VLSI Testing
(CMPE 646)
University of Maryland Baltimore County
Fall 2023

Project
Due: Dec 15, 2023

Final project of VLSI-Testing course compares test generation schemes for sequential circuits. In this project, you are to compare the random pattern test generation scheme to deterministic test generation scheme.

You are given a sequential circuit. In the first part, you compare random versus deterministic test patterns generation to detect all stuck-at faults in this circuit. In the second part of the project, you first insert flip-flops in a single scan chain and then compare the effectiveness of random and deterministic patterns with each other in testing the circuit.

Part-1: Sequential Circuit (No Scan Chain)

1-a) Deterministic Test Pattern Generation

In this part, you should generate a set of test patterns detecting all detectable faults in a given sequential circuit (using Synopsys Tetramax tool). Although the fault coverage of deterministic test-pattern generation schemes are high, these schemes suffer from high test-generation time.

The given circuit (*.v) is a sequential circuit. Using TetraMAX, perform full sequential test generation (no scan-chain is included in this circuit) and evaluate the following parameters.

- Total number of faults
- Fault-classes statistics (Number of detectable faults, Undetectable faults, etc)
- The number of test patterns generated to detect all stuck-at faults
- Fault coverage
- Time taken for test generation by the tool

Note that to perform sequential test generation, you can use the following commands

set_atpg –full_seq_atpg
run_atpg

1-b) Random Test Pattern Simulation

In random test pattern simulation, randomly generated test vectors are applied to the circuit. First random patterns are generated. Then, the circuit is fault simulated using these patterns to extract the faults detected by the randomly-generated patterns.

1. Using a tool/ language of your choice, generate a set of random test patterns for the circuits. Note that, the size of a single test pattern depends on the number of the primary inputs of the circuit.

For this project first generate a set of test vectors whose size is 30% of the number of patterns generated by the tool in the part-1, i.e., if Tmax generated 2000 patterns to detect all faults of the given circuit (in part 1-a), generate 600 randomly generated patterns for this part of the project. Perform fault simulation on the provided circuit using these randomly generated patterns and report:

- Total number of faults
- Fault-classes statistics (Number of detectable faults, Undetectable faults, etc)
- The number of test patterns you used for fault simulation
- Fault coverage
- Time taken for fault simulation by the tool
- List of the faults that were not detected by the random patterns.

1-c) Random Test Pattern Simulation

Repeat part 1-b two more times with more number of random patterns. One time with a set of random vectors whose size is 50% of the number of patterns generated by the tool in the part 1-a and the second time with a set of random vectors whose size is 80% of the number of patterns generated by the tool in the part 1-a.

Then, perform sequential test generation for the faults that were not detected by the randomly generated patterns (for all three parts) and report:

- Total number of faults considered in this step
- Fault-classes statistics (Number of detectable faults, Undetectable faults, etc)
- Fault coverage
- Time taken for test generation by the tool

Part-2: Sequential Circuit (One Scan-Chain)

Convert the flip-flops in the given circuit (*.v) to scan flip-flops (S_DFFX1) described in the library (mylib.v). The scan flip-flops contain a scan-input and scan enable signals. You need to make the necessary connections, especially connecting the output of the previous flip-flop to the scan-input of the flip-flop that follows. Do not forget to make necessary changes in the input and output parameter list (add two primary inputs (**SI, SE**) and one primary output (**SO**) to the circuit. The **SO** signal is fed with the output of the last flip-flop in the scan-chain and **SI** feeds the scan-input of the first flip-flop in the scan-chain).

2-a) Deterministic Test Pattern Generation

Insert all the flip-flops of the circuit in a scan-chain using a simple code of your choice. Then use TETRAMAX to perform ATPG for this circuit and report the following:

- Total number of faults
- The number of test patterns generated to detect all stuck-at faults
- Fault-classes statistics (Number of detectable faults, Undetectable faults, etc)
- Fault coverage
- Time taken for test generation by the tool

2-b & 2-c) Random Test Pattern Simulation

Consider the circuit with the single scan-chain (generated in section 2-a) and repeat what you performed in **Section 1-b** and **1-c** for this circuit. Note that you need to generate random input patterns for both primary inputs and initial values of flip-flops.

Part-3: Sequential Circuit (Multiple Scan-Chain)

Repeat Part 2 and this time distribute flip-flops in 10 scan-chains.

Deliverables:

1. A complete package including your circuit, scripts, results, …
2. Demo: You should give a demo of the project.
3. Project report: Write an extensive report comparing the results obtained along with your observations. Use tables, graphs, etc to elaborate on your observations. Compare part 1, 2, and 3 with each other in terms of number of test patterns, fault coverage, test generation time, and other metrics that you think are appropriate. Also in each part compare section (a) and section (b) and section (c).

Note 1: To define the clock signal you need to add the following command to your scripts:

add_clocks 0 { clk_sig } -shift -timing { 100 50 80 40 }

Note that clk_sig is the name of clock signal in your circuit. The parameters are as below:

1. The '0' indicates an active-high clock. (off state is 0).

2. -timing {period LE TE measure_time} defines the test cycle period associated with the clock. LE is the time of the leading edge of the clock, while TE is the time of the trailing edge of the clock. The measure time option specifies the time within the test cycle at which the design's output pins is measured.

Note 2: Add the following command to your scripts to avoid inserting faults inside the flip-flops.

add_nofaults -module flip-flp-type

In this command ***flip-flp-type*** is the type of the flip-flop you use in your design (e.g. DFFX1)

Note 3: In part-b you should define your scan structure in the script. You can use the following commands in your scripts:

add_scan_chain chain1 scan_data_in scan_data_out
add_scan_enables 1 scan_enable # means scan_enable is '1' during the test mode
add_pi_constraint 0 SI
add_pi_constraint 0 SE

You can find the details of these commands in the Tmax tutorial posted along with Lab assignments.

Part-4: Sequential Circuit (Partial Scan)

Insert 30% of flip-flops in one scan chain using a simple code of your choice (select them randomly). Then use TETRAMAX to perform ATPG for this circuit and report the following:

- Total number of faults
- The number of test patterns generated to detect all stuck-at faults
- Fault-classes statistics (Number of detectable faults, Undetectable faults, etc)
- Fault coverage
- Time taken for test generation by the tool

Repeat this part-4. This part place another 30% of flip-flops (not included in the first set) in the scan chain and answer all questions.

Note that to perform partial scan chain, you should look for required commands in Tetramax.