| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [PatternProcessor/src](http://docs.google.com/index.html) - PatternExploder.cpp (source / [functions](http://docs.google.com/PatternExploder.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 98 | 122 | 80.3 % | | Date: | 2012-03-23 |  | Functions: | 11 | 11 | 100.0 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 48 | 74 | 64.9 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : :   2 : : #include "PatternExploder.h"  3 : : #include "pattern.h"  4 : : #include "Logger.h"  5 : : #include "ErrorHandler.h"  6 : : #include "Sgs\_Define.h"  7 : : #include "SDDB.h"  8 : : #include "print.h"  9 : : #include "HealthCheckCounter.h"  10 : :   11 : : namespace nft {  12 : :   13 : 2311 : PatternExploder::PatternExploder() {  14 : : }  15 : :   16 : : //This will be called after vsbGetEP is called  17 : : void  18 : : PatternExploder::getParsedPattern (  19 : : GroupContext \*inPgc,   20 : : std::vector<PatternPosition> &outPositions,  21 : : EP ep,   22 : : int Nx, int Ny,   23 : : int Xe, int Ye,  24 : : const IntPosition &inCellPosition,  25 : : ParsedPattern &outParsedPattern) throw(DPL::Error \*)  26 : 5983 : {  27 : 5983 : HealthCheckCounter::instance()->incrCounter(  28 : : HealthCheckCounter::eUpdateFrequent);  29 : :   30 : 5983 : int32\_t Sx, Sy;  31 : :   32 : 5983 : outParsedPattern.setRepetition(InputPatternSize(Nx, Ny));  33 : 5983 : outParsedPattern.setPitch(PatternPosition(0, 0));   34 : : //Keeping the original Lx and Ly as slicing   35 : : //depends on original lengths   36 : 5983 : outParsedPattern.setSize(InputPatternSize(ep.L1, ep.L2));  37 : 5983 : outParsedPattern.setAI(inPgc->attributeInfo);  38 : 5983 : outParsedPattern.setCode(ep.PatCode);  39 : : //Map for special pattern codes  40 : 5983 : outParsedPattern.mapPatCode(ep.PatCode);  41 : 5983 : try  42 : : {  43 [ + + ][ + + ]: 5983 : switch (inPgc->ppc.WhichContext)  [ - + ][ + + ]  [ - + ]  44 : : {  45 : : case Normal:   46 : 5417 : outPositions.push\_back(  47 : : PatternPosition(ep.X0 + inCellPosition.getX(),  48 : : ep.Y0 + inCellPosition.getY()));  49 : 5417 : parseNR( outParsedPattern, inCellPosition,  50 : : inPgc, ep, Nx, Ny, Xe, Ye);  51 : 5417 : break;  52 : : case Optimize:   53 : 30 : outPositions.push\_back(  54 : : PatternPosition(ep.X0 + inCellPosition.getX(),  55 : : ep.Y0 + inCellPosition.getY()));  56 : 30 : parseOR( outParsedPattern, inCellPosition,  57 : : outPositions, inPgc, ep, Nx, Ny, Xe, Ye);  58 : 30 : break;  59 : : case Array:   60 : 115 : parseAR( outParsedPattern, inCellPosition,  61 : : outPositions, inPgc, ep, Nx, Ny, Xe, Ye);  62 : 115 : break;  63 : : case Composite:   64 : 421 : parseCR( outParsedPattern, inCellPosition,  65 : : outPositions, inPgc, ep, Nx, Ny, Xe, Ye);  66 : 421 : break;  67 : : default:  68 : 0 : break;  69 : : }  70 : : }  71 [ # # ]: 0 : catch( DPL::Error \*pChild )  72 : : {  73 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  74 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  75 : : ErrorHandler::eErrFunction,  76 : : pErr->getErrorMsg(  77 : : ErrorHandler::eErrFunction, "PatternExploder::parse\*"));  78 : : }  79 [ # # ]: 5983 : return;  80 : : }  81 : :   82 : : void  83 : : PatternExploder::parseNR ( ParsedPattern &outParsedPattern,  84 : : const IntPosition &inCellPosition,  85 : : GroupContext \*inPgc,  86 : : const EP &ep,  87 : : uint32\_t Nx,  88 : : uint32\_t Ny,  89 : : uint32\_t Xe,  90 : : uint32\_t Ye)   91 : 5921 : {  92 : 0 : return;  93 : : }  94 : :   95 : : void  96 : : PatternExploder::parseAR ( ParsedPattern &outParsedPattern,  97 : : const IntPosition &inCellPosition,  98 : : std::vector<PatternPosition> &outPositions,  99 : : GroupContext \*inPgc,  100 : : const EP &ep,  101 : : uint32\_t Nx,  102 : : uint32\_t Ny,  103 : : uint32\_t Xe,  104 : : uint32\_t Ye)   105 : 438 : {  106 : 438 : uint32\_t i, j;   107 : 438 : int xOtherPos, yOtherPos;  108 : : //Probably not needed as calculated in vsbGetEP   109 [ + + ][ + - ]: 438 : double pitchX = (Nx - 1)? (static\_cast<double>(Xe - ep.X0))/(Nx - 1) : 0;  [ + - ]  110 [ + + ][ + - ]: 438 : double pitchY = (Ny - 1)? (static\_cast<double>(Ye - ep.Y0))/(Ny - 1) : 0;  [ + - ]  111 : 438 : outParsedPattern.setPitch(PatternPosition(static\_cast<int>(pitchX), static\_cast<int>(pitchY)));  112 : 438 : CRContext \*crctx = &(inPgc->ppc.crctx) ;  113 : :   114 : : //Special handling for Linear Array  115 [ + + ][ + + ]: 438 : if(inPgc->ppc.arctx.PatArrangement == nft::Linear && crctx->CRType!=CRArray){  116 : 21 : PP pp;  117 : 21 : int vsbretval=-1;  118 : 21 : outPositions.push\_back(  119 : : PatternPosition(ep.X0 + inCellPosition.getX(),  120 : : ep.Y0 + inCellPosition.getY()));  121 : : //#ifdef ERROR\_SEED  122 : : // vsbretval = -2;  123 : : // while (-1 != vsbretval ) //<<Testcase\_Id:PatternExploder\_parserAR>>  124 : : //#else  125 : : //Now iterate over all the PPs if not CR Array  126 : 21 : while (-1 != (vsbretval = vsbGetPP(inPgc, &pp)) )  127 : : //#endif   128 : : {  129 [ - + ]: 124 : if (-1 > vsbretval)  130 : : {  131 : 0 : Logger::instance()->log(LogMessageTable::ePATTERN\_ERROR\_OR,  132 : : "Incorrect parsing of Linear Array in vsbGetPP");  133 : :   134 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  135 : 0 : pErr->errorAdd(NULL, (char\*)\_\_FILE\_\_, (char\*)\_\_FUNCTION\_\_, (int)\_\_LINE\_\_  136 : : ,ErrorHandler::eErrFunction,pErr->getErrorMsg((ErrorHandler::eErrFunction),"vsbGetPP") );  137 : : }  138 : : else  139 : : {  140 : : //Add PPs to the vector  141 : : PatternPosition patPos(pp.X0 + inCellPosition.getX(),  142 : 124 : pp.Y0 + inCellPosition.getY());  143 : 124 : e9k\_debug("Found pattern at position %d %d\n",  144 : : patPos.getX(), patPos.getY() );  145 : 124 : outPositions.push\_back(patPos);  146 : : }  147 : : }  148 : 21 : return;  149 : : }  150 : :   151 [ + - ]: 417 : if (Nx <= 1 && Ny <= 1)  152 : : {  153 : 417 : parseNR( outParsedPattern, inCellPosition,  154 : : inPgc, ep, Nx, Ny, Xe, Ye);  155 : 417 : return;  156 : : }else{  157 : 417 : SDDB \*pSDDB = SDDB::instance();  158 : 417 : try  159 : : {  160 : 417 : pSDDB->arrayPatternFound(ep, Nx, Ny, Xe, Ye, inCellPosition);  161 : : }  162 [ # # ]: 0 : catch( DPL::Error \*pChild )  163 : : {  164 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  165 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  166 : : ErrorHandler::eErrFunction,  167 : : pErr->getErrorMsg(  168 : : ErrorHandler::eErrFunction,  169 : : "SDDB::arrayPatternFound"));  170 : : }  171 : :   172 [ + - ][ + + ]: 1552 : for (i = 0; i < Nx; i++)  173 : : {  174 : : // Starting point of the ith element along the x direction  175 [ + - ]: 1135 : xOtherPos = (ep.X0 + static\_cast<int>(i \* pitchX + 0.5));   176 [ + - ][ + + ]: 2938 : for(j =0; j< Ny; j++)  177 : : {  178 : : // starting point of the j the element  179 [ + - ]: 1803 : yOtherPos = (ep.Y0 + static\_cast<int>(j \* pitchY + 0.5));   180 : : //Add other array pattern positions to the vector  181 : 1803 : outPositions.push\_back( PatternPosition(  182 : : xOtherPos + inCellPosition.getX(),  183 : : yOtherPos + inCellPosition.getY()));  184 : : }  185 : : }  186 : 417 : try  187 : : {  188 : 417 : pSDDB->patternFlatteningCompleted(outParsedPattern, outPositions);  189 : : }  190 [ # # ]: 0 : catch( DPL::Error \*pChild )  191 : : {  192 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  193 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  194 : : ErrorHandler::eErrFunction,  195 : : pErr->getErrorMsg(  196 : : ErrorHandler::eErrFunction,  197 : : "SDDB::patternFlatteningCompleted"));  198 : : }  199 : :   200 : : }  201 : 438 : return;  202 : : }  203 : :   204 : : void  205 : : PatternExploder::parseOR ( ParsedPattern &outParsedPattern,  206 : : const IntPosition &inCellPosition,  207 : : std::vector<PatternPosition> &outPositions,  208 : : GroupContext \*inPgc,  209 : : const EP &ep,  210 : : uint32\_t Nx,  211 : : uint32\_t Ny,  212 : : uint32\_t Xe,  213 : : uint32\_t Ye)  214 : 30 : {  215 : 30 : PP pp;  216 : 30 : int vsbretval=-1;  217 : :   218 : : //Now iterate over all the PPs  219 : : //#ifdef ERROR\_SEED //<<Testcase\_Id:PatternExploder\_parseOR>>  220 : : // vsbretval = -2;  221 : : // while (-1 != vsbretval )  222 : : //#else   223 : 30 : while (-1 != (vsbretval = vsbGetPP(inPgc, &pp)) )  224 : : //#endif   225 : : {  226 [ - + ]: 41 : if (-1 > vsbretval)  227 : : {  228 : 0 : Logger::instance()->log(LogMessageTable::ePATTERN\_ERROR\_OR,  229 : : "Incorrect parsing of OR in vsbGetPP");  230 : :   231 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  232 : 0 : pErr->errorAdd(NULL, (char\*)\_\_FILE\_\_, (char\*)\_\_FUNCTION\_\_, (int)\_\_LINE\_\_  233 : : ,ErrorHandler::eErrFunction,pErr->getErrorMsg((ErrorHandler::eErrFunction),"vsbGetPP") );  234 : : }  235 : : else  236 : : {  237 : : //Add PPs to the vector  238 : : PatternPosition patPos(pp.X0 + inCellPosition.getX(),  239 : 41 : pp.Y0 + inCellPosition.getY());  240 : 41 : e9k\_debug("Found pattern at position %d %d\n",  241 : : patPos.getX(), patPos.getY() );  242 : 41 : outPositions.push\_back(patPos);  243 : : }  244 : : }  245 : : return;  246 : : }  247 : :   248 : : void  249 : : PatternExploder::parseCR ( ParsedPattern &outParsedPattern,  250 : : const IntPosition &inCellPosition,  251 : : std::vector<PatternPosition> &outPositions,  252 : : GroupContext \*inPgc,  253 : : const EP &ep,  254 : : uint32\_t Nx,  255 : : uint32\_t Ny,  256 : : uint32\_t Xe,  257 : : uint32\_t Ye)   258 : 421 : {  259 : : //A CR pattern contains a list of pattern definitions  260 : : //This list of patterns is enclosed by a bounding box.  261 : : //This BB is placed at different locations identified  262 : : //by the list of pattern positions.  263 : 421 : CRContext \*crctx = &(inPgc->ppc.crctx) ;  264 : 421 : EP ep\_or = ep;  265 : 421 : PP pp;  266 : 421 : pp.X0 = ep\_or.X0;  267 : 421 : pp.Y0 = ep\_or.Y0;  268 : : //#ifdef ERROR\_SEED //<<Testcase\_Id:PatternExploder\_parseCR>>  269 : : // crctx->CRType = static\_cast<nft::CRContextType>(10);  270 : : //#endif   271 : 421 : int vsbretval = -1;  272 [ + + ][ + + ]: 421 : switch (crctx->CRType)  [ - + ][ - + ]  273 : : {  274 : : case CRArray:   275 [ - + ]: 236 : if (Nx <= 1 && Ny <= 1)  276 : : {  277 : 0 : parseCRSubContext(outParsedPattern, inCellPosition,  278 : : outPositions, inPgc, ep, Nx, Ny, Xe, Ye);  279 : : }  280 : : else  281 : : {  282 : : //This array of patterns may actually be an array of  283 : : //1. NR  284 : : //2. OR  285 : : //3. AR  286 : 236 : parseAR(outParsedPattern, inCellPosition,  287 : : outPositions, inPgc, ep, Nx, Ny, Xe, Ye);  288 : : }  289 : 421 : break;  290 : : case CRLinear:  291 : : case CROptimize:  292 : 287 : do  293 : : {  294 : 472 : ep\_or.X0 = pp.X0;  295 : 472 : ep\_or.Y0 = pp.Y0;   296 : 472 : parseCRSubContext(outParsedPattern, inCellPosition,  297 : : outPositions, inPgc, ep\_or, Nx, Ny, Xe, Ye);   298 : : //Add PPs to the vector  299 : : PatternPosition patPos(ep\_or.X0 + inCellPosition.getX()  300 : 472 : , inCellPosition.getY() + ep\_or.Y0);  301 : 472 : e9k\_debug("Found pattern at position %d %d\n",  302 : : patPos.getX(), patPos.getY() );  303 : 472 : outPositions.push\_back(patPos);  304 [ + + ]: 472 : if(Array == crctx->WhichSubContext)  305 : : {  306 : 472 : break;  307 : : }  308 : : else  309 : : {  310 : : }  311 : :   312 [ + + ]: 385 : } while(-1 != (vsbretval = vsbGetPP(inPgc, &pp)) );   313 : 98 : break;  314 : : default:  315 : 0 : Logger::instance()->log(LogMessageTable::eUTIL\_ERROR\_GENERAL\_MESSAGE,  316 : : "Incorrect ContextType in parseCR()");  317 : :   318 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  319 : 0 : pErr->errorAdd(NULL, (char\*)\_\_FILE\_\_, (char\*)\_\_FUNCTION\_\_, (int)\_\_LINE\_\_  320 : : ,ErrorHandler::eErrMessageString  321 : : ,pErr->getErrorMsg((ErrorHandler::eErrMessageString)  322 : : ,"Incorrect ContextType in parseCR()"));  323 : 421 : break;  324 : : }  325 : :   326 : : }  327 : :   328 : : void  329 : : PatternExploder::parseCRSubContext ( ParsedPattern &outParsedPattern,  330 : : const IntPosition &inCellPosition,  331 : : std::vector<PatternPosition> &outPositions,  332 : : GroupContext \*inPgc,  333 : : const EP &ep,  334 : : uint32\_t Nx,  335 : : uint32\_t Ny,  336 : : uint32\_t Xe,  337 : : uint32\_t Ye  338 : : )  339 : :   340 : 472 : {  341 : 472 : CRContext \*crctx = &(inPgc->ppc.crctx) ;  342 : : //#ifdef ERROR\_SEED //<<Testcase\_Id:PatternExploder\_parseCRSubContext>>  343 : : // crctx->WhichSubContext=static\_cast<nft::ContextType>(4);  344 : : //#endif   345 [ + + ][ + + ]: 472 : switch(crctx->WhichSubContext)  [ - + ][ + - ]  346 : : {  347 : : case Normal:   348 : 87 : parseNR(outParsedPattern, inCellPosition,  349 : : inPgc, ep, Nx, Ny, Xe, Ye);  350 : 87 : break;  351 : : case Optimize:   352 : : //Handled before  353 : 87 : break;  354 : : case Array:   355 : 87 : parseAR(outParsedPattern, inCellPosition,  356 : : outPositions, inPgc, ep, Nx, Ny, Xe, Ye);  357 : 87 : break;  358 : : default:  359 : 0 : Logger::instance()->log(LogMessageTable::eUTIL\_GENERAL\_MESSAGE,  360 : : "Invalid context in CR pattern");  361 : :   362 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  363 : 0 : pErr->errorAdd(NULL, (char\*)\_\_FILE\_\_, (char\*)\_\_FUNCTION\_\_, (int)\_\_LINE\_\_  364 : : ,ErrorHandler::eErrMessageString  365 : : ,pErr->getErrorMsg((ErrorHandler::eErrMessageString)  366 : : ,"Invalid context in CR pattern"));  367 : 472 : break;  368 : : }  369 : : }  370 : 2311 : PatternExploder::~PatternExploder() {  371 : : }  372 : :   373 : 1215 : } // namespace nft |

|  |
| --- |
| Generated by: [LCOV version 1.9](http://ltp.sourceforge.net/coverage/lcov.php) |