Unraveling the Complexities of Functional Coverage: An advanced guide to simplify your use model

Thom Ellis
Mentor Graphics Corporation
thomas_ellis@mentor.com

Rohit Jain
Mentor Graphics Corporation
rohit_jain@mentor.com

Top 3 Guidelines

1) For covergroup options, the most efficient settings will be:
   * option.per_instance=0
   * type_option.merge_instances=1

2) Get in the habit of always naming your covergroup instances.

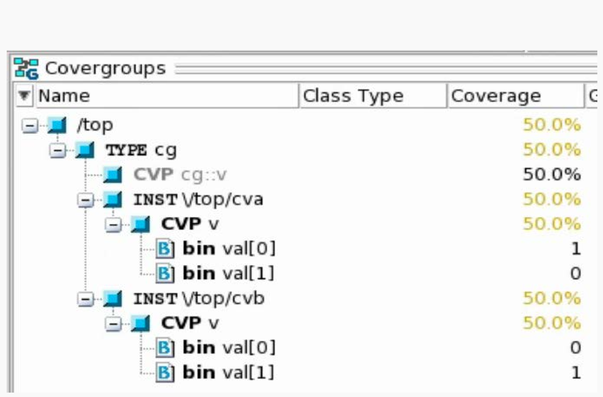
3) Explicitly define a list of cross bins whenever possible, rather than relying on a tools auto-expansion features.

1) Covergroup Options

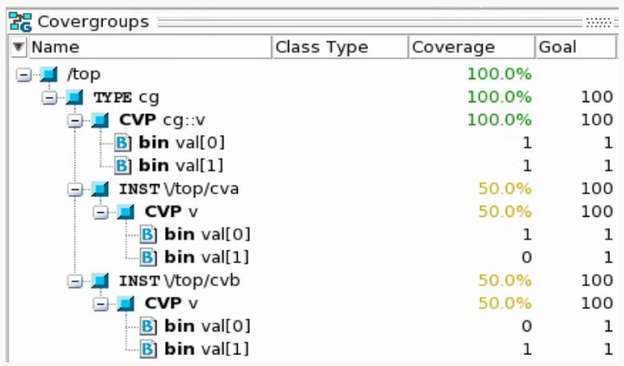
```
module top;
  covergroup cg (ref int v);
    option.per_instance = 0;
    type_option.merge_instances = 0;
    option.get_inst_coverage = 0;
    coverpoint v { bins val[] = { [0:1] }; }
  endgroup
  int a, b;
  cg cva = new(a);
  cg cvb = new(b);
  initial begin
    #1; a = 0; cva.sample();
    #1; b = 1; cvb.sample();
    #1; $display("cva=%.2f cvb=%.2f cva+cvb=%.2f",
                 cva.get_inst_coverage(), cvb.get_inst_coverage(),
                 cg::get_coverage());
  end
endmodule
```

Example Covergroup with Default Options

Covergroup with Default Options



option.per_instance=1

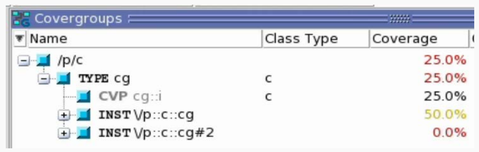


type_option.merge_instances=1

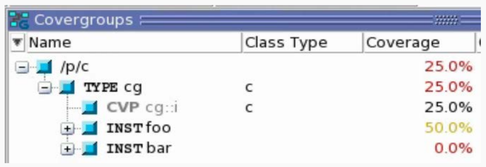
2) Naming Covergroups

```
package p;
  class c;
    int i;
    covergroup cg(string o_name);
      option.name = o_name;
      coverpoint i { bins ival[] = { [0:1] }; }
    endgroup
    function new(string cvg_name);
      cg = new(cvg_name);
    endfunction
    function void sample(int val);
      i = val;
      cg.sample();
    endfunction
  endclass
endpackage
module top;
  p::c cv = new("foo");
  p::c cv2 = new("bar");
  initial begin
    ...
end
endmodule
```

Adding User Names



Tool Named Instances



User Named Instances

3) Auto-bin Expansion

```
covergroup cvg;
  a_cov: coverpoint A iff (start) {
    bins a_1 = {1};
    bins a_2 = {2};
    bins a_3 = {3};
  }
  b_cov: coverpoint B iff (start) {
    bins b_1 = {1};
    bins b_2 = {2};
    bins b_3 = {3};
  }
  c_cov: coverpoint C iff (start) {
    bins c_1 = {1};
    bins c_2 = {2};
    bins c_3 = {3};
  }

  a_b_cross :cross a_cov, b_cov{
    bins a_1_b_1 = binsof(a_cov.a_1) && binsof(b_cov.b_1);
    // don't auto-expand cross-bins here
  }
  a_c_cross :cross a_cov, b_cov;
    // auto-expand cross-bins here
endgroup
```