Outline

- **SoC Manufacturing Test**
  - The testing problem
  - SoC testing costs
  - Design for Test (DFT)

- **SoC Testability Features**
  - Boundary Scan
  - P1500 standard

- **Built-In Self Test**
  - Functional Test Access Mechanism (TAM)
The Manufacturing Test Problem

Partitioning for SoC Test

- Partition according to test methodology:
  - Logic blocks
  - Memory blocks
  - Analog blocks

- Provide test access:
  - Boundary scan
  - Analog test bus

- Provide test-wrappers for cores

  Design for Test (DFT)
DFT Architecture for SOC

- User defined test access mechanism (TAM)
- Test source
- Functional inputs
- Test wrapper
- Instruction register control
- Serial instruction data
- Test access port (TAP)
- Test sink
- Functional outputs
- Module
- Module
- SOC inputs
- TDI, TCK, TMS, TRST, TDO
- SOC outputs

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
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 Example
Boundary Scan (IEEE 1149.1, JTAG)

- 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.

Additional DFT Components

- Test source: Provides test vectors via on-chip LFSR, counter, ROM, or off-chip ATE.

- Test sink: Provides output verification using on-chip signature analyzer, or off-chip ATE.

- Test access mechanism (TAM): User-defined test data communication structure; carries test signals from source to module, and module to sink; tests module interconnects via test-wrappers; TAM may contain bus, boundary-scan and analog test bus components.

- Test controller: Boundary-scan test access port (TAP); receives control signals from outside; serially loads test instructions in test-wrappers.

Source: H. Kerkhoff
Test Wrapper for a Core

- Logic added around a core to provide test access to the embedded core
- Test-wrapper provides for each core input terminal
  - An external test mode – Wrapper element observes core input terminal for interconnect test
  - An internal test mode – Wrapper element controls state of core input terminal for testing the logic inside core
- For each core output terminal
  - A normal mode – Host chip driven by core terminal
  - An external test mode – Host chip is driven by wrapper element for interconnect test
  - An internal test mode – Wrapper element observes core outputs for core test

A Test-Wrapper

![Diagram of test wrapper](image)

Source: H. Kerkhoff
ECE382M.20: SoC Design, Lecture 17 © J. A. Abraham
Goals of IEEE P1500

- Core test interface between embedded core and system chip
- Test reuse for embedded cores
- Testability guarantee for system interconnect and logic
- Improve efficiency of test between core users and core providers

Set-up of P1500 Architecture

Source: H. Kerkhoff
Core including Wrapper Cells

Outline

- **SoC Manufacturing Test**
  - The testing problem
  - SoC testing costs
  - Design for test (DFT)

- **SoC Testability Features**
  - Boundary Scan
  - P1500 standard

  - *Built-In Self Test*
    - Functional Test Access Mechanism (TAM)
Built-In Self Test (BIST)

- Increasing circuit complexity, tester cost
  - Interest in techniques which integrate some tester capabilities on the chip
  - Reduce tester costs

- Approach:
  - Pseudo-random (or pseudo-exhaustive) pattern generator (PRPG) on the chip
  - Compress test responses into “signature” using multi-input shift register (MISR)

 Integrating pattern generation and response evaluation on chip – BIST

Pseudo-Random Sequences

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

Can also be used to compress test responses

<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</td>
</tr>
</tbody>
</table>

(repeats)
Example of BIST

Technique called STUMPS (from IBM)

Test Access Mechanisms (TAMs)

- **Non-functional access**
  - Uses a kind of access to core not allowed during the normal functional operation
  - Generally based on scan chains or other design for test (DFT) structures → Slow serial access
  - Direct access to core test pins through external pins → Fast, but high pin overhead
  - Can also use the embedded processor as the test source/sink → Needs wrappers around the core under test

- **Functional access**
  - Embedded processor is the test source/sink → No DFT structures or wrappers around the cores
**Functional TAMs**

