Design for Test of Digital Systems

TDDC33

Erik Larsson

Department of Computer Science
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
- Test data compression, Built-In Self-Test; Logic BIST and memory BIST
- System Chip Test; test architectures, test planning, test scheduling, and power constraints
- System Test and Boundary Scan
System-on-Chip

- Viper 2.0 RevB
- Analog/Digital TV Processor
- 10mm x 10 mm (100 mm$^2$)
- ~10 M gates
- ~50 M transistors
- ~100 clock domains
System-on-Chip

- Processor Cores
  - ARM, MIPS, PowerPC
- Memories
  - SRAM, ROM, Flash, DRAM
- DSP Cores
- Peripherals
  - DMA Controllers, MMU
- Interface
  - PCI, USB, UART
- Multimedia
  - JPEG compression, MPEG decoder
- Networking
  - Ethernet controller
Modular Test Design

- Test Quality
  - Different parts (logic, memory, analog, RF) need different test methods
- Black-boxed Embedded Core
  - Implementation is not known, forced to use tests developed by provider
- Divide-and-Conquer
  - Very large SOCs are intractable for ATPG/FSim tools
  - Modular test approach allows concurrent development/engineering
- Test Reuse
  - Module will be reused in other designs
Challenges

- Distributed Design and Test Development
  - Standardized set of deliverables
- Test Access to Embedded Modules
  - Standardized on-chip test access hardware
  - Tools for test translation
- Chip-Level Test Optimization
  - Tools to evaluate trade-offs; minimal impact on design (extra silicon, delay) at minimizing test application time and ATE memory requirement
Scan Test Application

Test vectors: 10

Test vectors: 20
Non-modular Alternative

Test time = \((20 + 10 + 1) \times 20 + (20 + 10) = 650\)

Non-modular alternative:
Modular Alternative

Core 1:

Test time = \((20+1) \times 10 + (20)\) = 230

Test vectors: 10

Scan chain 1 (20 FFs)

Scan chain 0 (20 FFs)

Capture

Core 2:

Test time = \((10+1) \times 20 + (10)\) = 230

Test vectors: 20

Scan chain 1 (10 FFs)

Scan chain 0 (10 FFs)

Total test time: 460
### Scan Test Application

Scan in + capture + scan out\(\rightarrow\) (sc+1)*p+sc

**Non-modular alternative:**

Test time = \((20+10+1)*20+(10+20)\) = 650

**Modular alternative:**

Core1: \(sc=20, p=10\) \(\rightarrow\) \((20+1)*10+20=230\)

Core2: \(sc=10, p=20\) \(\rightarrow\) \((10+1)*20+10=230\)

Total = Time(Core1)+Time(Core2)=460 \(\text{(30\% cut)}\)
Generic Test Access Architecture

- Test pattern Source and Sink
  - Store/generate test stimuli and store/evaluate test responses
- Test Access Mechanism (TAM)
  - Transports test patterns to/from module under test (MUT)
- Test Wrapper
  - Provides test access to MUT
  - Isolates MUT at test
Test Planning

- Objectives: Optimizing test access to cores and scheduling tests
IEEE 1500 Core Test Standard

- **Goals**
  - Define a core test interface between an embedded core and the SOC
  - Core isolation and protection
  - Plug-and-play protocols

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

- Test 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 between cores
- IEEE 1500 Standard for Embedded Core Test
Test Wrapper

WSP: Wrapper Serial Port
WSI: Wrapper Serial Input
WSC: Wrapper Serial Control
WSO: Wrapper Serial Output
WPP: Wrapper Parallel Port
WPI: Wrapper Parallel Input
WPC: Wrapper Parallel Control
WPO: Wrapper Parallel Output
Test Wrapper

CTI: Cell Test Input
CTO: Cell Test Output
CFI: Cell Functional Input
CFO: Cell Functional Output
FI: Functional Input
FO: Functional Output
WFI: Wrapper Functional Input
WFO: Wrapper Functional Output
WBC: Wrapper Boundary Cell
WBR: Wrapper Boundary Register
WBY: Wrapper Bypass Register
WIR: Wrapper Instruction Register
WSP: Wrapper Serial Port
WSI: Wrapper Serial Input
WSC: Wrapper Serial Control
WSO: Wrapper Serial Output
WPP: Wrapper Parallel Port
WPI: Wrapper Parallel Input
WPC: Wrapper Parallel Control
WPO: Wrapper Parallel Output

