Remedy for an asynchronous weakness:  
A fully-testable interconnect fabric  

Aristides Efthymiou  
John Bainbridge  
Doug Edwards  

APT Group (Amulet)  
University of Manchester  
www.cs.man.ac.uk/apt
Testing - introduction

- Fabricated ICs may have defects
  - Completely different issue to design verification
  - The earlier these are discovered in the manufacture process, the more $$ are saved.
  - Functional testing hard to guarantee high fault coverage

- Fault models
  - Too many types of faults exist to model directly
  - Abstractions used, e.g. stuck-at fault model
    - A fault manifests itself as a gate input/output tied to 0 or 1.

The same patterns also test B sa0, output sa0
Scan testing

- Sequential circuits require more attention

- Using scan, registers used as inputs/outputs
  - Testing sequential circuits is simplified
  - The process is automated by tools for scan insertion, pattern generation
Testing asynchronous circuits

- Data path (bundled-data) is easy to test
- Control is hard:
  - State stored in ‘special’ sequential gates and/or feedback loops
  - State bits are not updated at the same time

Async. mode: \( \varphi_1 = \varphi_2 = 1 \)
Shift mode: 2 phase non-overlap
Eval. mode: \( \varphi_1 = 1, \varphi_2 = 0 \)
A new, improved test method

- Standard asynchronous method: break all feedback loops with scan-latches
  - A huge number of scan-latches required
- New method: systematic partial-scan
  - Place scan-latches only at ‘global’ loops
  - Sequential pattern generation for circuits with local loops

- Progress:
  - Automated sequential pattern generation for C-gates, a very important class of asynchronous sequential circuits
  - Successful application to an asynchronous interconnect fabric for systems-on-chip

- Future work:
  - Investigate general applicability of the method
Dual-slave scan latch

- State must be kept while the next pattern is being scanned in
  - A third latch is required compared to a standard scan-latch

![Latch Diagram]

- Asynchronous: \( en = 0, \varphi_2 = 0, \varphi_1 = \varphi_3 = 1 \)
- Scan: \( en = 1, \varphi_3 = 0, \varphi_1, \varphi_2 \) non-overlapping 2-phase clock
- Evaluate: \( en = 0, \varphi_3 = 1, \varphi_1 = \varphi_2 = 0 \)
- Load: \( en = 0, \varphi_1 = 1, \varphi_2 = \varphi_3 = 0 \)
Sequential patterns for C gates

Very common storage/synchronization element in asynchronous design

\[ q = ab + (a+b)q \]

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>q</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>q</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>q'</th>
<th>q</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Asynchronous interconnect (CHAIN)

- The basic CHAIN circuit: 1/5 pipeline latch
  - Only 1 scan-latch per pipe latch
  - Standard, full-scan approach requires 5 scan-latches
Local patterns handling

- Controlling the `A` input

- Fault propagation through C-gates

- B inputs depend on the previous C-gate output
  
  If expected, good-machine `a == q`,
  
  \[ B \leftarrow \neg a \]

  If expected, good-machine `a != q`,
  
  \[ B \leftarrow a \]
Experimental results

- The new test method was applied to the ASPIDA SoC interconnect network
  - 3 initiators (‘masters’)
  - 5 targets (‘slaves’)
- Automatic pattern generation
  - Input the topology, scan-chain
  - Output a pattern file
- Over 99.5% fault coverage
- Area improvement:
  - 1/3 scan-latches than std.
  - 1/2 including boundary
  - 60% less area (incl. boundary)
- Delay overhead unchanged
Summary

- Testing of asynchronous circuits
  - A novel technique offering significant area savings
  - Fully-testable asynchronous interconnect

- Extensions
  - Async. interconnect: M of N codes
    - Completion detection more complex than an OR gate
    - Encode/decode circuits need to be testable
  - Generalization of the algorithm