| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [LibVsb/src](http://docs.google.com/index.html) - vsb\_patternOR.cpp (source / [functions](http://docs.google.com/vsb_patternOR.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 73 | 95 | 76.8 % | | Date: | 2012-03-23 |  | Functions: | 3 | 3 | 100.0 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 19 | 28 | 67.9 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : /\*--------------------------------------------------------------------------  2 : : \* Copyright (c) 2003, NuFlare Technology Inc., Digital GlobalSoft Ltd.  3 : : \* All Rights Reserved.  4 : : \* These technical data are subject to export control laws of Japan  5 : : \* regulations, and diversion contrary thereto is prohibited.  6 : : \*   7 : : \* Module : VSB Optimize Representation Extraction Routines  8 : : \* Design Document : VSB Library - High Level Design (Rev 0.76)  9 : : \* Author : santhosh.kudva@digital.com  10 : : \* Date : 14th April, 2003  11 : : \*   12 : : \* File names prefixed with vsb\_ contain routines used privately by the library   13 : : \*   14 : : \* -------------------------------------------------------------------------\*/  15 : :   16 : : #include <stdio.h>  17 : : #include "dec\_types.h"  18 : : //#include "vsb\_types.h"  19 : : #include "vsb\_utils.h"  20 : : #include "vsb\_translate.h"  21 : : #include "context.h"  22 : : #include "PH4Flag.h"  23 : : #include "StrictAliasUnion.h"  24 : : #include "SDDB.h"  25 : : namespace nft {  26 : : /\* Private Decelerations \*/  27 : :   28 : : extern "C" int ORGetEP(GroupContext \*pgc, EP \*ep, int \*Nx, int \*Ny, int \*Xe, int \*Ye) ;  29 : : extern "C" int ORGetPP(GroupContext \*pgc, PP \*pp) ;  30 : :   31 : :   32 : : /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  33 : : \* Name: ORGetEP  34 : : \* Functionality: Extracts one EP from a OR2 Representation (OR2:PH1(NPEPiPP+)+)  35 : : \* Input: Pointers to GroupContext, EP, Number of X Repetitions, Number of Y   36 : : \* Repetitions, Array end position X direction, Array end position Y direction  37 : : \* Output: Returns an EP as a Array Representation, Nx=Ny=1  38 : : \* Return value: Success or failure of the operation  39 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  40 : : int ORGetEP(GroupContext \*pgc, EP \*ep, int \*Nx, int \*Ny, int \*Xe, int \*Ye)  41 : 182 : {  42 : 182 : ORContext \*orctx = NULL ;  43 : : //vsbPH1 \*ph1 = NULL ;  44 : 182 : vsbPH1 ph1 ;  45 : 182 : int SkipSize = 0 ;  46 : 182 : alias\_PH4\_1 alias\_PH4\_1Val;  47 : 182 : alias\_PH4\_2 alias\_PH4\_2Val;  48 : 182 : alias\_PH1 alias\_PH1Val;  49 : :   50 : 182 : orctx = &(pgc->ppc.orctx) ;  51 : :   52 [ + + ]: 182 : if(orctx->isEmpty)  53 : : {  54 : : //ph1 = (vsbPH1 \*)pgc->current ;  55 : 102 : uint temp ;  56 : : //temp = vsbGetInt(pgc->current) ;  57 : 102 : alias\_PH4\_1Val.value = alias\_PH4\_2Val.value = vsbGetInt(pgc->current) ;  58 : : //PH4 handling  59 : 102 : if(1 == PH4Flag::instance()->getValue())  60 : : {  61 : 0 : vsbPH4\_1 ph4;  62 : : //ph4 = \*(vsbPH4\_0 \*)(&temp) ;  63 : 0 : ph4 = alias\_PH4\_1Val.ph4 ;  64 [ # # ][ # # ]: 0 : switch(ph4.b)  65 : : {  66 : : case 0xb1:  67 : : {  68 : 0 : vsbPH4\_1 ph4\_1;  69 : : //ph4\_1= \*(vsbPH4\_1 \*)(&temp);  70 : 0 : ph4\_1 = alias\_PH4\_1Val.ph4 ;  71 : 0 : pgc->attributeInfo = ph4\_1.AI1;  72 : 0 : pgc->current = pgc->current + sizeof(vsbPH4\_1);  73 : 0 : break;  74 : : }  75 : : case 0xb2:  76 : : {  77 : 0 : vsbPH4\_2 ph4\_2;  78 : : //ph4\_2 = \*(vsbPH4\_2 \*)(&temp);  79 : 0 : ph4\_2 = alias\_PH4\_2Val.ph4 ;  80 : 0 : pgc->attributeInfo = ph4\_2.AI2;  81 : 0 : pgc->current = pgc->current + sizeof(vsbPH4\_2);  82 : 0 : break;  83 : : }  84 : : }  85 : : }  86 : : else  87 : : {  88 : 102 : pgc->attributeInfo=0;  89 : : }  90 : :   91 : : //temp = vsbGetInt(pgc->current) ;  92 : 102 : alias\_PH1Val.value = vsbGetInt(pgc->current) ;  93 : : //ph1 = \*(vsbPH1 \*)(&temp) ;  94 : 102 : ph1 = alias\_PH1Val.ph1;  95 : 102 : orctx->PatCode = ph1.code1 ;  96 : 102 : try{  97 : 102 : checkPatternCode(orctx->PatCode);  98 : : }  99 [ # # ]: 0 : catch(DPL::Error \*e)  100 : : {  101 : 0 : char msg[1024];  102 : 0 : sprintf(msg,"Invalid