- **Software-Based Self Test (SBST):** Use the intelligence of the embedded processor to test the SOC
- **At-speed tests are possible**
- **Cores in the SOC can be of three kinds**
  1. White box -- internals visible, structure changeable
  2. Grey box – all the internals visible, but structure of the core cannot be changed
  3. Black box – no internals visible, no change can be made on the core
- **Any methodology for testing black box cores should not depend on knowledge of the core’s internals**

**Why is Conventional Test Successful?**

- **Two innovations have allowed test to keep up with complex designs**
  - The stuck-at fault model
    - The model allows structural test generation, with a number of faults which is linear in the size of the circuit
  - Partitioning the circuit
    - Partitioning the circuit (with scan latches for example), alleviates the test problem so that test generation does not have to deal with the entire circuit

➤ Do these two assumptions hold for Deep SubMicron (DSM) circuits?
IC Technology

Minimum Feature Size

- Human hair, 100 μm
- Amoeba, 15 μm
- Red blood cell, 7 μm
- AIDS virus, 0.1 μm
- Buckyball, 0.001 μm

Source: Raul Camposano, Synopsys

Features Smaller than Wavelengths

Source: Raul Camposano, Synopsys
Random Dopant Fluctuations

![Graph showing the relationship between technology node and mean number of dopant atoms]

Defects in DSM Technologies

- Experiments on real chips (e.g., Stanford)
  - Stuck-at tests do not detect some defects unless they are applied at speed

- Resistive opens comprise the bulk of test escapes in one production line
  - Likely in copper interconnect – cause delay faults

- Delay faults identified as the cause of most test escapes on another line
  - Speed differences of up to a factor of 1.5 can exist between fast and slow devices - problems with “speed binning”

- Increasing possibility of shorts and crosstalk
Effects on Chip?

- Change in delays of paths
- Effects could be distributed across paths

![Graph showing at-speed and structural defects]

Source: Gelsinger

Solution:
- At-Speed tests
  - Tester Cost?
- Apply “Native Mode”?
  - Can use low-cost testers

Native-Mode Built-In Self Test

- Functional capabilities of processors can be used to replace BIST hardware – [UT Austin, ITC’1998]
  - Application to self-test of processors at Intel – FRITS method applied to Pentium 4, Itanium [ITC’2002]

```plaintext
for each data value D_i { 
  Shift_Right_Through_Carry(S);
  if (Carry) S = XOR(S, polynomial);
  S = XOR(S, D_i);
}
```

Hardware for MISR

Software implementation of MISR
Native-Mode Self Test for Processors

- Random instructions can be run from cache and results compressed into a signature
- Implementation in Intel FRITS system showed benefits for real chips (Pentium 4, Itanium)
- Technique can be used for self-test of an embedded processor in a System-on-Chip

➢ Is it possible to now use this processing capability to test other modules (digital, analog/mixed-signal and RF) on the SoC?
  - First, can the processor test be improved to detect realistic defects, e.g., small delays?

Are Random Tests Sufficient?

- Intel implementation involved code in the cache which generated random instruction sequences
- Interest in generating instructions targeting faults
  - Possible to generate instruction sequences which will test for an internal stuck-at fault in a module [Gurumurthy, Vasudevan and Abraham, ITC 2006]
- In order to deal with defects in DSM technologies, need to target small delay defects
  - Recent work: automatically generate instruction sequences which will target small delay defects in an internal module [Gurumurthy, Vemu, Abraham and Saab, European Test Symposium (ETS) 2007]
Approach to Testing Cores

- Uses functional TAM
- Uses pre-existing vectors
- Generates software to be loaded on to the embedded processor
  - Reverse driver that produces given test vectors for core

[ Gurumurthy, Sambamurthy and Abraham, \( \text{Int'l Test Synthesis Workshop (ITSW) 2008} \) ]

Summary

- **SoC Manufacturing Test**
  - Scan chains
  - JTAG Boundary Scan
  - Test wrappers for cores
  - Built-in self test (BIST)

- **Advanced SoC test topics**
  - Analog/Mixed-Signal (AMS) test
  - RF test
  - Micro-Electro-Mechanical Systems (MEMS) test