Test Generation

Design a circuit and apply a pre-calculating method and test

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

***Abstract*—In this document we are going to design a circuit that return max absolute of three different 4-bits inputs as output. Then we are going to discuss about a new pre-calculation method based on Statistics and Probability that help us reduce or testing vectors.**

Max absolute, Pre-calculation,

I. Introduction

In this document, first we are going to show our circuit RTL schematic and describe how out circuit operate (1) then we apply some deterministic inputs and check the result (2) after that we present a new pre-calculation test method and discuss about how it works (3) at the end we apply our test vectors based on our pre-calculation method and compare the results (4).

II. Schematic and Opreration

A. Schematics

First, we draw our hardware circuit schematic based on our Verilog codes in RTL. Our schematic is shown in Fig.2 and our positive maker module is shown in Fig.1.

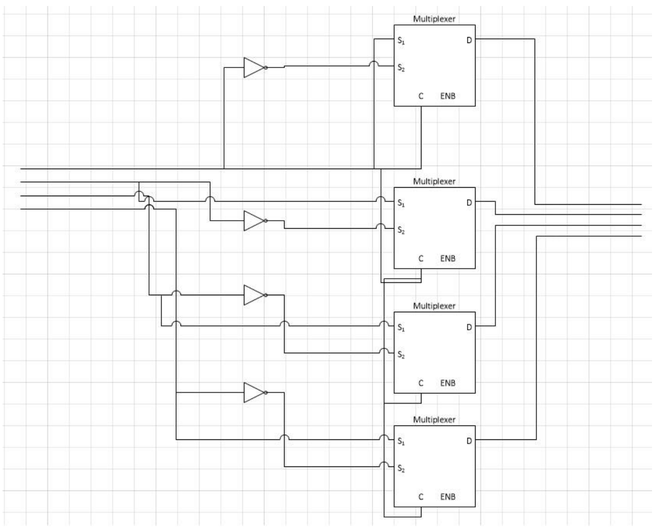


Fig.1 Positive Maker Schematic

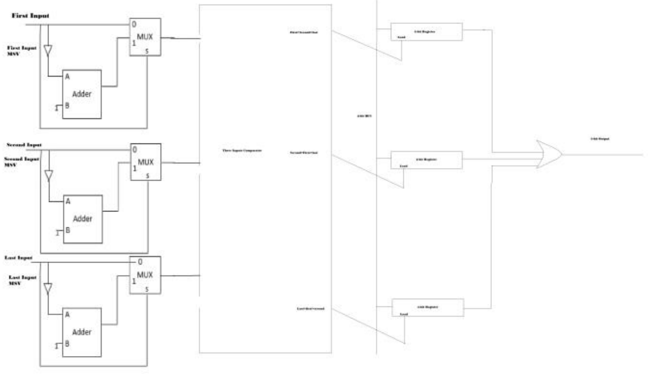


Fig.2 Circuit Schematic

B. Opreration

As we see we have 3 different 4-bits inputs in 2's complement. First we have to make all of our input a positive number. To make this thing happen we have to check our sign bit. If its sign bit is zero then we throw our number as our output but if our sign bit is 1 then we have to invert all bits and add our number with one.

III. Testbench and Test Phase

After describing our circuit and make it a module in Verilog it is time to make a test bench and test our module so we apply three different inputs : 1.(- 3) , 2.(7) , 3.(-2) and absolutes are 3 , 7 , 2. We expected our module choose (7) and return it.

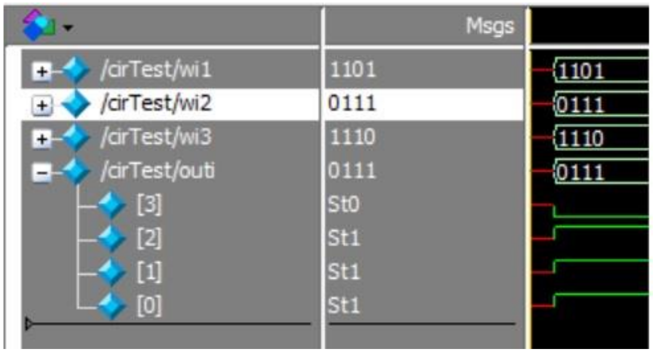
Our numbers 2's complement is shown below:

