

ELEC 7250 VLSI Testing

Final Project: Logic Simulation and Fault Diagnosis

Andrew J. White

- I. Introduction
- II. Purpose
- III. Process
 - a. Compiler
 - b. Fault simulation
 - c. Fault diagnosis
- IV. Results
- V. Conclusion
- VI. References

- I. Introduction

Technology for semiconductor manufacturing has progressed rapidly over the last few decades. With the increase in logic gates, test simulation along with diagnosis has become a very important issue in VLSI testing. Simulators, along with diagnostics software, needs to be efficient along with accurate as Moore's Law continues to hold true.

II. Purpose

This project targeted several areas in VLSI testing including compilation, simulation, and diagnosis. A compiler was to be written to flatten a hierarchical bench format circuit while other programs performed simulation and diagnosis of the combinational circuit.

III. Process

a. Compiler

Flattening the hierarchical bench format was achieved by using m-files generated with Matlab. The bench format is used to describe primarily ISCAS'85 circuits in a similar way to many other description languages such as VHDL and Verilog. An example bench format circuit for a 1-bit full adder is shown below in

Figure 1.

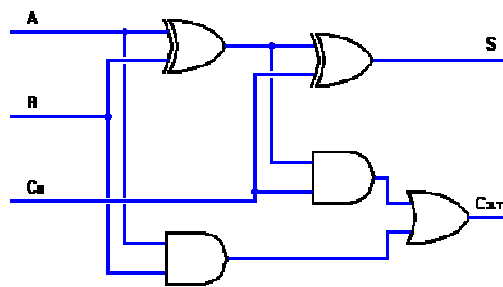


Figure 1. Above: 1-bit Full Adder Circuit
Right: Bench format for above circuit

```
#FA, 1-bit full adder
#3 inputs
#2 outputs
#0 inverter
#5 gates (2 XOR, 2 AND, 1 OR)
INPUT(A)
INPUT(B)
INPUT(CIN)
OUTPUT(S)
OUTPUT(COUT)
1 = AND(A,B)
2 = XOR(A,B)
3 = AND(C,2)
S = XOR(2,CIN)
COUT = OR(3,1)
```

The compiler was used to not only flatten hierarchical bench format descriptions but also to generate a simulation table which was used for fault simulation along with fault diagnosis.

Three different .m files were actually created to generate the flattened bench format file. The first .m file, convert.m, would take the input bench files and replace all input and output names with numbers. This enables me to still be able to create hierarchical bench format files using names for the inputs and outputs which make understanding the file much easier. Once convert.m converted the file, compiler.m would compile the file and generate the simulation table. Compiler.m used a function called replace, replace.m, which was used in replacing component names in the hierarchical description with the actual component. A simulation table for a C17 circuit is show below in Figure 2.

INPUT: 1	Gatename: NAND2	-----
INPUT: 2	Fanin: 3 6	Gatetype: NAND
INPUT: 3	Fanout: 11	Gatename: NAND5
INPUT: 6	-----	Fanin: 10 16
INPUT: 7	Gatetype: NAND	Fanout: 22
OUTPUT: 22	Gatename: NAND3	-----
OUTPUT: 23	Fanin: 2 11	Gatetype: NAND
Gatetype: NAND	Fanout: 16	Gatename: NAND6
Gatename: NAND1	-----	Fanin: 16 19
Fanin: 1 3	Gatetype: NAND	Fanout: 23
Fanout: 10	Gatename: NAND4	-----
-----	Fanin: 11 7	
Gatetype: NAND	Fanout: 19	

Figure 2. Simulation Table for C17 circuit

b. Fault simulation

The fault simulator was written using C programming. Using the simulation table generated from the compiler, the simulator would take the test vectors and apply them to the primary inputs. The simulator would then compute the outputs for the circuitry and propagate these to the primary outputs. Each gate was grouped into a level and the output was computed for each level. For Figure 3 shown below, in level 1, all of the primary inputs would be applied to gates 6 and 7 and their outputs would be determined. This would continue until all levels were processed.

