Chapter 6 Testability Analysis
Outline

- Introduction
- SCOAP
- COP
- High-level Testability
Testability Analysis

- **Applications**
  - To give early warnings about the test problems
    - Guide the selection of test points to improve testability.
    - Automate the “Design for Testability” problem
  - To provide guidance in ATPG
    - For example, determine the “hardest” & “easiest” inputs in backtrace of PODEM
- **Complexity should be simpler than ATPG and fault simulation**
  - Need to be linear or almost linear in terms of circuit size
- **Topology analysis**
  - Only the structure of the circuit is analyzed
  - No test vectors are involved
  - Only an approximation
    - reconvergent fanouts cause inaccuracy
Testability Measures

- **Controllability**
  - The difficulty of setting a particular logic signal to 0 or 1.

- **Observability**
  - The difficulty of observing the logic state of a signal.
SCOAP computes 6 numbers for each node $N$.

<table>
<thead>
<tr>
<th></th>
<th>0-controllability</th>
<th>1-controllability</th>
<th>Observability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combinational</td>
<td>$CC^0(N)$</td>
<td>$CC^1(N)$</td>
<td>$CO(N)$</td>
</tr>
<tr>
<td>Sequential</td>
<td>$SC^0(N)$</td>
<td>$SC^1(N)$</td>
<td>$SO(N)$</td>
</tr>
</tbody>
</table>
Combinational SCOAP Measures

- Combinational controllability
  - \( CC^0(N) \), \( CC^1(N) \)
  - Related to the minimum number of combinational node (PI or gate output) assignments required to justify a 0 or 1 on a node \( N \).

- Combinational observability
  - \( CO(N) \)
  - Related to the number of gates between \( N \) and PO’s, \textbf{and}

- the minimum number of PI assignments required to propagate the logical value on node \( N \) to a primary output.
## CC⁰(N) & CC¹(N)

<table>
<thead>
<tr>
<th>Input Configuration</th>
<th>CC⁰(y)</th>
<th>CC¹(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[\min[CC⁰(x₁), CC⁰(x₂)] + 1]</td>
<td>[CC¹(x₁) + CC¹(x₂) + 1]</td>
</tr>
<tr>
<td>(x₁) (\rightarrow) y (x₂)</td>
<td>[CC⁰(x₁) + CC⁰(x₂) + 1]</td>
<td>[\min[CC¹(x₁), CC¹(x₂)] + 1]</td>
</tr>
<tr>
<td></td>
<td>[\min[CC⁰(x₁) + CC⁰(x₂),\right.]</td>
<td>[\min[CC⁰(x₁) + CC¹(x₂),\right.]</td>
</tr>
<tr>
<td></td>
<td>(CC¹(x₁) + CC¹(x₂)]] + 1</td>
<td>(CC¹(x₁) + CC⁰(x₂)]] + 1</td>
</tr>
<tr>
<td>(x₁) (\rightarrow) y (x₂)</td>
<td>[CC¹(x) + 1]</td>
<td>[CC⁰(x) + 1]</td>
</tr>
</tbody>
</table>

**Primary inputs**

<table>
<thead>
<tr>
<th>Input</th>
<th>CC⁰(x)</th>
<th>CC¹(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Primary outputs</td>
<td>CO($x_1$)</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>$x_1$ $x_2$ $y$</td>
<td>CO($y$) + CC$^1$(x$_2$) + 1</td>
<td></td>
</tr>
<tr>
<td>$x_1$ $x_2$ $y$</td>
<td>CO($y$) + CC$^0$(x$_2$) + 1</td>
<td></td>
</tr>
<tr>
<td>$x_1$ $x_2$ $y$</td>
<td>CO($y$) + min[CC$^0$(x$_2$),CC$^1$(x$_2$)] + 1</td>
<td></td>
</tr>
<tr>
<td>$x_1$ $y$</td>
<td>CO($y$) + 1</td>
<td></td>
</tr>
<tr>
<td>$x_1$ $y_1$ $y_2$</td>
<td>min[CO($y_1$),CO($y_2$)]</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CO(N)
An Example – Controllability

CC^0/CC^1

CC^0/CC^1

CC^0/CC^1
An Example – Observability

$CC^0/CC^1$

$\begin{array}{|c|c|}
\hline
a & 4 \\
\hline
b & 6 \\
\hline
c & 4 \\
\hline
\end{array}$

$\begin{array}{|c|c|}
\hline
4 & 1/1 \\
\hline
6 & 1/1 \\
\hline
4 & 1/1 \\
\hline
3 & 2/2 \\
\hline
3 & 2/2 \\
\hline
3 & 1/1 \\
\hline
4 & 1/1 \\
\hline
5 & 1/1 \\
\hline
6 & 1/1 \\
\hline
5/7 & 2/6 \\
\hline
2/7 & 3 \\
\hline
0 & 5/7 \\
\hline
0 & 2/7 \\
\hline
0 & 0 \\
\hline
0 & 3/5 \\
\hline
\end{array}$
Sequential SCOAP Measures

- Sequential controllability
  - \( SC^0(N), SC^1(N) \)
  - Estimate the minimum number of sequential node (FF output) assignments required to justify a 0 or 1 on a node \( N \).

