ECE 6745 Complex Digital ASIC Design Topic 8: Testing and Verification

Christopher Batten

School of Electrical and Computer Engineering Cornell University

http://www.csl.cornell.edu/courses/ece6745

Part 1: ASIC Design Overview



Coverge Design For Test: Scan and BIST **Testing and Verification Specification** Formal Verification Test Vector Equivalence (RTL Sim) **RTL** 0 Coverag Formal Verification Test Vector Equivalence (GL Sim) Timing, Power, Noise Analysis Gate Level Lavout vs Schematic Test Vector Equivalence (Circuit Sim) Layout Timing, Power, Noise Analysis DRC, ERC, Manufacturing Tests Scan Testing BIST Chip

Due to time constraints, we will just focus on coverage and DFT.

Coverage Types

Coverage = measuring progress to complete design verification

Bug rate

- Indirect way to measure coverage
- When bug rate sags, find different ways to test

Code coverage

Line, path, toggle, FSM

Functional coverage

- Which features have been tested?
- cover property to identify interesting signal values or sequences of signal values
- cover group samples data values and transactions



always @(posedge clk or negedge reset_1) begin
 q <= d;
 q_1 <= !d;
 end
endmodule</pre>

Functional Coverage

- You can can use a directed test for features in design
- You can use random testing to make writing tests easier
- How do you know you are testing every feature?
- Functional Coverage: measure of which design features are tested



Functional Coverage Strategies

Gather information, not data

- Do we really need to test all 4B values of a 32-bit address?
- Force constrained random testing into "interesting" areas
- Only measure what you are going to use in converage analysis
 - Gathering functional coverage data can be expensive
 - Think critically about what to monitor/record
 - Measuring completeness
 - "Are we there yet?"
 - Use code coverage then bug rate to help determine when verification is starting to converge



Functional Coverage Example

```
program automatic test(busifc.TB ifc);
  class Transaction;
    rand bit [31:0] data;
   rand bit [ 2:0] dst;
                                  // Eight dst port numbers
  endclass
  Transaction tr:
                                   // Transaction to be sampled
  covergroup CovDst2;
    coverpoint tr.dst;
                                    // Measure coverage
  endgroup
  initial begin
    CovDst2 ck;
    ck = new():
                                    // Instantiate group
    repeat (32) begin
                                    // Run a few cycles
      @ifc.cb;
                                    // Wait a cycle
      tr = new();
      'SV RAND CHECK(tr.randomize); // Create a transaction
      ifc.cb.dst <= tr.dst;
                                         and transmit
                                    11
      ifc.cb.data <= tr.data;
                                    11
                                         onto interface
      ck.sample();
                                    // Gather coverage
    end
  end
endprogram
```

Adapted from [Spear'12]





Design For Test: Scan and BIST •

Scan Testing

- 1. Set up the scan chain configuration.
- 2. Shift values into the active scan chains.
- 3. Exit the scan configuration.
- 4. Apply stimulus to the inputs and measure the outputs. 8. Exit the scan configuration.
- 5. Pulse clocks to capture the test circuit response.
- 6. Set up the scan chain configuration.
- 7. Shift values out of the active scan chains.



Adapted from [Kapur'17]



Automatic Test Pattern Generation





Linear-Feedback Shift Registers



7 1 1 repeats forever

0

0

0

Q₂, **Y**

0

()

0

Adapted from [Weste'11]



Design For Test: Scan and BIST

Acknowledgments

- [Spear'12] C. Spear and G. Tumbush, "SystemVerilog for Verification," 3rd ed, Springer, 2012.
- [Weste'11] N. Weste and D. Harris, "CMOS VLSI Design: A Circuits and Systems Perspective," 4th ed, Addison Wesley, 2011.
- [Kapur'05] R. Kapur and K. Brisacher, "Next Generation Scan Synthesis," COMPILER, 2005. https://www.synopsys.com/news/pubs/ compiler/art2_scansynthe-may05.html