Restrict verification statement

Numbering based on P1800 Draft 4.

Objectives:

Currently SVA supports the verification statements **assert**, **assume** and **cover property**. In formal verification, for the tool to converge on a proof of a property or to initialize the design to a specific state, it is often necessary to constrain the state space. For this purpose, the assertion statement **restrict property** is introduced. It has the same semantics as assume property, however, in contrast to that statement, verification tools are not obliged to verify that the property holds.

Modify at the beginning 16.14

— **assert** to specify the property as a checker to ensure that the property holds for the design
— **assume** to specify the property as an assumption for the environment
— **cover** to monitor the property evaluation for coverage
— **restrict** to constrain the environment

Insert after 16.14.3.

(Note to editor: please renumber clauses)

16.14.4 Restrict statement

In formal verification, for the tool to converge on a proof of a property or to initialize the design to a specific state, it is often necessary to constrain the state space. For this purpose, the assertion statement **restrict property** is introduced. It has the same semantics as **assume property**, however, in contrast to that statement, the **restrict** statement is not verified in simulation and has no action block.

The statement has the following form

restrict property (property_spec **)** ;

There is no action block associated with the statement.

Example:

Suppose that when a control bit `ctr` has a value 0, an ALU performs an addition, and when it is 1, it performs a subtraction. It is required to formally verify that some behavior is correct when ALU does an addition (in another verification session it is possible to do the same for subtraction by changing the restriction). The behavior can thus be constrained using the statement

```
restrict property (@(posedge clk) ctr == '0);
```

It does not mean that `ctr` cannot be 1 in any test case in the simulation, this is not an error.

Change Syntax 16-16

```
procedural_assertion_statement ::=                                    // from A.6.10
        concurrent_assertion_statement
      | immediate_assert_statement
concurrent_assertion_item ::= [ block_identifier : ] concurrent_assertion_statement   // from A.2.10
concurrent_assertion_statement ::=
        assert_property_statement
      | assume_property_statement
      | cover_property_statement
      | restrict_property_statement
assert_property_statement::=
      assert property ( property_spec ) action_block
assume_property_statement::=
      assume property ( property_spec ) action_block
cover_property_statement::=
      cover property ( property_spec ) statement_or_null
cover_sequence_statement::=
      cover sequence ( [clocking_event ] [ disable iff ( expression_or_dist ) ]
            sequence_expr ) statement_or_null
      restrict property ( property_spec ) ;
```

Change Annex A.2.10 Assertion declaration

```
concurrent_assertion_statement ::=
        assert_property_statement
      | assume_property_statement
      | cover_property_statement
      | cover_sequence_statement
      | restrict_property_statement
assert_property_statement::=
      assert property ( property_spec ) action_block
assume_property_statement::=
      assume property ( property_spec ) action_block
cover_property_statement::=
      cover property ( property_spec ) statement_or_null
expect_property_statement ::=
      expect ( property_spec ) action_block
cover_sequence_statement::=
      cover sequence ( [clocking_event ] [ disable iff ( expression_or_dist ) ]
            sequence_expr ) statement_or_null
restrict_property_statement::=
      restrict property ( property_spec ) ;
```

...

Annex B, Table B1

Add keyword **restrict** to the table.

Change Annex J

From

```
/* concurrent assertions */
#define vpiAssert 686
#define vpiAssume 687
#define vpiCover 688
```

To

```
/* concurrent assertions */
#define vpiAssert 686
#define vpiAssume 687
#define vpiCover 688
#define vpiRestrict ?? (Editor please assign a code)
```

36.43 Concurrent assertions

Change

vpi diagram to add an oval with **restrict** to the concurrent assertion container.

Change

Details:

1) Clocking event is always the actual clocking event on which the assertion is being evaluated, regardless of whether this is explicit or implicit (inferred).

To

Details:

1) Clocking event is always the actual clocking event on which the assertion is being evaluated, regardless of whether this is explicit or implicit (inferred).

2) The restrict property statement has no pass and no fail action statement.

38.3.2 Obtaining static assertion information

Change

The following information about an assertion is considered to be static:
— Assertion name
— Instance in which the assertion occurs
— Module definition containing the assertion
— Assertion type
— Sequence
— Assert
— Assume
— Cover
— Property
— ImmediateAssert
— Assertion source information: the file, line, and column where the assertion is defined
— Assertion **`clocking`** block/expression

To

The following information about an assertion is considered to be static:
— Assertion name
— Instance in which the assertion occurs
— Module definition containing the assertion
— Assertion type
— Sequence
— Assert
— Assume
— Cover
— Restrict
— Property
— ImmediateAssert
— Assertion source information: the file, line, and column where the assertion is defined
— Assertion **`clocking`** block/expression