Testing

Testing is one of the most expensive parts of chips
- Logic verification accounts for > 50% of design effort for many chips
- Debug time after fabrication has enormous opportunity cost
- Shipping defective parts can sink a company

Example: Intel FDIV bug
- Logic error not caught until > 1M units shipped
- Recall cost $450M (!!!)

Logic Verification

Does the chip simulate correctly?
- Usually done at HDL level
- Verification engineers write test bench for HDL
  - Can’t test all cases
  - Look for corner cases
  - Try to break logic design

Ex: 32-bit adder
- Test all combinations of corner cases as inputs:
  - 0, 1, 2, 2^{31}-1, -1, -2^{31}, a few random numbers
- Good tests require ingenuity
Silicon Debug

- Test the first chips back from fabrication
  - If you are lucky, they work the first time
  - If not...
- Logic bugs vs. electrical failures
  - Most chip failures are logic bugs from inadequate simulation
  - Some are electrical failures
    - Crosstalk
    - Dynamic nodes: leakage, charge sharing
    - Ratio failures
  - A few are tool or methodology failures (e.g. DRC)
- Fix the bugs and fabricate a corrected chip

Manufacturing Test

- A speck of dust on a wafer is sufficient to kill chip
- Yield of any chip is < 100%
  - Must test chips after manufacturing before delivery to customers to only ship good parts
- Manufacturing testers are very expensive
  - Minimize time on tester
  - Careful selection of test vectors

Shmoo Plots

- How to diagnose failures?
  - Hard to access chips
    - Picoprobes
    - Electron beam
    - Laser voltage probing
    - Built-in self-test
- Shmoo plots
  - Vary voltage, frequency
  - Look for cause of electrical failures

Validation and Test of Manufactured Circuits

Goals of Design-for-Test (DFT)

- Make testing of manufactured part swift and comprehensive

DFT Mantra

- Provide controllability and observability

Components of DFT strategy

- Provide circuitry to enable test
- Provide test patterns that guarantee reasonable coverage
**Test Classification**

- Diagnostic test
  - used in chip/board debugging
  - defect localization
- “go/no go” or production test
  - Used in chip production
- Parametric test
  - \( x \in [v,i] \) versus \( x \in [0,1] \)
  - check parameters such as NM, Vt, tp, T

**Design for Testability**

Exhaustive test is impossible or unpractical

**Design for Test**

- Design the chip to increase observability and controllability
- If each register could be observed and controlled, test problem reduces to testing combinational logic between registers.
- Better yet, logic blocks could enter test mode where they generate test patterns and report the results automatically.

**Controllability/Observability**

- Combinational Circuits: controllable and observable - relatively easy to determine test patterns
- Sequential Circuits: State!
  - Turn into combinational circuits or use self-test
- Memory: requires complex patterns
  - Use self-test
Generating and Validating Test-Vectors

- Automatic test-pattern generation (ATPG)
  - for given fault, determine excitation vector (called test vector) that will propagate error to primary (observable) output
  - majority of available tools: combinational networks only
  - sequential ATPG available from academic research
- Fault simulation
  - determines test coverage of proposed test-vector set
  - simulates correct network in parallel with faulty networks
- Both require adequate models of faults in CMOS integrated circuits

Fault Models

Most Popular - “Stuck-at” model

\[ \alpha, \gamma: x_1 \text{ sa1} \]
\[ \beta: x_1 \text{ sa0 or } x_2 \text{ sa0} \]
\[ \gamma: Z \text{ sa1} \]

Covers almost all (other) occurring faults, such as opens and shorts.

Problem with stuck-at model: CMOS open fault

Sequential effect

Needs two vectors to ensure detection!

Problem with stuck-at model: CMOS short fault

Covers short circuit between \( V_{dd} \) and GND for \( A=C=0, B=1 \)

Possible approach:
Supply Current Measurement (IDDQ)
but: not applicable for gigascale integration

Other options: use stuck-open or stuck-short models
This requires fault-simulation and analysis at the switch or transistor level - Very expensive!
Test Pattern Generation

- Manufacturing test ideally would check every node in the circuit to prove it is not stuck.
- Apply the smallest sequence of test vectors necessary to prove each node is not stuck.
- Good observability and controllability reduces number of test vectors required for manufacturing test.
  - Reduces the cost of testing
  - Motivates design-for-test

