Accessing Embedded Instruments through the Life Cycle Using P1687

Farrokh Ghani Zadegan, Gunnar Carlsson, Erik Larsson
Motivation

Life Cycle of a Chip

On-chip instruments used in:

- **Chip-level:**
  - Debugging
  - Characterization
  - Burn-in
  - Manufacturing test

- **Board-level**
  - Bring-up
  - Manufacturing test

- **System-level**
  - Monitoring & fault management
  - In-field test

Photography supplied courtesy of Intel Corporation
Motivation

• An instrument might be used in several scenarios
• Example: A Memory Built-In-Self-Test might be used in
  • Characterization $\Rightarrow$ choosing most suitable algorithms
  • Manufacturing test $\Rightarrow$ detecting & repairing defective ICs
  • Burn-in $\Rightarrow$ causing activity in the chip and detecting infant mortality
  • PCB debugging $\Rightarrow$ taking a suspected memory error out of the equation
  • PCBA manufacturing test $\Rightarrow$ detecting defective ICs
  • Power-on-self-test & in-field test $\Rightarrow$ detecting defective ICs
Motivation

- How to access the instruments?
- IEEE P1687 standardizes the access through the JTAG TAP
  - On-chip instruments become accessible through the whole life cycle
- Many ways to build the on-chip instrument access infrastructure (network)
  - How to do it efficiently?
  - In this work we describe and compare six network design approaches w.r.t
    - Overall access time
    - Hardware overhead
    - Robustness toward new scenarios
- We believe the work is also applicable to IEEE 1149.1-2013
Outline

• Introduction & background
• Design approaches
• Experiments
• Conclusion
Overview of P1687

Instrument Interfaces

Access Network

Chip Boundary

Chip

Inst. A

Inst. B

Inst. C

TDI

TDO
Overview of P1687

Instrument Interfaces

Access Network

Chip Boundary

Chip

Inst. A

Inst. B

Inst. C

SIB: Segment Insertion Bit

TDI

TDO
Introduction & Background

Overview of P1687

Instrument Interfaces

Access Network

Chip Boundary

Chip

Inst. A

Inst. B

Inst. C

TDI

TDO

SCB: ScanMux Control Bit
Introduction & Background

• Prior work on network design for P1687 [Zadegan et al, DATE’11]
  • SIB-based
  • One scenario
    – sequential access (one instrument at a time)
    – concurrent access (all instruments at the same time)
  • Overhead
    – Not affected by length of shift-registers
    – Affected by number of accesses and schedule
    – Both vary by scenario
• Networks optimized for one scenario, not optimal for others
In this work,

- we assume given is
  - A number of scenarios, each assigned a weight $W_s$ by the designer
  - The number of accesses for each instrument per scenario $A_{i,s}$
  - For each scenario, sequential/concurrent access
- we compare six design approaches w.r.t.
  - Overall access time (OAT)
  - Hardware overhead
  - Robustness against new scenarios
- each design approaches is either
  - SIB-based, or
  - SCB-based (daisy-chained)
Outline

• Introduction & background
• Design approaches
• Experiments
• Conclusion
Design Approaches

• We consider the following approaches:
  • A regular scan-chain
  • Flat network
    – SIB-based
    – SCB-based (daisy-chained)
  • Hierarchical network
    – SIB-based
    – SCB-based (daisy-chained)
  • Multiple networks
    – SIB-based
    – SCB-based (daisy-chained)
Design Approaches

- A regular scan-chain

- Assume three instruments with 5, 3, and 10 accesses
- Instruments always on the scan-path
  - Overhead: dummy-bits shifted for inactive instruments
Design Approaches

- Flat network (SIB-based)
  - Instruments included in the scan-path when needed
    - Time overhead: SIBs on the scan-path
    - Hardware overhead: SIBs
  - Straightforward to design
Design Approaches

- Flat network (daisy-chained)

- Instruments included in the scan-path when needed
- Time and hardware overhead:
  - SCBs on the scan-path
  - Bypass registers
- Straightforward to design
Design Approaches

• Hierarchical network (SIB-based)

• Instruments included in the scan-path when needed
  • Overhead: SIBs on the scan-path
Design Approaches

- How to design the hierarchy for multiple scenarios?

