Object Oriented Implementation of Combinational Controllability and Observability Algorithms

Vaishali Dhare1 & Usha Mehta2
1,2Institute of Technology, Nirma University, Ahmedabad, India

Abstract: Testability measures are controllability and observability. Controllability guides the test generation algorithms while setting a value to primary input (PI) in line justification problem. When more than one path are available for error propagation Observability finds which path is to be select. This paper describes controllability functions (CC0, CC1) and observability function (ob) for combinational circuits. The controllability functions and observability function are implemented using object oriented language C++ for circuit having 2-fanin-fanout, 8-fanin-fanout, and 10-fanin-fanout combinational circuit. ISCAS netlist format is used. The ISCAS 85 benchmark c17 circuits are used for analysis. The flow charts and results for various combinational circuits are given in this paper.

Keywords: Testability Measures, Controllability, Observability, ISCAS.

1. INTRODUCTION

ATPG (Automatic Test Pattern Generation) is generally guided by the testability measures to choose a decision during justification and propagation [3]. Several heuristics, such as distance based testability measures, probabilistic measures, SCOAP measures [4], super-gate based measures, and correlation-based measures were introduced as testability measures. These measures serve as heuristics and represent the relative difficulty of justifying a gate value to a control input or propagating a fault effect to an observe point. The search process of any Test Generation algorithms involves two important decisions. The first one being to select one of the several unsolved problems existing at a certain stage in the execution of the algorithm, the second type is to select one possible way to solve the selected problem.

Selection criteria differ mainly by the cost functions they are used to measure “difficulty”. Typically cost functions are of two types:

1. Controllability: For a digital circuit it is defined as the difficulty of setting a particular logic signal to state 0 or 1.

2. Observability: For a digital circuit it is defined as the difficulty of observing the state of a logic signal. Controllability measures can be used both to select the most difficult line-justification problem, and then to select among the unspecified inputs of G the one that it is easiest to set to the controlling value of the gate. Observability measures can be used to select the gate from the D-frontier whose input error is the easiest to observe. Goldstein was the first one to implement a computer program to calculate those controllability and Observability values..

Testability analysis usually has two significant attributes [14]:

- It involves topological analysis, but no test vectors. It is static type of analysis;
- It has linear complexity, because otherwise testability analysis is pointless and one might as well automatic test pattern generation (ATPG) or fault simulation.

In this paper we discuss the testability measures combinational controllability 0-CC0, combinational controllability 1-CC1, and combinational observability –CO (Ob). All above testability measures are implemented using C++ for 2-fanin-fanout, 8-fanin-fanout and 10-fanin-fanout combinational circuits.

2. CONTROLLABILITY

If a and b are inputs of a gate and z is the output then the following are the controllability 0-CC0, controllability 1-CC1 values for input a and b. Values for different gates are shown in figure 1.

If net is fanout branch then the Controllability 0-CC0 and Controllability1-CC1 values are the same as the stem Controllability 0-CC0 and Controllability1-CC1 value respectively.

The controllability values for each node of C17 ISCAS85 Benchmark are shown in figure 2. Each line is labeled with (CC0, CC1) pair to show its controllabilities.

*Corresponding Author: 1vaishalidhare@gmail.com,
2usha.mehta@nirmauni.ac.in
Since all are NAND gates the value for Controllability 0-CC0 and Controllability 1-CC1 are calculated as follows:

CC0 (Z) = min (CC0 (a), CC0 (b)) + 1
CC1 (Z) = CC1 (a) + CC1 (b) + 1

CC0 (Z) = CC1 (a) + CC1 (b) + 1
CC1 (Z) = min (CC0 (a), CC0 (b)) + 1

CC0 (Z) = min (CC0 (a), CC0 (b), CC0 (a) + CC1 (b)) + 1
CC1 (Z) = CC0 (a) + CC0 (b), CC0 (a) + CC1 (b) + 1

We proceed from primary inputs up to primary output in the following ways:

CC0 (10) = CC1 (1) + CC1 (8) + 1 = 3
CC1 (10) = min (CC0 (1), CC0 (8)) + 1 = 2
CC2 (11) = CC1 (9) + CC1 (6) + 1 = 3
CC1 (11) = min (CC0 (9), CC0 (6)) + 1 = 2
CC0 (16) = CC1 (9) + CC1 (6) + 1 = 4
CC1 (16) = min (CC0 (9), CC0 (6)) + 1 = 2
CC1 (20) = CC1 (15) + CC1 (7) + 1 = 4
CC1 (22) = CC2 (15) + CC2 (7) + 1 = 5
CC0 (19) = CC1 (15) + CC1 (7) + 1 = 4
CC2 (22) = CC1 (10) + CC1 (20) + 1 = 5

Figure 1: Controllability 0-CC0 and Controllability 1-CC1 Values for different Gates

3. OBSERVABILITY

The observability value calculation for various gates is shown in figure 3.

