Scan Testing

Digital Systems Testing and Testable Design- ECE 438 - Fall 1397 Homework 5

Mohammad Hashemi
Univercity of tehran
Tehran,Iran
md.hashemi@ut.ac.ir

***Abstract*—In this document we are going to design a four function module that calculates SIN and COS and SINH and COSH by using Taylor series. Then we are going to test it with scan method and apply it by Verilog simulation and achieve our goal coverage.**

Cos, COSH, four function, scan method

I. Introduction

In this paper first we are going to design and unfold our circuit. So we apply our Verilog code to our net generator and reveal our circuit net-list. Then: (1) we unfold our net-list to remove our register by adding some inputs and outputs to make our circuit combinational after that (2) we try to generate an algorithm that generates n sets of tests. (3) Now it is time to apply our test with partial scan. (4) Then we made our virtual tester with Verilog and test our module. (5) Finally we calculate our coverage.

II. Design Phase

A. *Unfolding the Circuit*

First, we remove our registers and replace them with inputs and outputs. Our circuit has 36 registers so we have to add 36 inputs and outputs. The changes are in netlist_forFuncUnfold.v.

B. *Collapsing*

Our circuit is now unfold and turn to a combinational circuit and there is no sequential problem and we can behave our circuit as a combination circuit. So we apply our collapsing with Faultinjection PLI and collapse our faults. There is a problem that collapsing in this phase will collapse whole circuit and may inject some faults on a wire that we do not mention it but if we fine a test set that tests our Adder and Multiplier and table it will also test rest of our circuit and give us coverage.

III. Test Set Generation

As our circuit can calculate sin and cos we can use the formula below to check our multiplier and adder.

$$\mathrm{Sin}(x) = \mathrm{Cos}(90 - x) \qquad (1)$$

A. *Determenistic test set*

The first way to apply our tests is to use our table and put our table value to the formula (1). Then use control value of sin and cos and compare the results. After that we apply n of our test to the circuit and achieve our coverage.

B. *Random test set*

The other way of applying our test is use random test and we can calculate the coverage with the factor of n.

C. *Check table generation*

If we apply an x to our circuit we expect the output will be result of the x Taylor series. We can use a Verilog code or C++ code to make it and then read from the file that generated by or code to test our circuit by compare the results.

IV. Partial Scan

For apply partial scan we set our input to any random amount and apply clock after the specified clock cycle we expect the output to be Taylor series of our input. For having better timing we add some redundancy include 4 pins before and after each module such as adder and multiplier and apply 0 bit and check the output pin. For adder if the output sets to 1 there is no fault but if it remains 0 there is a fault accrued in adder. For multiplier after apply 0 and one clock cycle the output must be 0 else there is a fault accrued. This method is fast but needs pins and may not possible in real physical chips to probe a part of circuit or have a pin out of the chip. The pin locations are shown in Fig.1.

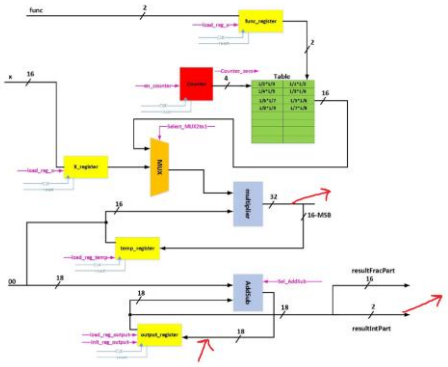

Fig.1 pins location

We can add a multiplexer that has NBART and set our inputs manually by select T selector. The new circuit schematic shown in Fig.2.This way we can make our circuit testable.

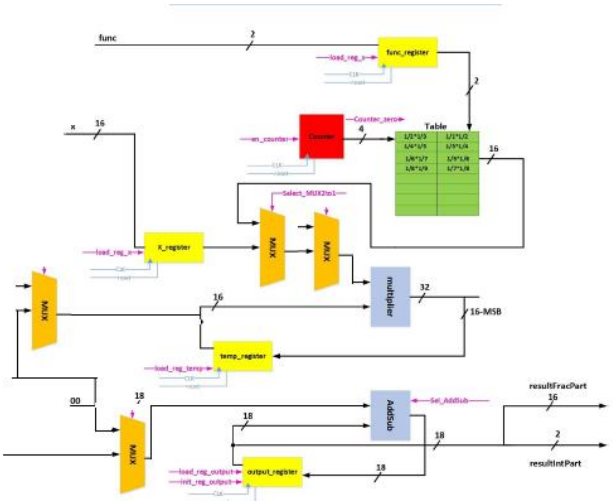

Fig.2 Testable circuit by adding MUX

V. VIRTUAL TESTING

For testing our circuit we have to use Verilog simulation. We deploy a test bench and apply our test. To make our circuit more testable we have to change our circuit inputs and add 3 more inputs of 8 bits to each module such as adder and multiplier. After that we apply some random test to check our adder and multiplier if they operate correct then we select our table and apply our table amounts to our circuit and check the output this method can prove our table operate correct if our multiplier and adder both work correctly. At the end we apply collapsing to our netlist that generate by net-gen and apply our test to obtain coverage. We can see the reports in reports category.

VI. REPORTS

After create our virtual tester we apply the collapsing function to our net-list and apply our random tests. The coverage is shown at Fig.

REFERENCES

[1] Proffesor Navabi Fall 2017 Lectures