Pattern code 0x%x\n",orctx->PatCode);  103 : :   104 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  105 : 0 : pErr->errorAdd(NULL, (char\*)\_\_FILE\_\_, (char\*)\_\_FUNCTION\_\_, (int)\_\_LINE\_\_  106 : : ,ErrorHandler::eErrFunction,pErr->getErrorMsg(ErrorHandler::eErrFunction  107 : : ,"Invalid Pattern Code Error while reading OR pattern"));  108 : : }  109 : :   110 : 102 : orctx->L1WordLength = ph1.L1WordLength + 1 ;  111 : 102 : orctx->L2WordLength = ph1.L2WordLength + 1 ;  112 : 102 : orctx->PPDirection = WhichDirection[ph1.PPDirection] ;  113 : 102 : orctx->NumFieldsInEP = vsbNumEPFields(orctx->PatCode) ;  114 : 102 : orctx->NumOR1 = ph1.NumEP ;  115 : 102 : orctx->CurrOR1Num = 0 ;  116 : 102 : orctx->CurrPPNum = 1 ;  117 : 102 : orctx->isEmpty = false ;  118 : :   119 : 182 : pgc->current += sizeof(vsbPH1) + SizeOfOption[ph1.isOptionAdded] ;  120 : : }  121 : :   122 [ + + ]: 182 : if(orctx->CurrOR1Num == orctx->NumOR1)  123 : : {  124 : 80 : orctx->isEmpty = true ;  125 : 80 : return -1 ;  126 : : }   127 : :   128 : 102 : orctx->NumPP = vsbGetShort(pgc->current);   129 : :   130 : 102 : SDDB \*pSDDB = SDDB::instance();  131 : 102 : try  132 : : {  133 : 102 : pSDDB->EPReadComplete(orctx->NumPP);  134 : : }  135 [ # # ]: 0 : catch( DPL::Error \*pChild )  136 : : {  137 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  138 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  139 : : ErrorHandler::eErrFunction,  140 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  141 : : "SDDB::EPReadComplete"));  142 : : }  143 : :   144 : 102 : orctx->CurrPPNum = 1 ;  145 : 102 : pgc->current += sizeof(short int) ;  146 : :   147 : 102 : SkipSize = vsbBufferToEP(pgc->current, orctx->NumFieldsInEP, pgc->X0Y0WordLength, orctx->L1WordLength, orctx->L2WordLength, ep) ;  148 : :   149 : 102 : ep->PatCode = orctx->PatCode ;  150 : 102 : \*Nx = \*Ny = 1 ;  151 : 102 : \*Xe = ep->X0 ;  152 : 102 : \*Ye = ep->Y0 ;  153 : :   154 : 102 : pgc->ppc.ep = \*ep ;  155 : :   156 : : // Copy pre-translated EP to CRContext  157 [ + + ]: 102 : if(Composite == pgc->ppc.WhichContext){  158 : 72 : pgc->ppc.crctx.ReprEP = \*ep ;  159 : 72 : pgc->ppc.crctx.ReprNx = \*Nx ;  160 : 72 : pgc->ppc.crctx.ReprNy = \*Ny ;  161 : 72 : pgc->ppc.crctx.ReprXe = \*Xe ;  162 : 72 : pgc->ppc.crctx.ReprYe = \*Ye ;  163 : : }  164 [ + + ][ + + ]: 30 : else if ( ( 1.0F != pgc->TranslateFactor ||  165 : : pgc->mirror))  166 : : {  167 [ - + ]: 14 : if( 0 > TranslateEP(pgc, ep, Xe, Ye) )  168 : 0 : return -1 ;  169 : : }  170 : :   171 : 102 : orctx->CurrOR1Num++ ;  172 : 102 : pgc->current += SkipSize ;  173 : :   174 : 182 : return 0 ;  175 : : }  176 : :   177 : : /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  178 : : \* Name: ORGetPP  179 : : \* Functionality: Extracts one PP from a OR2 Representation (OR2:PH1(NPEPiPP+)+)  180 : : \* Input: Pointers to GroupContext and PP  181 : : \* Output: Returns an PP   182 : : \* Return value: Success or failure of the operation  183 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  184 : :   185 : : int ORGetPP(GroupContext \*pgc, PP \*pp)  186 : 215 : {  187 : 215 : ORContext \*orctx = NULL ;  188 : 215 : int SkipSize = 0 ;  189 : :   190 : 215 : orctx = &(pgc->ppc.orctx) ;  191 : :   192 [ + + ]: 215 : if(orctx->CurrPPNum < orctx->NumPP)  193 : : {  194 : 113 : pp->X0 = pgc->ppc.ep.X0 ;  195 : 113 : pp->Y0 = pgc->ppc.ep.Y0 ;  196 : :   197 : 113 : SkipSize = vsbBufferToPP(pgc->current, orctx->PPDirection, pgc->X0Y0WordLength, pp) ;  198 : :   199 : : // Copy pre-translated PP to CRContext  200 [ + + ]: 113 : if(Composite == pgc->ppc.WhichContext){  201 : 72 : pgc->ppc.crctx.ReprPP.X0 = pp->X0 ;  202 : 72 : pgc->ppc.crctx.ReprPP.Y0 = pp->Y0 ;  203 : : }  204 [ + + ][ + + ]: 41 : else if ( ( 1.0F != pgc->TranslateFactor ||  205 : : pgc->mirror))   206 : : {  207 : 12 : if( 0 > TranslatePP(pgc, pp) )  208 : 113 : return -1 ;  209 : :   210 : : }  211 : :   212 : 113 : pgc->current += SkipSize ;  213 : 113 : orctx->CurrPPNum++ ;  214 : 113 : return 0 ;  215 : : }  216 : :   217 : 215 : return -1 ;  218 : : }  219 : :   220 : 1215 : } //namespace nft |

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