSubmatrix
  - FFPACK, [341](#), [376](#)
- RowRankProfileSubmatrix\_modular\_double
  - ffpack.C, [942](#)
  - ffpack\_c.h, [973](#)
- RowRankProfileSubmatrixIndices
  - FFPACK, [339](#), [375](#)
- RowRankProfileSubmatrixIndices\_modular\_double
  - ffpack.C, [942](#)
  - ffpack\_c.h, [973](#)
- ruint< K >, [554](#)
- run\_with\_field
  - benchmark-charpoly.C, [765](#)
  - benchmark-fdot.C, [772](#)
  - test-charpoly.C, [1058](#)
  - test-echelon.C, [1062](#)
  - test-fdot.C, [1065](#)
  - test-fgemm-check.C, [1066](#)
  - test-fgemm.C, [1068](#)
  - test-fgemv.C, [1070](#)
  - test-fger.C, [1072](#)
  - test-fgesv.C, [1073](#)
  - test-finit.C, [1074](#)
  - test-fsyr2k.C, [1077](#)
  - test-fsyrk.C, [1079](#)
  - test-fsytrf.C, [1080](#)
  - test-ftrmm.C, [1082](#)
  - test-ftrmv.C, [1083](#)
  - test-ftrsm.C, [1085](#)
  - test-ftrssyr2k.C, [1086](#)
  - test-ftrstr.C, [1087](#)
  - test-ftrsv.C, [1088](#)
  - test-ftrtri.C, [1090](#)
  - test-io.C, [1092](#)
  - test-lu.C, [1095](#)
  - test-minpoly.C, [1098](#)
  - test-nullspace.C, [1099](#)
  - test-rankprofiles.C, [1102](#)
  - test-solve.C, [1108](#)
- run\_with\_Integer
  - test-fdot.C, [1065](#)
- saxpy\_
  - config-blas.h, [802](#)
- ScalAndReduce
  - FFLAS::Protected, [222](#)
- scalar\_t
  - FieldSimd< \_Field >, [443](#)
  - NoSimd< T >, [519](#)
  - Simd128\_impl< true, true, false, 2 >, [567](#)
  - Simd128\_impl< true, true, false, 4 >, [576](#)
  - Simd128\_impl< true, true, false, 8 >, [584](#)
  - Simd128\_impl< true, true, true, 2 >, [594](#)
  - Simd128\_impl< true, true, true, 4 >, [602](#)
  - Simd128\_impl< true, true, true, 8 >, [611](#)
  - Simd256\_impl< true, false, true, 8 >, [622](#)
  - Simd256\_impl< true, true, false, 2 >, [629](#)
  - Simd256\_impl< true, true, false, 4 >, [639](#)
  - Simd256\_impl< true, true, false, 8 >, [655](#)
  - Simd256\_impl< true, true, true, 2 >, [663](#)
  - Simd256\_impl< true, true, true, 4 >, [673](#), [674](#)
  - Simd256\_impl< true, true, true, 8 >, [688](#)
  - Simd512\_impl< true, false, true, 8 >, [698](#)
  - Simd512\_impl< true, true, false, 8 >, [706](#)
  - Simd512\_impl< true, true, true, 8 >, [715](#)
- ScalFunctions< Element, Enable >, [554](#)
- ScalFunctions< Element, typename enable\_if<
  - is\_floating\_point< Element >::value >::type
  - >, [555](#)
- add, [556](#)
- addin, [556](#)
- ceil, [556](#)
- div, [557](#)
- eq, [558](#)
- floor, [556](#)
- fmadd, [557](#)
- fmaddin, [557](#)
- fmsub, [557](#)
- fmsubin, [557](#)
- fnmadd, [557](#)

- fnmaddin, [557](#)
- greater, [558](#)
- greater\_eq, [558](#)
- lesser, [558](#)
- lesser\_eq, [558](#)
- mul, [556](#)
- mulin, [557](#)
- round, [556](#)
- sub, [556](#)
- subin, [556](#)
- vand, [555](#)
- vandnot, [556](#)
- vor, [555](#)
- vxor, [555](#)
- zero, [555](#)
- ScalFunctions< Element, typename enable\_if<  
is\_integral< Element >::value >::type >,  
[558](#)  
add, [560](#)  
addin, [560](#)  
eq, [563](#)  
fmadd, [561](#)  
fmaddin, [561](#)  
fmaddx, [561](#)  
fmaddxin, [561](#)  
fmsub, [561](#)  
fmsubin, [561](#)  
fmsubx, [562](#)  
fmsubxin, [562](#)  
fnmadd, [562](#)  
fnmaddin, [562](#)  
fnmaddx, [562](#)  
fnmaddxin, [562](#)  
greater, [563](#)  
greater\_eq, [563](#)  
lesser, [563](#)  
lesser\_eq, [563](#)  
mul, [560](#)  
mulhi, [561](#)  
mullo, [560](#)  
mulx, [561](#)  
round, [559](#)  
sll, [563](#)  
sra, [562](#), [563](#)  
srl, [563](#)  
sub, [560](#)  
subin, [560](#)  
vand, [559](#)  
vandnot, [560](#)  
vor, [559](#)  
vxor, [560](#)  
zero, [559](#)
- scalp
  - FFLAS::vectorised, [290](#)
  - FFLAS::vectorised::unswitch, [291](#), [292](#)
- schedule\_bini.inl, [1041](#)
- \_\_FFLASFFPACK\_fgemm\_bini\_INL, [1041](#)
- schedule\_winograd.inl, [1042](#)
- \_\_FFLASFFPACK\_fgemm\_winograd\_INL, [1042](#)
- \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_INL,  
[1043](#)
- schedule\_winograd\_acc\_ip.inl, [1043](#)
- \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_ip\_INL,  
[1043](#)
- schedule\_winograd\_ip.inl, [1044](#)
- \_\_FFLASFFPACK\_fgemm\_winograd\_ip\_INL, [1044](#)
- scopy\_
  - config-blas.h, [804](#)
- sdot\_
  - config-blas.h, [803](#)
- second\_component
  - Compose< H1, H2 >, [420](#)
- Self
  - Coo< ValT, IdxT >, [426](#), [430](#)
- Self\_t
  - MMHelper< FFPACK::RNSInteger< E >, Algo-  
Trait, ModeCategories::DefaultTag, ParSeq-  
Trait >, [501](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, Al-  
goTrait, ModeCategories::DefaultTag, ParSeq-  
Trait >, [504](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
Dest >, ParSeqTrait >, [506](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
ElementCategories::RNSElementTag >,  
ParSeqTrait >, [507](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag,  
ParSeqTrait >, [509](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
Trait >, [497](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [742](#)
- SELL
  - FFLAS, [79](#)
- sell.h, [1044](#)
- sell\_pspmv.inl, [1045](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL,  
[1045](#)
- sell\_spmv.inl, [1045](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL,  
[1046](#)
- sell\_utils.inl, [1046](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL,  
[1047](#)
- SELL\_ZO
  - FFLAS, [79](#)
- Sequential, [564](#)
  - numthreads, [564](#)
  - operator<<, [564](#)
  - Sequential, [564](#)
- set
  - Simd128\_impl< true, true, false, 2 >, [568](#)
  - Simd128\_impl< true, true, false, 4 >, [576](#)
  - Simd128\_impl< true, true, false, 8 >, [585](#)
  - Simd128\_impl< true, true, true, 2 >, [594](#)
  - Simd128\_impl< true, true, true, 4 >, [603](#)

- Simd128\_impl< true, true, true, 8 >, [611](#)
- Simd256\_impl< true, false, true, 8 >, [622](#)
- Simd256\_impl< true, true, false, 2 >, [630](#)
- Simd256\_impl< true, true, false, 4 >, [640](#), [643](#), [646](#)
- Simd256\_impl< true, true, false, 8 >, [655](#)
- Simd256\_impl< true, true, true, 2 >, [664](#)
- Simd256\_impl< true, true, true, 4 >, [674](#), [680](#)
- Simd256\_impl< true, true, true, 8 >, [689](#)
- Simd512\_impl< true, false, true, 8 >, [699](#)
- Simd512\_impl< true, true, false, 8 >, [706](#), [709](#)
- Simd512\_impl< true, true, true, 8 >, [716](#)
- set1
  - Simd128\_impl< true, true, false, 2 >, [567](#)
  - Simd128\_impl< true, true, false, 4 >, [576](#)
  - Simd128\_impl< true, true, false, 8 >, [585](#)
  - Simd128\_impl< true, true, true, 2 >, [594](#)
  - Simd128\_impl< true, true, true, 4 >, [603](#)
  - Simd128\_impl< true, true, true, 8 >, [611](#)
  - Simd256\_impl< true, false, true, 8 >, [622](#)
  - Simd256\_impl< true, true, false, 2 >, [630](#)
  - Simd256\_impl< true, true, false, 4 >, [640](#), [643](#)
  - Simd256\_impl< true, true, false, 8 >, [655](#)
  - Simd256\_impl< true, true, true, 2 >, [664](#)
  - Simd256\_impl< true, true, true, 4 >, [674](#), [680](#)
  - Simd256\_impl< true, true, true, 8 >, [689](#)
  - Simd512\_impl< true, false, true, 8 >, [699](#)
  - Simd512\_impl< true, true, false, 8 >, [706](#)
  - Simd512\_impl< true, true, true, 8 >, [716](#)
- set\_numthreads
  - Parallel< C, P >, [521](#)
- setErrorStream
  - Failure, [440](#)
- setNorm
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [502](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [504](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [508](#)
- setOutBounds
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [499](#)
- sgemm\_
  - config-blas.h, [806](#)
- sgemv\_
  - config-blas.h, [804](#)
- sger\_
  - config-blas.h, [804](#)
- shuffle
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [588](#)
  - Simd128\_impl< true, true, true, 2 >, [595](#)
  - Simd128\_impl< true, true, true, 4 >, [604](#)
  - Simd128\_impl< true, true, true, 8 >, [613](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [647](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [665](#)
  - Simd256\_impl< true, true, true, 4 >, [675](#), [682](#)
  - Simd256\_impl< true, true, true, 8 >, [690](#)
  - Simd512\_impl< true, false, true, 8 >, [700](#)
  - Simd512\_impl< true, true, false, 8 >, [709](#)
  - Simd512\_impl< true, true, true, 8 >, [717](#)
- shuffle\_twice
  - Simd256\_impl< true, true, false, 4 >, [647](#)
  - Simd256\_impl< true, true, true, 4 >, [675](#), [682](#)
- sigma
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [746](#)
- signbits
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd256\_impl< true, true, false, 8 >, [661](#)
  - Simd256\_impl< true, true, true, 8 >, [695](#)
  - Simd512\_impl< true, true, false, 8 >, [712](#)
  - Simd512\_impl< true, true, true, 8 >, [722](#)
- Simd
  - fflas\_simd.h, [916](#)
- simd
  - FieldSimd< \_Field >, [443](#)
- SIMD wrapper, [46](#)
- simd.doxy, [1047](#)
- Simd128
  - simd128.inl, [1048](#)
- simd128.inl, [1047](#)
  - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL, [1047](#)
  - Simd128, [1048](#)
- simd128\_double.inl, [1048](#)
  - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL, [1048](#)
- simd128\_float.inl, [1048](#)
  - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL, [1048](#)
- Simd128\_impl< ArithType, Int, Signed, Size >, [565](#)
- Simd128\_impl< true, false, true, 4 >, [565](#)
  - type\_string, [565](#)
- Simd128\_impl< true, false, true, 8 >, [565](#)
  - type\_string, [565](#)
- Simd128\_impl< true, true, false, 2 >, [566](#)
  - add, [571](#)
  - addin, [571](#)
  - alignment, [574](#)
  - blend, [571](#)
  - compliant, [570](#)
  - eq, [573](#)
  - fmadd, [572](#)
  - fmaddin, [572](#)
  - fmaddx, [569](#)
  - fmaddxin, [569](#)
  - fmsub, [572](#)



