Annex I

(normative)

"normative" moved by editor (IEEE style requirement)

sv_vpi_user.h

~~(normative)~~

Editor Note: Erratum #158 replaced all of this file. All other changes were made on top of #158.

sv_vpi_user.h

```
/************************************************************************
* sv_vpi_user.h
*
* ~~Accellera~~ SystemVerilog VPI extensions.
*
* This file contains the constant definitions, structure definitions, and
* routine declarations used by the Verilog PLI procedural interface VPI
* access routines.
*
************************************************************************/

/************************************************************************
* NOTE:
* The constant values 600 through 999 are reserved for use in this file.
* - the range 600-749 is reserved for SV VPI model extensions
* - the range 750-779 is reserved for the Coverage VPI
* - the range 780-799 is reserved for the Assertion VPI
* - the range 800-899 is reserved for the Reader VPI
* Overlaps in the numerical ranges are permitted for different categories
*  of identifiers; e.g.
*  - object types
*  - properties
*  - callbacks
************************************************************************/

#ifndef SV_VPI_USER_H
#define SV_VPI_USER_H

#include <vpi_user.h>

#ifdef __cplusplus
extern "C" {
#endif
}

/****************************** OBJECT TYPES *******************************/
#define vpiPackage              600
#define vpiInterface            601
#define vpiProgram              602
#define vpiInterfaceArray       603
#define vpiProgramArray         604
#define vpiTypespec             605
#define vpiModport              606
#define vpiInterfaceTfDecl      607
```

"Accellera" deleted by editor

```
#define vpiRefObj                608

/* variables */
#define vpiVarBitVar vpiRegBit                                  erratum #044
#define vpiLongIntVar            610
#define vpiShortIntVar           611
#define vpiIntVar                612
#define vpiShortRealVar          613
#define vpiByteVar               614
#define vpiClassVar              615
#define vpiStringVar             616
#define vpiEnumVar               617
#define vpiStructVar             618
#define vpiUnionVar              619
#define vpiBitVar                620
#define vpiLogicVar              vpiRegVar
#define vpiArrayVar              vpiRegArray

/* typespecs */
#define vpiLongIntTypespec       625
#define vpiShortRealTypespec     626
#define vpiByteTypespec          627
#define vpiShortIntTypespec      628
#define vpiIntTypespec           629
#define vpiClassTypespec         630
#define vpiStringTypespec        631
#define vpiVarBitTypespec        632                            erratum #295
#define vpiEnumTypespec          633
#define vpiEnumConst             634
#define vpiIntegerTypespec       635
#define vpiTimeTypespec          636
#define vpiRealTypespec          637
#define vpiStructTypespec        638
#define vpiUnionTypespec         639
#define vpiBitTypespec           640
#define vpiLogicTypespec         641
#define vpiArrayTypespec         642
#define vpiVoidTypespec          643
#define vpiMemberTypespec vpiTypespecMember 644                 erratum #123

#define vpiClockingBlock         650
#define vpiClockingIODecl        651
#define vpiClassDefn             652
#define vpiConstraint            653
#define vpiConstraintOrdering    654

#define vpiDistItem              645
#define vpiAliasStmt             646
#define vpiThread                647
#define vpiMethodFuncCall        648
#define vpiMethodTaskCall        649

/* concurrent assertions */
#define vpiAssertProperty        650                            erratum #265
#define vpiAssumeProperty        651
#define vpiCoverProperty         652

#define vpiDisableCondition      653
#define vpiClockingEvent         654
```

```
/* property decl, spec */
#define vpiPropertyDecl           655
#define vpiPropertySpec           656
#define vpiPropertyExpr           657
#define vpiMulticlockSequenceExpr 658
#define vpiClockedSeq             659
#define vpiPropertyInst           660
#define vpiSequenceDecl           661
#define vpiSequenceSpec           662
#define vpiActualArgExpr          663
#define vpiSequenceInst           664
#define vpiImmediateAssert        665
#define vpiReturn                 666
/* pattern */
#define vpiAnyPattern             667
#define vpiTaggedPattern          668
#define vpiStructPattern          669
/* do .. while */
#define vpiDoWhile                670
/* waits */
#define vpiOrderedWait            671
#define vpiWaitFork               672
/* disables */
#define vpiDisableFork            673
#define vpiExpectStmt             674
#define vpiForeachStmt            675
#define vpiFinal                  676
#define vpiExtend                 677
#define vpiDistribution           678
#define vpiIdentifier             679
#define vpiArrayNet               vpiNetArray
#define vpiEnumNet                680
#define vpiIntegerNet             681
#define vpiLogicNet               vpiNet
#define vpiTimeNet                682
#define vpiStructNet              683


/******************************** METHODS *********************************/
/************* methods used to traverse 1 to 1 relationships **************/
#define vpiActual                 700

#define vpiTypedefAlias           701

#define vpiIndexTypespec          702
#define vpiBaseTypespec           703
#define vpiElemTypespec           704

#define vpiDefInputSkew           706
#define vpiDefOutputSkew          707
#define vpiClockingSkew           708

#define vpiActualDefn             710
#define vpiLhs                    711
#define vpiRhs                    712
#define vpiOrigin                 713
#define vpiPrefix                 714
#define vpiWith                   715
```

