Course Outline

- Introduction: Manufacturing, Wafer sort, Final test, Board and System Test, Defects, and Faults
- Test generation; combinational and sequential test generation
- Design-for-Test techniques; test point insertion, scan, enhanced scan
- Built-In Self-Test; Logic BIST and memory BIST
- Test data compression
- System-on-Chip Test
- System Test and Boundary Scan
- Diagnosis; scan-chain and logic diagnosis

Making fault free electronic products

Printed Circuit Board (PCB)
Probing for Test

Test Objectives

- Given a Printed Circuit Board (PCB) composed of a set of components (ICs) where each component is tested good.
- The main objectives are to ensure that all components are:
  - correct (the desired ICs are selected)
  - mounted correctly at the right place on the board and
  - ensuring that interconnections are functioning according to specification
- Problems that may occur:
  - A component does not contain logic
  - A component is not placed where it should be,
  - A component is at its place but turned wrongly,
  - A component is correct but the interconnection is not correct, for example due to bad soldering.

Bed-of-Nails

Comparing SOC Test and Board Test

SOC Test
- Cores are not tested prior to chip fabrication
- Focus on both core test (internal test) and connection test (external test)

Board Test
- All components (chips) are tested prior board fabrication
- Main focus on test of interconnections (external test)
IEEE 1500 Core Test Standard

- **Goals**
  - Define test interface between core and SOC
  - Core isolation
  - Plug-and-play protocols

- **Scope**
  - Standardize core isolation protocols and test modes
  - TAM design
  - Type of test to be applied
  - Test scheduling

IEEE 1500 Wrapper

- **Interface** between module and the rest of the chip; makes it possible access core and isolate core from rest of the system.

- **Test modes**
  - Normal: Functional mode
  - InTest: test of module itself
  - ExTest: test of interconnection to other core

IEEE 1500 Standard for Embedded Core Test
Test Wrapper

- Test wrapper is in functional mode; hence the test wrapper is transparent (invisible)

Test Wrapper: WS_Bypass

- Test data (test stimuli and test responses) are bypassed.

Test Wrapper: WS_EXTEST

- Testing of user defined logic (or interconnections).
Objective

Wrapper cells are programmed to perform internal test, testing of the core itself.

Boundary Scan (IEEE std. 1149.1)

- The Joint European Test Action Group (JETAG), formed in mid-80, became Joint Test Action Group (JTAG) in 1988 and formed the IEEE std 1149.1. The standard consists of:
  - Test Access Port (TAP)
  - TAP Controller (TAPC),
  - Instruction Register (IR), and
  - Data Registers (DR)

Boundary Scan

IEEE Std. 1149.1 and IEEE Std. 1500

(Use-defined WPP = WPI+WPO+WPC)
Boundary Scan

Boundary Scan cell

Boundary Scan chain

PCB

Core Logic

Core Logic

Core Logic

TAP Controller

TAP Controller

TAP Controller

TAP Controller

Boundary Scan cell (BSC)

MODE
IN
SN/RDR

OUT

SI

1

0

SO

ClockDR

UpdateDR

MUX

WBC

CFO

CFI

CTI

CTO

SI

Std. 1149

OUT

BSC

IN

Std. 1500

Boundary Scan

Boundary Scan

Boundary Scan

Boundary Scan

Boundary Scan Cell (BSC)

Boundary Scan Register

Bypass Register

Device ID Register

Design Specific Data Registers

Data Registers (DR)

Reset

Instruction Decode

Instruction Register (IR)

TAP Controller

TDO

TDI

ClockIR

ShiftIR

UpdateIR

TMS

TCK

TRST

Optional registers and signals are shown in dotted lines
Boundary Scan Description Language (BSDL)

- BSDL is a subset of VHDL that describes how JTAG (IEEE 1149.1) is implemented in a particular device.
- For a device to be JTAG compliant, it must have an associated BSDL file.
- Use BSDL file to work out how to access a device in the JTAG chain.
- BSDL files contain the following elements:
  - Entity Description: Statements naming the device.
  - Generic Parameter: A value such as a package type.
  - Port Description: Describes the pins on the device.
  - Use Statements: References external definitions.
  - Pin Mapping(s): Maps logical signals in the device to physical pins.
  - Scan Port Identification: Defines the pins to access the JTAG capabilities (TDI, TDO, etc. - the Test Access Port).
  - Instruction Register Description: Signals to access JTAG device modes.
  - Register Access Description: Which register is placed between TDI and TDO for each JTAG instruction.
  - Boundary Register Description: List of the boundary scan cells and their functionality.
Mapping component data to the BScan Chain

"Manual" composition of the scan data arrays – Knowledge of position in the overall chain is required

Component data for one or more DR registers

BYPASS

PCB
Ring Architecture with Shared TMS

Ring Architecture with Separate TMS

Star Architecture

Multi-drop Architecture
### IEEE 1149 Standard Family

<table>
<thead>
<tr>
<th>Number</th>
<th>Main objectives</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1149.1</td>
<td>Testing of digital chips and interconnects between chips</td>
<td>Std. 1149.1-1990</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Std. 1149.1a-1993</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Std. 1149.1b-1994 (BSDL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Std. 1149.1-2001</td>
</tr>
<tr>
<td>1149.2</td>
<td>Extended digital serial interface</td>
<td>Discontinued</td>
</tr>
<tr>
<td>1149.3</td>
<td>Direct access testability interface</td>
<td>Discontinued</td>
</tr>
<tr>
<td>1149.4</td>
<td>Mixed-signal test bus</td>
<td>Std. 1149.4-1999</td>
</tr>
<tr>
<td>1149.5</td>
<td>Standard module test and maintenance (MTM) bus</td>
<td>Std. 1149.5-1995 (not endorsed by IEEE since 2003)</td>
</tr>
<tr>
<td>1149.6</td>
<td>High-speed network interface protocol</td>
<td>Std. 1149.6-2003</td>
</tr>
<tr>
<td>1149.7</td>
<td>Reduced-Pin and Enhanced-Functionality Test Access Port</td>
<td>Std. 1149.7-2009</td>
</tr>
</tbody>
</table>

