Sources & Background

- JTAG Boundary Scan is from IEEE Standard 1149.1
- Most of slides here are based on the document “Boundary Scan Tutorial” by Ben Bennetts, for ASSET InterTech Inc.,

The old way

In-Circuit & Functional Board Test

Bed-Of-Nails (MDA, ICT) Functional

Problem with modern packaging styles

DIP  PGA  SOIC  TSOP
SOJ  PLCC  QFP  BGA
Problem with multi-layer PCB

Motivation of Boundary Scan

- Basic motivation was miniaturization of device packaging, leading to …
- surface mount packaging styles, leading to …
- double sided boards, leading to …
- multi-layer boards, leading to …
- a reduction of physical access test lands for traditional bed-of-nail in-circuit testers

- Problem: how to test for manufacturing defects in the future?
- Solution: add boundary-scan registers to the devices

Principle of Boundary Scan

- Test Data In (TDI)
- Test Clock (TCK)
- Test Mode Select (TMS)
- Test Data Out (TDO)

Each boundary-scan cell can:
- **Capture** data on its parallel input PI
- **Update** data onto its parallel output PO
- **Serially scan** data from SO to its neighbour’s SI
- **Behave transparently:** PI passes to PO
- **Note:** all digital logic is contained inside the boundary-scan register

The Boundary Scan Path
Basic Boundary Scan Cell

Defect Coverage: Bed-of-nails

Defect Coverage: Exttest

Defect Coverage: Intest
**1149.1 Chip Architecture**

- **Boundary-Scan Register**
- **Internal Register**
- **Any Digital Chip**
- **Identification Register**
- **Instruction Register**

**Mandatory Instructions and Reset Modes**

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Target (Active) Register</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extest</td>
<td>Boundary Scan</td>
<td>Formerly All-0s</td>
</tr>
<tr>
<td>Bypass</td>
<td>Bypass</td>
<td>All-1s</td>
</tr>
<tr>
<td>Sample</td>
<td>Boundary Scan</td>
<td>Undefined</td>
</tr>
<tr>
<td>Preload</td>
<td>Boundary Scan</td>
<td>Undefined</td>
</tr>
</tbody>
</table>

- **IR \(\geq 2\)**
- **Reset:**
  - **TMS** \(1, 5 \times \text{TCK}\)

**Target Register Modes**

- **Boundary-Scan Register**
- **Internal Register**
- **Any Digital Chip**
- **Identification Register**
- **Instruction Register**

**Open-Circuit TDI, TMS and TRST**?

- An open-circuit TDI, TMS or TRST must go to logic-1. Why?
- TDI: Bypass instruction is loaded: safe instruction
- TMS: TAP controller placed into Test Logic Reset state after 5 TCKs, max: safe state
- TRST not asserted. Rest of 1149.1 logic still usable
Instruction Register

- DR select and control signals routed to selected target register
- Decode Logic
- Hold register (Holds current instruction)
- Scan Register (Scan-in new instruction, scan-out capture bits)
- Higher order bits: current instruction, status bits, informal ident, results of a power-up self test, ...

Standard Instructions

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Selected Data Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mandatory:</td>
<td></td>
</tr>
<tr>
<td>Exttest</td>
<td>Boundary scan (formerly all-0s code)</td>
</tr>
<tr>
<td>Bypass</td>
<td>Bypass (initialised state, all-1s code)</td>
</tr>
<tr>
<td>Sample</td>
<td>Boundary scan (device in functional mode)</td>
</tr>
<tr>
<td>Preload</td>
<td>Boundary scan (device in functional mode)</td>
</tr>
<tr>
<td>Optional:</td>
<td></td>
</tr>
<tr>
<td>Inext</td>
<td>Boundary scan</td>
</tr>
<tr>
<td>Idcode</td>
<td>Identification (initialised state if present)</td>
</tr>
<tr>
<td>Usercode</td>
<td>Identification (for PLDs)</td>
</tr>
<tr>
<td>Runbist</td>
<td>Result register</td>
</tr>
<tr>
<td>Clamp</td>
<td>Bypass (output pins in safe state)</td>
</tr>
<tr>
<td>HighZ</td>
<td>Bypass (output pins in high-Z state)</td>
</tr>
</tbody>
</table>

NB. All unused instruction codes must default to Bypass

Extest Instructions

- Boundary-scan register selected
- Used to apply patterns to the interconnect structures on the board
- Boundary-scan cells have permission to write to their outputs (device in test mode)

Bypass Instruction

- Bypass register selected
- Used to allow quick passage through this device to another device connected in the chain
Sample and Preload Instruction

- Boundary scan register selected
- Used to Preload known values in the boundary scan cells.
- Also used to Sample (Capture) mission-mode signals into the boundary-scan cells.
- Device in functional mode, not test mode

Intest Instruction

- Boundary scan register selected
- Used to apply patterns to the device itself
- Boundary scan cells have permission to write to their outputs (device in test mode)

Idcode Instruction

- Optional Identification register selected, if available, else Bypass register selected
- Used to capture internal 32-bit identification code (manufacturer, part number, version number) and then shift out through TDO

Usercode Instruction

- Optional Identification register selected, if available, else Bypass register selected
- Use to capture an alternative 32-bit identification code for dual personality devices e.g. PLDs
RunBISt Instruction