- fmsubin, [572](#)
- fmsubx, [570](#)
- fmsubxin, [570](#)
- fnmadd, [572](#)
- fnmaddin, [572](#)
- fnmaddx, [570](#)
- fnmaddxin, [570](#)
- gather, [568](#)
- greater, [569](#)
- greater\_eq, [569](#)
- hadd\_to\_scal, [570](#)
- lesser, [569](#)
- lesser\_eq, [569](#)
- load, [568](#)
- loadu, [568](#)
- mod, [573](#)
- mul, [572](#)
- mulhi, [569](#)
- mullo, [571](#)
- mulx, [569](#)
- round, [573](#)
- scalar\_t, [567](#)
- set, [568](#)
- set1, [567](#)
- shuffle, [571](#)
- sll, [570](#)
- sll128, [573](#)
- sra, [568](#)
- srl, [570](#)
- srl128, [573](#)
- store, [568](#)
- storeu, [568](#)
- stream, [568](#)
- sub, [571](#)
- subin, [571](#)
- type\_string, [573](#)
- unpackhi, [571](#)
- unpacklo, [571](#)
- valid, [570](#)
- vand, [573](#)
- vandnot, [574](#)
- vect\_size, [574](#)
- vect\_t, [567](#)
- vor, [573](#)
- vxor, [574](#)
- zero, [573](#)
- Simd128\_impl< true, true, false, 2 >::Converter, [420](#)
- t, [420](#)
- v, [420](#)
- Simd128\_impl< true, true, false, 4 >, [574](#)
- add, [579](#)
- addin, [580](#)
- alignment, [582](#)
- blend, [579](#)
- compliant, [579](#)
- eq, [581](#)
- fmadd, [580](#)
- fmaddin, [580](#)
- fmaddx, [578](#)
- fmaddxin, [578](#)
- fmsub, [581](#)
- fmsubin, [581](#)
- fmsubx, [578](#)
- fmsubxin, [578](#)
- fnmadd, [580](#)
- fnmaddin, [581](#)
- fnmaddx, [578](#)
- fnmaddxin, [578](#)
- gather, [576](#)
- greater, [577](#)
- greater\_eq, [577](#)
- hadd\_to\_scal, [579](#)
- lesser, [577](#)
- lesser\_eq, [577](#)
- load, [576](#)
- loadu, [576](#)
- mod, [581](#)
- mul, [580](#)
- mulhi, [577](#)
- mullo, [580](#)
- mulx, [578](#)
- round, [581](#)
- scalar\_t, [576](#)
- set, [576](#)
- set1, [576](#)
- shuffle, [579](#)
- sll, [579](#)
- sll128, [582](#)
- sra, [577](#)
- srl, [579](#)
- srl128, [582](#)
- store, [577](#)
- storeu, [577](#)
- stream, [577](#)
- sub, [580](#)
- subin, [580](#)
- type\_string, [581](#)
- unpackhi, [579](#)
- unpacklo, [579](#)
- valid, [579](#)
- vand, [582](#)
- vandnot, [582](#)
- vect\_size, [582](#)
- vect\_t, [576](#)
- vor, [582](#)
- vxor, [582](#)
- zero, [582](#)
- Simd128\_impl< true, true, false, 4 >::Converter, [421](#)
- t, [421](#)
- v, [421](#)
- Simd128\_impl< true, true, false, 8 >, [583](#)
- add, [588](#)
- addin, [588](#)
- alignment, [592](#)
- blend, [588](#)
- compliant, [587](#)



- eq, [590](#)
- fmadd, [589](#)
- fmaddin, [589](#)
- fmaddx, [586](#)
- fmaddxin, [587](#)
- fmsub, [589](#)
- fmsubin, [590](#)
- fmsubx, [587](#)
- fmsubxin, [587](#), [590](#)
- fnmadd, [589](#)
- fnmaddin, [589](#)
- fnmaddx, [587](#)
- fnmaddxin, [587](#)
- gather, [585](#)
- get, [588](#)
- greater, [586](#)
- greater\_eq, [586](#)
- hadd\_to\_scal, [587](#)
- lesser, [586](#)
- lesser\_eq, [586](#)
- load, [585](#)
- loadu, [585](#)
- mask\_high, [590](#)
- mod, [590](#)
- mul, [589](#)
- mulhi\_fast, [590](#)
- mullo, [586](#)
- mulx, [586](#)
- round, [590](#)
- scalar\_t, [584](#)
- set, [585](#)
- set1, [585](#)
- shuffle, [588](#)
- signbits, [591](#)
- sll, [588](#)
- sll128, [591](#)
- sra, [586](#)
- srl, [588](#)
- srl128, [591](#)
- store, [585](#)
- storeu, [585](#)
- stream, [585](#)
- sub, [589](#)
- subin, [589](#)
- type\_string, [591](#)
- unpackhi, [588](#)
- unpacklo, [588](#)
- valid, [587](#)
- vand, [591](#)
- vandnot, [591](#)
- vect\_size, [592](#)
- vect\_t, [585](#)
- vor, [591](#)
- vxor, [591](#)
- zero, [591](#)
- Simd128\_impl< true, true, false, 8 >::Converter, [421](#)
  - t, [421](#)
  - v, [421](#)
- Simd128\_impl< true, true, true, 2 >, [592](#)
  - add, [596](#)
  - addin, [596](#)
  - alignment, [600](#)
  - blend, [596](#)
  - compliant, [594](#)
  - eq, [598](#)
  - fmadd, [597](#)
  - fmaddin, [597](#)
  - fmaddx, [597](#)
  - fmaddxin, [597](#)
  - fmsub, [598](#)
  - fmsubin, [598](#)
  - fmsubx, [598](#)
  - fmsubxin, [598](#)
  - fnmadd, [597](#)
  - fnmaddin, [597](#)
  - fnmaddx, [598](#)
  - fnmaddxin, [598](#)
  - gather, [594](#)
  - greater, [598](#)
  - greater\_eq, [599](#)
  - hadd\_to\_scal, [599](#)
  - lesser, [599](#)
  - lesser\_eq, [599](#)
  - load, [594](#)
  - loadu, [594](#)
  - mod, [599](#)
  - mul, [596](#)
  - mulhi, [596](#)
  - mullo, [596](#)
  - mulx, [597](#)
  - round, [599](#)
  - scalar\_t, [594](#)
  - set, [594](#)
  - set1, [594](#)
  - shuffle, [595](#)
  - sll, [595](#)
  - sll128, [600](#)
  - sra, [595](#)
  - srl, [595](#)
  - srl128, [600](#)
  - store, [595](#)
  - storeu, [595](#)
  - stream, [595](#)
  - sub, [596](#)
  - subin, [596](#)
  - type\_string, [599](#)
  - unpackhi, [595](#)
  - unpacklo, [595](#)
  - valid, [594](#)
  - vand, [600](#)
  - vandnot, [600](#)
  - vect\_size, [600](#)
  - vect\_t, [593](#)
  - vor, [600](#)
  - vxor, [600](#)
  - zero, [599](#)

- Simd128\_impl< true, true, true, 2 >::Converter, [421](#)
  - t, [422](#)
  - v, [422](#)
- Simd128\_impl< true, true, true, 4 >, [601](#)
  - add, [604](#)
  - addin, [604](#)
  - alignment, [609](#)
  - blend, [604](#)
  - compliant, [603](#)
  - eq, [607](#)
  - fmadd, [605](#)
  - fmaddin, [605](#)
  - fmaddx, [606](#)
  - fmaddxin, [606](#)
  - fmsub, [606](#)
  - fmsubin, [607](#)
  - fmsubx, [607](#)
  - fmsubxin, [607](#)
  - fnmadd, [606](#)
  - fnmaddin, [606](#)
  - fnmaddx, [606](#)
  - fnmaddxin, [606](#)
  - gather, [603](#)
  - greater, [607](#)
  - greater\_eq, [607](#)
  - hadd\_to\_scal, [608](#)
  - lesser, [607](#)
  - lesser\_eq, [607](#)
  - load, [603](#)
  - loadu, [603](#)
  - mod, [608](#)
  - mul, [605](#)
  - mulhi, [605](#)
  - mullo, [605](#)
  - mulx, [605](#)
  - round, [608](#)
  - scalar\_t, [602](#)
  - set, [603](#)
  - set1, [603](#)
  - shuffle, [604](#)
  - sll, [604](#)
  - sll128, [608](#)
  - sra, [604](#)
  - srl, [604](#)
  - srl128, [608](#)
  - store, [603](#)
  - storeu, [603](#)
  - stream, [603](#)
  - sub, [605](#)
  - subin, [605](#)
  - type\_string, [608](#)
  - unpackhi, [604](#)
  - unpacklo, [604](#)
  - valid, [602](#)
  - vand, [608](#)
  - vandnot, [609](#)
  - vect\_size, [609](#)
  - vect\_t, [602](#)
  - vor, [608](#)
  - vxor, [609](#)
  - zero, [608](#)
- Simd128\_impl< true, true, true, 4 >::Converter, [422](#)
  - t, [422](#)
  - v, [422](#)
- Simd128\_impl< true, true, true, 8 >, [609](#)
  - add, [613](#)
  - addin, [613](#)
  - alignment, [618](#)
  - blend, [613](#)
  - compliant, [611](#)
  - eq, [616](#)
  - fmadd, [614](#)
  - fmaddin, [614](#)
  - fmaddx, [614](#)
  - fmaddxin, [614](#)
  - fmsub, [615](#)
  - fmsubin, [615](#)
  - fmsubx, [615](#)
  - fmsubxin, [615](#)
  - fnmadd, [614](#)
  - fnmaddin, [615](#)
  - fnmaddx, [615](#)
  - fnmaddxin, [615](#)
  - gather, [612](#)
  - get, [612](#)
  - greater, [616](#)
  - greater\_eq, [616](#)
  - hadd\_to\_scal, [616](#)
  - lesser, [616](#)
  - lesser\_eq, [616](#)
  - load, [612](#)
  - loadu, [612](#)
  - mask\_high, [616](#)
  - mod, [617](#)
  - mul, [614](#)
  - mulhi\_fast, [616](#)
  - mullo, [614](#)
  - mulx, [614](#)
  - round, [616](#)
  - scalar\_t, [611](#)
  - set, [611](#)
  - set1, [611](#)
  - shuffle, [613](#)
  - signbits, [617](#)
  - sll, [612](#)
  - sll128, [617](#)
  - sra, [613](#)
  - srl, [612](#)
  - srl128, [617](#)
  - store, [612](#)
  - storeu, [612](#)
  - stream, [612](#)
  - sub, [613](#)
  - subin, [613](#)
  - type\_string, [617](#)
  - unpackhi, [613](#)

