## 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 may be specified anywhere 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. Furthermore, modules, interfaces or programs that are declared within the scope where a default disable is specified may redefine the default disable expression.

The following rules apply for the disable condition 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, 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.

c) 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);

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);

// Only enable condition and clocking event are inferred from an always block
// Assertion a8 is equivalent to
//    @(posedge clk)  !bit'(rst!='b0) |-> (a |=> b)

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

endmodule
```

In assertion `a8` the inferred enabling condition is from the **else** clause of the **if - else** statement and thus it has to respects the interpretation of a four-valued expression in the **if** condition. One of such forms is as indicated in `a8`, however other forms may be used. For example, `((rst != 'b0) !== 1'b1)`. For synthesizable forms of the enabling condition where only two-valued interpretation of signals is used, the enabling condition for the assertion can be `!rst`.

### 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 **;**