ELEC 7250 PROJECT REPORT:
LOGIC SIMULATION AND FAULT DIAGNOSIS.

JINS DAVIS ALEXANDER.

Objective.

- To implement a logic simulator to verify combinational circuits given a set of input vectors and the expected output responses.
- To introduce a design error in the circuit and list the failing vectors and primary outputs where the errors are observed.
- To diagnose the design error.

Algorithm.

- Each primary input, primary output and gate are represented as a structure or a node.
- The primary inputs contain input name and input vector values.
- The primary outputs contain output name and the expected output response.
- Gate nodes also contain the following information : gate name , gate type , fan – in list, fan – in values , number of fan-in's, fan-out name , fan – out value
- Simple search method was used to link each gate to each other according to the given net list and hence produce a configuration of net connections. For every gate, the gate fan – in list was searched for a match with the fan – out names of the remaining gates. If a match was found then the latter gate was adjudged to be connected to the former.
- The simulation procedure can be understood from the detailed flowchart given in Fig 1.

Implementation and User Information.

The benchmark net list (hierarchical or flattened) is first converted into a simulation file that lists all information about the input, output and gates (including fan – in and fan – out information). This was implemented in matlab. The logic simulation and diagnosis has been implemented in C. The user will have to give three files as inputs:

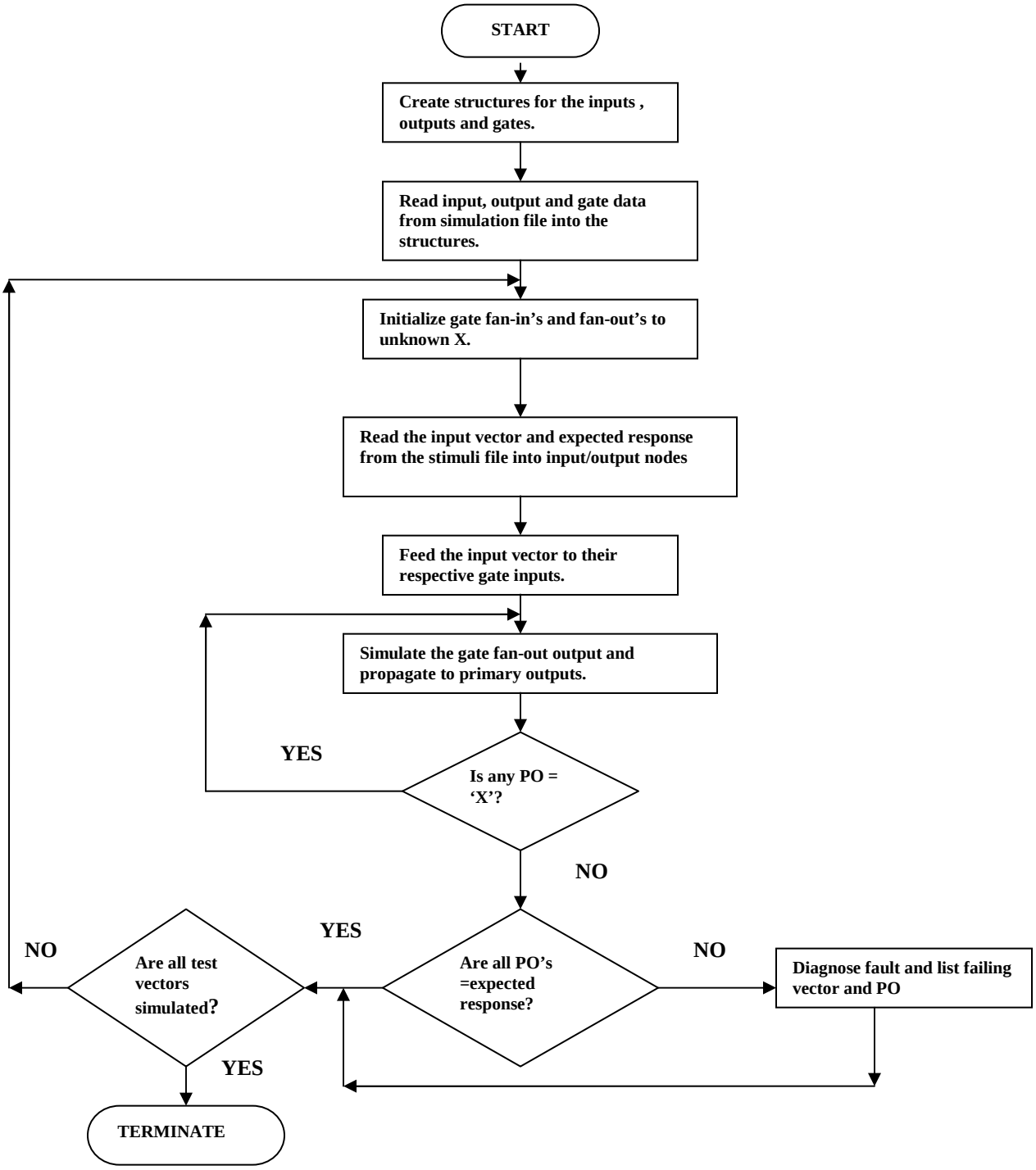
1. Simulation file.
2. Stimuli file (which contains the input vectors and the expected responses).
3. Fault list (fault dictionary for diagnosis).

The simulator provides the following results:

1. Any failing vectors, erroneous primary outputs and the expected response with the simulated response.

2. The possible error paths from the erroneous primary output back traced to the primary inputs.
3. Possible faults determined from the fault dictionary.
4. CPU Execution Time.