erratum #213

erratum #333

```
#define vpiProperty               718

#define vpiValueRange             720
#define vpiPattern                721
#define vpiWeight                 722

/************ methods used to traverse 1 to many relationships ************/
#define vpiTypedef                725
#define vpiImport                 726
#define vpiDerivedClasses         727

#define vpiMethods                730
#define vpiSolveBefore            731
#define vpiSolveAfter             732

#define vpiWaitingProcesses       734

#define vpiMessages               735
#define vpiMembers                736
#define vpiLoopVars               737

#define vpiConcurrentAssertions   740
#define vpiMatchItem              741

/************ methods both 1-1 and 1-many relations **********************/
#define vpiInstance               745


/*************************************************************************/
/************************ generic object properties **********************/
/*************************************************************************/

#define vpiTop                    600

#define vpiUnit                   602

#define vpiAccessType             604
#define vpiForkJoinAcc         1
#define vpiExternAcc           2
#define vpiDPIExternAcc        3
#define vpiDPIImportAcc        4

#define vpiArrayType              606
#define vpiStaticArray         1
#define vpiDynamicArray        2
#define vpiAssocArray          3
#define vpiQueueArray          4
#define vpiArrayMember            626 607

#define vpiIsRandomized           608

#define vpiRandType               610
#define vpiNotRand             1
#define vpiRand                2
#define vpiRandC               3

#define vpiConstantVariable       612
#define vpiMember                 615
```

erratum #062 (location and value changed by editor)

```
#define vpiVisibility             620
#define vpiPublicVis            1
#define vpiProtectedVis         2
#define vpiLocalVis             3

#define vpiNullConst              622
#define vpiOneStepConst           623

#define vpiAlwaysType             624
#define vpiAlwaysComb           1
#define vpiAlwaysFF             2
#define vpiAlwaysLatch          3

#define vpiDistType               625
#define vpiEqualDist            1 /* constraint equal distribution */
#define vpiDivDist              2 /* constraint divided distribution */

#define vpiPacked                 630
#define vpiTagged                 632
#define vpiRef                    633
#define vpiDefaultSkew            634
#define vpiVirtual                635
#define vpiUserDefined            636
#define vpiIsConstraintEnabled    638

#define vpiClassType              640
#define vpiMailboxClass         1
#define vpiSemaphoreClass       2
#define vpiUserDefineClass      3

#define vpiMethod                 645
#define vpiValid                  646
#define vpiActive                 647
#define vpiIsClockInferred        649

#define vpiQualifier              650
#define vpiNoQualifier          0
#define vpiUniqueQualifier      1
#define vpiPriorityQualifier    2
#define vpiTaggedQualifier      3
#define vpiRandQualifier        8

/******************************** Operators ********************************/
#define vpiImplyOp              50 /* -> implication operator */
#define vpiNonOverlapImplyOp    51 /* |=> non-overlapped implication */
#define vpiOverlapImplyOp       52 /* |-> overlapped implication operator */
#define vpiUnaryCycleDelayOp    53 /* binary cycle delay (##) operator */
#define vpiCycleDelayOp         54 /* binary cycle delay (##) operator */
#define vpiIntersectOp          55 /* intersection operator */
#define vpiFirstMatchOp         56 /* first_match operator */
#define vpiThroughoutOp         57 /* throught operator */
#define vpiWithinOp             58 /* within operator */
#define vpiRepeatOp             59 /* [=] non-consecutive repetition */
#define vpiConsecutiveRepeatOp  60 /* [*] consecutive repetition */
#define vpiGotoRepeatOp         61 /* [->] goto repetition */

#define vpiPostIncOp            62 /* ++ post-increment */
#define vpiPreIncOp             63 /* ++ pre-increment */
#define vpiPostDecOp            64 /* -- post-decrement */
```