- unpacklo, [613](#)
- valid, [611](#)
- vand, [617](#)
- vandnot, [618](#)
- vect\_size, [618](#)
- vect\_t, [611](#)
- vor, [617](#)
- vxor, [618](#)
- zero, [617](#)
- Simd128\_impl< true, true, true, 8 >::Converter, [422](#)
- t, [422](#)
- v, [422](#)
- simd128\_int16.inl, [1048](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL, [1049](#)
- simd128\_int32.inl, [1049](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL, [1049](#)
- simd128\_int64.inl, [1049](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL, [1049](#)
- vect\_t, [1049](#)
- Simd128fp\_base, [618](#)
- type\_string, [618](#)
- Simd128i\_base, [619](#)
- sll128, [619](#)
- srl128, [619](#)
- type\_string, [619](#)
- vand, [620](#)
- vandnot, [620](#)
- vect\_t, [619](#)
- vor, [620](#)
- vxor, [620](#)
- zero, [619](#)
- Simd256
- simd256.inl, [1050](#)
- simd256.inl, [1050](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL, [1050](#)
- Simd256, [1050](#)
- simd256\_double.inl, [1050](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL, [1050](#)
- simd256\_float.inl, [1051](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL, [1051](#)
- Simd256\_impl< ArithType, Int, Signed, Size >, [620](#)
- Simd256\_impl< true, false, true, 4 >, [620](#)
- Simd256\_impl< true, false, true, 8 >, [621](#)
- add, [624](#)
- addin, [624](#)
- alignment, [627](#)
- blend, [624](#)
- blendv, [624](#)
- ceil, [627](#)
- compliant, [622](#)
- div, [625](#)
- eq, [626](#)
- floor, [627](#)
- fmadd, [625](#)
- fmaddin, [625](#)
- fmsub, [625](#)
- fmsubin, [625](#)
- fnmadd, [625](#)
- fnmaddin, [625](#)
- gather, [623](#)
- greater, [626](#)
- greater\_eq, [626](#)
- hadd, [627](#)
- hadd\_to\_scal, [627](#)
- lesser, [626](#)
- lesser\_eq, [626](#)
- load, [623](#)
- loadu, [623](#)
- mod, [627](#)
- mul, [624](#)
- mulin, [624](#)
- round, [627](#)
- scalar\_t, [622](#)
- set, [622](#)
- set1, [622](#)
- store, [623](#)
- storeu, [623](#)
- stream, [623](#)
- sub, [624](#)
- subin, [624](#)
- unpackhi\_twice, [623](#)
- unpacklo\_twice, [623](#)
- valid, [622](#)
- vand, [626](#)
- vandnot, [626](#)
- vect\_size, [627](#)
- vect\_t, [622](#)
- vor, [626](#)
- vxor, [626](#)
- zero, [622](#)
- Simd256\_impl< true, true, false, 2 >, [628](#)
- add, [634](#)
- addin, [634](#)
- alignment, [636](#)
- blend\_twice, [634](#)
- compliant, [633](#)
- eq, [635](#)
- fmadd, [635](#)
- fmaddin, [635](#)
- fmaddx, [632](#)
- fmaddxin, [632](#)
- fmsub, [635](#)
- fmsubin, [635](#)
- fmsubx, [632](#)
- fmsubxin, [632](#)
- fnmadd, [635](#)
- fnmaddin, [635](#)
- fnmaddx, [632](#)
- fnmaddxin, [632](#)
- gather, [630](#)

- greater, 631
- greater\_eq, 631
- hadd\_to\_scal, 632
- half\_t, 630
- lesser, 631
- lesser\_eq, 631
- load, 630
- loadu, 630
- mod, 636
- mul, 634
- mulhi, 631
- mullo, 634
- mulx, 632
- round, 636
- scalar\_t, 629
- set, 630
- set1, 630
- shuffle, 633
- simdHalf, 629
- sll, 633
- sra, 631
- srl, 633
- store, 630
- storeu, 631
- stream, 631
- sub, 634
- subin, 634
- type\_string, 636
- unpackhi, 633
- unpackhi\_twice, 633
- unpacklo, 633
- unpacklo\_twice, 633
- unpacklohi, 634
- valid, 633
- vect\_size, 636
- vect\_t, 629
- zero, 636
- Simd256\_impl< true, true, false, 2 >::Converter, 423
  - t, 423
  - v, 423
- Simd256\_impl< true, true, false, 4 >, 636
  - add, 648
  - addin, 648
  - alignment, 653
  - blend, 648
  - compliant, 646
  - eq, 651
  - fmadd, 649
  - fmaddin, 649, 650
  - fmaddx, 642, 644
  - fmaddxin, 642, 645
  - fmsub, 650
  - fmsubin, 651
  - fmsubx, 642, 645
  - fmsubxin, 642, 645
  - fmadd, 650
  - fmaddin, 650
  - fmaddx, 642, 645
- fnmaddxin, 642, 645
- gather, 640, 643
- greater, 641, 644
- greater\_eq, 641, 644
- hadd\_to\_scal, 642, 645
- half\_t, 640
- lesser, 641, 644
- lesser\_eq, 641, 644
- load, 640, 643
- loadu, 640, 643
- mod, 651
- mul, 649
- mulhi, 641, 644
- mullo, 649
- mulx, 642, 644
- round, 651
- scalar\_t, 639
- set, 640, 643, 646
- set1, 640, 643
- shuffle, 647
- shuffle\_twice, 647
- simdHalf, 639
- sll, 646
- sra, 641, 644
- srl, 646
- store, 640, 643
- storeu, 641, 643
- stream, 641, 643
- sub, 648
- subin, 648, 649
- type\_string, 652
- unpackhi, 647
- unpackhi\_twice, 647
- unpacklo, 647
- unpacklo\_twice, 647
- unpacklohi, 647
- valid, 645
- vand, 652
- vandnot, 652
- vect\_size, 653
- vect\_t, 639, 640
- vor, 652
- vxor, 652
- zero, 652
- Simd256\_impl< true, true, false, 4 >::Converter, 423
  - t, 423
  - v, 423
- Simd256\_impl< true, true, false, 8 >, 653
  - add, 659
  - addin, 659
  - alignment, 661
  - blend, 659
  - compliant, 658
  - eq, 660
  - fmadd, 660
  - fmaddin, 660
  - fmaddx, 657
  - fmaddxin, 657

- fmsub, [660](#)
- fmsubin, [660](#)
- fmsubx, [657](#)
- fmsubxin, [657](#)
- fnmadd, [660](#)
- fnmaddin, [660](#)
- fnmaddx, [657](#)
- fnmaddxin, [657](#)
- gather, [655](#)
- get, [658](#)
- greater, [656](#)
- greater\_eq, [656](#)
- hadd\_to\_scal, [658](#)
- half\_t, [655](#)
- lesser, [656](#)
- lesser\_eq, [656](#)
- load, [655](#)
- loadu, [655](#)
- mask\_high, [661](#)
- mod, [661](#)
- mul, [659](#)
- mulhi\_fast, [661](#)
- mullo, [656](#)
- mulx, [657](#)
- round, [661](#)
- scalar\_t, [655](#)
- set, [655](#)
- set1, [655](#)
- shuffle, [658](#)
- signbits, [661](#)
- simdHalf, [655](#)
- sll, [658](#)
- sra, [656](#)
- srl, [658](#)
- store, [656](#)
- storeu, [656](#)
- stream, [656](#)
- sub, [659](#)
- subin, [659](#)
- type\_string, [661](#)
- unpackhi, [659](#)
- unpackhi\_twice, [658](#)
- unpacklo, [658](#)
- unpacklo\_twice, [658](#)
- unpacklohi, [659](#)
- valid, [658](#)
- vect\_size, [661](#)
- vect\_t, [655](#)
- zero, [661](#)
- Simd256\_impl< true, true, false, 8 >::Converter, [423](#)
  - t, [424](#)
  - v, [423](#)
- Simd256\_impl< true, true, true, 2 >, [662](#)
  - add, [666](#)
  - addin, [666](#)
  - alignment, [670](#)
  - blend\_twice, [666](#)
  - compliant, [664](#)
  - eq, [669](#)
  - fmadd, [667](#)
  - fmaddin, [667](#)
  - fmaddx, [667](#)
  - fmaddxin, [668](#)
  - fmsub, [668](#)
  - fmsubin, [668](#)
  - fmsubx, [669](#)
  - fmsubxin, [669](#)
  - fnmadd, [668](#)
  - fnmaddin, [668](#)
  - fnmaddx, [668](#)
  - fnmaddxin, [668](#)
  - gather, [664](#)
  - greater, [669](#)
  - greater\_eq, [669](#)
  - hadd\_to\_scal, [669](#)
  - half\_t, [663](#)
  - lesser, [669](#)
  - lesser\_eq, [669](#)
  - load, [664](#)
  - loadu, [665](#)
  - mod, [670](#)
  - mul, [667](#)
  - mulhi, [667](#)
  - mullo, [667](#)
  - mulx, [667](#)
  - round, [669](#)
  - scalar\_t, [663](#)
  - set, [664](#)
  - set1, [664](#)
  - shuffle, [665](#)
  - simdHalf, [664](#)
  - sll, [665](#)
  - sra, [665](#)
  - srl, [665](#)
  - store, [665](#)
  - storeu, [665](#)
  - stream, [665](#)
  - sub, [666](#)
  - subin, [667](#)
  - type\_string, [670](#)
  - unpackhi, [666](#)
  - unpackhi\_twice, [666](#)
  - unpacklo, [666](#)
  - unpacklo\_twice, [665](#)
  - unpacklohi, [666](#)
  - valid, [664](#)
  - vect\_size, [670](#)
  - vect\_t, [663](#)
  - zero, [670](#)
- Simd256\_impl< true, true, true, 2 >::Converter, [424](#)
  - t, [424](#)
  - v, [424](#)
- Simd256\_impl< true, true, true, 4 >, [670](#)
  - add, [676](#), [682](#)
  - addin, [676](#), [682](#)
  - alignment, [686](#)

- blend, [676](#)
- compliant, [674](#), [680](#)
- eq, [679](#), [684](#)
- fmadd, [677](#), [683](#)
- fmaddin, [677](#), [683](#)
- fmaddx, [678](#), [683](#)
- fmaddxin, [678](#), [683](#)
- fmsub, [678](#), [684](#)
- fmsubin, [679](#), [684](#)
- fmsubx, [679](#), [684](#)
- fmsubxin, [679](#), [684](#)
- fnmadd, [678](#), [683](#)
- fnmaddin, [678](#), [683](#)
- fnmaddx, [678](#), [684](#)
- fnmaddxin, [678](#), [684](#)
- gather, [674](#), [681](#)
- greater, [679](#), [685](#)
- greater\_eq, [679](#), [685](#)
- hadd\_to\_scal, [680](#), [685](#)
- half\_t, [673](#), [674](#)
- lesser, [679](#), [685](#)
- lesser\_eq, [679](#), [685](#)
- load, [674](#), [681](#)
- loadu, [675](#), [681](#)
- mod, [680](#), [685](#)
- mul, [677](#), [682](#)
- mulhi, [677](#), [683](#)
- mullo, [677](#), [682](#)
- mulx, [677](#), [683](#)
- round, [680](#), [685](#)
- scalar\_t, [673](#), [674](#)
- set, [674](#), [680](#)
- set1, [674](#), [680](#)
- shuffle, [675](#), [682](#)
- shuffle\_twice, [675](#), [682](#)
- simdHalf, [673](#), [674](#)
- sll, [675](#), [681](#)
- sra, [675](#), [682](#)
- srl, [675](#), [681](#)
- store, [675](#), [681](#)
- storeu, [675](#), [681](#)
- stream, [675](#), [681](#)
- sub, [677](#), [682](#)
- subin, [677](#), [682](#)
- type\_string, [685](#), [686](#)
- unpackhi, [676](#)
- unpackhi\_twice, [676](#)
- unpacklo, [676](#)
- unpacklo\_twice, [676](#)
- unpacklohi, [676](#)
- valid, [674](#), [680](#)
- vand, [686](#)
- vandnot, [686](#)
- vect\_size, [686](#)
- vect\_t, [673](#)
- vor, [686](#)
- vxor, [686](#)
- zero, [685](#), [686](#)
- Simd256\_impl< true, true, true, 4 >::Converter, [424](#)
  - t, [424](#)
  - v, [424](#)
- Simd256\_impl< true, true, true, 8 >, [687](#)
  - add, [691](#)
  - addin, [691](#)
  - alignment, [695](#)
  - blend, [691](#)
  - compliant, [689](#)
  - eq, [694](#)
  - fmadd, [692](#)
  - fmaddin, [692](#)
  - fmaddx, [692](#)
  - fmaddxin, [692](#)
  - fmsub, [693](#)
  - fmsubin, [693](#)
  - fmsubx, [693](#)
  - fmsubxin, [693](#)
  - fnmadd, [692](#)
  - fnmaddin, [693](#)
  - fnmaddx, [693](#)
  - fnmaddxin, [693](#)
  - gather, [689](#)
  - get, [689](#)
  - greater, [694](#)
  - greater\_eq, [694](#)
  - hadd\_to\_scal, [694](#)
  - half\_t, [688](#)
  - lesser, [694](#)
  - lesser\_eq, [694](#)
  - load, [689](#)
  - loadu, [689](#)
  - mask\_high, [694](#)
  - mod, [695](#)
  - mul, [692](#)
  - mulhi\_fast, [694](#)
  - mullo, [692](#)
  - mulx, [692](#)
  - round, [694](#)
  - scalar\_t, [688](#)
  - set, [689](#)
  - set1, [689](#)
  - shuffle, [690](#)
  - signbits, [695](#)
  - simdHalf, [689](#)
  - sll, [690](#)
  - sra, [690](#)
  - srl, [690](#)
  - store, [690](#)
  - storeu, [690](#)
  - stream, [690](#)
  - sub, [691](#)
  - subin, [691](#)
  - type\_string, [695](#)
  - unpackhi, [691](#)
  - unpackhi\_twice, [690](#)
  - unpacklo, [691](#)
  - unpacklo\_twice, [690](#)