(User-defined WPP = WPI+WPO+WPC)

Core

Test enable

WBC

CTI

CFI

CTO

WFI

WFO

WPP

WPI

WPO

WPC

WSI

WSC

WBR

WIR

WBY

SelectWIR

ShiftWR

CaptureWR

UpdateWR
Wrapper Boundary Cell

Input cell
- From chip
- From WSI/WPI

Output cell
- From core
- From WSI/WPI

To core
To WSO/WPO

To chip
To WSO/WPO

Clock (Clk)

Shift (shift)

Write Clock Input (wci)

Flip-Flop (FF)
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

Wrapper cells are programmed to control the WFO and observe (capture) the WFI
Test Wrapper: WS_INTEST

Wrapper cells are programmed to control FI and observe (capture) the FO of the core.
Multiplexed TAM
Direct Access TAM
Daisy-Chained TAM
Architecture Design

ATE channels

Multiplex TAM

Test time

Direct Access TAM

Daisy-Chained TAM

CoreA  CoreC  CoreB

CoreA

CoreC

CoreB

CoreA+CoreB+CoreC  B+C  B
Architecture Design

Multiplex TAM + Direct Access TAM

Flexible Architecture

Test bus 1
Test bus 2
Problem

- For a given SoC:
  - form wrapper chains out of the scan-chains and the wrapper cells at every core
  - connect the wrapper chains to TAMs, and
  - assign a time for testing each core,
- such that the total test time is minimized.
Architecture Design
Wrapper Design

Test time \( T = (sc+1)p+sc \)

\[ T = (400+1) \times 10 + 400 = 4410 \]

\[ T = (200+1) \times 10 + 200 = 2210 \]
Wrapper Design

1. Minimize length of longest wrapper scan in/out chain
2. Minimize number of wrapper scan chains
Test Wrapper Optimization

Priority 1: Balanced Wrapper Scan Chains

Minimize length of longest wrapper scan in/out chain
Reducing TAM Width

Priority 2: Minimize wrapper scan chains created

Scan chain – 32 FF

4 Wrapper scan chains

Scan chain – 32 FF

2 Wrapper scan chains
Core To TAM Assignment

[Diagram showing a system-on-chip (SoC) with various components labeled: Mem 1, Mem 2, Logic 1, Logic 2, CPU, A, B, C, D, E, TAM 1, TAM 2, TAM 3.]
Test Bus Architecture

- Combination of multiplexing and distribution
- Supports only serial schedule
- Core-external testing is cumbersome or impossible
Test Scheduling
Test Scheduling
Test Scheduling
Problem

- For a given SoC:
  - form wrapper chains out of the scan-chains and the wrapper cells at every core
  - connect the wrapper chains to TAMs, and
  - assign a time for testing each core,

- such that the total (or expected) test time is minimized.
Test Scheduling
Test Scheduling
Test Scheduling
Test Scheduling

ATE (produced response)

Fault at module E

ATE (expected response)
Abort-on-fail testing

Over a large set of ICs, minimize overall test time
Abort-on-fail testing

ATE (produced response)

Fault at module E

ATE (expected response)
Abort-on-fail testing

Without Abort-on-Fail

With Abort-on-Fail

Time to determine a possible fault in module E
Abort-on-fail testing

Idea

- Spend less time on faulty circuits
- If the test fails, it is aborted early
- Low-yielding and short tests should be performed early

Problem

- Find a test schedule that minimizes the expected test time.

Assumptions

- Abort-on-fail: when a fault occurs, testing terminates.
- Defect probability for each testable unit is given.
- Sequential testing and concurrent testing.

- Urban Ingelsson, Sandeep Goel, Erik Larsson, and Erik Jan Marinissen, Test Scheduling for Modular SOCs in an Abort-on-Fail Environment, European Test Symposium (ETS’05), 2005, pages 8-13
Example

- Test time: 15.

