| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [LibVsb/src](http://docs.google.com/index.html) - pattern.cpp (source / [functions](http://docs.google.com/pattern.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 33 | 36 | 91.7 % | | Date: | 2012-03-23 |  | Functions: | 3 | 3 | 100.0 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 18 | 24 | 75.0 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   2 : : \* Copyrighted 2007 by Interra Systems, USA  3 : : \* All rights reserved   4 : : \*  5 : : \* Date of Creation: Fri Jul 20 15:53:41 IST 2007  6 : : \* File Name: pattern.cpp  7 : : \* Purpose:   8 : : \* Author:   9 : : \*  10 : : \* Revision History:   11 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/   12 : :   13 : : /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  14 : : \* Header Include section   15 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  16 : : #include <stdio.h>  17 : : #include "pattern.h"  18 : : #include "vsb\_utils.h"  19 : : //#include "Error.h"  20 : : #include "SDDB.h"  21 : : #include "print.h"  22 : : #include "HealthCheckCounter.h"  23 : : namespace nft {  24 : : /\* Globals \*/  25 : :   26 : : /\* Private Decelerations \*/  27 : :   28 : : extern bool selfDiag\_PMD ;  29 : : extern "C" int NRGetEP(GroupContext \*pgc, EP \*ep, int \*Nx, int \*Ny, int \*Xe, int \*Ye) ;  30 : : extern "C" int ORGetEP(GroupContext \*pgc, EP \*ep, int \*Nx, int \*Ny, int \*Xe, int \*Ye) ;  31 : : extern "C" int ARGetEP(GroupContext \*pgc, EP \*ep, int \*Nx, int \*Ny, int \*Xe, int \*Ye) ;  32 : : extern "C" int CRGetEP(GroupContext \*pgc, EP \*ep, int \*Nx, int \*Ny, int \*Xe, int \*Ye) ;   33 : :   34 : : extern "C" int ORGetPP(GroupContext \*pgc, PP \*pp) ;  35 : : extern "C" int ARGetPP(GroupContext \*pgc, PP \*pp) ;  36 : : extern "C" int CRGetPP(GroupContext \*pgc, PP \*pp) ;   37 : :   38 : : #if (!defined(\_\_SUNPRO\_CC) || \_\_SUNPRO\_CC < 0x530)  39 : : typedef int (\* EPFunction)(GroupContext \*, EP \*, int \*, int \*, int \*, int \*) ;  40 : : typedef int (\* PPFunction)(GroupContext \*, PP \*) ;   41 : : #else  42 : : extern "C" typedef int (\* EPFunction)(GroupContext \*, EP \*, int \*, int \*, int \*, int \*) ;  43 : : extern "C" typedef int (\* PPFunction)(GroupContext \*, PP \*) ;   44 : : #endif  45 : :   46 : : static EPFunction WhichEPFunction[5]={NULL, NRGetEP, ORGetEP, ARGetEP, CRGetEP} ;  47 : : static PPFunction WhichPPFunction[5]={NULL, NULL, ORGetPP, ARGetPP, CRGetPP} ;  48 : :   49 : : static const int padding = 4 ;  50 : :   51 : :   52 : : /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  53 : : \* Name: vsbGetEP  54 : : \* Functionality: Calls the required representation EP extraction function based  55 : : \* on representation type.  56 : : \* Input: Pointers to GroupContext, EP, Number of X Repetitions, Number of Y   57 : : \* Repetitions, Array end position X direction, Array end position Y direction  58 : : \* Output: An Elementary pattern  59 : : \* Return value: Success or failure of the operation  60 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  61 : : int vsbGetEP(GroupContext \*pgc, EP \*ep, int \*Nx, int \*Ny, int \*Xe, int \*Ye)   62 : 7268 : {  63 : 7268 : HealthCheckCounter::instance()->incrCounter(  64 : : HealthCheckCounter::eUpdateFrequent);  65 : 7268 : ContextType ctype ;  66 : 7268 : int status = -1 ;  67 : : //char code = 0 ;  68 : :   69 [ + + ]: 7268 : if(pgc->current + padding <= pgc->end)  70 : : {  71 [ + + ]: 6062 : if(NoContext == pgc->ppc.WhichContext)  72 : : {  73 : 1300 : ctype = GetContextType(pgc->current) ;  74 : 1300 : pgc->ppc.WhichContext = ctype ;  75 : :   76 [ + - ][ + + ]: 1300 : switch(ctype)  [ - + ]  77 : : {  78 : : case Normal :   79 : : case Optimize :  80 : 1250 : case Array : pgc->ppc.nrctx.isEmpty = true ; break ;  81 : 50 : case Composite : pgc->ppc.crctx.isEmpty = true ; break ;  82 : : default:  83 : 0 : return -1;  84 : : }  85 : : }   86 : : else  87 : : {  88 : 4762 : ctype = pgc->ppc.WhichContext ;  89 : : }  90 : 6062 : status = WhichEPFunction[ctype](pgc, ep, Nx, Ny, Xe, Ye) ;  91 : :   92 [ - + ]: 6030 : if( -1 > status )  93 : 0 : return status ;  94 : :   95 [ + + ]: 6030 : if(-1 == status)  96 : : {  97 : 449 : pgc->ppc.WhichContext = NoContext ;  98 : :   99 [ + + ]: 449 : if(pgc->current + padding <= pgc->end)  100 : : {  101 : 402 : pgc->ppc.WhichContext = GetContextType(pgc->current) ;  102 : :   103 [ + - ][ + + ]: 402 : switch(pgc->ppc.WhichContext)  [ - + ]  104 : : {  105 : : case Normal :   106 : : case Optimize :  107 : 393 : case Array : pgc->ppc.nrctx.isEmpty = true ; break ;  108 : 9 : case Composite : pgc->ppc.crctx.isEmpty = true ; break ;  109 : : default:  110 : 0 : return -1;  111 : : }  112 : 402 : status = WhichEPFunction[pgc->ppc.WhichContext](pgc, ep, Nx, Ny, Xe, Ye) ;  113 : :   114 : : }  115 : :   116 : : }   117 : : }  118 : 7236 : return status ;  119 : : }  120 : :   121 : : /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  122 : : \* Name: vsbGetPP  123 : : \* Functionality: Calls the required representation PP extraction function based  124 : : \* on Context type.  125 : : \* Input: Pointers to GroupContext and PP   126 : : \* Output: A pattern position  127 : : \* Return value: Success or failure of the operation  128 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  129 : : int vsbGetPP(GroupContext \*pgc, PP \*pp)  130 : 601 : {  131 : 601 : int status = -1 ;  132 : :   133 [ + - ]: 601 : switch(pgc->ppc.WhichContext)  134 : : {  135 : 601 : case NoContext : break ;  136 : 601 : case Normal : break ;  137 : 601 : default : status = WhichPPFunction[pgc->ppc.WhichContext](pgc, pp) ;  138 : 601 : break ;  139 : : }  140 : :   141 : 601 : return status ;  142 : : }  143 : 1215 : } //namespace nft |

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