### IJTAG P1687

- **IJTAG P1687 Statement of Scope:**
  - This standard will develop a methodology for access to embedded test and debug features, (but not the features themselves) via the IEEE 1149.1 Test Access Port (TAP) and additional signals that may be required. The elements of the methodology include a description language for the characteristics of the features and for communication with the features, and requirements for interfacing to the features.

### Analysis of IEEE P1687 network

- The control data for the SIB is transported on the same wire as test data
- Control data is transferred to the status register when the JTAG state machine does “apply and capture”
- “CUC” : Apply Capture Update Cycle : 5 clock cycles in the FSM

### Overall picture

```
CoreA
  SCAN 1687 DEBUG
  1 1 1 1 4 4 9 9
  7 1 7 1

CoreB
  1687 MBIST
  1 1 1 1 4 4 9 9
  7 1

CoreC
  LBIST 1687 DEBUG
  1 1 1 1 4 4 9 9
  7 1

CoreD
  SCAN
  1 1 1 1 4 4 9 9
  7 1
```

Farrokh Ghani Zadegan, Urban Ingelsson, Gunnar Carlsson and Erik Larsson Test Time Analysis for IEEE P1687, IEEE 19th Asian Test Symposium(ATS2010), Shanghai, China, Dec. 2010
Interface and scan-path

The infrastructure

Analysis: Hierarchical architecture
Concurrent test schedule

<table>
<thead>
<tr>
<th>Scan-sequence</th>
<th>Scanned Bits</th>
<th>Inc. CUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set-up</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>Sequence 1</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>Sequence 2</td>
<td>6</td>
<td>18</td>
</tr>
</tbody>
</table>
### Analysis: Hierarchical Architecture

#### Concurrent Test Schedule

```
<table>
<thead>
<tr>
<th>Scan-sequence</th>
<th>Scanned Bits</th>
<th>Inc. CUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIBs</td>
<td>SC1</td>
<td>SC2</td>
</tr>
<tr>
<td>Set-up</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Sequence 1</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Sequence 2</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Sequence 3-6</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Sequence 7-13</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Test application time Σx=223
Making fault free electronic products

Test specification

Test Preparation

Production Test

In-Field Test

A Fault Management View

NMS

displays & correlates alarms

OSS

I/f: Mur, Mub

Sub-network Management

I/f: Mun, standard Alarm IRP

Element Management of Network Elements

NE

alarm log

alarm list

alarm correlation within NE

Sub-system

Network Management

Network Management System

NMS

displays & correlates alarms

OSS

I/f: Mun, standard Alarm IRP

Element Management of Network Elements

NE

alarm log

alarm list

alarm correlation within NE

Embedded deterministic test

- Printed Circuit Boards with mounted ICs
- In-field test (boot-up, regular checks)
  - Detects manufacturing test escapes
  - Detects faults that are due to wear-out, environmental impact etc.
- In-field diagnosis
  - To determine which IC has the fault
  - Requires on-system test evaluation
- Key problem: Storing the test data on the system
  - High memory requirements
  - Inflexible in applying different tests
- The proposed solution
  - An embedded test controller to manipulate test data based on commands
  - With the help of structural information about the system
  - Reduces memory requirements and provides flexibility
  - Supports diagnosis
  - Based on IEEE 1149.1 JTAG

Embedded test-specific circuitry

Online generation and application of test data

Offline test data generation
Test data stored on-system

- We consider deterministic test data generated offline
  - Memory requirements
  - Diagnosis

Commands from test manager

CPU

Memory

Test controller

PCB

Mudassar Majeed, Daniel Ahlström, Urban Ingelsson, Gunnar Carlsson and Erik Larsson, "Efficient embedding of deterministic test data", IEEE 19th Asian Test Symposium(ATS2010), Shanghai, China, Dec. 2010
IEEE 1149.1

Basic idea behind the solution

Example

Example

PROCEDURE check_id USES maindata;
  IRSCAN 14, $09FE;
  DRSCAN 43, $0000000000000000, CAPTURE result_43[], COMPARISON $714140935654093, $0fffffff0fffffff, test_fail;
ENDPROC; 'check_id

PROCEDURE instance;
  INSTANTIATE s200 device1, device3;
  INSTANTIATE f02s device2;
ENDPROC; 'instance

PROCEDURE check_id;
  PARALLEL;
    CALL device1.id_check;
    CALL device3.id_check;
    CALL device2.id_check;
  ENDPARALLEL;
ENDPROC; 'check_id

CLASS f02s;
  PROCEDURE id_check;
    IRSCAN 8, $FE;
    DRSCAN 32, $00000000,
    CAPTURE result_32[],
    COMPARISON $f5045093, $0fffffff,
    test_fail;
  ENDPROC; 'id_check
ENDCLASS;

CLASS s200;
  PROCEDURE id_check;
    IRSCAN 6, $09;
    DRSCAN 31, $00000000,
    CAPTURE result_31[],
    COMPARISON $71414093, $0fffffff,
    test_fail;
  ENDPROC; 'id_check
ENDCLASS;