For each instrument $A_{i,total} = \sum_{s \in S}(A_{i,s} \times W_s)$

- Use methods in [Zadegan et al, DATE’11]
Design Approaches

- Hierarchical network (daisy-chained)

- Instruments included in the scan-path when needed

Overhead:
- SCBs on the scan-path
- Bypass registers
Design Approaches

- **Multiple networks**

- One optimized network per scenario
- Two alternatives:
  - SIB-based
  - SCB-based (daisy-chained)
Outline

• Introduction & background
• Design approaches
• Experiments
• Conclusion
Experiments

• Comparing design approaches:
  • Overall access time (OAT)
    – Weighted sum of the access time for each scenario
  • Hardware overhead
  • Robustness for new scenarios
## Experiments

### Instruments and Access Scenarios

<table>
<thead>
<tr>
<th>Instruments</th>
<th>Scenarios (weights)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Count</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>
# Experiments

## Overall Access Time (for known scenarios)

<table>
<thead>
<tr>
<th>Approach</th>
<th>Weighted overall access time ($\times 10^6$)</th>
<th>Sum ($\times 10^6$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$S_1(1)$ Seq.</td>
<td>$S_2(100)$ Con.</td>
</tr>
<tr>
<td>Regular scan-chain</td>
<td>8427</td>
<td>20.3</td>
</tr>
<tr>
<td>Flat Network</td>
<td>525</td>
<td>4.7</td>
</tr>
<tr>
<td>Flat daisy-chained</td>
<td>525</td>
<td>4.5</td>
</tr>
<tr>
<td>Hierarchical network</td>
<td>152</td>
<td>4.1</td>
</tr>
<tr>
<td>Hierarchical daisy-chain</td>
<td>152</td>
<td>3.9</td>
</tr>
<tr>
<td>Multiple networks</td>
<td>151</td>
<td>3.8</td>
</tr>
<tr>
<td>Multiple daisy-chains</td>
<td>151</td>
<td>3.7</td>
</tr>
</tbody>
</table>
## Experiments

### Hardware Overhead

<table>
<thead>
<tr>
<th>Approach</th>
<th>Flip-flops</th>
<th>Muxes</th>
<th>Buffers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular scan-chain</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Flat Network</td>
<td>200</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>Flat daisy-chained</td>
<td>302</td>
<td>101</td>
<td>0</td>
</tr>
<tr>
<td>Hierarchical network</td>
<td>286</td>
<td>143</td>
<td>0</td>
</tr>
<tr>
<td>Hierarchical daisy-chain</td>
<td>517</td>
<td>187</td>
<td>0</td>
</tr>
<tr>
<td>Multiple networks</td>
<td>1172</td>
<td>586</td>
<td>400</td>
</tr>
<tr>
<td>Multiple daisy-chains</td>
<td>1940</td>
<td>677</td>
<td>400</td>
</tr>
</tbody>
</table>
## Experiments

### Overall Access Time (for new scenarios)

<table>
<thead>
<tr>
<th>Approach</th>
<th>S6 (seq.)</th>
<th>S7 (con.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>OAT (x10^6)</td>
<td>Potential improvement</td>
</tr>
<tr>
<td>Regular scan-chain</td>
<td>2006</td>
<td>0%</td>
</tr>
<tr>
<td>Flat Network</td>
<td>125</td>
<td>0%</td>
</tr>
<tr>
<td>Flat daisy-chained</td>
<td>125</td>
<td>0%</td>
</tr>
<tr>
<td>Hierarchical network</td>
<td>40</td>
<td>17.56%</td>
</tr>
<tr>
<td>Hierarchical daisy-chain</td>
<td>40</td>
<td>17.55%</td>
</tr>
<tr>
<td>Multiple networks</td>
<td>36</td>
<td>11.78%</td>
</tr>
<tr>
<td>Multiple daisy-chains</td>
<td>36</td>
<td>11.79%</td>
</tr>
</tbody>
</table>
Conclusion

- P1687 to access the on-chip instruments
- Instruments used in multiple scenarios
  - Different schedules
  - Different number of accesses
- Network optimized for one scenario, not optimal for others
- We compared six approaches, regarding
  - Overall access time
  - Hardware overhead
  - Robustness against new scenarios
Conclusion

Observations from the comparison

- Use of multiple networks results in the least OAT
  - at the cost of extra hardware
- Daisy-chains perform better for concurrent access
  - Slightly higher hardware overhead compared with their SIB-based counterparts
- A single hierarchical network resulted in low OAT
  - with a relatively low hardware overhead
Accessing Embedded Instruments through the Life Cycle Using P1687

Farrokh Ghani Zadegan, Gunnar Carlsson, Erik Larsson