VLSI Testing and Design for Testability Assignment 4

魏子翔 311510173

1. How to compile codes

   Go to the folder /podem, then enter "*make*" in command line to compile all codes.

2. The algorithm and idea of my codes

   A. Implement the fault generation function with checkpoint theorem

   I accumulated the number of branches which fanouts are more than 1.

```
for(unsigned i=0; i<Netlist.size(); i++)
    if(Gate(i)->No_Fanout() > 1)
        num_checkpoint++;
```

   Then, we can get the number of checkpoint

   To get the percentage of collapsed faults, I apply the function to get the number of faults before collapsing.

```
GenerateALLFaultList();
```

   Finally, we can calculate the answer.

   B. Bridging faults

   In function ***GenerateBridgingFaultList()***, I create another queue to store the gates which are not PO. First, I put the adjacent nets intobridging fault list, which contains AND bridging and OR bridging faults. Finally, visit all the members in the list, then we can get all the results.

3. Several case results

   A. Implement the fault generation function with checkpoint theorem

Circuit	Original faults	Check point theorem	Percentage of collapsed faults
c17	36	22	38.8889%
c432	1110	620	44.1441%
c880	2104	1014	51.8061%
c1355	2716	1618	40.6456%
c5315	10094	5502	45.4924%
c7552	19456	8098	58.3779%

   According to the table above, we can see that almost more than 50% faults are collapsed in each case. The checkpoint theorem can effectively reduce the number of faults.

B. Bridging faults

Circuit	Bridging faults	Original faults	Percentage of collapsed faults
c17	16	36	55.5556%
c432	518	1110	53.3333%
c880	1150	2104	45.3422%
c1355	1138	2716	58.1002%
c5315	4828	10094	52.1696%
c7552	11642	19456	40.1624%

With the exception of circuit c880.bench and c7553.bench, the number of collapsed faults from bridging faults method is relatively small compared to the checkpoint theorem.

The example of c17.bench

```
(G1, G2, AND)
(G1, G2, OR)
(G2, G3, AND)
(G2, G3, OR)
(G3, G4, AND)
(G3, G4, OR)
(G4, G5, AND)
(G4, G5, OR)
(net17, n60, AND)
(net17, n60, OR)
(n60, net14, AND)
(n60, net14, OR)
(net18, net25, AND)
(net18, net25, OR)
(G16, G17, AND)
(G16, G17, OR)
```