Design for Testability (DfT)

- Board level testing first forced (ca 1970) the issue of circuit node control and observation
- Structured DfT incorporates specific hardware for a test mode operation
  - DfT hardware has to be tested and generally separately
  - Inactive during mission mode (normal operation)
  - Routinely combined with additional on-chip circuitry to complete Built-In-Self-Test (BIST)
- Recall Bennetts 1984 working definition of testability
  
  A circuit is testable if a set of test patterns (vectors) can be generated, evaluated and applied in such a way as to satisfy pre-defined levels of performance, defined in terms of fault-detection fault-location, and test application criteria, within a pre-defined cost budget and time-scale.
- A cost element controlled DfT by is the ease of test
Design for Testability (DfT)

- Ad-hoc schemes may be the easiest on design they can be the most difficult to improve
  - Testability analysis is a last step to design not an integral one
- BIST eases both test application and analysis by allocating more of the test to design
- The middle step is early inclusion of design structures to ease test with externally applied vectors and external capture and comparison
- Scan design forms basis for much of the DfT

Functional Model of Scan Register
Design for Testability (DfT)

- The scan circuitry in test mode is a linked shift register
  - Full scan — all flip-flops (latches) are linked into one of the serial registers
  - Partial scan — flip-flops (e.g., near PIs) are removed from serial register

- As a test strategy, scan is a divide-and-conquer
  - The scan (shift) registers isolate smaller blocks of combinational logic
  - Logic to set sequential states (the flip-flops) is reduced to a sequence of shifted 1s and 0s

- Scan segmentation naturally increases both node control and node observation

- ATPG simplified by divide-and-conquer scan segmentation

- Most common is scan latch, serial test input, test output and test control

Design for Testability (DfT)

- Convert sequential testing problem to combinational testing
  - Performance, area and timing overhead ranges from 5% - 15%
    - Area — routing and adding latch
    - Performance — power requirements and yield loss from failing DfT circuitry
    - Timing — latch MUX delay and added capacitance

- Test time limited by number of scan cells and test clock
  - Most (all) vector patterns initialized by shift cycles
  - Scan clocks typically optimized for area not frequency (5-20 MHz)
  - Scan chain balance (same number of cells) minimizes scan in/out time (maximum efficiency)
Design for Testability (DfT)

- Increases pin-count or multiplexing of other I/Os
  - Ordering scan elements can reduce test time
  - Ordering scan elements can increase area or performance costs

- Test mode MUX selects Scan; Normal mode MUX selects Data
- Input scan data controls register contents

Tests using PI functional input can be combined with scan input

Output scan data observes register contents

Internal node latched as output in scan chain is input to “downstream” logic

Summary of Scan Functions

<table>
<thead>
<tr>
<th>Mode</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operate</td>
<td>Normal transparent operation of scan cell</td>
</tr>
<tr>
<td>Scan Sample</td>
<td>Shift normal data to serial scan chain</td>
</tr>
<tr>
<td>Scan Load/Shift</td>
<td>Shift register operation</td>
</tr>
<tr>
<td>Scan Apply</td>
<td>Scan data controls value to downstream logic</td>
</tr>
</tbody>
</table>

- Latch versus flip-flop, clocks and control vary by specific scan register design
  - Circuits with multiple clock domains use per domain scan chains
Design for Testability (DfT)

- Master-slave latch and FF reduce race conditions in test mode by increasing probability shift operations are race free

- Combinational logic settles to final shift values (after shift completion)

- Sequential logic states are set by final shift not racing combinational logic

- Scan controllability and observability combined with three key steps
  - Activation - force tested node to opposite value
  - Propagation - propagate the transfer effects of forced tested node to one or more POs
  - Justification - resolve internal nodes from activation and propagation steps to PIs

Design for Testability (DfT)

- Add scan cells to nodes with low controllability or observability (test insertion)

Third scan cell is a test point insertion
Design for Testability (DfT)

- Circuits may not lend themselves to scan
  - Limits on power requirements during shift
  - Register timing and asynchronous timing
- In full Scan every design latch is one link in a serial shift register
- Partial Scan designs relax this requirement
  - Reduced fault coverage
  - Increased ATPG cost
  - During scan out non-scan latches may continuously updated
  - Scan-in 'X' values waste memory and reduces controllability
  - Scan-capture of uninitialized states reduces observability

- Unbalanced scan chain lengths introduce similar reductions in control and observation
- Why use a DfT approach like scan?
  - Functional testing (patterns) are generated to test behavior not structure of design
    - SoC and segmented IP based designs limit the application of functional patterns
  - Divide-and-conquer approach eases fault coverage grading
- Al Crouch has an alternative view and cites misconceptions of scan
  - Scan negatively impacts performance
  - Scan negatively impacts area
  - Scan requires deeper test memories
Design for Testability (DfT)

