| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [LibVsb/include](http://docs.google.com/index.html) - context.h (source / [functions](http://docs.google.com/context.h.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 15 | 15 | 100.0 % | | Date: | 2012-03-23 |  | Functions: | 0 | 0 | - | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 2 | 2 | 100.0 % | |  | |
|  |

|  |
| --- |
| 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 13:14:36 IST 2007  6 : : \* File Name: context.h  7 : : \* Purpose:   8 : : \* Author:   9 : : \*  10 : : \* Revision History:   11 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/   12 : : #ifndef \_CONTEXT\_H\_  13 : : #define \_CONTEXT\_H\_  14 : :   15 : : /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  16 : : \* Header Include section   17 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  18 : : #include "dec\_types.h"  19 : : #include "vsb\_types.h"  20 : : #include "Position.h"  21 : : #include "ThreadID.h"  22 : : #include "Memory.h"  23 : : /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  24 : : \* Class Declarations section   25 : : \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/   26 : :   27 : : namespace nft {  28 : : typedef enum \_vsbLocDirection {XYDirxn, XDirxn, YDirxn} LocDirection ;  29 : : typedef enum \_vsbContextType {NoContext, Normal, Optimize, Array, Composite} ContextType ;  30 : : typedef enum \_vsbCRContextType {CROptimize, CRArray, CRLinear} CRContextType ;  31 : :   32 : : typedef enum \_PatternArrangement  33 : : {  34 : : Horizontal,  35 : : Vertical,  36 : : Matrix,  37 : : Linear,  38 : : NoArrangement  39 : : } PatternArrangement ;  40 : :   41 : : typedef enum \_ArrayType  42 : : {  43 : : Term,  44 : : Pitch  45 : : } ArrayType ;  46 : :   47 : : typedef struct \_vsbNRContext  48 : : {  49 : : bool isEmpty ;  50 : : unsigned char PatCode ;  51 : : int L1WordLength ;  52 : : int L2WordLength ;  53 : : int NumEP ;  54 : : int CurrEPNum ;  55 : : int NumFieldsInEP ;  56 : : } NRContext ;  57 : :   58 : : typedef struct \_vsbORContext  59 : : {  60 : : bool isEmpty ;  61 : : unsigned char PatCode ;  62 : : int L1WordLength ;  63 : : int L2WordLength ;  64 : : int NumOR1 ;  65 : : int CurrOR1Num ;  66 : : int NumPP ;  67 : : int CurrPPNum ;  68 : : int NumFieldsInEP ;  69 : : LocDirection PPDirection ;  70 : : } ORContext ;  71 : :   72 : : typedef struct \_vsbARContext  73 : : {  74 : : bool isEmpty ;  75 : : double PitchLinearX ;  76 : : double PitchLinearY ;  77 : : int NumLinearReps ;  78 : : int CurrLinearReps ;  79 : : PatternArrangement PatArrangement ;  80 : : } ARContext ;  81 : :   82 : : typedef struct \_vsbCRContext  83 : : {  84 : : bool isEmpty ;  85 : : bool isPatternOver ;  86 : : bool isReprEPOver ;  87 : : CRContextType CRType ;  88 : : PP gPP0;  89 : : EP ReprEP ; // EP of a Sub Representation  90 : : int ReprNx ;  91 : : int ReprNy ;  92 : : int ReprXe ;  93 : : int ReprYe ;  94 : : vsbPP ReprPP ; // PP of Sub Representation,  95 : : // ARGetPP can return PP as a double, because Pitch need not be integral   96 : : int NumRepX ; // NumRepX from CR AP[x|y]  97 : : int NumRepY ; // NumRepX from CR AP[x|y]  98 : : int Xe ; // Xe computed/extracted from AP[x|y]  99 : : int Ye ; // Ye computed/extracted from AP[x|y]  100 : : int CurrRepX ;  101 : : int CurrRepY ;  102 : : double PitchX ;  103 : : double PitchY ;  104 : : int NumPP ;  105 : : int CurrPP ;  106 : : int NumRepresentations ;  107 : : int CurrRepresentations ;  108 : : Byte \*StartReprPtr ; // Representation Start Pointer  109 : : Byte \*CurrReprPtr ;  110 : : Byte \*StartPPptr ;  111 : : Byte \*CurrPPptr ;  112 : : Byte \*EndPPptr ;  113 : : Byte \*EndCRptr ;  114 : : LocDirection PPDirection ;  115 : : PatternArrangement PatArrangement ;  116 : : ContextType WhichSubContext ;  117 : : } CRContext ;  118 : :   119 : : typedef struct \_vsbPatternContext  120 : : {  121 : : double X0Offset ;  122 : : double Y0Offset ;  123 : : EP ep ; // Required for Mirroring PP :-(  124 : : ContextType WhichContext ;  125 : : CRContext crctx ;  126 : : union  127 : : {  128 : : NRContext nrctx ;  129 : : ORContext orctx ;  130 : : ARContext arctx ;  131 : : } ;  132 : : } PatternContext ;  133 : :   134 : : typedef struct \_vsbGroupContext  135 : : {  136 : : PatternContext ppc ;  137 : : double scale ;  138 : : int SourceAU ;  139 : : int isAligned ;  140 : : int mirror ;  141 : : double TranslateFactor ;  142 : : int X0Y0WordLength ;  143 : : unsigned int CellPatDx ;  144 : : unsigned int CellPatDy ;  145 : : Byte \*start ;  146 : : Byte \*current ;  147 : : Byte \*end ;  148 : : unsigned int attributeInfo;  149 : : } GroupContext ;  150 : :   151 : : typedef struct \_vsbCellContext  152 : : {  153 : : GroupContext pgc ;  154 : : Byte \*start ;  155 : : Byte \*current ;  156 : : Byte \*end ;  157 : : //Function:  158 : : public:  159 : : int getRefIDNext();  160 : : } CellContext ;  161 : :   162 : : typedef struct \_vsbContext  163 : : {  164 : : CellContext pcl ;  165 : : //constructor  166 : : } Context ;  167 : :   168 : : struct Cell : public NftNewDeleteBase  169 : : {  170 : : typedef Cell \* Pointer;  171 : : int refid ;  172 : : CellPlacementType PlacementType ;  173 : : CellLocationMethod RefType;  174 : : GroupContext \*pgc ; //Added ANINDITA  175 : : IntPosition startPos ;  176 : : union   177 : : {  178 : : SingleCell sc ;  179 : : ArrayCell ac ;  180 : : } ;  181 : : size\_t mSegmentDataSize;  182 : : ThreadID mTargetThreadID;  183 : :   184 : : Cell()  185 : 2601 : :refid(),  186 : : PlacementType(),  187 : : RefType(),  188 : : pgc(),  189 : : startPos(),  190 : : mSegmentDataSize(),  191 : 48635 : mTargetThreadID() {  192 : : }  193 : :   194 : 2545 : ~Cell() {  195 : 2545 : delete pgc;  196 : : }  197 : :   198 : : void arrayToSingle()  199 : : {  200 : : switch(RefType)  201 : : {  202 : : case ARRAY\_NEAR:  203 : : RefType = SINGLE\_NEAR;  204 : : break;  205 : : case ARRAY\_VALUE:  206 : : RefType = SINGLE\_VALUE;  207 : : break;  208 : : default: //nothing to do  209 : : break;  210 : : }  211 : : }  212 : : inline bool isReference() const  213 : : {  214 : : return (SINGLE\_NEAR == RefType || ARRAY\_NEAR == RefType  215 : : );  216 : : }  217 : : inline bool isArray() const  218 [ + + ]: 2522 : {  219 : : return (ARRAY\_VALUE == RefType || ARRAY\_NEAR == RefType  220 : : );  221 : : }  222 : : IntPosition getCellStartPosition()  223 : 1261 : {  224 : 1261 : if(isArray())  225 : 54 : {  226 : : return IntPosition(ac.xb, ac.yb);  227 : : }  228 : : else  229 : 1207 : {  230 : : return IntPosition(sc.xb, sc.yb);  231 : : }  232 : : }  233 : : inline void setSegmentDataLength( size\_t size )  234 : 1261 : {  235 : 1261 : mSegmentDataSize = size;  236 : : }  237 : : inline size\_t getSegmentDataSize()  238 : 1307 : {  239 : : return mSegmentDataSize;  240 : : }  241 : :   242 : : inline void setTargetThreadID( ThreadID id )  243 : 1244 : {  244 : 1244 : mTargetThreadID = id;  245 : : }  246 : :   247 : : inline ThreadID getTargetThreadID()  248 : 1244 : {  249 : : return mTargetThreadID;  250 : : }  251 : : };  252 : :   253 : :   254 : : }//namespace nft  255 : : #endif /\* \_CONTEXT\_H\_ \*/ |

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