- unpacklohi, [691](#)
- valid, [689](#)
- vect\_size, [695](#)
- vect\_t, [688](#)
- zero, [695](#)
- Simd256\_impl< true, true, true, 8 >::Converter, [425](#)
- t, [425](#)
- v, [425](#)
- simd256\_int16.inl, [1051](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL, [1051](#)
- simd256\_int32.inl, [1051](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL, [1052](#)
- simd256\_int64.inl, [1052](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL, [1052](#)
- vect\_t, [1052](#)
- Simd256fp\_base, [695](#)
- Simd256i\_base, [696](#)
- type\_string, [696](#)
- vect\_t, [696](#)
- zero, [696](#)
- Simd512
- simd512.inl, [1053](#)
- simd512.inl, [1052](#)
- \_\_FFLASFFPACK\_simd512\_INL, [1053](#)
- Simd512, [1053](#)
- simd512\_double.inl, [1053](#)
- \_\_FFLASFFPACK\_simd512\_double\_INL, [1053](#)
- simd512\_float.inl, [1053](#)
- \_\_FFLASFFPACK\_simd512\_float\_INL, [1053](#)
- Simd512\_impl< ArithType, Int, Signed, Size >, [697](#)
- Simd512\_impl< true, false, true, 4 >, [697](#)
- type\_string, [697](#)
- Simd512\_impl< true, false, true, 8 >, [697](#)
- add, [701](#)
- addin, [701](#)
- alignment, [703](#)
- blend, [700](#)
- blendv, [700](#)
- ceil, [703](#)
- compliant, [699](#)
- div, [701](#)
- eq, [702](#)
- floor, [703](#)
- fmadd, [701](#)
- fmaddin, [701](#)
- fmsub, [702](#)
- fmsubin, [702](#)
- fnmadd, [702](#)
- fnmaddin, [702](#)
- gather, [699](#)
- greater, [702](#)
- greater\_eq, [703](#)
- hadd, [703](#)
- hadd\_to\_scal, [703](#)
- lesser, [702](#)
- lesser\_eq, [702](#)
- load, [699](#)
- loadu, [699](#)
- mul, [701](#)
- mulin, [701](#)
- round, [703](#)
- scalar\_t, [698](#)
- set, [699](#)
- set1, [699](#)
- shuffle, [700](#)
- store, [700](#)
- storeu, [700](#)
- stream, [700](#)
- sub, [701](#)
- subin, [701](#)
- type\_string, [703](#)
- unpackhi\_twice, [700](#)
- unpacklo\_twice, [700](#)
- valid, [699](#)
- vect\_size, [703](#)
- vect\_t, [698](#)
- zero, [699](#)
- Simd512\_impl< true, true, false, 8 >, [704](#)
- add, [710](#)
- addin, [710](#)
- alignment, [713](#)
- blend, [710](#)
- compliant, [709](#)
- eq, [712](#)
- fmadd, [711](#)
- fmaddin, [711](#)
- fmaddx, [708](#)
- fmaddxin, [708](#)
- fmsub, [711](#)
- fmsubin, [711](#)
- fmsubx, [708](#)
- fmsubxin, [708](#)
- fnmadd, [711](#)
- fnmaddin, [711](#)
- fnmaddx, [708](#)
- fnmaddxin, [708](#)
- gather, [706](#)
- greater, [707](#)
- greater\_eq, [707](#)
- hadd\_to\_scal, [709](#)
- half\_t, [706](#)
- lesser, [707](#)
- lesser\_eq, [707](#)
- load, [706](#)
- loadu, [706](#)
- mask\_high, [712](#)
- maskstore, [707](#)
- mod, [712](#)
- mul, [711](#)
- mulhi\_fast, [712](#)
- mullo, [708](#)
- mulx, [708](#)
- round, [712](#)

- scalar\_t, 706
- set, 706, 709
- set1, 706
- shuffle, 709
- signbits, 712
- simdHalf, 706
- sll, 709
- sra, 707
- srl, 709
- store, 707
- storeu, 707
- stream, 707
- sub, 710
- subin, 710
- type\_string, 712
- unpackhi, 710
- unpackhi\_twice, 709
- unpacklo, 710
- unpacklo\_twice, 709
- unpacklohi, 710
- valid, 709
- vand, 713
- vandnot, 713
- vect\_size, 713
- vect\_t, 706
- vor, 712
- vxor, 713
- zero, 712
- Simd512\_impl< true, true, false, 8 >::Converter, 425
  - t, 425
  - v, 425
- Simd512\_impl< true, true, true, 8 >, 713
  - add, 718
  - addin, 718
  - alignment, 723
  - blend, 718
  - compliant, 716
  - eq, 721
  - fmadd, 719
  - fmaddin, 719
  - fmaddx, 719
  - fmaddxin, 719
  - fmsub, 720
  - fmsubin, 720
  - fmsubx, 720
  - fmsubxin, 720
  - fmadd, 719
  - fmaddin, 720
  - fmaddx, 720
  - fmaddxin, 720
  - gather, 716
  - greater, 721
  - greater\_eq, 721
  - hadd\_to\_scal, 721
  - half\_t, 715
  - lesser, 721
  - lesser\_eq, 721
  - load, 716
  - loadu, 716
  - mask\_high, 721
  - maskstore, 717
  - mod, 722
  - mul, 719
  - mulhi\_fast, 721
  - mullo, 719
  - mulx, 719
  - round, 721
  - scalar\_t, 715
  - set, 716
  - set1, 716
  - shuffle, 717
  - signbits, 722
  - simdHalf, 715
  - sll, 717
  - sra, 717
  - srl, 717
  - store, 716
  - storeu, 717
  - stream, 717
  - sub, 718
  - subin, 718
  - type\_string, 722
  - unpackhi, 718
  - unpackhi\_twice, 717
  - unpacklo, 718
  - unpacklo\_twice, 717
  - unpacklohi, 718
  - valid, 715
  - vand, 722
  - vandnot, 722
  - vect\_size, 723
  - vect\_t, 715
  - vor, 722
  - vxor, 722
  - zero, 722
- Simd512\_impl< true, true, true, 8 >::Converter, 425
  - t, 426
  - v, 425
- simd512\_int32.inl, 1053
  - \_\_FFLASFFPACK\_simd512\_int32\_INL, 1054
- simd512\_int64.inl, 1054
  - \_simd512\_int64\_INL, 1054
  - vect\_t, 1054
- Simd512fp\_base, 723
  - type\_string, 723
- Simd512i\_base, 723
  - type\_string, 724
  - vand, 724
  - vandnot, 724
  - vect\_t, 724
  - vor, 724
  - vxor, 724
  - zero, 724
- SIMD\_INT
  - fflas\_simd.h, 915
- simd\_modular.inl, 1054



- SimdChooser< T, bool, bool >, [725](#)
- SimdChooser< T, false, b >, [725](#)
  - value, [725](#)
- SimdChooser< T, true, false >, [725](#)
  - value, [725](#)
- SimdChooser< T, true, true >, [725](#)
  - value, [726](#)
- simdHalf
  - Simd256\_impl< true, true, false, 2 >, [629](#)
  - Simd256\_impl< true, true, false, 4 >, [639](#)
  - Simd256\_impl< true, true, false, 8 >, [655](#)
  - Simd256\_impl< true, true, true, 2 >, [664](#)
  - Simd256\_impl< true, true, true, 4 >, [673](#), [674](#)
  - Simd256\_impl< true, true, true, 8 >, [689](#)
  - Simd512\_impl< true, true, false, 8 >, [706](#)
  - Simd512\_impl< true, true, true, 8 >, [715](#)
- SimdSparseMatrix
  - FFLAS, [75](#)
- simdToType< T >, [726](#)
- Single, [726](#)
- size
  - Info, [475](#), [476](#)
  - RNSInteger< RNS >, [544](#)
  - RNSIntegerMod< RNS >, [549](#)
- SZOF\_\_INT64
  - config.h, [810](#)
- SZOF\_CHAR
  - config.h, [810](#)
- SZOF\_INT
  - config.h, [810](#)
- SZOF\_LONG
  - config.h, [810](#)
- SZOF\_LONG\_LONG
  - config.h, [810](#)
- SZOF\_SHORT
  - config.h, [810](#)
- sll
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [563](#)
  - Simd128\_impl< true, true, false, 2 >, [570](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [588](#)
  - Simd128\_impl< true, true, true, 2 >, [595](#)
  - Simd128\_impl< true, true, true, 4 >, [604](#)
  - Simd128\_impl< true, true, true, 8 >, [612](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [646](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [665](#)
  - Simd256\_impl< true, true, true, 4 >, [675](#), [681](#)
  - Simd256\_impl< true, true, true, 8 >, [690](#)
  - Simd512\_impl< true, true, false, 8 >, [709](#)
  - Simd512\_impl< true, true, true, 8 >, [717](#)
- sll128
  - Simd128\_impl< true, true, false, 2 >, [573](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [600](#)
- Simd128\_impl< true, true, true, 4 >, [608](#)
- Simd128\_impl< true, true, true, 8 >, [617](#)
- Simd128i\_base, [619](#)
- Solve
  - FFPACK, [333](#), [334](#), [373](#)
- solve.C, [1054](#)
  - main, [1055](#)
- Solve\_modular\_double
  - ffpack.C, [940](#)
  - ffpack\_c.h, [971](#)
- solveLB
  - FFPACK, [349](#), [374](#)
- solveLB2
  - FFPACK, [349](#), [374](#)
- solveLB2\_modular\_double
  - ffpack.C, [940](#)
  - ffpack\_c.h, [971](#)
- solveLB\_modular\_double
  - ffpack.C, [940](#)
  - ffpack\_c.h, [971](#)
- Sparse< \_Field, SparseMatrix\_t::COO >, [726](#)
  - col, [727](#)
  - dat, [727](#)
  - delayed, [727](#)
  - Field, [727](#)
  - kmax, [727](#)
  - m, [727](#)
  - maxrow, [728](#)
  - n, [727](#)
  - nElements, [727](#)
  - nnz, [727](#)
  - row, [727](#)
- Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [728](#)
  - col, [728](#)
  - cst, [728](#)
  - dat, [729](#)
  - delayed, [729](#)
  - Field, [728](#)
  - kmax, [729](#)
  - m, [729](#)
  - maxrow, [729](#)
  - n, [729](#)
  - nElements, [729](#)
  - nnz, [729](#)
  - row, [729](#)
- Sparse< \_Field, SparseMatrix\_t::CSR >, [729](#)
  - col, [731](#)
  - dat, [731](#)
  - delayed, [730](#)
  - Field, [730](#)
  - kmax, [730](#)
  - m, [730](#)
  - maxrow, [731](#)
  - n, [730](#)
  - nElements, [731](#)
  - nnz, [730](#)
  - st, [731](#)
  - stend, [731](#)

- Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 731
  - col, 732
  - dat, 732
  - delayed, 732
  - Field, 732
  - kmax, 732
  - m, 732
  - maxrow, 732
  - n, 732
  - nElements, 732
  - nMOnes, 733
  - nnz, 732
  - nOnes, 733
  - nOthers, 733
  - st, 732
- Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 733
  - col, 734
  - cst, 734
  - dat, 735
  - delayed, 734
  - Field, 733
  - kmax, 734
  - m, 734
  - maxrow, 734
  - n, 734
  - nElements, 734
  - nnz, 734
  - st, 734
  - stend, 734
- Sparse< \_Field, SparseMatrix\_t::ELL >, 735
  - col, 736
  - dat, 736
  - delayed, 735
  - Field, 735
  - kmax, 735
  - ld, 736
  - m, 736
  - maxrow, 736
  - n, 736
  - nElements, 736
  - nnz, 736
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 736
  - chunk, 737
  - col, 738
  - dat, 738
  - delayed, 737
  - kmax, 737
  - ld, 737
  - m, 737
  - maxrow, 737
  - n, 737
  - nChunks, 738
  - nElements, 737
  - nnz, 737
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 738
  - chunk, 739
  - col, 739
  - cst, 738
  - dat, 739
  - delayed, 738
  - kmax, 739
  - ld, 739
  - m, 739
  - maxrow, 739
  - n, 739
  - nChunks, 739
  - nElements, 739
  - nnz, 739
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 740
  - col, 741
  - cst, 740
  - dat, 741
  - delayed, 740
  - Field, 740
  - kmax, 740
  - ld, 741
  - m, 740
  - maxrow, 741
  - n, 741
  - nElements, 741
  - nnz, 741
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 741
  - dat, 742
  - delayed, 742
  - Field, 742
  - kmax, 742
  - m, 742
  - maxrow, 742
  - mone, 743
  - n, 742
  - nElements, 742
  - nnz, 742
  - one, 743
  - Self\_t, 742
- Sparse< \_Field, SparseMatrix\_t::SELL >, 743
  - chunk, 744
  - chunkSize, 745
  - col, 745
  - dat, 745
  - delayed, 744
  - Field, 743
  - kmax, 744
  - m, 744
  - maxrow, 744
  - n, 744
  - nChunks, 744
  - nElements, 744
  - nnz, 744
  - perm, 744
  - sigma, 744
  - st, 745
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 745
  - chunk, 746
  - chunkSize, 747
  - col, 747
  - cst, 746

- dat, [747](#)
  - delayed, [746](#)
  - Field, [746](#)
  - kmax, [746](#)
  - m, [746](#)
  - maxrow, [746](#)
  - n, [746](#)
  - nChunks, [746](#)
  - nElements, [747](#)
  - nnz, [747](#)
  - perm, [747](#)
  - sigma, [746](#)
  - st, [747](#)
  - Sparse< Field, SparseMatrix\_t, IdxT, PtrT >, [726](#)
  - sparse\_delete
    - FFLAS, [147](#), [148](#), [150–152](#), [155](#)
  - sparse\_init
    - FFLAS, [147–152](#), [155](#)
  - sparse\_matrix\_traits.h, [1055](#)
  - sparse\_print
    - FFLAS, [148](#), [151](#), [155](#)
  - SparseMatrix\_t
    - FFLAS, [79](#)
  - SpecRankProfile
    - FFPACK, [372](#)
  - SpecRankProfile\_modular\_double
    - ffpack.C, [939](#)
    - ffpack\_c.h, [970](#)
  - splitt
    - parallel.h, [1031](#)
  - SPLITTER
    - parallel.h, [1031](#)
  - splitting\_0
    - parallel.h, [1030](#)
  - splitting\_1
    - parallel.h, [1031](#)
  - splitting\_2
    - parallel.h, [1031](#)
  - splitting\_3
    - parallel.h, [1031](#)
  - SpMat< Field, flag >, [747](#)
    - \_coo, [747](#)
    - \_csr, [747](#)
    - \_ell, [748](#)
  - sra
    - ScalFunctions< Element, typename enable\_if<
      - is\_integral< Element >::value >::type >, [562](#), [563](#)
    - Simd128\_impl< true, true, false, 2 >, [568](#)
    - Simd128\_impl< true, true, false, 4 >, [577](#)
    - Simd128\_impl< true, true, false, 8 >, [586](#)
    - Simd128\_impl< true, true, true, 2 >, [595](#)
    - Simd128\_impl< true, true, true, 4 >, [604](#)
    - Simd128\_impl< true, true, true, 8 >, [613](#)
    - Simd256\_impl< true, true, false, 2 >, [631](#)
    - Simd256\_impl< true, true, false, 4 >, [641](#), [644](#)
    - Simd256\_impl< true, true, false, 8 >, [656](#)
    - Simd256\_impl< true, true, true, 2 >, [665](#)
  - Simd256\_impl< true, true, true, 4 >, [675](#), [682](#)
  - Simd256\_impl< true, true, true, 8 >, [690](#)
  - Simd512\_impl< true, true, false, 8 >, [707](#)
  - Simd512\_impl< true, true, true, 8 >, [717](#)
- srl
  - ScalFunctions< Element, typename enable\_if<
    - is\_integral< Element >::value >::type >, [563](#)
  - Simd128\_impl< true, true, false, 2 >, [570](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [588](#)
  - Simd128\_impl< true, true, true, 2 >, [595](#)
  - Simd128\_impl< true, true, true, 4 >, [604](#)
  - Simd128\_impl< true, true, true, 8 >, [612](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [646](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [665](#)
  - Simd256\_impl< true, true, true, 4 >, [675](#), [681](#)
  - Simd256\_impl< true, true, true, 8 >, [690](#)
  - Simd512\_impl< true, true, false, 8 >, [709](#)
  - Simd512\_impl< true, true, true, 8 >, [717](#)
- srl128
  - Simd128\_impl< true, true, false, 2 >, [573](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [600](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd128i\_base, [619](#)
- sscal\_
  - config-blas.h, [805](#)
- st
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [731](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [732](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [745](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [747](#)
- Static\_error\_check
  - Static\_error\_check< bool >, [748](#)
- Static\_error\_check< bool >, [748](#)
  - Static\_error\_check, [748](#)
- Static\_error\_check< false >, [748](#)
- StatsMatrix, [748](#)
  - averageCol, [750](#)
  - averageColDifference, [750](#)
  - averageRow, [750](#)
  - averageRowDifference, [751](#)
  - coldim, [749](#)
  - denseCols, [751](#)
  - denseRows, [751](#)
  - deviationCol, [750](#)
  - deviationColDifference, [750](#)
  - deviationRow, [750](#)
  - deviationRowDifference, [751](#)
  - maxCol, [750](#)
  - maxColDifference, [750](#)
  - maxRow, [749](#)

- maxRowDifference, [750](#)
- minCol, [750](#)
- minColDifference, [750](#)
- minRow, [749](#)
- minRowDifference, [750](#)
- nDenseCols, [751](#)
- nDenseRows, [751](#)
- nEmptyCols, [751](#)
- nEmptyColsEnd, [751](#)
- nEmptyRows, [751](#)
- nMOnes, [749](#)
- nnz, [749](#)
- nOnes, [749](#)
- nOthers, [749](#)
- rowdim, [749](#)
- STDC\_HEADERS
  - config.h, [810](#)
- stend
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [731](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [734](#)
- store
  - Simd128\_impl< true, true, false, 2 >, [568](#)
  - Simd128\_impl< true, true, false, 4 >, [577](#)
  - Simd128\_impl< true, true, false, 8 >, [585](#)
  - Simd128\_impl< true, true, true, 2 >, [595](#)
  - Simd128\_impl< true, true, true, 4 >, [603](#)
  - Simd128\_impl< true, true, true, 8 >, [612](#)
  - Simd256\_impl< true, false, true, 8 >, [623](#)
  - Simd256\_impl< true, true, false, 2 >, [630](#)
  - Simd256\_impl< true, true, false, 4 >, [640](#), [643](#)
  - Simd256\_impl< true, true, false, 8 >, [656](#)
  - Simd256\_impl< true, true, true, 2 >, [665](#)
  - Simd256\_impl< true, true, true, 4 >, [675](#), [681](#)
  - Simd256\_impl< true, true, true, 8 >, [690](#)
  - Simd512\_impl< true, false, true, 8 >, [700](#)
  - Simd512\_impl< true, true, false, 8 >, [707](#)
  - Simd512\_impl< true, true, true, 8 >, [716](#)
- storeu
  - Simd128\_impl< true, true, false, 2 >, [568](#)
  - Simd128\_impl< true, true, false, 4 >, [577](#)
  - Simd128\_impl< true, true, false, 8 >, [585](#)
  - Simd128\_impl< true, true, true, 2 >, [595](#)
  - Simd128\_impl< true, true, true, 4 >, [603](#)
  - Simd128\_impl< true, true, true, 8 >, [612](#)
  - Simd256\_impl< true, false, true, 8 >, [623](#)
  - Simd256\_impl< true, true, false, 2 >, [631](#)
  - Simd256\_impl< true, true, false, 4 >, [641](#), [643](#)
  - Simd256\_impl< true, true, false, 8 >, [656](#)
  - Simd256\_impl< true, true, true, 2 >, [665](#)
  - Simd256\_impl< true, true, true, 4 >, [675](#), [681](#)
  - Simd256\_impl< true, true, true, 8 >, [690](#)
  - Simd512\_impl< true, false, true, 8 >, [700](#)
  - Simd512\_impl< true, true, false, 8 >, [707](#)
  - Simd512\_impl< true, true, true, 8 >, [717](#)
- stream
  - Simd128\_impl< true, true, false, 2 >, [568](#)
  - Simd128\_impl< true, true, false, 4 >, [577](#)
  - Simd128\_impl< true, true, false, 8 >, [585](#)
- Simd128\_impl< true, true, true, 2 >, [595](#)
- Simd128\_impl< true, true, true, 4 >, [603](#)
- Simd128\_impl< true, true, true, 8 >, [612](#)
- Simd256\_impl< true, false, true, 8 >, [623](#)
- Simd256\_impl< true, true, false, 2 >, [631](#)
- Simd256\_impl< true, true, false, 4 >, [641](#), [643](#)
- Simd256\_impl< true, true, false, 8 >, [656](#)
- Simd256\_impl< true, true, true, 2 >, [665](#)
- Simd256\_impl< true, true, true, 4 >, [675](#), [681](#)
- Simd256\_impl< true, true, true, 8 >, [690](#)
- Simd512\_impl< true, false, true, 8 >, [700](#)
- Simd512\_impl< true, true, false, 8 >, [707](#)
- Simd512\_impl< true, true, true, 8 >, [717](#)
- strmm\_
  - config-blas.h, [806](#)
- strsm\_
  - config-blas.h, [805](#)
- sub
  - FFLAS::vectorised, [288](#)
  - FieldSimd< \_Field >, [445](#)
  - RNSIntegerMod< RNS >, [550](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [556](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [560](#)
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [580](#)
  - Simd128\_impl< true, true, false, 8 >, [589](#)
  - Simd128\_impl< true, true, true, 2 >, [596](#)
  - Simd128\_impl< true, true, true, 4 >, [605](#)
  - Simd128\_impl< true, true, true, 8 >, [613](#)
  - Simd256\_impl< true, false, true, 8 >, [624](#)
  - Simd256\_impl< true, true, false, 2 >, [634](#)
  - Simd256\_impl< true, true, false, 4 >, [648](#)
  - Simd256\_impl< true, true, false, 8 >, [659](#)
  - Simd256\_impl< true, true, true, 2 >, [666](#)
  - Simd256\_impl< true, true, true, 4 >, [677](#), [682](#)
  - Simd256\_impl< true, true, true, 8 >, [691](#)
  - Simd512\_impl< true, false, true, 8 >, [701](#)
  - Simd512\_impl< true, true, false, 8 >, [710](#)
  - Simd512\_impl< true, true, true, 8 >, [718](#)
- sub\_r
  - FieldSimd< \_Field >, [445](#)
- subin
  - FieldSimd< \_Field >, [445](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [556](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [560](#)
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [580](#)
  - Simd128\_impl< true, true, false, 8 >, [589](#)
  - Simd128\_impl< true, true, true, 2 >, [596](#)
  - Simd128\_impl< true, true, true, 4 >, [605](#)
  - Simd128\_impl< true, true, true, 8 >, [613](#)
  - Simd256\_impl< true, false, true, 8 >, [624](#)