(-3) = 1101

7 = 0111

(-2) = 1110

The output waveform is shown in Fig.3.



```
                 Msgs
/cirTest/wi1     1101    1101
/cirTest/wi2     0111    0111
/cirTest/wi3     1110    1110
/cirTest/outi    0111    0111
  [3]            St0
  [2]            St1
  [1]            St1
  [0]            St1
```

Fig.3 Test Bench output waveform

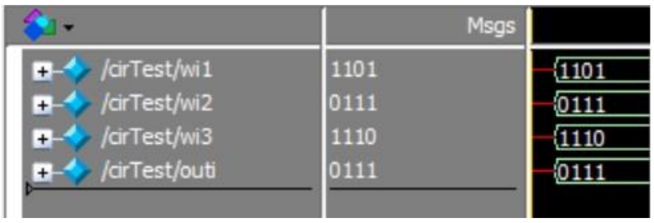
As we see our module works right and set our output to 7 now let’s try a negative number. This time we apply -7, 3, and 2.

(-7) = 1001

3 = 0011

2 = 0010

We expected our module returns 7. The result is shown in Fig.4.



```
                 Msgs
/cirTest/wi1     1101     1101
/cirTest/wi2     0111     0111
/cirTest/wi3     1110     1110
/cirTest/outi    0111     0111
```

Fig.4 Output waveform

IV. Pre-Calculation Method

After design and test our module we are going to describe our algorithm of new pre-calculation method. We consider any test (purely random test) can detect each fault with chance of 50%. Our most important factor here is our number of lines that we show them with n. We have $2 \times n$ faults includes s@1 and s@0 for each line. To calculate our coverage we need our test detects m of n Faults. If we increase the number of our test vectors our chance to achieve our goal coverage will increase. For each test vector we have formula below:

$$p(C) = \binom{n}{m} p^m (1-p)^{n-m}$$

For more vectors our probability is fallow formula below:

$$p(Ct) = p(C1) \Delta p(C2) \Delta p(C3) \Delta ...$$

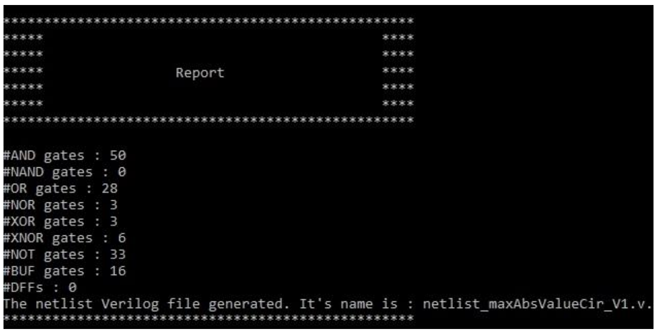
$$p(C1) \Delta p(C2) = p(C1) + p(C2) - p(C1) \cap p(C2)$$

For simulating this formula we use Math lab and plot the result. Our results are shown in Fig.5. As we know the third term of our formula is very low so we can calculate approximately by removing this term and make our function linear. By removing that term we have:

$$p(Ct) = p(C1) + p(C2) + p(C3) + ...$$

V. Pre-Calculation Method

After pre-calculation we apply our test vectors to our circuit in Verilog and compare our coverage. First we apply our circuit to NetlistGen_main.exe and recover our netlist (Fig.6).



```
**************************************************
*****                                         ****
*****                                         ****
*****                Report                   ****
*****                                         ****
*****                                         ****
**************************************************

#AND gates : 50
#NAND gates : 0
#OR gates : 28
#NOR gates : 3
#XOR gates : 3
#XNOR gates : 6
#NOT gates : 33
#BUF gates : 16
#DFFs : 0
The netlist Verilog file generated. It's name is : netlist_maxAbsValueCir_V1.v.
**************************************************
```

Fig.6 Netlist Generated by NetlistGen_main.exe

We have 246 wires (Lines). We apply n=246 and coverage 80% that means our m is 197. After that we apply one test vector and we hope our coverage return 80% with chance of . Results are in Fig.7.

VI. Refrences

[1] Professor Navabi Lectures