- Scan is a DC fault test
- Scan is slow and serialized
- Scan is complicated and limits effectiveness of design tools

- Many delay test alternatives require additional hardware
  - Adjust circuit timing
  - Apply and capture vector pair \( V_1, V_2 \)

- Delay test techniques
  - Slow clock combinational
    - Limited to circuits with no internal state
    - May reduce supply voltage to reduce timing requirements
    - Input and output clock controlled separately

Alternative 1 enhance scan testing to support delay test
- Load scan with slow clock
- Transfer final scan state to hold latch (\( V_1 \))
- Scan in \( V_2 \)
- Latch result

Alternative 2 enhance scan
- Broad-side scan use combination logic for \( V_2 \)

Alternative 3 enhance scan
- Derive \( V_2 \) from scan shift

- At-speed scan
- Variable clock non-scan
- At-speed functional
Delay Test with Scan

- Increasing common use of scan is delay fault testing
- Delay test using scan counter point to running functional vectors at full system clock rates (at-speed functional)
  - Limited number of vectors
  - At-speed functional delay coverage low (non-robust)
- Delay test asynchronous circuits (no scan) remains active area of research
- Conventional scan testing could be called DC scan or stuck-at scan
- Key to success of scan delay test is the clock
  - The slow-clock combinational delay test separate clock signals for PIs and POs
  - Scan allows similar method for circuits with internal state

Delay Test with Scan

- Initial sensitization path is shifted slowly
- Results are captured with the faster (shorter delay) system clock
  - Slow scan clock examines the results
- Clock relation measures delays between scan chains
- Two different usage patterns of scan and system clock
  - Launch-off-shift — scan clock is generated exactly one at-speed clock cycle after the last shift clock (also called scan-shift-delay)
    - \( V_1 \) pattern from the next to last shift
    - \( V_2 \) pattern from last shift
  - Broadside — after the pattern is scanned in, two or more at-speed capture clocks are applied to the scan flop (also called launch-from-capture)
Delay Test with Scan

- $V_1$ pattern from shift
- $V_2$ pattern from upstream combinational logic

![Diagram of Delay Test with Scan](image)

- Launch-off-shift uses the scan clock and test mode select like a system clock
  - Good for partial scan where the combination of scan and system latches must work together
  - Scan clock and test mode select require similar distribution (routing) and drive as system clock (more overhead)

---

Delay Test with Scan

- Broadside is easier as the scan enable is always run slow
  - If clocks are external the ATE provides them
  - If clocks are internal a clock control must support the control of small number (one or two) pulses
  - Can use more that two consecutive at-speed clock pulses
Two methods have different delay fault coverage

- **Launch-from-shift**
  - Broadside ATPG generates $V_1$ pattern and has to compute the correct $V_2$ pattern from logic for every path.
  - Transition delay vectors first and then generating stuck-at vectors for faults that were missed by transition delay vectors.
  - Transition delay fault patterns three to five times larger than stuck-at patterns.
  - Transition delay testing is used to test as many paths as possible at speed.
  - Reduces the number of required stuck at fault scan vectors.

- **Delay Test with Scan**
  - Path delay testing needs pattern to sensitize the path of interest and sets up the values to be sent through the entire on-path and off-path.
  - Input values launch from scan flop then propagate to the capture scan flop.
  - Scan vector controls all inputs for all paths.
  - Capture bits in scan are at end of tested path.
  - Path delay vectors are inefficient as small paths are tested per vector pair.
  - Maximizes launch points and capture points.

**Maximizes launch points and capture points**
Scan Diagnosis

- Scan Diagnostics comes in two flavors
  - Scan chain fail diagnostics when a link in a scan chain fails to shift
  - Scan cell(s) capture $D$ or $\bar{D}$ bits from the combinational logic

- Value of diagnostics in three short examples
  - Yield takes a sudden (unexpected) downturn in production
  - Initial silicon bringup when debug diagnosis is chasing design fails and process design fails
  - Use scan to capture internal states of functional states

- Combinational logic
  - Each scan bit shift corresponds to the logic cone of the scan-flop

Scan Diagnosis

- Faults in the cone of logic are registered on the scan capture
- Diagnosis is the intersection of multiple patterns and multiple capture cycles
- Match a pattern of fails to a specific set of simulation results
- Exact matching of fault to scan fail pattern is not normally obtained
- As a result successful diagnosis some “fails” are removed

- Scan chain fails and dictionaries
  - Collect is a reverse engineering approach
  - Cause-effect in place of effect-cause
  - Matching of test failures to stored failure signatures
Scan Diagnosis

- Store the simulation in contrast to resimulation
- Limited dictionary to particular fault types
- Particularly, suited to a failure within the scan chain