Path Sensitization

Goals: Determine input pattern that makes a fault controllable (triggers the fault, and makes its impact visible at the output nodes)

Techniques Used: D-algorithm, Podem

Test Example

- $A_3$: (0110), (1110)
- $A_2$: (1010), (1110)
- $A_1$: (0100), (0110)
- $A_0$: (0110), (0111)
- $n1$: (1110), (0110)
- $n2$: (0110), (0100)
- $n3$: (0101), (0110)
- $Y$: (0110), (1110)

Minimum set: {0100, 0101, 0110, 0111, 1010, 1110}

Test Approaches

- Ad-hoc testing
- Scan-based Test
- Self-Test

Problem is getting harder
  - increasing complexity and heterogeneous combination of modules in system-on-a-chip.
  - Advanced packaging and assembly techniques extend problem to the board level
**Ad-hoc Test**

Inserting multiplexer improves testability

**Scan**

- Convert each flip-flop to a scan register
  - Only costs one extra multiplexer
- Normal mode: flip-flops behave as usual
- Scan mode: flip-flops behave as shift register

- Contents of flops can be scanned out and new values scanned in

**Scan-based Test**

**Scannable Flip-flops**
**Polarity-Hold SRL (Shift-Register Latch)**

- System Data \( D \)
- System Clock \( C \)
- Scan Data \( S \)
- Shift A Clock \( A \)
- Shift B Clock \( B \)

Introduced at IBM and set as company policy

**Scan-based Test — Operation**

**Scan-Path Testing**

Partial-Scan can be more effective for pipelined datapaths

**Boundary Scan**

- Testing boards is also difficult
  - Need to verify solder joints are good
    - Drive a pin to 0, then to 1
    - Check that all connected pins get the values
  - Through-hold boards used "bed of nails"
  - SMT and BGA boards cannot easily contact pins
  - Build capability of observing and controlling pins into each chip to make board test easier
Boundary Scan (JTAG)

Board testing becomes as problematic as chip testing

Boundary Scan Interface

- Boundary scan is accessed through five pins
  - TCK: test clock
  - TMS: test mode select
  - TDI: test data in
  - TDO: test data out
  - TRST*: test reset (optional)

- Chips with internal scan chains can access the chains through boundary scan for unified test strategy.

Boundary Scan Example

Built-in Self-test

- Built-in self-test lets blocks test themselves
  - Generate pseudo-random inputs to comb. logic
  - Combine outputs into a syndrome
  - With high probability, block is fault-free if it produces the expected syndrome
**Self-test**

Rapidly becoming more important with increasing chip-complexity and larger modules

**PRSG**

- **Linear Feedback Shift Register**
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*

<table>
<thead>
<tr>
<th>Step</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>111</td>
</tr>
<tr>
<td>1</td>
<td>110</td>
</tr>
<tr>
<td>2</td>
<td>101</td>
</tr>
<tr>
<td>3</td>
<td>010</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
</tr>
<tr>
<td>5</td>
<td>001</td>
</tr>
<tr>
<td>6</td>
<td>011</td>
</tr>
<tr>
<td>7</td>
<td>111 (repeats)</td>
</tr>
</tbody>
</table>

**Linear-Feedback Shift Register (LFSR)**

**Signature Analysis**

Counts transitions on single-bit stream
- Compression in time
BILBO

- Built-in Logic Block Observer
  - Combine scan with PRSG & signature analysis

Memory Self-Test

Patterns: Writing/Reading 0s, 1s,
         Walking 0s, 1s
         Galloping 0s, 1s

BILBO Application

ScanIn

ScanOut

In

Out

Combinational Logic

Combinational Logic

FSM

Memory Under Test

Signature Analysis

data-in

address &
R/W control

data-out
Low Cost Testing

- If you don’t have a multimillion dollar tester:
  - Build a breadboard with LED’s and switches
  - Hook up a logic analyzer and pattern generator
  - Or use a low-cost functional chip tester

Summary

- Think about testing from the beginning
  - Simulate as you go
  - Plan for test after fabrication

- “If you don’t test it, it won’t work! (Guaranteed)”

TestosterICs

- Ex: TestosterICs functional chip tester
  - Designed by clinic teams and David Diaz at HMC
  - Reads your IRSIM test vectors, applies them to your chip, and reports assertion failures