erratum #267

```
#define vpiPreDecOp            65 /* -- pre-decrement */

#define vpiMatchOp             66 /* match() operator */
#define vpiCastOp              67 /* type'() operator */
#define vpiIffOp               68 /* iff operator */
#define vpiWildEqOp            69 /* =?= ==? operator */
#define vpiWildNeqOp           70 /* !?= !=? operator */

#define vpiStreamLROp          71 /* left-to-right streaming {>>} operator */
#define vpiStreamRLOp          72 /* right-to-left streaming {<<} operator */

#define vpiMatchedOp           73 /* the .matched sequence operation */
#define vpiEndedOp             74 /* the .ended sequence operation */

/************************* STRUCTURE DEFINITIONS *************************/

/***************************** structure *****************************/

/***************************** CALLBACK REASONS *****************************/
#define cbStartOfThread       600 /* callback on thread creation */
#define cbEndOfThread         601 /* callback on thread termination */
#define cbEnterThread         602 /* callback on re-entering thread */
#define cbStartOfFrame        603 /* callback on frame creation */
#define cbEndOfFrame          604 /* callback on frame exit */
#define cbTypeChange          605 /* callback on variable type/size change */

/************************* FUNCTION DECLARATIONS *************************/


/*****************************************************************************/
/***************************** Coverage VPI ******************************/
/*****************************************************************************/
/* NOTE: Coverage VPI has been assigned the numerical range 750-779 */

/* coverage control */
#define vpiCoverageStart          750
#define vpiCoverageStop           751
#define vpiCoverageReset          752
#define vpiCoverageCheck          753
#define vpiCoverageMerge          754
#define vpiCoverageSave           755

/* coverage type properties */
#define vpiAssertCoverage         760
#define vpiFsmStateCoverage       761
#define vpiStatementCoverage      762
#define vpiToggleCoverage         763

/* coverage status properties */
#define vpiCovered                765
#define vpiCoverMax               766
#define vpiCoveredCount           767
/* assertion-specific coverage status properties */
#define vpiAssertAttemptCovered   770
#define vpiAssertSuccessCovered   771
#define vpiAssertFailureCovered   772
/* FSM-specific coverage status properties */
#define vpiFsmStates              775
#define vpiFsmStateExpression     776
```

erratum #213

```
/* FSM handle types */
#define vpiFsm                    758
#define vpiFsmHandle              759


/************************************************************************/
/***************************** Assertion VPI ****************************/
/************************************************************************/
/* NOTE: Assertion VPI has been assigned the numerical range 780-799 */

/* assertion types */
#define vpiSequenceType           780
#define vpiAssertType             781
#define vpiCoverType              782
#define vpiPropertyType           783
#define vpiImmediateAssertType    784

/* assertion callback types */
#define cbAssertionStart          606
#define cbAssertionSuccess        607
#define cbAssertionFailure        608
#define cbAssertionStepSuccess    609
#define cbAssertionStepFailure    610
#define cbAssertionDisable        611
#define cbAssertionEnable         612
#define cbAssertionReset          613
#define cbAssertionKill           614
/* assertion "system" callback types */
#define cbAssertionSysInitialized 615
#define cbAssertionSysStart       616
#define cbAssertionSysStop        617
#define cbAssertionSysEnd         618
#define cbAssertionSysReset       619

/* assertion control constants */
#define vpiAssertionDisable       620
#define vpiAssertionEnable        621
#define vpiAssertionReset         622
#define vpiAssertionKill          623
#define vpiAssertionEnableStep    624
#define vpiAssertionDisableStep   625
#define vpiAssertionClockSteps    626
#define vpiAssertionSysStart      627
#define vpiAssertionSysStop       628
#define vpiAssertionSysEnd        629
#define vpiAssertionSysReset      630

/* assertion related structs and types */
typedef struct t_vpi_source_info {
   PLI_BYTE8 *fileName;
   PLI_INT32 startLine;
   PLI_INT32 startColumn;
   PLI_INT32 endLine;
   PLI_INT32 endColumn;
} s_vpi_source_info, *p_vpi_source_info;

typedef struct t_vpi_assertion_info {
   PLI_BYTE8 *assertName;           /* name of assertion */
```