<table>
<thead>
<tr>
<th>Core</th>
<th>Test time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>
Sequential testing

- Without abort-on-fail: 15

<table>
<thead>
<tr>
<th>Core</th>
<th>Test time</th>
<th>Pass probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>0.7</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>0.8</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0.9</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>0.95</td>
</tr>
</tbody>
</table>

- Expected test time: 13.6

- Expected test time: 9.5
Sequential testing

At time point $\tau_1$:

- Probability to pass, $p_1 = 0.7$.
- Weighted probability to pass, $\tau_1 p_1$: $2 \times 0.7 = 1.4$
- Probability to fail $(1-p_1) = 0.3$.
- Weighted probability to fail, $\tau_1 (1-p_1)$: $2 \times 0.3 = 0.6$
Sequential testing

At time point $\tau_1 + \tau_2$:
- Probability to pass -> pass test at core 1 and test at core 2.
- Probability to fail -> pass test at core 1 and fail test at core 2.
Sequential testing

\[ p_1 = 0.7 \]
\[ p_2 = 0.8 \]
\[ p_3 = 0.9 \]
\[ p_4 = 0.95 \]

At \( \tau_1 \) : \( \tau_1 x (1-p_1)=0.6 \)

At \( \tau_1 + \tau_2 \) : \( (\tau_1+\tau_2)xp_1(1-p_2)=0.84 \)

At \( \tau_1 + \tau_2 + \tau_3 \) : \( (\tau_1+\tau_2+\tau_3)xp_1xp_2(1-p_3)=0.50 \)

At \( \tau_1 + \tau_2 + \tau_3 + \tau_4 \) : \( (\tau_1+\tau_2+\tau_3+\tau_4)xp_1xp_2xp_3(1-p_4)=0.38 \)
\[ (\tau_1+\tau_2+\tau_3+\tau_4)xp_1xp_2xp_3xp_4=7.2 \]

Total expected time : \( 7.2+0.5+0.84+0.6=9.5 \)
Concurrent testing

Probability to pass at time point $\tau_{11}$ (=2):
- Core 1: $p_1 = 0.7$  
- Core 2: $p_{2k}, k = \tau_{11}/\tau_2 = 0.82/4$  
- Core 3: $p_{3k}, k = \tau_{11}/\tau_3 = 0.92/3$

Probability to fail at time point $\tau_{11}$ (=2): $\tau_{11} x ((1-p_{11}) x p_{21} x p_{31})$
Constraints to consider

- Power consumption
  - An SOC is designed according to functional power consumption
  - In testing, switch as many “sites” as possible in order to test as much as possible in a short time
  - Power consumed during testing is
    - Higher and different from “functional”-mode power
  - Burn an IC or getting wrong results

- SoC Test Planning including Test Data Compression
IC Test

ATE

TEST STIMULUS

SOC

PRODUCED RESPONDS
Power-Aware Test Approaches

- Design SOC to handle test power consumption
- Design SOC with test power reducing techniques
- SOC test planning to handle test power consumption

Test planning is a low-cost alternative to:
- Explore ordering of tests to lower the test application time
- Guide the search for bottlenecks where
  - design for low-power techniques are to be included or
  - (over) design for test power is needed
Test Data Compression

- W TAM wires are expanded two m scan chains (m>>W)
- Reduces test time and test data volume
Test Data Compression for Core-based SOC

- Major drawback
  - High number of TAM wires \((m)\) are routed to all cores
Test Data Compression for Core-based SOCs

- TAM wires (W) are partitioned into test buses
- Cores are connected to test buses
- Few TAM wires (W) are routed to the cores
- Decoder design at core-level
Analysis of Compression Techniques

![Graph 1: Test-data volume vs. No. of TAM wires](image1)

- Vector Repeat
- Selective Encoding
- Selective Encoding & Vector Repeat

![Graph 2: Test time vs. No. of TAM wires](image2)

- Vector Repeat
- Selective Encoding
- Selective Encoding & Vector Repeat

No. of TAM wires (w)

Test-data volume (Mbits)

Test time (clock cycles)
Analysis of Compression Techniques

- At core-level define
  - number of TAM wires \( (w) \),
  - number of wrapper-chains \( (m_i) \),
  - compression technique, and
  - decoder
- such that the core’s test time and test-data volume are minimized

- At SOC-level define
  - number of test buses,
  - width \( (w_j) \) of test buses,
  - core’s assignment to test buses, and
  - compression technique, and
  - decoder
- such that the SOC’s test time and test-data volume are minimized.