- Simd256\_impl< true, true, false, 2 >, [634](#)
- Simd256\_impl< true, true, false, 4 >, [648](#), [649](#)
- Simd256\_impl< true, true, false, 8 >, [659](#)
- Simd256\_impl< true, true, true, 2 >, [667](#)
- Simd256\_impl< true, true, true, 4 >, [677](#), [682](#)
- Simd256\_impl< true, true, true, 8 >, [691](#)
- Simd512\_impl< true, false, true, 8 >, [701](#)
- Simd512\_impl< true, true, false, 8 >, [710](#)
- Simd512\_impl< true, true, true, 8 >, [718](#)
- subin\_r
  - FieldSimd< \_Field >, [445](#)
- subp
  - FFLAS::vectorised, [287](#)
- support\_fast\_mod< double >, [752](#)
- support\_fast\_mod< float >, [752](#)
- support\_fast\_mod< int64\_t >, [752](#)
- support\_fast\_mod< T >, [751](#)
- support\_simd< T >, [753](#)
- support\_simd\_add< T >, [753](#)
- support\_simd\_mod< T >, [753](#)
- swapval
  - FFPACK, [386](#)
- SYNCH\_GROUP
  - parallel.h, [1026](#)
- SysTimer
  - FFLAS, [77](#)
- T
  - limits< char >, [484](#)
  - limits< double >, [485](#)
  - limits< float >, [486](#)
  - limits< Givaro::Integer >, [486](#)
  - limits< int >, [487](#)
  - limits< long >, [488](#)
  - limits< long long >, [488](#)
  - limits< Reclnt::rint< K > >, [489](#)
  - limits< Reclnt::ruint< K > >, [490](#)
  - limits< short int >, [490](#)
  - limits< signed char >, [491](#)
  - limits< unsigned char >, [492](#)
  - limits< unsigned int >, [493](#)
  - limits< unsigned long >, [493](#)
  - limits< unsigned long long >, [494](#)
  - limits< unsigned short int >, [495](#)
- t
  - Simd128\_impl< true, true, false, 2 >::Converter, [420](#)
  - Simd128\_impl< true, true, false, 4 >::Converter, [421](#)
  - Simd128\_impl< true, true, false, 8 >::Converter, [421](#)
  - Simd128\_impl< true, true, true, 2 >::Converter, [422](#)
  - Simd128\_impl< true, true, true, 4 >::Converter, [422](#)
  - Simd128\_impl< true, true, true, 8 >::Converter, [422](#)
  - Simd256\_impl< true, true, false, 2 >::Converter, [423](#)
  - Simd256\_impl< true, true, false, 4 >::Converter, [423](#)
  - Simd256\_impl< true, true, false, 8 >::Converter, [424](#)
  - Simd256\_impl< true, true, true, 2 >::Converter, [424](#)
  - Simd256\_impl< true, true, true, 4 >::Converter, [424](#)
  - Simd256\_impl< true, true, true, 8 >::Converter, [425](#)
  - Simd512\_impl< true, true, false, 8 >::Converter, [425](#)
  - Simd512\_impl< true, true, true, 8 >::Converter, [426](#)
- TASK
  - parallel.h, [1025](#)
- tBC
  - test-lu.C, [1096](#)
  - test-permutations.C, [1101](#)
- test
  - test-maxdelayeddim.C, [1097](#)
  - test-simd.C, [1107](#)
- test-charpoly-check.C, [1056](#)
  - ENABLE\_CHECKER\_charpoly, [1057](#)
  - main, [1057](#)
  - printPolynomial, [1057](#)
  - TIME\_CHECKER\_CHARPOLY, [1057](#)
- test-charpoly.C, [1057](#)
  - launch\_test, [1057](#)
  - main, [1058](#)
  - run\_with\_field, [1058](#)
- test-compressQ.C, [1058](#)
  - Field, [1058](#)
  - main, [1059](#)
  - printvect, [1059](#)
- test-det-check.C, [1059](#)
  - ENABLE\_CHECKER\_Det, [1059](#)
  - main, [1059](#)
  - TIME\_CHECKER\_Det, [1059](#)
- test-det.C, [1060](#)
  - main, [1060](#)
  - test\_det, [1060](#)
- test-echelon.C, [1060](#)
  - \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE, [1061](#)
  - \_\_FFLASFFPACK\_PLUQ\_THRESHOLD, [1061](#)
  - \_\_FFLASFFPACK\_SEQUENTIAL, [1061](#)
  - main, [1063](#)
  - run\_with\_field, [1062](#)
  - test\_colechelon, [1061](#)
  - test\_redcoechelon, [1062](#)
  - test\_redrowechelon, [1062](#)
  - test\_rowechelon, [1062](#)
- test-fadd.C, [1063](#)
  - main, [1064](#)
  - test\_fadd, [1063](#)
  - test\_faddin, [1063](#)
  - test\_fsub, [1064](#)

- test\_fsubin, 1064
- test-fdot.C, 1064
  - check\_fdot, 1065
  - ENABLE\_ALL\_CHECKINGS, 1065
  - main, 1065
  - run\_with\_field, 1065
  - run\_with\_Integer, 1065
- test-fgemm-check.C, 1065
  - ENABLE\_ALL\_CHECKINGS, 1066
  - launch\_MM\_dispatch, 1066
  - main, 1066
  - run\_with\_field, 1066
- test-fgemm.C, 1067
  - check\_MM, 1067
  - ENABLE\_CHECKER\_fgemm, 1067
  - launch\_MM, 1068
  - launch\_MM\_dispatch, 1068
  - main, 1069
  - run\_with\_field, 1068
- test-fgemv.C, 1069
  - check\_MV, 1069
  - launch\_MV, 1070
  - launch\_MV\_dispatch, 1070
  - main, 1070
  - run\_with\_field, 1070
- test-fger.C, 1071
  - check\_fger, 1071
  - launch\_fger, 1072
  - launch\_fger\_dispatch, 1072
  - main, 1072
  - run\_with\_field, 1072
  - TIME, 1071
- test-fgesv.C, 1073
  - main, 1074
  - run\_with\_field, 1073
  - test\_rect\_fgesv, 1073
  - test\_square\_fgesv, 1073
- test-finit.C, 1074
  - main, 1075
  - run\_with\_field, 1074
  - test\_freduce, 1074
- test-fscal.C, 1075
  - main, 1076
  - test\_fscal, 1075, 1076
  - test\_fscaln, 1076
- test-fsyr2k.C, 1076
  - check\_fsyr2k, 1077
  - ENABLE\_ALL\_CHECKINGS, 1077
  - main, 1077
  - run\_with\_field, 1077
- test-fsyrk.C, 1078
  - check\_fsyrk, 1078
  - check\_fsyrk\_bkdiag, 1079
  - check\_fsyrk\_diag, 1078
  - ENABLE\_ALL\_CHECKINGS, 1078
  - main, 1079
  - run\_with\_field, 1079
- test-fsytrf.C, 1079
  - main, 1080
  - operator<<, 1080
  - run\_with\_field, 1080
  - test\_generic\_fsytrf, 1080
  - test\_RPM\_fsytrf, 1080
- test-ftrmm.C, 1081
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1081
  - check\_ftrmm, 1081
  - main, 1082
  - run\_with\_field, 1082
- test-ftrmv.C, 1082
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1082
  - check\_ftrmv, 1083
  - ENABLE\_ALL\_CHECKINGS, 1082
  - main, 1083
  - run\_with\_field, 1083
- test-ftrsm-check.C, 1083
  - ENABLE\_ALL\_CHECKINGS, 1083
  - main, 1084
- test-ftrsm.C, 1084
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1084
  - check\_ftrsm, 1085
  - ENABLE\_ALL\_CHECKINGS, 1084
  - main, 1085
  - run\_with\_field, 1085
- test-ftrssyr2k.C, 1085
  - check\_ftrssyr2k, 1086
  - ENABLE\_ALL\_CHECKINGS, 1086
  - main, 1086
  - run\_with\_field, 1086
- test-ftrstr.C, 1086
  - check\_ftrstr, 1087
  - ENABLE\_ALL\_CHECKINGS, 1087
  - main, 1087
  - run\_with\_field, 1087
- test-ftrsv.C, 1088
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1088
  - check\_ftrsv, 1088
  - ENABLE\_ALL\_CHECKINGS, 1088
  - main, 1089
  - run\_with\_field, 1088
- test-ftrtri.C, 1089
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1089
  - check\_ftrtri, 1090
  - ENABLE\_ALL\_CHECKINGS, 1089
  - main, 1090
  - run\_with\_field, 1090
- test-interfaces-c.c, 1090
  - main, 1090
- test-invert-check.C, 1090
  - ENABLE\_ALL\_CHECKINGS, 1091
  - main, 1091
- test-io.C, 1091
  - main, 1092
  - run\_with\_field, 1092
- test-lu.C, 1092
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1093
  - \_\_LUDIVINE\_CUTOFF, 1093

- BASECASE\_K, 1093
- launch\_test, 1095
- main, 1095
- mvcnt, 1096
- run\_with\_field, 1095
- tBC, 1096
- test\_LUdivine, 1093
- test\_pluq, 1094
- tgemm, 1096
- timtot, 1096
- tperm, 1095
- trest, 1096
- ttrsm, 1096
- verifPLUQ, 1094
- test-maxdelayeddim.C, 1096
  - main, 1097
  - MAX\_WITH\_SIZE\_T, 1096
  - test, 1097
- test-minpoly.C, 1097
  - check\_minpoly, 1097
  - main, 1098
  - run\_with\_field, 1098
- test-multifile1.C, 1098
- test-multifile2.C, 1098
  - main, 1098
- test-nullspace.C, 1098
  - checkingMessage, 1099
  - main, 1100
  - readOrRandomMatrixWithRankAndRandomRPM, 1099
  - run\_with\_field, 1099
  - test\_nullspace, 1099
- test-permutations.C, 1100
  - checkMonotonicApplyP, 1100
  - main, 1100
  - tBC, 1101
  - tgemm, 1101
  - timtot, 1101
  - tperm, 1100
  - trest, 1101
  - ttrsm, 1101
- test-pluq-check.C, 1101
  - ENABLE\_ALL\_CHECKINGS, 1101
  - main, 1101
- test-rankprofiles.C, 1102
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1102
  - main, 1102
  - run\_with\_field, 1102
- test-rpm.C, 1103
  - checkRPM, 1103
  - checkSymmetricRPM, 1103
  - main, 1103
- test-simd.C, 1103
  - check\_eq, 1106
  - eval\_func\_on\_array, 1106, 1107
  - generate\_random\_vector, 1106
  - integer, 1105
  - main, 1107
  - REGISTER\_TYPE\_NAME, 1105, 1106
  - test, 1107
  - test\_impl, 1107
  - TEST\_ONE\_OP, 1105
  - test\_op, 1107
  - TypeName, 1105
- test-solve.C, 1107
  - check\_solve, 1108
  - main, 1108
  - run\_with\_field, 1108
- test-utils.h, 1108
- test\_colechelon
  - test-echelon.C, 1061
- test\_det
  - test-det.C, 1060
- test\_fadd
  - test-fadd.C, 1063
- test\_faddin
  - test-fadd.C, 1063
- test\_freduce
  - test-finit.C, 1074
- test\_fscal
  - test-fscal.C, 1075, 1076
- test\_fscalin
  - test-fscal.C, 1076
- test\_fsub
  - test-fadd.C, 1064
- test\_fsubin
  - test-fadd.C, 1064
- test\_generic\_fsytrf
  - test-fsytrf.C, 1080
- test\_impl
  - test-simd.C, 1107
- test\_LUdivine
  - test-lu.C, 1093
- test\_nullspace
  - test-nullspace.C, 1099
- TEST\_ONE\_OP
  - test-simd.C, 1105
- test\_op
  - test-simd.C, 1107
- test\_pluq
  - test-lu.C, 1094
- test\_rect\_fgesv
  - test-fgesv.C, 1073
- test\_redcoechelon
  - test-echelon.C, 1062
- test\_redrowechelon
  - test-echelon.C, 1062
- test\_rowechelon
  - test-echelon.C, 1062
- test\_RPM\_fsytrf
  - test-fsytrf.C, 1080
- test\_square\_fgesv
  - test-fgesv.C, 1073
- tfn\_minus, 753
  - operator(), 754
- tfn\_minus\_eq, 754