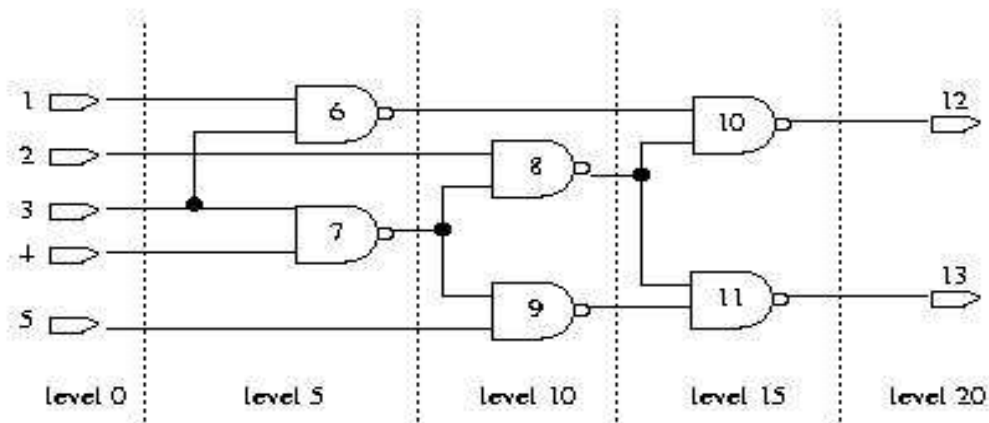


Figure 3. C17 Level Description

The outputs of the simulation would then be compared with the expected outputs. Any difference between the simulated and expected outputs would yield a fault and the corresponding test vector along with failing primary output would be noted.

c. Fault diagnosis

A widely used fault diagnosis method is effect-cause diagnosis.

Effect-cause diagnosis is the process of tracing the path from the incorrect

primary output to the origin. Although path tracing is not very practical for industrial uses because of the many fault candidates it generates, it is somewhat sufficient for this project. By back-tracking through the circuit, the faulty wire or gate which caused the incorrect output for the circuit should be fairly accurate. A proposed algorithm along with a rating procedure used to ‘guess’ which gate/wire is faulty can be found at [2].

IV. Results

Several results were gathered for measuring performance of the simulator designed. The first set of results is shown below in Figure 4 and represents the computational time based on the number of input vectors. The results are almost perfectly linear.

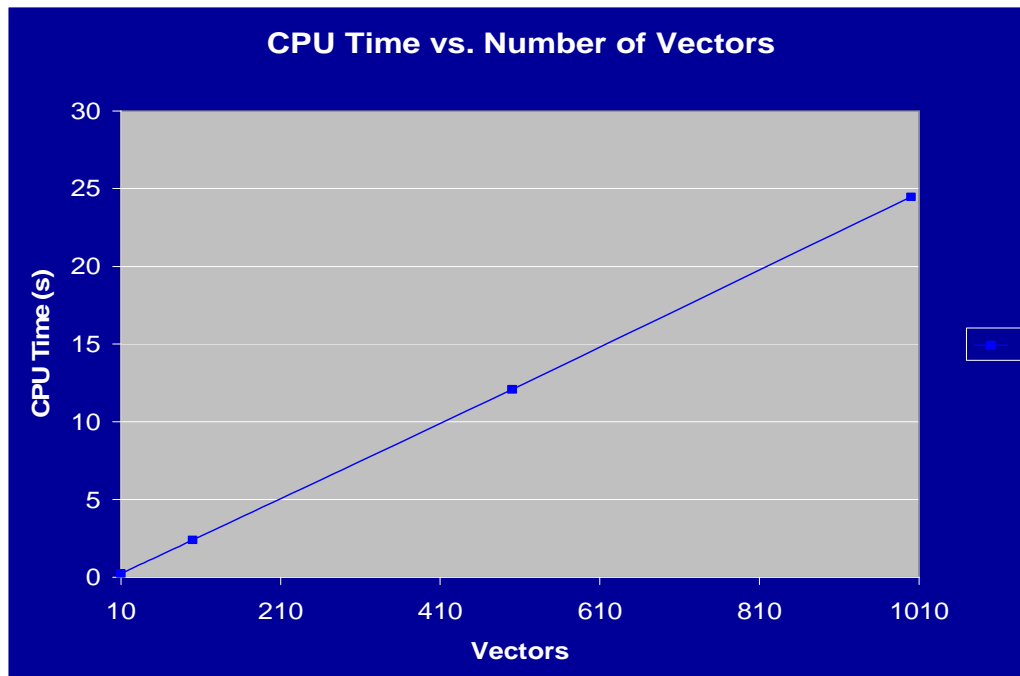


Figure 4. CPU Time versus Number of Input Vectors

The next set of results, shown below in Figure 5, depicts the computational

time based on the size of the circuit. The amount of gates in the circuit definitely has an impact on the computation time.

