

Mantis 2088: Allow checker construct to include covergroups

Motivation

Mantis 1900 added a new construct called a checker to encapsulate related verification constructs like assertions into one entity. This proposal extends the checker to allow covergroup constructs to be included. This is required to make the checker more usable in verification libraries.

Note: Some changes to VPI diagrams may be required.

Note to editor: This proposal relies on the following other language-extending Mantis items:

- 1900: checker

16.18.1 Overview

Note to editor: from Mantis 1900 Text

REPLACE

The checkers may contain concurrent assertions, free variable declarations and assignments, structural procedures, function declarations, **let** declarations, sequences and properties, and generate regions. The following procedures are allowed in a checker (see 16.18.4):

WITH

The checkers may contain concurrent assertions, free variable declarations and assignments, structural procedures, function declarations, **let** declarations, sequences and properties, **covergroups**, and generate regions. The following procedures are allowed in a checker (see 16.18.4):

16.18.2 Checker declaration

Mantis 1900

REPLACE

```
checker_or_generate_item_declaration ::=
    data_declaration
    | function_declaration
    | assertion_item_declaration           Mantis 1728: concurrent_assertion_item_declaration
    | overload_declaration
    | genvar_declaration
    | default clocking clocking_identifier ;
    | default disable iff expression_or_dist ; Mantis 1674
    | ;
```

WITH

```
checker_or_generate_item_declaration ::=
    data_declaration
    | function_declaration
    | assertion_item_declaration           Mantis 1728: concurrent_assertion_item_declaration
    | covergroup_declaration
    | overload_declaration
    | genvar_declaration
    | default clocking clocking_identifier ;
```

```
| default disable iff expression_or_dist ; Mantis 1674  
| ;
```

REPLACE

A checker body may contain the following elements:

- Declaration of **let**, sequences, properties and functions.
- Concurrent assertions.
- Free variables and their assignments (see 16.18.5).
- Default clocking and disable declarations.
- **initial_check** and **always_check** procedures (see 16.18.4).
- Generate blocks, containing any of the above elements.

WITH

A checker body may contain the following elements:

- Declaration of **let**, sequences, properties and functions.
- Concurrent assertions.
- [Covergroup declarations and instances](#).
- Free variables and their assignments (see 16.18.5).
- Default clocking and disable declarations.
- **initial_check** and **always_check** procedures (see 16.18.4).
- Generate blocks, containing any of the above elements.

ADD NEW SECTION

Note to editor: Shift subsequent section numbers

16.18.6 Covergroups in checkers

One or more **covergroup** declarations or instances (see 18.3) are permitted within a checker. These declarations and instances shall not appear in any procedural block in the checker. A **covergroup** may reference any variable visible in its scope, including checker formal arguments and checker variables. For example:

```
checker my_check(logic clk, active);  
  checkvar bit active_d1 = 1'b0;  
  
  always_check @(posedge clk) begin  
    active_d1 <= active;  
  end  
  
  covergroup cg_active @(posedge clk);  
    cp_active : coverpoint active
```

```

    {
        bins idle = { 1'b0 };
        bins active = { 1'b1 };
    }

    cp_active_d1 : coverpoint active_d1
    {
        bins idle = { 1'b0 };
        bins active = { 1'b0 };
    }
endgroup

cg_active cg_active_1 = new();
endchecker : my_check

```

18.3 Defining the coverage model: covergroup

REPLACE

The **covergroup** construct is a user-defined type. The type definition is written once, and multiple instances of that type can be created in different contexts. Similar to a class, once defined, a **covergroup** instance can be created via the **new()** operator. A **covergroup** can be defined in a package, module, program, interface, or class.

WITH

The **covergroup** construct is a user-defined type. The type definition is written once, and multiple instances of that type can be created in different contexts. Similar to a class, once defined, a **covergroup** instance can be created via the **new()** operator. A **covergroup** can be defined in a package, module, program, interface, **checker**, or class.

A.1.8 Checker items

Mantis 1900

REPLACE

```

checker_or_generate_item_declaration ::=
    data_declaration
    | function_declaration
    | assertion_item_declaration
    | overload_declaration
    | genvar_declaration
    | default clocking clocking_identifier ;
    | default disable iff expression_or_dist ;
    ;

```

Mantis 1728: ~~concurrent~~-assertion_item_declaration

Mantis 1674

WITH

```

checker_or_generate_item_declaration ::=
    data_declaration
    | function_declaration
    | assertion_item_declaration
    | covergroup_declaration
    | overload_declaration
    | genvar_declaration

```

Mantis 1728: ~~concurrent~~-assertion_item_declaration

```
| default clocking clocking_identifier ;  
| default disable iff expression_or_dist ; Mantis 1674  
| ;
```