- Sequential observability
  - \( SO(N) \)
  - Related to the number of FF’s between \( N \) and PO’s, and
  - the minimum number of FF assignments required to propagate the logical value on node \( N \) to a primary output.
Computing the Sequential SCOAP Measures

- Computation of $SC^0(N)$, $SC^1(N)$, and $SO(N)$ is similar to that of $CC^0(N)$, $CC^1(N)$, and $CO(N)$.

- The differences are
  - One increments the sequential measures by 1 only when signals propagate from FF inputs to Q or Q’, or backwards.
  - Several iterations may be required for the controllability numbers to converge.
## Computing SC^0(N) and SC^1(N)

<table>
<thead>
<tr>
<th>x_1 \rightarrow y</th>
<th>SC^0(y)</th>
<th>SC^1(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{AND}</td>
<td>min[SC^0(x_1), SC^0(x_2)]</td>
<td>SC^1(x_1) + SC^1(x_2)</td>
</tr>
<tr>
<td>\text{OR}</td>
<td>SC^0(x_1) + SC^0(x_2)</td>
<td>min[SC^1(x_1), SC^1(x_2)]</td>
</tr>
<tr>
<td>\text{AND}</td>
<td>min[SC^0(x_1) + SC^0(x_2), SC^1(x_1) + SC^1(x_2)]</td>
<td>min[SC^0(x_1) + SC^1(x_2), SC^1(x_1) + SC^0(x_2)]</td>
</tr>
<tr>
<td>x \rightarrow y</td>
<td>SC^1(x)</td>
<td>SC^0(x)</td>
</tr>
</tbody>
</table>

### Primary inputs

<table>
<thead>
<tr>
<th>x_1 \rightarrow y</th>
<th>SC^0(y)</th>
<th>SC^1(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{AND}</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
### SO(N)

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$y$</td>
<td>SO(y) + SC$^1$(x_2)</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$y$</td>
<td>SO(y) + SC$^0$(x_2)</td>
</tr>
<tr>
<td>$x_1$</td>
<td>$y$</td>
<td>SO(y) + min[SC$^0$(x_2), SC$^1$(x_2)]</td>
</tr>
<tr>
<td>$x_1$</td>
<td>$y$</td>
<td>SO(y)</td>
</tr>
<tr>
<td>$x_1$</td>
<td>$y_1$</td>
<td>min[SO(y_1), SO(y_2)]</td>
</tr>
</tbody>
</table>

**Primary outputs**

| Output | 0 |
Flip-Flop

\[ CC^1(Q) = CC^1(D) + CC^1(CLK) + CC^0(CLK) + CC^0(R) \]
\[ SC^1(Q) = SC^1(D) + SC^1(CLK) + SC^0(CLK) + SC^0(R) + 1 \]

\[ CC^0(Q) = \min[CC^1(R) + CC^0(CLK),\]
\[ CC^0(D) + CC^1(CLK) + CC^0(CLK) + CC^0(R)] \]
\[ SC^0(Q) = \min[SC^1(R) + SC^0(CLK),\]
\[ SC^0(D) + SC^1(CLK) + SC^0(CLK) + SC^0(R)] + 1 \]
\[ CO(D) = CO(Q) + CC^1(CLK) + CC^0(CLK) + CC^0(R) \]
\[ SO(D) = SO(Q) + SC^1(CLK) + SC^0(CLK) + SC^0(R) + 1 \]
Computing Testability Measures for Sequential Circuits

1. For all PI’s, set $CC^0 = CC^1 = 1$ and $SC^0 = SC^1 = 0$.

2. For all other nodes, set $CC^0 = CC^1 = \infty$ and $SC^0 = SC^1 = \infty$.

3. Propagate controllability measures from PI’s to PO’s. Iterate until the controllability numbers stabilize.

4. For all PO’s, set $CO = SO = 0$.

5. For all other nodes, set $CO = SO = \infty$.

6. Propagate observability from PO’s to PI’s.
Controllability Computation

Assuming no RST can occur
Controllability Computation – 2nd Iteration

CC\(^0\)/CC\(^1\), SC\(^0\)/SC\(^1\)

1/1, 0/0

a

1/1, 0/0

2/2, 0/0

2/2, 0/0

2/\(\infty\), 0/\(\infty\)

2/14, 0/1

9/17, 1/2

9/\(\infty\), 1/\(\infty\)

1/9, 0/1

2/\(\infty\), 0/\(\infty\)