erratum #265

```
   vpiHandle instance;              /* instance containing assertion */
   PLI_BYTE8 defname;               /* name of module/interface containing
                                     * the assertion */
   vpiHandle clock;                 /* clocking expression */
   PLI_INT32 assertionType;        /* vpiSequenceTypeInst, vpiAssertType, vpiAssume
                                     * vpiCoverType, vpiPropertyTypeInst,
                                     * vpiImmediateAssertType */
   s_vpi_source_info sourceInfo;
} s_vpi_assertion_info, *p_vpi_assertion_info;

typedef struct t_vpi_assertion_step_info {
   PLI_INT32 matched_expression_count;
   vpiHandle *matched_exprs;                /* array of expressions */
   p_vpi_source_info *exprs_source_info;    /* array of source info */
   PLI_INT32 stateFrom, stateTo;            /* identify transition */
} s_vpi_assertion_step_info, *p_vpi_assertion_step_info;

typedef struct t_vpi_attempt_info {
   union {
      vpiHandle failExpr;
      p_vpi_assertion_step_info step;
   } detail;
   s_vpi_time attemptStartTime;     /* Time attempt triggered */
} s_vpi_attempt_info, *p_vpi_attempt_info;

/* typedef for vpi_register_assertion_cb callback function */
typedef PLI_INT32 (vpi_assertion_callback_func)(
   PLI_INT32 reason,                /* callback reason */
   p_vpi_time cb_time,              /* callback time */
   vpiHandle assertion,             /* handle to assertion */
   p_vpi_attempt_info info,         /* attempt related information */
   PLI_BYTE8 *user_data             /* user data entered upon registration */
);

/* assertion specific VPI functions */
PLI_INT32 vpi_get_assertion_info (assert_handle, p_vpi_assertion_info);

vpiHandle vpi_register_assertion_cb(
   vpiHandle assertion,             /* handle to assertion */
   PLI_INT32 reason,                /* reason for which callbacks needed */
   vpi_assertion_callback_func *cb_rtn,
   PLI_BYTE8 *user_data             /* user data to be supplied to cb */
);


/*****************************************************************************/
/****************************** Reader VPI ***********************************/
/*****************************************************************************/
/* NOTE: Reader VPI has been assigned the numerical range 800-899 */


/********** Reader types ***********/
#define vpiTrvsObj              800 /* Data traverse object */
#define vpiCollection           810 /* Collection of VPI handle */
#define vpiObjCollection        811 /* Collection of traversable design objs */
#define vpiTrvsCollection       812 /* Collection of vpiTrvsObjs */

/********* Reader methods *********/
```

erratum #265

```
/* Check */
#define vpiIsLoaded             820 /* Object data is loaded check */
#define vpiHasDataVC            821 /* Traverse object has at least one VC
                                     * at some point in time in the
                                     * database check */

#define vpiHasVC                822 /* Has VC at specific time check */
#define vpiHasNoValue           823 /* Has no value at specific time check */
#define vpiBelong               824 /* Belongs to extension check */

/* Access */
#define vpiAccessLimitedInteractive  830 /* Interactive access */
#define vpiAccessInteractive         831 /* interactive with history access */
#define vpiAccessPostProcess         832 /* Database access */

/* Member of a collection */
#define vpiMember               840 /* Member of a collection */

/* Iteration on instances for loaded */
#define vpiDataLoaded           850 /* Use in vpi_iterate() */

/* Control Traverse/Check Time */
#define vpiMinTime              860 /* Min time */
#define vpiMaxTime              864 /* Max time */
#define vpiPrevVC               868 /* Previous Value Change (VC) */
#define vpiNextVC               870 /* Next Value Change (VC) */
#define vpiTime                 874 /* Time jump */

/********** routines **********/


/* general form of the vpi extension loading function is:
 * PLI_INT32 vpi_load_extension PROTO_PARAMS((PLI_BYTE8 *extension_name, ...))
 */

/********** Reader routines **********/

/* load extension form for the reader extension */
PLI_INT32 vpi_load_extension PROTO_PARAMS((PLI_BYTE8 *extension_name,
                                           PLI_BYTE8 *name,
                                           vpiType mode, ...))

PLI_INT32 vpi_close PROTO_PARAMS((PLI_INT32 tool,
                                  vpiType prop,
                                  PLI_BYTE8* name));

PLI_INT32 vpi_load_init PROTO_PARAMS((vpiHandle objCollection,
                                      vpiHandle scope,
                                      PLI_INT32 level));

PLI_INT32 vpi_load PROTO_PARAMS((vpiHandle h));

PLI_INT32 vpi_unload PROTO_PARAMS((vpiHandle h));

vpiHandle vpi_create PROTO_PARAMS((vpiType prop,
                                   vpiHandle h,
                                   vpiHandle obj));

vpiHandle vpi_goto PROTO_PARAMS((vpiType prop,
```

```
                                    vpiHandle obj,
                                    p_vpi_time time_p,
                                    PLI_INT32 *ret_code));

vpiHandle vpi_filter PROTO_PARAMS((vpiHandle h,
                                     PLI_INT32 ft,
                                     PLI_INT32 flag));


#ifdef __cplusplus
}
#endif

#endif
```