ADD

17.14 Disable resolution

Note to editor: Shift the numeration of the following subsections accordingly.

Note to the editor: Add a Syntax Box containing the following text:

```
module_or_generate_item_declaration ::=                    // from A.1.4
        ...
        | default clocking clocking_identifier ;
        | default disable expression_or_dist ;
```

One can specify a default disabling condition for all assertions within any given module, interface, or program. The syntax for the default disable specification statement is as follows:

```
module_or_generate_item_declaration ::=
        ...
        | default disable expression_or_dist ;
```

Only one default disable can be specified in a module, interface, or program. Specifying a default disable more than once in the same module, interface, or program shall result in a compilation error.

A default disable is valid only within the scope containing the default disable specification. This scope includes the module, interface or program that contains the declaration as well as any nested modules or interfaces. It does not include instantiated modules or interfaces.

The following rules apply for the reset resolution:

a) If an assertion has a **disable iff** clause, then the disable specified in this clause will be used, and any **`default disable`** statement will be ignored.

b) If an assertion does not contain a **disable iff** clause and the assertion is in an **always** block or **always_ff** block that has asynchronous resets defined (see Note to the editor - please insert reference to $inferred_... functions), then the disabling value is inferred from the reset expression. Asynchronous resets are detected if and only if the event control expression of the **always** or **always_ff** block has the following form:

```
edge_identifier1 expression1 [ iff expression ] or edge_identifier2
expression2 or ...  or edge_identifierN expressionN
```

The disabling condition consists of an OR (||) operation over the target expressions determined by the edge expressions. That is, if `edge_identifieri` is **posedge** then the target expression is `expression2` and it is `!expressioni` if `edge_identifieri` is **negedge**.

For example, if the event control on an always block is `@(`**posedge** `clk` **or negedge** `rst` **or posedge** `set)`, the inferred disable condition is `!rst || set.`

c) If an assertion does not contain a **disable iff** clause and is not within an **always** or **always_ff** block that has the above form of event control expression, but the assertion is within the scope of a **default**

disable statement, then the disabling value for the assertion is inferred from the **default disable** statement.

d) Otherwise, no inference is performed (this is equivalent to the inference of a `1'b0` reset value).

Below are two example modules illustrating the application of these rules.

```
module examples_with_default (input logic a, b, clk, rst, rst1);

default disable rst;

property p1;
    disable iff (rst1) a |=> b;
endproperty

// Disable condition is rst1 - explicitly specified within a1
a1 : assert property (@(posedge clk) disable iff (rst1) a |=> b);

// Disable condition is rst1 - explicitly specified within p1
a2 : assert property (@(posedge clk ) p1);

// Disable condition is rst - no explicit specification, inferred from
// default disable statement
a3 : assert property (@(posedge clk) a |=> b);

// Disable condition is 1'b0 . This is the only way to
// cancel the effect of default disable.
a4 : assert property (@(posedge clk) disable iff (1'b0) a |=> b);

// disable condition and clocking event are inferred from an always block
// as @(posedge clk) disable iff (!rst), masking the effect of default disable
always @(posedge clk or negedge rst)
if (!rst)
   ...
else begin
   a8 : assert property (    a |=> b);
   ...
end

endmodule

module examples_without_default (input logic a, b, clk, rst);

property p2;
    disable iff (rst) a |=> b;
endproperty

// Disable condition is rst - explicitly specified within a5
a5 : assert property (@(posedge clk ) disable iff (rst) a |=> b);

// Disable condition is rst - explicitly specified within p2
a6 : assert property (@ (posedge clk ) p2);

// No Disable condition
a7 : assert property (@ (posedge clk ) a |=> b);

// disable condition, enable condition and clocking event are inferred from an always block
// as @(posedge clk) disable iff (rst) !rst |-> (a |=> b)

always @(posedge clk or posedge rst)
if (rst)
   ...
else begin
   a8 : assert property ( a |=> b);
   ...
end
```

endmodule

15.11 Default clocking

Change in Syntax 15-3 from

module_or_generate_item_declaration ::= *// from A.1.4*
...
| **default clocking** clocking_identifier ;

to

module_or_generate_item_declaration ::= *// from A.1.4*
...
| **default clocking** clocking_identifier ;
...

A.1.4 Module items

REPLACE

module_or_generate_item_declaration ::=
package_or_generate_item_declaration
| genvar_declaration
| clocking_declaration
| **default clocking** clocking_identifier **;**

WITH

module_or_generate_item_declaration ::=
package_or_generate_item_declaration
| genvar_declaration
| clocking_declaration
| **default clocking** clocking_identifier **;**
| **default disable** expression_or_dist **;**