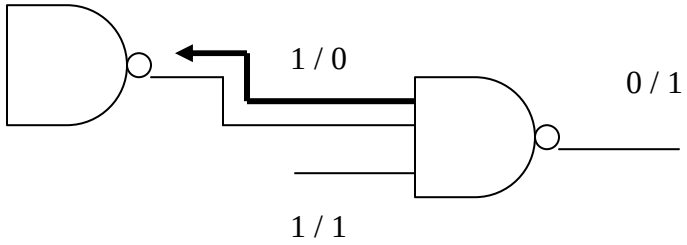
Fig 1. Flow Chart:



Design Verification and Diagnosis.

Back trace:

A method for diagnosis explored in this project was the means of back tracing from the erroneous primary outputs back to the inputs. By doing so, probable erroneous paths can be found. For example consider the following NAND gate:



Here for the good circuit we get a ‘0’ at the output of the NAND gate. For that both the fan – in’s should have a value of ‘1’. For the faulty circuit however the output of the NAND gate is a ‘1’. Thus at least one of the fan –in’s has a value of ‘0’ which is a faulty value. Hence we search which fan – in has the faulty value ‘0’ and trace that path. In the above case that would be path 1 and this is an erroneous path. Thus by comparing the expected response with the actual simulated response and back trace we can find all the probable erroneous paths.
Since there could be a number of paths to a given erroneous PO, sometimes a good path maybe considered as erroneous by the simulator. However it has been observed in this project that during simulation using a number of test vectors, the actual error path is detected the most number of times. Thus we can a set of faulty path lists in which the path which is a subset of all faulty path lists is found to be the actual error path (or highest probably error path).

Fault Dictionary:

The above back trace method can provide the actual error path as a diagnosis solution. However it isn’t possible to pinpoint the fault whether it‘s a wrong gate, or a faulty connection or a stuck at fault by just knowing which path is erroneous. For this reason a fault dictionary, containing the possible faults that can occur along with the vectors that detect them can help to increase the diagnosis resolution. Thus, if any test vector fails during simulation, the simulator will look up the fault dictionary and compare to check if the failing vector is listed. If so, the corresponding fault will be considered as a possible error to the given circuit. Thus a set of possible faults are obtained from the fault dictionary. If any of these faults are present on the possible error paths determined during back trace, then the simulator can determine with a better diagnostic resolution, the actual error of the circuit.

Performance and Results.

If net list is levelized, the complexity will be proportional to the number of gates. However the actual complexity of the logic simulator implemented is N*N where N is the number of gates. This is in agreement with results obtained for ISCAS'85 circuits simulated for 1000 random vectors (Fig 3). If the circuit is unlevelized, then the worst case complexity will be the depth of circuit times N*N.

Fig 2. Results for 4 bit adder (with an exhaustive set of vectors):

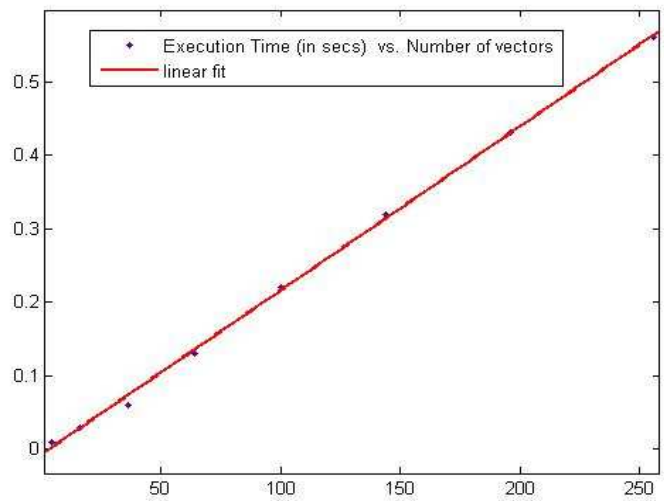
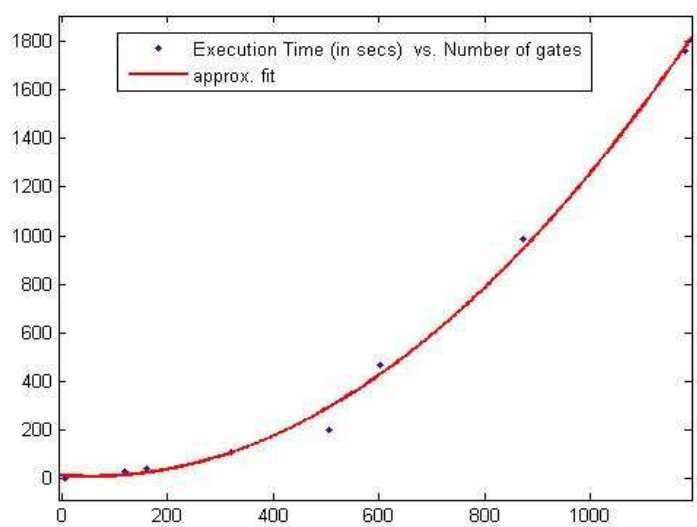


Fig 3. Results for ISCAS'85 circuits (simulated for 1000 random vectors):



Conclusion.

The logic simulator and diagnosis methods have been implemented successfully with the results documented. It has been observed that the performance is proportional to the N * N where N is the number of gates in the circuit. Better search algorithms can give a simulation complexity of N. Fault dictionary and back trace are useful methods circuit diagnosis. It has been observed that a combination of these two methods can give better diagnosis resolution and help determine the actual error much more accurately.

References.

1. Bushnell, M.L., Agrawal, V.D., Essentials of Electronic Testing for Digital, Memory and Mixed Signal VLSI Circuits. New York: Springer, 2000.
2. Dr. V.D. Agrawal's ELEC 7250 slides.