## 功能覆盖率

 (2011-08-01 10:19:00)

[IMG_256转载▼](http://blog.sina.com.cn/s/javascript:;)

|  |  |
| --- | --- |
| 标签： [杂谈](http://search.sina.com.cn/?c=blog&q=%D4%D3%CC%B8&by=tag" \t "http://blog.sina.com.cn/s/_blank) | 分类： [ASIC](http://blog.sina.com.cn/s/articlelist_1035826164_7_1.html" \t "http://blog.sina.com.cn/s/_blank) |

1.       功能覆盖率是基于验证计划而设计的。当功能覆盖率高，而代码覆盖率低的时候，验证计划不够完备。反之，检查设计是否完整的实现了所有的功能。

2.       功能覆盖率在于收集信息而不是数据，coverpoint多了会影响执行的速度，如何关闭功能覆盖率的执行呢？

在默认的条件下，VCS输出simv.vdb目录，里面包含了code, assertion, functional coverage数据。代码覆盖率的数据分析，需要在编译的时候，增加-cm line+tgl等选项。功能覆盖率不需要编译选项。

-cm\_dir 编译，执行选项。改变simv.vdb目录

-cm\_name 编译，执行选项，改变测试的文件名。

使用urg命令，得到覆盖率的报告。

3.       功能覆盖率的语法：

Covergroup cg\_name;      //这里可以加@posedge clk，使其在每个时钟周期都自动采样

         Coverpoint\_name : Coverpoint var\_exprepression;

Endgroup;

这里的var\_exprepression可以是本地的变量，使用层次结构的变量，变量的一部分，表达式，甚至是ref变量。

Ref variable

Example:

**covergroup** (**ref** **int** r\_v) cg;  
    cover\_ref: **coverpoint**  r\_v;  
  **endgroup**

**通用的covergroup实例**

**covergroup**  cg(**ref** **int** array, **int** low, **int** high ) @(clk);  
    **coverpoint**// sample variable passed by reference  
      {  
       **bins** s = { [low : high] };  
      }  
  **endgroup**  
    **int** A, B;  
  rgc1 = new( A, 0, 50 );// cover A in range 0 to 50  
  rgc2 = new( B, 120, 600 );// cover B in range 120 to 600

4.       Coverage Filter

**covergroup** cg;  
    **coverpoint** cp\_varib **iff**(!reset);  // filter condition只有当!reset为真的时候，覆盖点才有效。  
  **endgroup**

**5.** **仓**

**Bins是声明语句而不是程序语句，需要使用{}包围起来，结尾没有；**

**一些示例：**

**{bins len[] = {[0:23]};}** **自动创建了24个仓，名称为len\_00,…,len\_17**

**{bins lo={[1:3]，5}；}** **创建了名为lo的仓，包含,1,2,3,5**

**{bins neg={$:-1};}  $表示极值。**

**{bins d = default;}** **剩余的所有值**

**{wildcard bins even={3’b??0}; }**

**{ignore\_bins hi={[6,7]};}** **忽略6，7这两个仓**

**{illegal\_bins hi={[6, 7]};}当出现这两个仓，将报错。**

**6.** **TRANSITION BINS**

**{bins** tran\_34 = (3=>4);} // Single Value Transition

**bins** tran\_345 = (3=>4>=5); **Sequence Of Transitions**

**bins** trans[] = (3,4=>5,6); **Set Of Transitions**

**bins** trans\_3 = (3[\*5]); **Consecutive Repetitions**

**bins** trans\_3[] = (3[\*3:5]); **Range Of Repetition**

**bins** trans\_3 = (1=>3[->3]=>5); **Goto Repetition，类似**1,6,3,6,3,6,3,5这样的序列

**bins** trans\_3 = (1=>3[=2]=>5); **Non Consecutive Repetition，类似**1,6,3,6,3,6,5这样的序列

7.    **交叉覆盖率**

**int** i,j;  
   **covergroup** ct;  
      **coverpoint** i { **bins** i[] = { [0:1] }; }  
      **coverpoint** j { **bins** j[] = { [0:1] }; }  
      x1: **cross** i,j;  
      x2: **cross** i,j {  
          **bins** i\_zero = **binsof**(i) **intersect** { 0 };  
      }  
   **endgroup**  
  
Cross x1 has the following bins:  
<i[0],j[0]>  
<i[1],j[0]>  
<i[0],j[1]>  
<i[1],j[1]>  
Cross x2 has the following bins:  
i\_zero  
<i[1],j[0]>  
<i[1],j[1]>

**8.** **Options**

**a)** **Weight，权重，默认值为1**

**b)** **Goal**

**c)** **Name**

**d)** **Comment**

**e)** **At\_least**

**f)** **Detect\_overlap**

**g)** **Auto\_bin\_max**

**h)** **Cross\_num\_print\_missing**

**i)** **Per\_instance**

**j)** **Get\_inst\_coverage**

**9.** **方法**

**a)** **Sample**

**b)** **Get\_coverage.计算当前的覆盖率值。这是一个静态函数，返回特定的覆盖率项的所有实例的覆盖率的总和。可是使用两种方法调用，类型：：get\_coverage或者实例名字.get\_coverage。可以有两个输入ref类型的参数，第一个返回覆盖到的仓的数量，第二个参数返回该覆盖项的仓的数量。**

**c)** get\_inst\_coverage得到某个实例的覆盖率值，只能使用**实例名字.get\_coverage调用。**

**d)** void set\_inst\_name(string)

**e)** void start()

**f)** void stop()

**g)** $set\_coverage\_db\_name ( name )，SV的系统函数。设置仿真的数据库文件名字

**h)** $load\_coverage\_db ( name )，SV的系统函数，加载仿真的数据库文件，用于累加覆盖率值。

**i)** $get\_coverage ( )，SV的系统函数，得到当前仿真环境内所有覆盖项的值。

**10.采样**

**a)** **使用sample方法**

**在Callback基类中，使用sample进行采样。将callback基类的实例放置在driver的类中。**

**在事务类中，进行采样。**

**在其他方便的场合使用sample方法。**

**b)** **使用事件触发的覆盖组**

**@(posedge clk);使用时钟边沿，每个周期采样**

**@(trans\_ready);等待某信号触发**

**@(event\_ok);等待某事件触发。**

|  |
| --- |
| vcs -lca +v2k -sverilog -cm line+cond+fsm+tgl -cm\_dir CovData/vcs/test\_1.cm +define+marco=VCS+ test\_1.sv ./simv -cm line+cond+fsm+tgl -cm\_dir CovData/vcs/test\_1.cm +ntb\_random\_seed=666666 2>&1 |tee log/vcs/test\_1.log urg -dir CovData/vcs/test\_1.vdb -metric group -report FcovReport/vcs/test\_1 -format text |