VLSI Testing PA1 Report

R05921058
楊承翰

1. Testcase result

circuit number	number of test vector pairs	number of gates	number of total faults	number of detected faults	number of undetected faults	fault coverage
C499	68	554	2390	2277	113	95.27%
C1355	63	554	2726	1702	1024	62.44%
C6288	42	4800	17376	17109	267	98.46%
C7552	289	5679	19456	19144	312	98.40%

2. Code explanation
   a. In the TODO of fault detection, set the **detect** flag of a fault to be true if the output of fault-free circuit and faulty circuit are different.

```
for (i = 0; i < num_of_fault; i++) {
  if ((w->wire_value2 & Mask[i]) != (w->wire_value1 & Mask[i]))
    simulated_fault_list[i]->detect = TRUE;
}
```

   b. In the TODO of fault injection, set the corresponding bits of **wire_value2** to zeros or ones depending on the fault type.

```
if (fault == STUCK0)
  faulty_wire->wire_value2 &= ~(3 << (bit_position << 1));
else if (fault == STUCK1)
  faulty_wire->wire_value2 |= (3 << (bit_position << 1));
```

3. Speed up technique

   For faster parallel simulation, we can use a 64-bit variable (i.e. **long long** or **unsigned long long**) to store the simulation pattern.

```
struct WIRE {
  ...
  long long wire_value1;
  long long wire_value2;
  long long fault_flag;
  ...
};
```

We also need to change other variables and function prototypes to **unsigned long long**. Some constant also need to be expanded to 64 bits.

```
#define num_of_pattern 32
unsigned long long Mask[32] = { ... };
unsigned long long Unknown[32] = { ... };

unsigned long long
PINV(value)
unsigned long long value;
{
  return ((((value & 0x55555555555555ll) << 1) ^ 0xaaaaaaaaaaaaaaaall) |
          (((value & 0xaaaaaaaaaaaaaaaall) >> 1) ^ 0x55555555555555ll));
}/* end of PINV */

...
```

The improvement is not very obvious because the test cases are too small. The following table shows the runtime (in seconds) of the golden program (**golden_atpg**), my program before improvement (**atpg**) and my program after improvement (**atpg_improve**).

circuit number	golden_atpg	atpg	atpg_improve
C17	0.002	0.002	0.002
C499	0.020	0.018	0.017
C1355	0.021	0.018	0.014
C6288	0.149	0.130	0.107
C7552	0.416	0.364	0.327