Testability Measures

- Origins
- Controllability and observability
- SCOAP measures
  - Sources of correlation error
  - Combinational circuit example
  - Sequential circuit example
- Test vector length prediction
- High-Level testability measures
- Summary

Purpose

- Need approximate measure of:
  - Difficulty of setting internal circuit lines to 0 or 1 by setting primary circuit inputs
  - Difficulty of observing internal circuit lines by observing primary outputs
- Uses:
  - Analysis of difficulty of testing internal circuit parts – redesign or add special test hardware
  - Guidance for algorithms computing test patterns – avoid using hard-to-control lines
  - Estimation of fault coverage
  - Estimation of test vector length

Origins

- Control theory
- Rutman 1972 -- First definition of controllability
- Goldstein 1979 -- SCOAP
  - First definition of observability
  - First elegant formulation
  - First efficient algorithm to compute controllability and observability
- Parker & McCluskey 1975
  - Definition of Probabilistic Controllability
- Brglez 1984 -- COP
  - 1st probabilistic measures
- Seth, Pan & Agrawal 1985 – PREDICT
  - 1st exact probabilistic measures

Testability Analysis

- Involves Circuit Topological analysis, but no test vectors and no search algorithm
  - Static analysis
- Linear computational complexity
  - Otherwise, is pointless – might as well use automatic test-pattern generation and calculate:
    - Exact fault coverage
    - Exact test vectors

Types of Measures

- SCOAP – Sandia Controllability and Observability Analysis Program
- Combinational measures:
  - *CC0* – Difficulty of setting circuit line to logic 0
  - *CC1* – Difficulty of setting circuit line to logic 1
  - *CO* – Difficulty of observing a circuit line
- Sequential measures – analogous:
  - *SC0*
  - *SC1*
  - *SO*

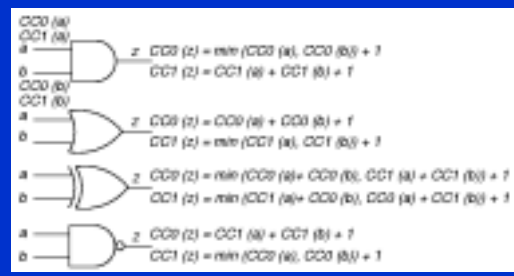
Range of SCOAP Measures

- Controllabilities – 1 (easiest) to infinity (hardest)
- Observabilities – 0 (easiest) to infinity (hardest)
- Combinational measures:
  - Roughly proportional to # circuit lines that must be set to control or observe given line
- Sequential measures:
  - Roughly proportional to # times a flip-flop must be clocked to control or observe given line

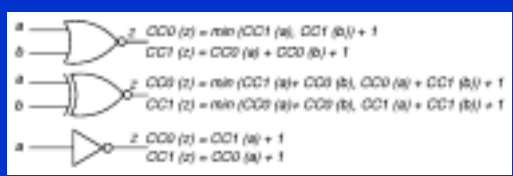
Goldstein's SCOAP Measures

- AND gate O/P 0 controllability:
  output_controllability = min (input_controllabilities) + 1
- AND gate O/P 1 controllability:
  output_controllability = Σ (input_controllabilities) + 1
- XOR gate O/P controllability
  output_controllability = min (controllabilities of each input set) + 1
- Fanout Stem observability:
  Σ or min (some or all fanout branch observabilities)

Controllability Examples

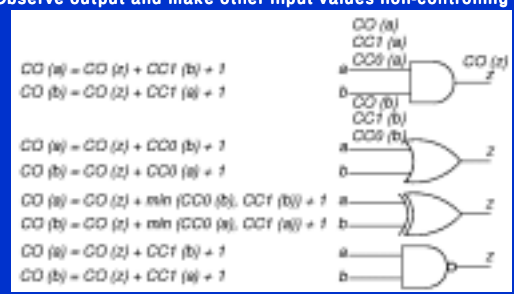


More Controllability Examples



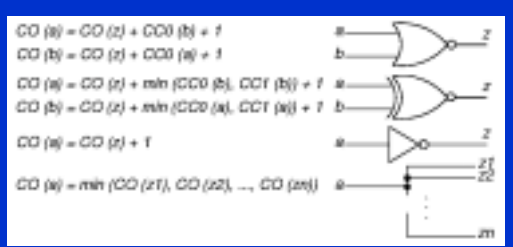
Observability Examples

To observe a gate input:
Observe output and make other input values non-controlling