- operator(), 754
- tfn\_mul, 754
  - operator(), 754
- tfn\_mul\_eq, 755
  - operator(), 755
- tfn\_plus, 755
  - operator(), 755
- tfn\_plus\_eq, 755
  - operator(), 755
- tgemm
  - test-lu.C, 1096
  - test-permutations.C, 1101
- THREADS
  - benchmark-fgemm-rns.C, 775
- Threads, 756
- threads\_fgemm
  - FFPACK, 363
- threads\_ftsm
  - FFPACK, 364
- THREED
  - benchmark-fgemm-rns.C, 775
- ThreeD, 756
- THREEDA
  - benchmark-fgemm-rns.C, 775
- ThreeDAdaptive, 756
- ThreeDInPlace, 756
- THREEDIIP
  - benchmark-fgemm-rns.C, 775
- TIME
  - test-fger.C, 1071
- TIME\_CHECKER\_CHARPOLY
  - test-charpoly-check.C, 1057
- TIME\_CHECKER\_Det
  - test-det-check.C, 1059
- Timer
  - FFLAS, 76
- timer.h, 1109
- timtot
  - test-lu.C, 1096
  - test-permutations.C, 1101
- tmain
  - benchmark-fgemm-mp.C, 773
- tperm
  - test-lu.C, 1095
  - test-permutations.C, 1100
- trest
  - test-lu.C, 1096
  - test-permutations.C, 1101
- trinv\_left
  - FFPACK, 314, 367
- trinv\_left\_modular\_double
  - ffpack.C, 931
  - ffpack\_c.h, 964
- TRSMBound
  - FFLAS::Protected, 217
- TRSMHelper
  - TRSMHelper< ReclterTrait, ParSeqTrait >, 757
- TRSMHelper< ReclterTrait, ParSeqTrait >, 756
- parseq, 757
- pMMH, 757
- TRSMHelper, 757
- TTimer
  - arithprog.C, 763
  - autotune/charpoly.C, 793
  - autotune/pluq.C, 1033
  - benchmark-dgemm.C, 767
  - benchmark-dgetrf.C, 768
  - benchmark-dgetri.C, 768
  - benchmark-dsytrf.C, 769
  - benchmark-dtrsm.C, 770
  - benchmark-dtrtri.C, 771
  - fsyrk.C, 1010
  - fsytrf.C, 1011
  - fttrtri.C, 1012
  - winograd.C, 1110
- ttrsm
  - test-lu.C, 1096
  - test-permutations.C, 1101
- TWOD
  - benchmark-fgemm-rns.C, 775
- TwoD, 758
- TWODA
  - benchmark-fgemm-rns.C, 775
- TwoDAdaptive, 758
- type
  - Argument, 405
  - associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, 406
  - associatedDelayedField< const Givaro::Modular< T, X > >, 406
  - associatedDelayedField< const Givaro::ModularBalanced< T > >, 407
  - associatedDelayedField< const Givaro::ZRing< T > >, 407
  - associatedDelayedField< Field >, 405
  - CompactElement< double >, 417
  - CompactElement< Element >, 416
  - CompactElement< float >, 417
  - CompactElement< int16\_t >, 417
  - CompactElement< int32\_t >, 417
  - CompactElement< int64\_t >, 418
  - is\_simd< T >, 477
- TYPE\_BOOL
  - args-parser.h, 761
- TYPE\_DOUBLE
  - args-parser.h, 762
- TYPE\_INT
  - args-parser.h, 762
- TYPE\_INTEGER
  - args-parser.h, 762
- type\_integer
  - args-parser.h, 762
- TYPE\_INTLIST
  - args-parser.h, 762
- TYPE\_LONGLONG
  - args-parser.h, 762



- TYPE\_NONE
  - args-parser.h, [762](#)
- TYPE\_STR
  - args-parser.h, [762](#)
- type\_string
  - NoSimd< T >, [519](#)
  - Simd128\_impl< true, false, true, 4 >, [565](#)
  - Simd128\_impl< true, false, true, 8 >, [565](#)
  - Simd128\_impl< true, true, false, 2 >, [573](#)
  - Simd128\_impl< true, true, false, 4 >, [581](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [599](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd128fp\_base, [618](#)
  - Simd128i\_base, [619](#)
  - Simd256\_impl< true, true, false, 2 >, [636](#)
  - Simd256\_impl< true, true, false, 4 >, [652](#)
  - Simd256\_impl< true, true, false, 8 >, [661](#)
  - Simd256\_impl< true, true, true, 2 >, [670](#)
  - Simd256\_impl< true, true, true, 4 >, [685](#), [686](#)
  - Simd256\_impl< true, true, true, 8 >, [695](#)
  - Simd256i\_base, [696](#)
  - Simd512\_impl< true, false, true, 4 >, [697](#)
  - Simd512\_impl< true, false, true, 8 >, [703](#)
  - Simd512\_impl< true, true, false, 8 >, [712](#)
  - Simd512\_impl< true, true, true, 8 >, [722](#)
  - Simd512fp\_base, [723](#)
  - Simd512i\_base, [724](#)
- TYPE\_UINT64
  - args-parser.h, [762](#)
- TypeName
  - test-simd.C, [1105](#)
- uint16\_t
  - instrset.h, [1020](#)
- uint32\_t
  - instrset.h, [1020](#)
- uint64\_t
  - instrset.h, [1021](#)
- uint8\_t
  - instrset.h, [1020](#)
- unfit
  - FFLAS::Protected, [224](#)
- unpackhi
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [588](#)
  - Simd128\_impl< true, true, true, 2 >, [595](#)
  - Simd128\_impl< true, true, true, 4 >, [604](#)
  - Simd128\_impl< true, true, true, 8 >, [613](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [647](#)
  - Simd256\_impl< true, true, false, 8 >, [659](#)
  - Simd256\_impl< true, true, true, 2 >, [666](#)
  - Simd256\_impl< true, true, true, 4 >, [676](#)
  - Simd256\_impl< true, true, true, 8 >, [691](#)
  - Simd512\_impl< true, true, false, 8 >, [710](#)
  - Simd512\_impl< true, true, true, 8 >, [718](#)
- unpackhi\_twice
  - Simd256\_impl< true, false, true, 8 >, [623](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [647](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [666](#)
  - Simd256\_impl< true, true, true, 4 >, [676](#)
  - Simd256\_impl< true, true, true, 8 >, [690](#)
  - Simd512\_impl< true, false, true, 8 >, [700](#)
  - Simd512\_impl< true, true, false, 8 >, [709](#)
  - Simd512\_impl< true, true, true, 8 >, [717](#)
- unpacklo
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [588](#)
  - Simd128\_impl< true, true, true, 2 >, [595](#)
  - Simd128\_impl< true, true, true, 4 >, [604](#)
  - Simd128\_impl< true, true, true, 8 >, [613](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [647](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [666](#)
  - Simd256\_impl< true, true, true, 4 >, [676](#)
  - Simd256\_impl< true, true, true, 8 >, [691](#)
  - Simd512\_impl< true, true, false, 8 >, [710](#)
  - Simd512\_impl< true, true, true, 8 >, [718](#)
- unpacklo\_twice
  - Simd256\_impl< true, false, true, 8 >, [623](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [647](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [665](#)
  - Simd256\_impl< true, true, true, 4 >, [676](#)
  - Simd256\_impl< true, true, true, 8 >, [690](#)
  - Simd512\_impl< true, false, true, 8 >, [700](#)
  - Simd512\_impl< true, true, false, 8 >, [709](#)
  - Simd512\_impl< true, true, true, 8 >, [717](#)
- unpacklohi
  - Simd256\_impl< true, true, false, 2 >, [634](#)
  - Simd256\_impl< true, true, false, 4 >, [647](#)
  - Simd256\_impl< true, true, false, 8 >, [659](#)
  - Simd256\_impl< true, true, true, 2 >, [666](#)
  - Simd256\_impl< true, true, true, 4 >, [676](#)
  - Simd256\_impl< true, true, true, 8 >, [691](#)
  - Simd512\_impl< true, true, false, 8 >, [710](#)
  - Simd512\_impl< true, true, true, 8 >, [718](#)
- UnparametricTag, [758](#)
- updateD
  - FFPACK::Protected, [399](#)
- USE\_OPENMP
  - config.h, [811](#)
- UserTimer
  - FFLAS, [77](#)
- utils.h, [1109](#)
- v
  - Simd128\_impl< true, true, false, 2 >::Converter, [420](#)

- Simd128\_impl< true, true, false, 4 >::Converter, [421](#)
- Simd128\_impl< true, true, false, 8 >::Converter, [421](#)
- Simd128\_impl< true, true, true, 2 >::Converter, [422](#)
- Simd128\_impl< true, true, true, 4 >::Converter, [422](#)
- Simd128\_impl< true, true, true, 8 >::Converter, [422](#)
- Simd256\_impl< true, true, false, 2 >::Converter, [423](#)
- Simd256\_impl< true, true, false, 4 >::Converter, [423](#)
- Simd256\_impl< true, true, false, 8 >::Converter, [423](#)
- Simd256\_impl< true, true, true, 2 >::Converter, [424](#)
- Simd256\_impl< true, true, true, 4 >::Converter, [424](#)
- Simd256\_impl< true, true, true, 8 >::Converter, [425](#)
- Simd512\_impl< true, true, false, 8 >::Converter, [425](#)
- Simd512\_impl< true, true, true, 8 >::Converter, [425](#)
- val
  - Coo< Field >, [429](#)
  - Coo< ValT, IdxT >, [427](#), [430](#)
- valid
  - NoSimd< T >, [519](#)
  - Simd128\_impl< true, true, false, 2 >, [570](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [587](#)
  - Simd128\_impl< true, true, true, 2 >, [594](#)
  - Simd128\_impl< true, true, true, 4 >, [602](#)
  - Simd128\_impl< true, true, true, 8 >, [611](#)
  - Simd256\_impl< true, false, true, 8 >, [622](#)
  - Simd256\_impl< true, true, false, 2 >, [633](#)
  - Simd256\_impl< true, true, false, 4 >, [645](#)
  - Simd256\_impl< true, true, false, 8 >, [658](#)
  - Simd256\_impl< true, true, true, 2 >, [664](#)
  - Simd256\_impl< true, true, true, 4 >, [674](#), [680](#)
  - Simd256\_impl< true, true, true, 8 >, [689](#)
  - Simd512\_impl< true, false, true, 8 >, [699](#)
  - Simd512\_impl< true, true, false, 8 >, [709](#)
  - Simd512\_impl< true, true, true, 8 >, [715](#)
- VALUE
  - parallel.h, [1027](#)
- value
  - AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >, [403](#)
  - AlgoChooser< ModeT, ParSeq >, [403](#)
  - AreEqual< X, X >, [404](#)
  - AreEqual< X, Y >, [404](#)
  - compatible\_data\_type< Field >, [418](#)
  - compatible\_data\_type< Givaro::ZRing< double >, >, [418](#)
  - compatible\_data\_type< Givaro::ZRing< float >, >, [419](#)
  - ElementTraits< double >, [434](#)
  - ElementTraits< Element >, [433](#)
  - ElementTraits< FFPACK::rns\_double\_elt >, [434](#)
  - ElementTraits< float >, [434](#)
  - ElementTraits< Givaro::Integer >, [435](#)
  - ElementTraits< int16\_t >, [435](#)
  - ElementTraits< int32\_t >, [435](#)
  - ElementTraits< int64\_t >, [436](#)
  - ElementTraits< int8\_t >, [436](#)
  - ElementTraits< Reclnt::rint< K > >, [436](#)
  - ElementTraits< Reclnt::rmint< K, MG > >, [437](#)
  - ElementTraits< Reclnt::ruint< K > >, [437](#)
  - ElementTraits< uint16\_t >, [437](#)
  - ElementTraits< uint32\_t >, [438](#)
  - ElementTraits< uint64\_t >, [438](#)
  - ElementTraits< uint8\_t >, [438](#)
  - has\_minus\_eq\_impl< C >, [467](#)
  - has\_minus\_impl< C >, [468](#)
  - has\_mul\_eq\_impl< C >, [468](#)
  - has\_mul\_impl< C >, [468](#)
  - has\_operation< T >, [469](#)
  - has\_plus\_eq\_impl< C >, [469](#)
  - has\_plus\_impl< C >, [469](#)
  - is\_simd< T >, [477](#)
  - ModeTraits< Field >, [511](#)
  - ModeTraits< Givaro::Modular< Element, Compute > >, [511](#)
  - ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >, [512](#)
  - ModeTraits< Givaro::Modular< int16\_t, Compute > >, [512](#)
  - ModeTraits< Givaro::Modular< int32\_t, Compute > >, [512](#)
  - ModeTraits< Givaro::Modular< int8\_t, Compute > >, [513](#)
  - ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >, [513](#)
  - ModeTraits< Givaro::Modular< uint16\_t, Compute > >, [513](#)
  - ModeTraits< Givaro::Modular< uint32\_t, Compute > >, [514](#)
  - ModeTraits< Givaro::Modular< uint8\_t, Compute > >, [514](#)
  - ModeTraits< Givaro::ModularBalanced< Element > >, [514](#)
  - ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, [515](#)
  - ModeTraits< Givaro::ModularBalanced< int16\_t > >, [515](#)
  - ModeTraits< Givaro::ModularBalanced< int32\_t > >, [516](#)
  - ModeTraits< Givaro::ModularBalanced< int8\_t > >, [516](#)
  - ModeTraits< Givaro::Montgomery< T > >, [516](#)
  - ModeTraits< Givaro::ZRing< double > >, [517](#)