Figure 2: ISCAS85 C17 Benchmark Circuit along with Controllability 0-CC0 and Controllability 1-CC1 Values
4. IMPLEMENTATION

The testability measures controllability 0 (CC0), controllability 1 (CC1) and observability (ob) is calculated using C++ language and stored in the form of array contains net number, CC0, CC1, Ob. The program is generic which means used for any combinational circuit provided that the circuits should be in ISCAS (International Symposium on Circuit And systems) netlist format. In main program two functions are developed, one is controllability function to find CC0, CC1 values and other is observability function to find the observability values. Firstly We Initialized the output array which contains netno, controllability 0 (CC0), controllability 1 (CC1) and Observability (observ), c_0 (CC0) and c_1 (CC1) to “0” since minimum value of it is “1” and Observability to “–1” since it has minimum value as ‘0’. Search primary input (PI) from 3rd array of the netlist which contains net numbers which are primary input, assign c_0 and c_1 to net number i.e. PI as (1, 1). Call controllability function for each net. The Controllability function is developed separately, shown in figure 5. The C part in the flow chart of figure 4 shows the controllability function. To check whether controllability of all net are calculated or not, check whether output array for c_0 and c1 are non zero or not. If c_0 and c_1 is found “0” then repeat the process until c_0 and c_1 contains non zero value. Search primary output (PO) from 4th array of netlist which contains “0” value for primary output. Assign observability ob to “0”. Call observability function (ob) for each net. Observability function is developed separately shown in figure 6. The O part of figure 4 shows the Observability function. To check whether the observability for all net are calculated or not, check whether the output array for obs contains -1 or not. If it contains “-1” then repeat the process until it becomes non “-1” value. Write netno c_0, c_1, ob into output text file. The output text file contains the final result stored in the form of array which is discussed in results section.

The controllability value for gate is calculated using calculations given in figure 1. We proceed from primary input which are set 1 for both the values CC0, CC1 initially upto primary output. If the primary output reaches and out if output array contains non zero value then the function exit.
The observability function is shown in figure 6. Firstly it finds out the net indices. If that net is primary output then return since we already stored Observability to “0” value. If it not primary output then it may be from or gate. If it is gate then it finds two inputs to that gate.

Then function will calculate the observability for net under consideration using following rules

If gate type is

- ‘and’ : \( \text{ob} = \text{obs(output index)} + c_1 \text{ sip} + 1 \)
- ‘nand’ : \( \text{obs(e)} = \text{obs(output index)} + c_1 \text{ sip} + 1 \)
- ‘or’ : \( \text{obs(e)} = \text{obs(output index)} + c_0 \text{ sip} + 1 \)
- ‘nor’ : \( \text{obs(e)} = \text{obs(output index)} + c_0 \text{ sip} + 1 \)

Where e is net index for the net under consideration, output_index is the index for the net which is gate output, \( c_1 \text{ sip} \) is the controllability 1 of second input to that gate and \( c_0 \text{ sip} \) is the controllability 0 of the second input to that gate.

If net under consideration is stem then assign minimum observability out of the two observabilities for branches to stem. For that purpose we have developed simple minimum function separately to find minimum of among all fanout branches.

5. RESULTS AND DISCUSSION

The snap shot of text file is shown in figure 9 which is generated output of the program written in C++ for ISCAS 85 C17 benchmark circuit.

The first column of output windows shown in figure 9, 10, 11 contains the net number, second column contains controllability 0-CC0 value, third column contains controllability 1-CC1 value and forth column is for observability ob for corresponding net number.
6. CONCLUSION

The testability measures Controllability and Observability are successfully implemented using C++ language. The programs are generic, we can calculate testability measures for any combinational circuit with 2-fanin-fanout, 8-fanin-fanout, and 10-fanin-fanout gates. The results for ISCAS 85 C17 circuit matches with manual calculation. The sample circuits of 8-fanin-fanout, 10-fanin-fanout are taken for analysis purpose, the results for these circuits are matches with manual calculation of testability measures.

REFERENCES