More Observability Examples

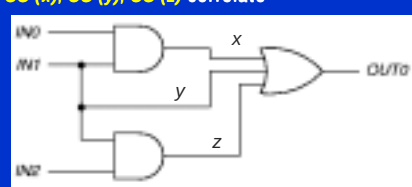
To observe a fanout stem:
Observe it through branch with best observability



Error Due to Stems & Reconverging Fanouts

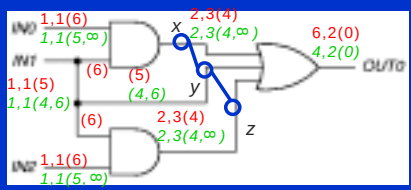
SCOAP measures wrongly assume that controlling or observing *x, y, z* are independent events

- *CC0 (x), CC0 (y), CC0 (z)* correlate
- *CC1 (x), CC1 (y), CC1 (z)* correlate
- *CO (x), CO (y), CO (z)* correlate

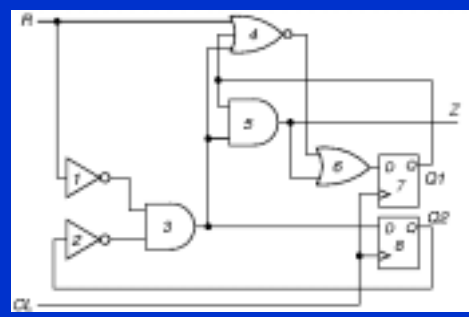


Correlation Error Example

- Exact computation of measures is NP-Complete and impractical
- Italicized (green) measures show correct values – SCOAP measures are in red or bold CC0,CC1 (CO)



Sequential Example

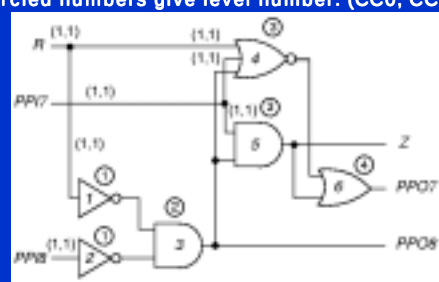


Levelization Algorithm 6.1

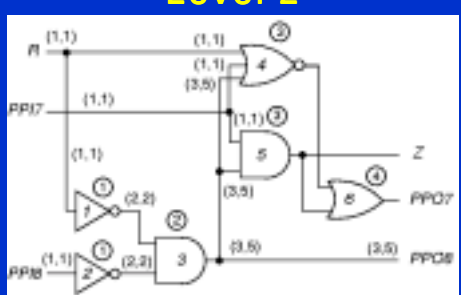
- Label each gate with max # of logic levels from primary inputs or with max # of logic levels from primary output
- Assign level # 0 to all *primary inputs* (PIs)
- For each PI fanout:
  - Label that line with the PI level number, &
  - Queue logic gate driven by that fanout
- While queue is not empty:
  - Dequeue next logic gate
  - If all gate inputs have level #'s, label the gate with the maximum of them + 1;
  - Else, requeue the gate

Controllability Through Level 0

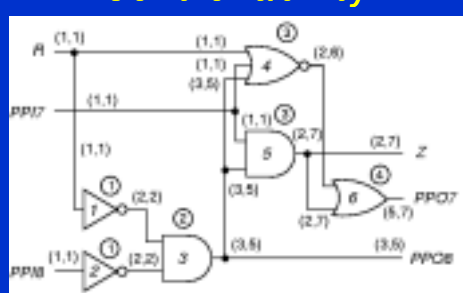
Circled numbers give level number. (CC0, CC1)