4/\(\infty\), 0/\(\infty\)

7/\(\infty\), 0/\(\infty\)

7/15, 0/1

3/\(\infty\), 0/\(\infty\)

3/9, 0/1

3/\(\infty\), 0/\(\infty\)

4/\(\infty\), 0/\(\infty\)

5/\(\infty\), 1/\(\infty\)

5/11, 1/2

b

1/1, 0/0

2/\(\infty\), 0/\(\infty\)

5/\(\infty\), 1/\(\infty\)

5/11, 1/2
Controllability Computation – 3rd iteration

CC₀/CC₁, SC₀/SC₁
**COP [F. Brglez, ’84]**

- $C_x$: the probability of $x$ being 1.
- $O_x$: the probability of $x$ being observed at a PO.

<table>
<thead>
<tr>
<th></th>
<th>$C_x$</th>
<th>$O_a$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a \quad b \quad x$</td>
<td>$C_x = C_a \times C_b$</td>
<td>$O_a = O_x \times C_b$</td>
</tr>
<tr>
<td>$a \quad b \quad x$</td>
<td>$C_x = 1 - (1 - C_a) \times (1 - C_b)$</td>
<td>$O_a = O_x \times (1 - C_b)$</td>
</tr>
<tr>
<td>$x \quad a \quad b$</td>
<td>$C_x = C_a = C_b$</td>
<td>$O_x = 1 - (1 - O_a) \times (1 - O_b)$</td>
</tr>
</tbody>
</table>
An Example – Controllability

COP values

0.5 0.5 0.25
0.5 0.5 0.5

Actual contrallabilities

0.5 0.25
0.5 0.5
0.5 0.5
0.5 0.5
0.5 0.5

0.4375
0.5
An Example – Observability

COP values

Actual observabilities
**PODEM: Example (1/3)**

*Initial objective=(G5,1).*

G5 is an AND gate $\rightarrow$ Choose the hardest-1 $\rightarrow$ Back-trace to (G1,1).

G1 is an AND gate $\rightarrow$ Choose the hardest-1 $\rightarrow$ Arbitrarily, back-trace to (A,1). A is a PI $\rightarrow$ Implication $\rightarrow$ G3=0.
PODEM: Example (2/3)

The initial objective satisfied? No! → Current objective=(G5,1).
G5 is an AND gate → Choose the hardest-1 → Back-trace to (G1,1).
G1 is an AND gate → Choose the hardest-1
→ Arbitrarily, back-trace to (B,1). B is a PI → Implication → G1=1, G6=0.
PODEM: Example (3/3)

The initial objective satisfied? No! → Current objective=(G5,1).
The value of G1 is known → Back-trace to (G4,0).
The value of G3 is known → Back-trace to (G2,0).
A, B is known → Back-trace to (C,0).
C is a PI → Implication → G2=0, G4=0, G5=D, G7=D.

No backtracking !!
Initial objective=(G5,1).
Choose path G5-G4-G2-A \(\rightarrow A=0\).
Implication for \(A=0\) \(\rightarrow\) G1=0, G5=0 \(\rightarrow\) Backtracking to \(A=1\).
Implication for \(A=1\) \(\rightarrow\) G3=0.
If The Backtracing Is Not Guided (2/3)

The initial objective satisfied? No! \(\rightarrow\) **Current objective\(=\)(G5,1).**

Choose path G5-G4-G2-B \(\rightarrow\) B=0.

Implication for B=0 \(\rightarrow\) G1=0, G5=0 \(\rightarrow\) **Backtracking to B=1.**

Implication for B=1 \(\rightarrow\) G1=1, G6=0.
The initial objective satisfied? No! → Current objective=(G5,1).
Choose path G5-G4-G2-C → C=0.
Implication for C=0 → G2=0, G4=0, G5=D, G7=D.

Two times of backtracking!!
High-Level Testability Analysis

- Based on behavioral level circuit model.
- Usually part of the behavior synthesis program.
- To improve the testability at earlier design stage.
Data Flow Graph (DFG)

- Each node corresponds to a register.
- Each arc represents a combinational path between two registers.
A High-Level Testability Measure – Sequential Depth

- The length of a sequential path between two nodes is the number of arcs along the path.
- The sequential depth between a pair of registers is the length of the shortest path between them.

![Diagram of sequential depth](image)

- R1 → R1 : 0
- R2 → R1 : 1
- R3 → R1 : 2
- R4 → R1 : 2
- a → g : 2
- b → g : 3
- d → g : 4
- e → g : 4
Testability Enhancement

- Improve controllability and observability of registers.
  - Whenever possible, allocate a register to at least one PI or PO.
- Reduce the sequential depth between a controllable and an observable registers.

```
R1 → R1 : 0
R2 → R1 : 1
R3 → R1 : 2
R4 → R1 : 2
```
An Example