VLSI Testing Final Project

Fault Diagnosis

Introduction:

As the best test expert in our *NTU-ATPG* company, your team are asked to implement a diagnosis tool for single stuck-at faults and multiple stuck-at faults. To test our tool, we will have to first generate *fail log* files. Then you will need to diagnose these fail log files and give a ranked list of suspects. The failing output will then be diagnosed by your tool. This project will be graded based on *diagnosis accuracy*, and *diagnosis resolution,* and *run time*.

Example:

For the C17 circuit and test pattern applied. Suppose we test the C17 circuit and we get a Fail Log file for a certain CUD (circuit under diagnosis).

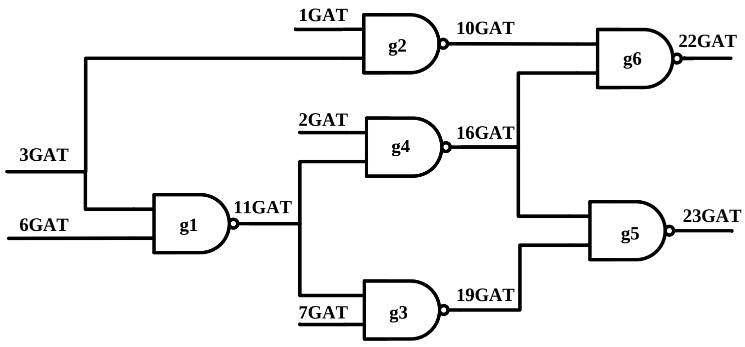

Figure 1. *c17* circuit

```
#Circuit Summary:
#---------------
#number of inputs = 5
#number of outputs = 2
#number of gates = 6
#number of wires = 11
T'00110'
T'01000'
T'00001'
```

Figure 2. *c17.pat*

To emulate the faulty CUT output, you should generate a Fail Log file,

```
./atpg -genFailLog <pattern file>  <circuit file> -fault <wire> <gate> <io> <fault type>
```

The generated Fail Log is a text file in the following format:

```
<pattern index> <faulty output wire> <expect value> <faulty value> # <pattern>
…
```

For example, if we run the following command on c17,

```
./atpg -genFailLog ../patterns/c17.pat ../sample_circuits/c17.ckt -fault 16GAT g4 GO SA0
```

```
>c17-001.failLog
```

After execution, your fail log should be like this:

```
vector [0]   22GAT   expect L, observe H     #   T'00110'
vector [0]   23GAT   expect L, observe H     #   T'00110'
vector [2]   22GAT   expect L, observe H     #   T'00001'
```

Figure 3. *Example Fail Log File*

Next, you can run the diagnosis tool by the following command,

```
./atpg -diag ../patterns/c17.pat ../sample_circuits/c17.ckt  ../FailLog/c17-001.failLog
```

The following is an example diagnosis report. The suspect faults are ranked by their score. (you can define your own score.) Please perform equivalent fault collapsing. Please put all equivalent faults at the end of each line.

```
#Circuit Summary:
#---------------
#number of inputs = 5
#number of outputs = 2
#number of gates = 6
#number of wires = 11
#number of vectors = 3
#number of failing outputs = 3

Ranked suspect faults
No.1    16GAT g4 GO SA0,    TFSF=3, TPSF=0, TFSP=0, score=100.0[equivalent faults: x SA0 , y SA1 …]
No.2    22GAT g6 GO SA0,    TFSF=2, TPSF=0, TFSP=1, score=66.7 [equivalent faults:    …]

…

# run time = 0.20 s
```

Figure 4. *Example Diagnosis Report*

Assignments:

1. You are required to implement a -genFailLog function.
2. You are required to implement a stuck-at fault diagnosis tool. You will be given fail log files: some of them are single stuck-at faults; the other of them are multiple stuck-at faults. Please fill in the following table. Please note that you may not find a perfect suspect fault for some fail logs. That means, the ranked number one suspect fault score may be lower than 100.

Diagnosis accuracy is defined as : $\frac{number\ of\ correctly\ diagnosed\ faults\ in\ your\ report}{number\ of\ total\ faults\ injected\ by\ TA}$

Correctly diagnosed faults mean those faults ranked within top 5 in your diagnosis report.

Diagnosis resolution is defined as : $\frac{number\ of\ diagnosed\ faults\ in\ your\ report}{number\ of\ correctly\ diagnosed\ faults}$

Please note that, all equivalent faults are counted as one for both equations.

3. Write your report in a paper-like format. It includes, introduction, background, proposed techniques, experimental results, discussion and conclusion. In your experimental results, please fill in the following table.

Circuit number	Diagnosis accuracy	Diagnosis resolution	Run time
C432			
C499			
C880			
C1355			
C2670			
C3540			
C6288			
C7552			

At the end, please **write clearly specific contribution** of each members (*e.g.* which lines or which function in which .cpp file). General things such as reading papers or attending discussion, are not counted as contribution.

Grading:

40% diagnosis accuracy

30% diagnosis resolution

20% run time

10% presentation and report

Submission deadlines:

1) Please do presentation in class. Give professor a **hardcopy of your ppt in the class of 6/9** so that professor can give you remarks.

2) You are required to **submit your code to COOL on 6/22 midnight (NO DELAY)**. Please make a directory ***<team_number>_project*** and copy 3 items /*src*, *report.pdf, readme*(optional) into directory. Then submit a single *.*tgz* file to COOL system. Include everything so that your code can be easily compiled using 'make'. You can use the following command to compress a whole directory:

```
tar -zcvf <team_number>_project.tgz <dir>
```

Here's a refence file strcture:

1_project/

```
├──src/         #Including *.cpp, *.h and makefile only
├──report.pdf   #Fill the table above and highlight your change of code
└──readme       #Optional
```

3) Please demo your program to professor **<u>on 6/23, 9AM-12 in EE2-339</u>**.

Reference

[Ye 2010]Jing Ye, Yu Hu, Xiaowei Li "Diagnosis of multiple arbitrary faults with mask and reinforcement effect," DATE 2010.

[Desineni 2006] Desineni, R., Poku, O. ; Blanton, R.D. "A Logic Diagnosis Methodology for Improved Localization and Extraction of Accurate Defect Behavior," ITC 2006.

[Yu ITC'08] X. Yu, R.D. Blanton "An Effective and Flexible Multiple Defect Diagnosis Methodology Using Error Propagation Analysis," ITC 2008.

[Yu DAC'08] X. Yu, R.D. Blanton "Multiple defect diagnosis using no assumptions on failing pattern characteristics, " DAC 2008.

NOTE:

Copying other source code can result in zero grade for all students involved.