Controllability Through Level 2

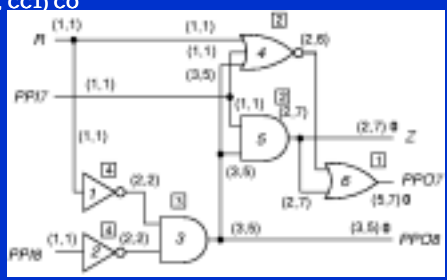


Final Combinational Controllability

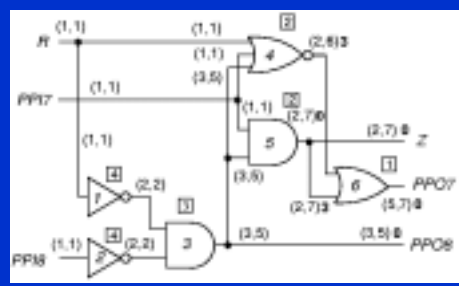


Combinational Observability for Level 1

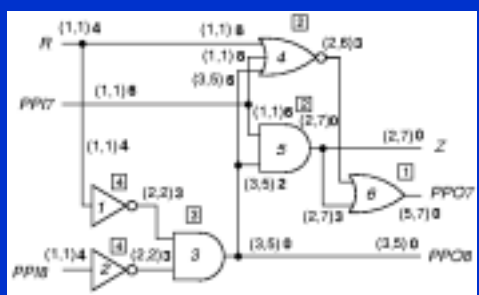
Number in square box is level from *primary outputs* (POs).
(CC0, CC1) CO



Combinational Observabilities for Level 2



Final Combinational Observabilities

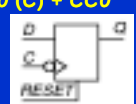


Sequential Measure Differences

- Combinational
  - Increment *CC0*, *CC1*, *CO* whenever you pass through a gate, either forwards or backwards
- Sequential
  - Increment *SC0*, *SC1*, *SO* only when you pass through a flip-flop, either forwards or backwards, to *Q*, *Q*, *D*, *C*, *SET*, or *RESET*
- Both
  - Must iterate on feedback loops until controllabilities stabilize

D Flip-Flop Equations

- Assume a synchronous RESET line.
- *CC1* (*Q*) = *CC1* (*D*) + *CC1* (*C*) + *CC0* (*C*) + *CC0* (*RESET*)
- *SC1* (*Q*) = *SC1* (*D*) + *SC1* (*C*) + *SC0* (*C*) + *SC0* (*RESET*) + 1
- *CC0* (*Q*) = min [*CC1* (*RESET*) + *CC1* (*C*) + *CC0* (*C*), *CC0* (*D*) + *CC1* (*C*) + *CC0* (*C*)]
- *SC0* (*Q*) is analogous
- *CO* (*D*) = *CO* (*Q*) + *CC1* (*C*) + *CC0* (*C*) + *CO* (*RESET*)
- *SO* (*D*) is analogous



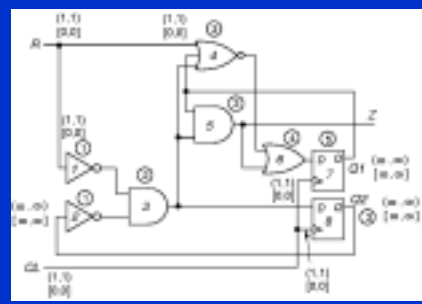
D Flip-Flop Clock and Reset

- *CO* (*RESET*) = *CO* (*Q*) + *CC1* (*Q*) + *CC1* (*RESET*) + *CC1* (*C*) + *CC0* (*C*)
- *SO* (*RESET*) is analogous
- Three ways to observe the clock line:
  1. Set *Q* to 1 and clock in a 0 from *D*
  2. Set the flip-flop and then reset it
  3. Reset the flip-flop and clock in a 1 from *D*
- *CO* (*C*) = min [*CO* (*Q*) + *CC1* (*Q*) + *CC0* (*D*) + *CC1* (*C*) + *CC0* (*C*), *CO* (*Q*) + *CC1* (*Q*) + *CC1* (*RESET*) + *CC1* (*C*) + *CC0* (*C*), *CO* (*Q*) + *CC0* (*Q*) + *CC0* (*RESET*) + *CC1* (*D*) + *CC1* (*C*) + *CC0* (*C*)]
- *SO* (*C*) is analogous

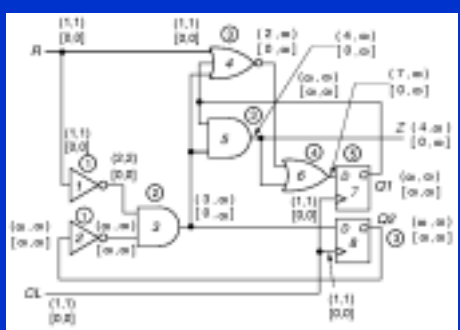
Algorithm 6.2 Testability Computation

1. For all PIs, *CC0* = *CC1* = 1 and *SC0* = *SC1* = 0
2. For all other nodes, *CC0* = *CC1* = *SC0* = *SC1* = ∞
3. Go from PIs to POS, using *CC* and *SC* equations to get controllabilities -- Iterate on loops until *SC* stabilizes -- convergence guaranteed
4. For all POs, set *CO* = *SO* = ∞
5. Work from POs to PIs, Use *CO*, *SO*, and controllabilities to get observabilities
6. Fanout stem (*CO*, SO) = min branch (CO, SO)
7. If a *CC* or *SC* (*CO* or *SO*) is ∞, that node is uncontrollable (unobservable)

