SocV p6d Arvind Singh Rathore
R10943157

If a signal is redundant, then it's stuck at fault is untestable. However, whether the fault can be propagated to output or not, depends on other inputs of gates. For example, if fault stuck at 0 for "or" gates, it won't influence the result, on the other hand, if fault stuck at 1 for "or" gate, it won't influence either.

For this case, g9 is an "and" gate, g5 stuck-at-1 will cause error. If g5 stuck-at-1 is undetectable we can remove g5.

To test it is redundant, we must satisfy condition of activation and sensitization. The difference function plays a role of sensitization, here, we need to construct activation function, which is ~g5.

I then checked ~g5 and difference, got the result equals const 0.

```
BddNode g95  = g8 & f & g5;
BddNode g95n = g8 & f & g5n;

BddNode redundancyCheck = ~g5 & (g95 ^ g95n);
cout <<"Redundancy Check" << redundancyCheck << endl;
```

```
Redundancy Check[0](-) 0x6d7c20 (28)

==> Total #BddNodes : 1
```

Therefore g5 to g9 wire redundant