- Control registers for initiating internal BIST (Memory or Logic)
- Pass/fail register targeted as final selected register

Clamp Instruction

- Known values are pre-loaded into boundary scan cells using **Preload** instruction
- **Clamp** drives these values to the output pins but leaves Bypass register as the selected register

HighZ Instruction

- Control-to-Z values are pre-loaded into high-Z control cells using the **Preload** instruction
- **HighZ** drives these values to the three-state controls causing them to go to their high-Z drive state but leaves Bypass register as the selected register

Test Access Port (TAP)

- Test Data In (TDI) Serial data in
  Sampled on rising edge
  Default = 1
- Test Data Out (TDO) Serial data out
  Sampled on falling edge
  Default = Z (only active during a shift operation)
- Test Mode Select Input Control (TMS) Sampled on rising edge
  Default = 1
- Test Clock (TCK) Dedicated clock
  Any frequency
- Test Reset (TRST*) Optional async reset
  Active low
  Default = 1
TAP Controller

16-state FSM
TAP Controller
(Moore machine)

TMS
TCK
TRST*

ClockDR
ShiftDR
UpdateDR
Reset*
Select
ClockIR
ShiftIR
UpdateIR
Enable

TAP Controller State Diagram

Bypass Register

- One-bit shift register, selected by the **Bypass** instruction
- Captures a hard-wired 0
- Note: in the *Test-Logic/Reset* state, the Bypass register is the default register if no Identification Register present

Identification Register

- 32-bit shift register
- Selected by *Idcode* and *Usercode* instruction
- No parallel output
- Captures a hard-wired 32-bit word
- Main function: identify device owner and part number
- Note: *Idcode* is power-up instruction if Identification Register is present, else **Bypass**
### Boundary Scan Register

- Shift register with boundary-scan cells on:
  - device input pins
  - device output pins
  - control of three-state outputs
  - control of bidirectional cells

- Selected by the **Extest, Intest, Preload** and **Sample** instructions

### Boundary Scan Cell OZ

On all device signal IO, control of three-state: dual-mode input signal or additional scan cell

### Boundary Scan Cell IO

On control of bidirectional IO: dual-mode input signal

### Application at Board Level

1. BS infrastructure integrity test
2. Interconnect test
3. Non-BS device test, including RAM test
Board Defects

- Missing component, wrong component, mis-oriented component, broken track, shorted tracks, pin-to-solder open circuit, pin-to-pin solder shorts
- Number of 2-net short circuit faults between k interconnects = k(k-1)/2
- Equivalent fault models for shorts: bridging of type wired-AND and wired-OR
- Open circuits are modelled down-stream as stuck-at-1 or stuck-at-0 faults

Example of faults

Generating Open and Short Test

Scan-in stimulus 1011

How many tests are needed?

Determine a set of tests to detect all:
- open circuits, modelled as any net s-a-1 and s-a-0, and
- short circuits, modelled as all 2-net shorts of type wired-AND and wired-OR i.e. {1,2}, {1,3}, {1,4}, {2,3}, {2,4}, {3,4}
Building the tests

<table>
<thead>
<tr>
<th>Chip 1</th>
<th>Chip 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 1</td>
<td>Net 1</td>
</tr>
<tr>
<td>0 1 0</td>
<td>Net 2</td>
</tr>
<tr>
<td>0 1 1</td>
<td>Net 3</td>
</tr>
<tr>
<td>1 0 0</td>
<td>Net 4</td>
</tr>
</tbody>
</table>

- Each code has at least one 1, one 0
- 2 forbidden codes: all-1s, all-0s
- 2-net shorts: each code is unique

Determine a set of tests to detect all:
- open circuits, modelled as any net s-a-1 and s-a-0, and
- short circuits, modelled as all 2-net shorts of type wired-AND and wired-OR
  i.e. {1,2}, {1,3}, {1,4}, {2,3}, {2,4}, {3,4}

Number of Tests

Number of tests
= Number of bits in the code
= \(\text{ceil} \ log_2(k + 2)\), \(k\) = number of interconnects
= 13 for \(k\) = 8000 interconnects

It's so simple, it's beautiful!!

Testing non-Boundary Scan cluster

- On modern boards, most non-boundary-scan devices are simple pass-thru devices e.g. line drivers
- Consequently, tests for presence, orientation and bonding are easily generated and easily applied via the embracing boundary-scan devices

Combining BS and Nails

- Use ICT nails to access uncontrollable/observable cluster-internal nets
- Select the real-nail locations on non-BS nets according to access to:
  - strategic disables for guarding or preventing bus conflicts,
  - buried nets in non-embraced clusters,
  - other key control signals e.g. O_Enab, Bidir or 3-state control signals
RAM Array Testing via Boundary Scan

Tool Flow for Boundary Scan Tests

Hardware Requirement

Where are we today?
Spread of Design-for-Test (DFT)

- **Chip**
  - Boundary-Scan Technology
  - 5001-00 Emulation (eJTAG)
  - P1500-02(?) Embedded Core Test

- **Board**
  - 1149.1-99/03/94/01 Interconnect Test
  - Chip DFT Re-use (Int. Scan, BIST, I/PDC)
  - 1149.4-99 Mixed-Signal Bus

- **System/Field Service**
  - Texas Inst. Natl Semi Back-Plane Bus
  - 1532-00 In-System Configuration