Framework for Massively Parallel Testing at Wafer and Package Test

Kee Sup Kim
Samsung Electronics
Key Message

- Massively parallel testing
  - Possible
  - Positive Return
- DFT Can be exciting
Outline

- Introduction
- Massively parallel test
- Return on investment
- Conclusion
Landscape

- Whole Wafer Touch Down
  - Some DRAM products
- More IC’s are becoming low power
- Low Number of pins needed for testing
Wafer probe technology

- Wentwoth Labs, Electroglas, FormFactor Inc., Tokyo Electron, SV Probe etc.
- 300mm Dram Wafer – 1 touchdown
  - 300mm wafer probe – 150K probes
- Comparable performance in logic
- Parallelism on a test board
Parallelism in Testing

- Test Controller
  - No full pin access
  - Network on chip
- Reduced pin tests for Scan
Parallelism in Testing

- Lox Power: 0.5 - 1 W
- 100X – 1000X parallelism
Parallelism in Testing

- Replicate ATE channels
- Not scalable
Parallelism in Testing

- Replicate ATE channels
- Not scalable
Parallelism in Testing

- Replicate ATE channels
- Not scalable
Parallelism in Testing

- Complexity → Interconnection network
- Scalable
- Optimize for speed, cost, etc.
Outline

- Introduction
- Related work
- Massively parallel test
  - Configuration
  - Initialization
  - Test data communication
  - Test result collection
- Return on investment
- Conclusion
Configuration

Also applicable to a standard test board
Network Types

- Mesh
  - Most Straight Forward
  - Good Redundancy
- Taurus
  - Even better redundancy
- Bus
  - Can save on pins
- Tree
Test Router

- North Port
- Tester Port
- West Port
- East Port
- South Port
- Control Logic & Status Registers
- Registers
Challenges

• Identical Devices
  • how do you tell one from another?

• How about DC tests?

• Would you get speed up?
  • If inputs are the same but outputs are all different?

• How about Vdd to GND short?
  • It can make all dies fail

• What if a node is bad?
  • Would you lose good chips because of a few bad?
Test Phases

- Initialization
- Test content delivery
- Result Collection
- Diagnosis

Network algorithms are implemented in a “Test Router”
Initialization

• Unique identification of all die’s
  • Dies are the same
  • Same process, same masks
  • No hardwired ID

• Relative position w.r.t. root die
Initialization – Identification
Initialization – Identification
Initialization – Identification

Status registers

<table>
<thead>
<tr>
<th>ID x cor</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID y cor</td>
</tr>
</tbody>
</table>
Initialization – Breadth First Tree

Status registers

<table>
<thead>
<tr>
<th>ID x cor</th>
<th>ID y cor</th>
<th>Parent</th>
<th>Children</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Initialization – Breadth First Tree

<table>
<thead>
<tr>
<th>Status registers</th>
<th>Parent</th>
<th>Tester</th>
</tr>
</thead>
<tbody>
<tr>
<td>Children</td>
<td>South, West</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status registers</th>
<th>Parent</th>
<th>North</th>
</tr>
</thead>
<tbody>
<tr>
<td>Children</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>
Initialization – Breadth First Tree

- Fault tolerance
Initialization

{
    Root sends ID message to everyone;
    Once message is received from neighbor,
        set self-ID based on direction of message;
    Send ID to neighbors who is not the parent;
    Simple & consistent tie breaking algorithm;
}

Reducing Unique Traffic

- Inputs – Common
- Outputs – Different
- Solution
  - Send input & output patterns to each die & do on-die comparison
  - Initialization & actual testing becomes pipelined
Test Data Communication

- Test patterns – pipelined
- Use the breadth first tree
- Required data:
  - Input pattern
  - Expected output
  - Output Mask
Test Data Communication

- Test patterns – pipelined
- Use the breadth first tree
- Required data:
  - Input pattern
  - Expected output
  - Output Mask
Test Data Communication

- Test patterns – pipelined
- Use the breadth first tree
- Required data:
  - Input pattern
  - Expected output
  - Output Mask
Test Data Communication

- Test patterns – pipelined
- Use the breadth first tree

Required data:
- Input pattern
- Expected output
- Output Mask

#1 #2 #3 #4

ATE
Test Data Communication

- Test patterns – pipelined
- Use the breadth first tree
- Required data:
  - Input pattern
  - Expected output
  - Output Mask
Test Data Communication

- Test patterns – pipelined
- Use the breadth first tree
- Required data:
  - Input pattern
  - Expected output
  - Output Mask

#3 #4 ATE
Test Data Communication

- Test patterns – pipelined
- Use the breadth first tree
- Required data:
  - Input pattern
  - Expected output
  - Output Mask

ATE #4
Test Result Communication

- Test finishes – all dies have test result info
- Local info
- Transmit to tester over root
- Use the breadth first tree (reverse order)
  - Every die sends info to parent
  - Until root is reached
Result Communication in Parallel

Each DUT {
    
    Grant message token to next child;
    Receive message from Child;
    Update fail map w/ new data;
} until message from all children received;

Update fail map with own fail info;
Wait for “message token” from parent;
Communicate fail map to parent;
}
Test Result Communication
Test Result Communication
Test Result Communication
Diagnosis

- Done one at a time
Outline

- Introduction
- Related work
- Massively parallel test
- Return on investment
- Conclusion
Return on Investment

- Total # of 5mm x 5mm dice on 12in wafer
  - About 2500
  - 2500X reduction in test time?

- Return
  - Even with power limitation, 100X reduction should be possible

- Cost
  - Area for Enhancements to Test Controller (50K gates – first cut estimate) .0024 cents per die
  - SRAM for Die result map is shared from existing SRAM
Will Amdahl prevail?

- Serial Test Time small
- Additional time to address diagnosis information
  - with 8bit width, & 80% yield
  - 67.5K Cycles @ 200 MHz
    - .03% increase in test time (when total test time is 2 sec)
    - .006% increase in (when total test time is 10 sec)

- Initialization time
  - About 1K cycle
  - Minuscule compared to total test time
Total Savings (Plug in your numbers)

<table>
<thead>
<tr>
<th>Per Die Savings due to 100x test time reduction</th>
<th>40 cents</th>
</tr>
</thead>
<tbody>
<tr>
<td>For life of 200M devices</td>
<td>$80M</td>
</tr>
</tbody>
</table>
Outline

- Introduction
- Related work
- Massively parallel test
- Return on investment
- Conclusion
Conclusion

- Framework for Massively Parallel Testing
  - Wafer Sort
  - Packaged Test
- Promising Results
  - Area Overhead Very small
  - Serial testing portion very small
  - Yield loss minimal
What about challenges?

- Power to Ground Short
- What if “root” is bad?
  - Will the solution complicate the router?
  - How do I know when to invest in this solution?
- What if there are “bad neighborhood”?
- What if your device volume is low
- Can you use it during package testing?
  - What would be the challenge?
- What if you can’t afford pins?
- What would you do after this project?