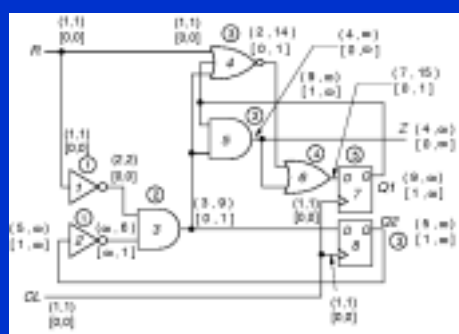
Sequential Example Initialization



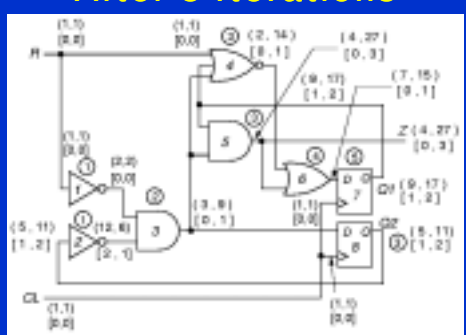
After 1 Iteration



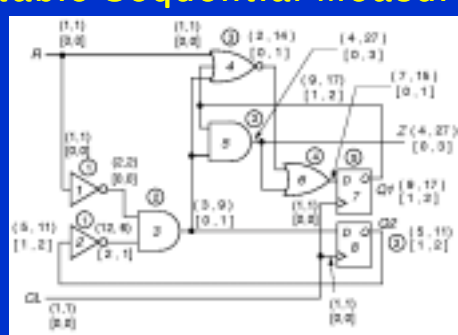
After 2 Iterations



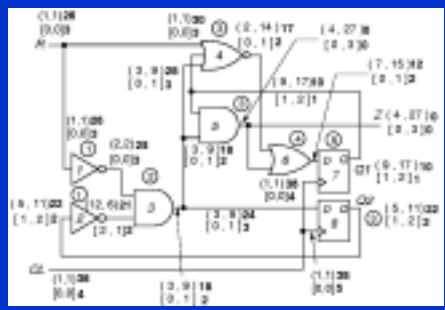
After 3 Iterations



Stable Sequential Measures



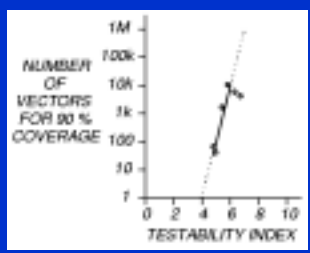
Final Sequential Observabilities



Test Vector Length Prediction

- First compute *testabilities* for stuck-at faults
  - T (x sa0) = $CC1$ (x) + CO (x)
  - T (x sa1) = $CC0$ (x) + CO (x)
  - *Testability index* = $\log \frac{\Sigma T(f_i)}{f_i}$

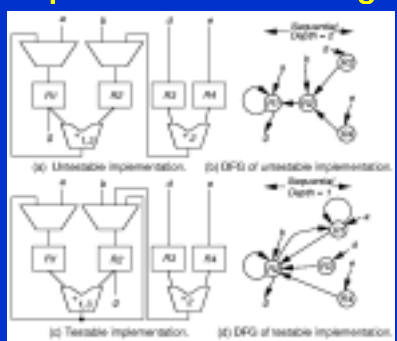
Number Test Vectors vs. Testability Index



High Level Testability

- Build *data path control graph* (DPCG) for circuit
- Compute sequential depth -- # arcs along path between PIs, registers, and POs
- Improve Register Transfer Level Testability with redesign

Improved RTL Design



Summary

Summary

- Testability approximately measures:
  - Difficulty of setting circuit lines to 0 or 1
  - Difficulty of observing internal circuit lines
- Uses:
  - Analysis of difficulty of testing internal circuit parts
    - Redesign circuit hardware or add special test hardware where measures show bad controllability or observability
  - Guidance for algorithms computing test patterns – avoid using hard-to-control lines
  - Estimation of fault coverage – 3-5 % error
  - Estimation of test vector length