- ModeTraits< Givaro::ZRing< float > >, [517](#)
- ModeTraits< Givaro::ZRing< Givaro::Integer > >, [517](#)
- need\_field\_characteristic< Field >, [518](#)
- need\_field\_characteristic< Givaro::Modular< Field > >, [518](#)
- need\_field\_characteristic< Givaro::ModularBalanced< Field > >, [519](#)
- SimdChooser< T, false, b >, [725](#)
- SimdChooser< T, true, false >, [725](#)
- SimdChooser< T, true, true >, [726](#)
- vand
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [555](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [559](#)
  - Simd128\_impl< true, true, false, 2 >, [573](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [600](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd128i\_base, [620](#)
  - Simd256\_impl< true, false, true, 8 >, [626](#)
  - Simd256\_impl< true, true, false, 4 >, [652](#)
  - Simd256\_impl< true, true, true, 4 >, [686](#)
  - Simd512\_impl< true, true, false, 8 >, [713](#)
  - Simd512\_impl< true, true, true, 8 >, [722](#)
  - Simd512i\_base, [724](#)
- vandnot
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [556](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [560](#)
  - Simd128\_impl< true, true, false, 2 >, [574](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [600](#)
  - Simd128\_impl< true, true, true, 4 >, [609](#)
  - Simd128\_impl< true, true, true, 8 >, [618](#)
  - Simd128i\_base, [620](#)
  - Simd256\_impl< true, false, true, 8 >, [626](#)
  - Simd256\_impl< true, true, false, 4 >, [652](#)
  - Simd256\_impl< true, true, true, 4 >, [686](#)
  - Simd512\_impl< true, true, false, 8 >, [713](#)
  - Simd512\_impl< true, true, true, 8 >, [722](#)
  - Simd512i\_base, [724](#)
- VEC\_ADD
  - FFLAS::vectorised, [286](#)
- VEC\_SUB
  - FFLAS::vectorised, [287](#)
- vect\_size
  - FieldSimd< \_Field >, [448](#)
  - NoSimd< T >, [520](#)
  - Simd128\_impl< true, true, false, 2 >, [574](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
- Simd128\_impl< true, true, false, 8 >, [592](#)
- Simd128\_impl< true, true, true, 2 >, [600](#)
- Simd128\_impl< true, true, true, 4 >, [609](#)
- Simd128\_impl< true, true, true, 8 >, [618](#)
- Simd256\_impl< true, false, true, 8 >, [627](#)
- Simd256\_impl< true, true, false, 2 >, [636](#)
- Simd256\_impl< true, true, false, 4 >, [653](#)
- Simd256\_impl< true, true, false, 8 >, [661](#)
- Simd256\_impl< true, true, true, 2 >, [670](#)
- Simd256\_impl< true, true, true, 4 >, [686](#)
- Simd256\_impl< true, true, true, 8 >, [695](#)
- Simd512\_impl< true, false, true, 8 >, [703](#)
- Simd512\_impl< true, true, false, 8 >, [713](#)
- Simd512\_impl< true, true, true, 8 >, [723](#)
- vect\_t
  - FieldSimd< \_Field >, [443](#)
  - NoSimd< T >, [519](#)
  - Simd128\_impl< true, true, false, 2 >, [567](#)
  - Simd128\_impl< true, true, false, 4 >, [576](#)
  - Simd128\_impl< true, true, false, 8 >, [585](#)
  - Simd128\_impl< true, true, true, 2 >, [593](#)
  - Simd128\_impl< true, true, true, 4 >, [602](#)
  - Simd128\_impl< true, true, true, 8 >, [611](#)
  - simd128\_int64.inl, [1049](#)
  - Simd128i\_base, [619](#)
  - Simd256\_impl< true, false, true, 8 >, [622](#)
  - Simd256\_impl< true, true, false, 2 >, [629](#)
  - Simd256\_impl< true, true, false, 4 >, [639](#), [640](#)
  - Simd256\_impl< true, true, false, 8 >, [655](#)
  - Simd256\_impl< true, true, true, 2 >, [663](#)
  - Simd256\_impl< true, true, true, 4 >, [673](#)
  - Simd256\_impl< true, true, true, 8 >, [688](#)
  - simd256\_int64.inl, [1052](#)
  - Simd256i\_base, [696](#)
  - Simd512\_impl< true, false, true, 8 >, [698](#)
  - Simd512\_impl< true, true, false, 8 >, [706](#)
  - Simd512\_impl< true, true, true, 8 >, [715](#)
  - simd512\_int64.inl, [1054](#)
  - Simd512i\_base, [724](#)
- verification\_PLUQ
  - benchmark-pluq.C, [788](#)
- verifPLUQ
  - test-lu.C, [1094](#)
- VERSION
  - config.h, [811](#)
- vor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [555](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [559](#)
  - Simd128\_impl< true, true, false, 2 >, [573](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [600](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd128i\_base, [620](#)

- Simd256\_impl< true, false, true, 8 >, [626](#)
- Simd256\_impl< true, true, false, 4 >, [652](#)
- Simd256\_impl< true, true, true, 4 >, [686](#)
- Simd512\_impl< true, true, false, 8 >, [712](#)
- Simd512\_impl< true, true, true, 8 >, [722](#)
- Simd512i\_base, [724](#)
- vxor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [555](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [560](#)
  - Simd128\_impl< true, true, false, 2 >, [574](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [600](#)
  - Simd128\_impl< true, true, true, 4 >, [609](#)
  - Simd128\_impl< true, true, true, 8 >, [618](#)
  - Simd128i\_base, [620](#)
  - Simd256\_impl< true, false, true, 8 >, [626](#)
  - Simd256\_impl< true, true, false, 4 >, [652](#)
  - Simd256\_impl< true, true, true, 4 >, [686](#)
  - Simd512\_impl< true, true, false, 8 >, [713](#)
  - Simd512\_impl< true, true, true, 8 >, [722](#)
  - Simd512i\_base, [724](#)
- WAIT
  - parallel.h, [1026](#)
- Winograd, [758](#)
  - FFLAS::BLAS3, [198](#)
- winograd.C, [1110](#)
  - balanced, [1110](#), [1111](#)
  - DOUBLE\_TO\_FLOAT\_CROSSOVER, [1110](#)
  - GFOPS, [1110](#)
  - main, [1111](#)
  - TTimer, [1110](#)
- Winograd\_L\_S
  - FFLAS::BLAS3, [201](#)
- Winograd\_LR\_S
  - FFLAS::BLAS3, [201](#)
- Winograd\_R\_S
  - FFLAS::BLAS3, [202](#)
- WinogradAcc\_2\_24
  - FFLAS::BLAS3, [199](#)
- WinogradAcc\_2\_27
  - FFLAS::BLAS3, [199](#)
- WinogradAcc\_3\_21
  - FFLAS::BLAS3, [199](#)
- WinogradAcc\_3\_23
  - FFLAS::BLAS3, [198](#)
- WinogradAcc\_L\_S
  - FFLAS::BLAS3, [201](#)
- WinogradAcc\_LR
  - FFLAS::BLAS3, [200](#)
- WinogradAcc\_R\_S
  - FFLAS::BLAS3, [200](#)
- WinogradCalc
  - FFLAS::Protected, [219](#)
- WinogradPar, [758](#)
- WinogradSteps
  - FFLAS::Protected, [218](#)
- WinogradThreshold
  - FFLAS::Protected, [217](#), [218](#)
- WinoPar
  - FFLAS::BLAS3, [197](#)
- WINOTHRESHOLD
  - fflas.h, [840](#)
- WRITE
  - parallel.h, [1026](#)
- write
  - RNSInteger< RNS >, [546](#)
  - RNSIntegerMod< RNS >, [551](#)
- write\_field
  - Matio.h, [1024](#)
- write\_matrix
  - benchmark-fgemv-mp.C, [777](#)
  - RNSIntegerMod< RNS >, [551](#)
- write\_matrix\_long
  - RNSIntegerMod< RNS >, [552](#)
- writeCommandString
  - FFLAS, [191](#)
- writeDnsFormat
  - FFLAS, [154](#)
- WriteMatrix
  - FFLAS, [191](#), [193](#)
- WritePermutation
  - FFLAS, [193](#)
- zero
  - FFLAS, [79](#)
  - FieldSimd< \_Field >, [445](#), [446](#)
  - RNSInteger< RNS >, [546](#)
  - RNSIntegerMod< RNS >, [553](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [555](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [559](#)
  - Simd128\_impl< true, true, false, 2 >, [573](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [599](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd128i\_base, [619](#)
  - Simd256\_impl< true, false, true, 8 >, [622](#)
  - Simd256\_impl< true, true, false, 2 >, [636](#)
  - Simd256\_impl< true, true, false, 4 >, [652](#)
  - Simd256\_impl< true, true, false, 8 >, [661](#)
  - Simd256\_impl< true, true, true, 2 >, [670](#)
  - Simd256\_impl< true, true, true, 4 >, [685](#), [686](#)
  - Simd256\_impl< true, true, true, 8 >, [695](#)
  - Simd256i\_base, [696](#)
  - Simd512\_impl< true, false, true, 8 >, [699](#)
  - Simd512\_impl< true, true, false, 8 >, [712](#)
  - Simd512\_impl< true, true, true, 8 >, [722](#)
  - Simd512i\_base, [724](#)
- ZOSparseMatrix

FFLAS, [74](#)