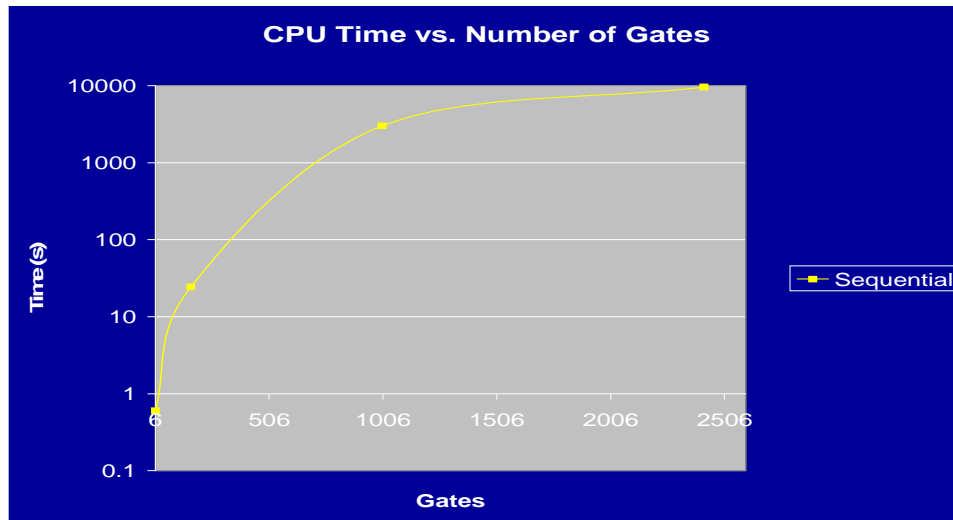
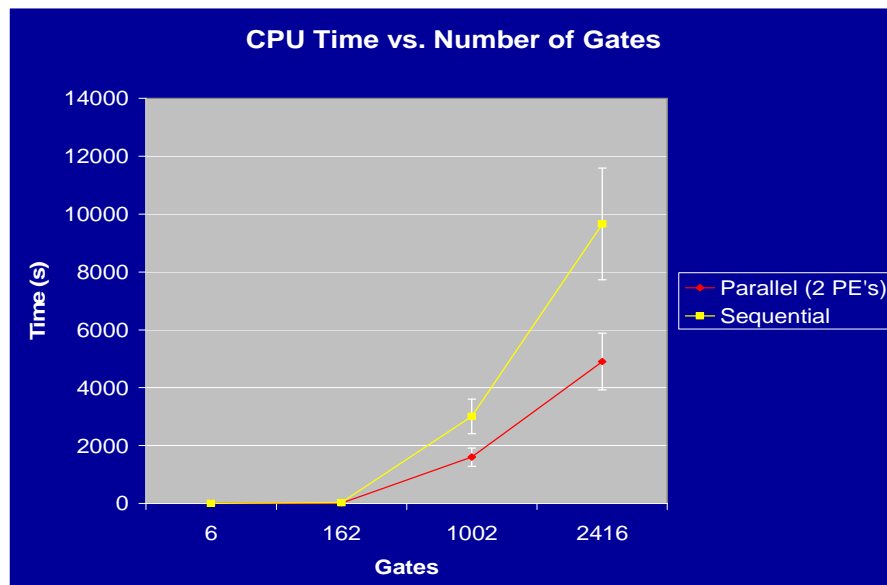


Figure 5. CPU Time vs. Number of Gates.

Due to the high computation times for larger circuits, the simulator was parallelized to achieve faster results. The main processing element, PE₀, would read in the simulation table and distribute it to the other PE's along with an evenly divided amount of input vectors. All of the PE's would then perform the simulation using the same sequential algorithm in parallel. Performance figures for this approach are shown below in Figure 6.



V. Conclusion

Writing a compiler to handle a hierarchical description of a circuit can be quite a complex task in itself. The addition of fault simulation can also be cumbersome. However, accurate fault diagnosis is the hardest part of the problem and much research has been and will be done in this area. The minimal results obtained for such a simple simulator really illustrate the importance of an efficient algorithm along with parallel processing to minimize computation times.

VI. References

1. [Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits](#) . Bushnell and Agrawal, Springer, 2000.
2. An improved fault diagnosis algorithm based on path tracing with dynamic circuit extraction. Shigeta, K., Ishiyama, T. [Test Conference, 2000. Proceedings. International](#) 3-5 Oct. 2000 Page(s):235 – 244.
3. [Comprehensive Fault Diagnosis of Combinational Circuits](#). Lavo, D. Dissertation University of California: Santa Cruz. September 2002.