20. Design for Testability

Jacob Abraham

Department of Electrical and Computer Engineering
The University of Texas at Austin

VLSI Design
Fall 2020

November 5, 2020

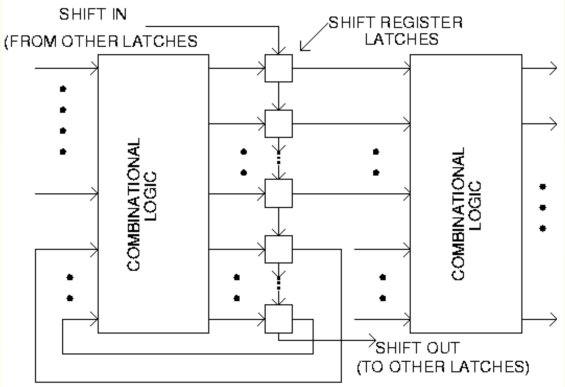
Design for Testability (DFT)

- Reduce costs associated with testing complex circuit
- Design circuit so that it will be easier to test
- Increase **accessibility** of internal nodes
  - **Controllability:** ability to establish specific signal value at each internal node by setting inputs
  - **Observability:** ability to determine internal values by controlling inputs and observing outputs
- Ensure **predictable** circuit responses
- Tradeoffs
  - Technical: area, I/O pins, performance
  - Economic: design time, yield, time to revenue

ECE Department, University of Texas at Austin Lecture 20. Design for Testability Jacob Abraham, November 5, 2020 1 / 38

Testable Sequential Circuits

Sequential circuits are very difficult to test



Design the internal memory elements to be part of a shifter register chain to provide controllability, observability through serial shifts

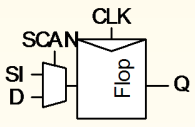
With **scan chain**, problem of testing any circuit reduces to testing the combinational logic

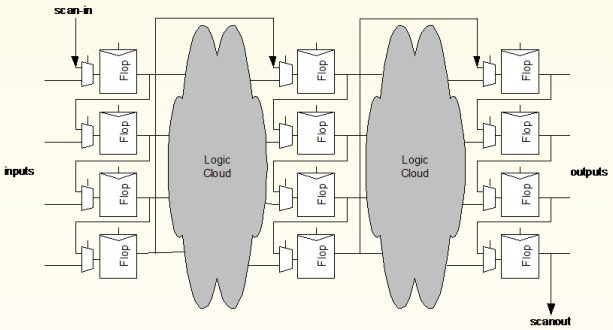
Level-Sensitive Scan Design (LSSD)

- Structured DFT developed at IBM
  - All internal storage implemented in **hazard-free polarity-hold latches (SRLs)**, part of a **scan chain**
- Latches controlled by **two or more non-overlapping clocks**, with rules for clocking
  - All clock inputs to SRLs must be in their "off" states when primary inputs (PIs) are "off"
  - Clock signal at any clock input to an SRL must be controlled from one or more clock PIs
  - No clock can be ANDed with another clock
  - Clock PIs cannot feed data inputs to latches, either directly or through combinational logic

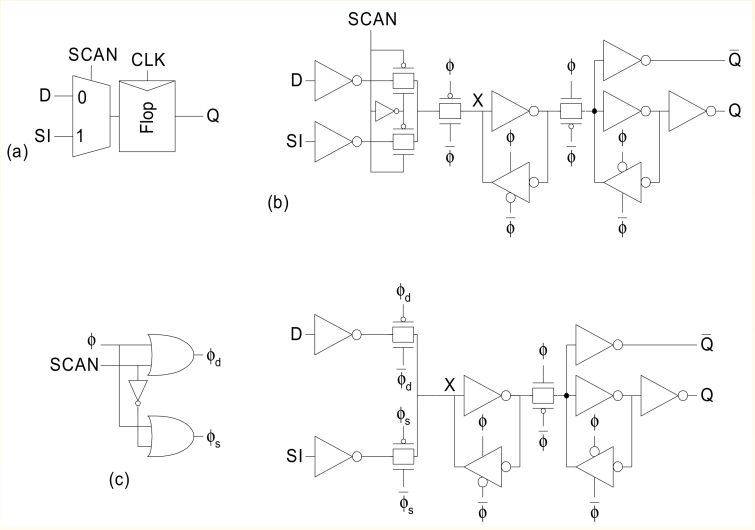
Scan Chains

- Convert each flip-flop to a scan register
  - Only costs one extra multiplexer
- Normal mode: flip-flops behave as usual
- Scan mode: flip-flops behave as shift register
- Contents of flops can be scanned out and new values scanned in





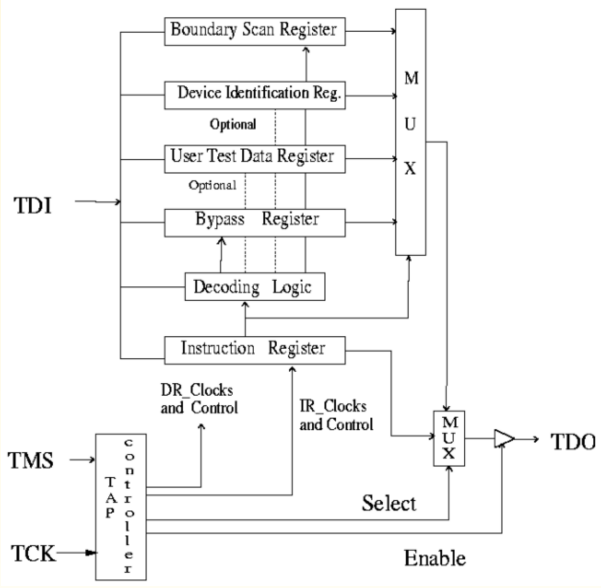
Scannable Flip-Flops



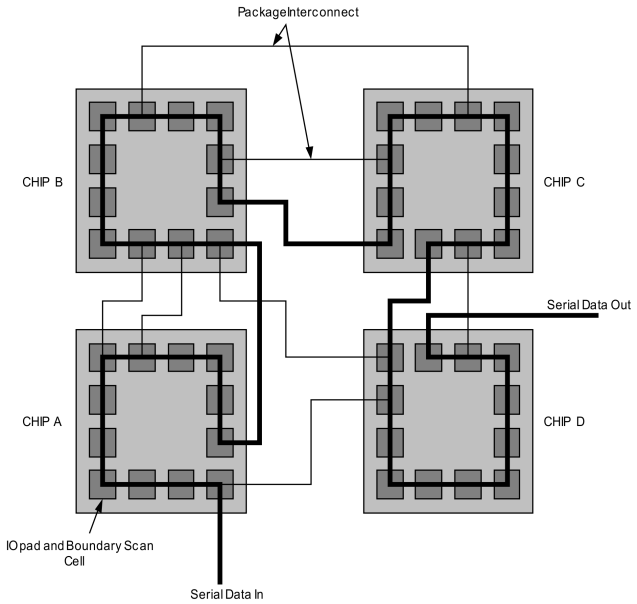
Boundary Scan

- Testing boards is also difficult
  - Need to verify solder joints are good
  - Drive a pin to 0, then to 1
  - Check that all connected pins get the values
- Single-sided PC boards with "through-hole" construction used "bed of nails" to contact pins of chip on the back side of the board
- SMT and BGA boards cannot easily contact pins
- Build capability of observing and controlling pins into each chip to make board test easier

Boundary Scan (IEEE 1149.1)



Boundary Scan Example



Boundary Scan Interface

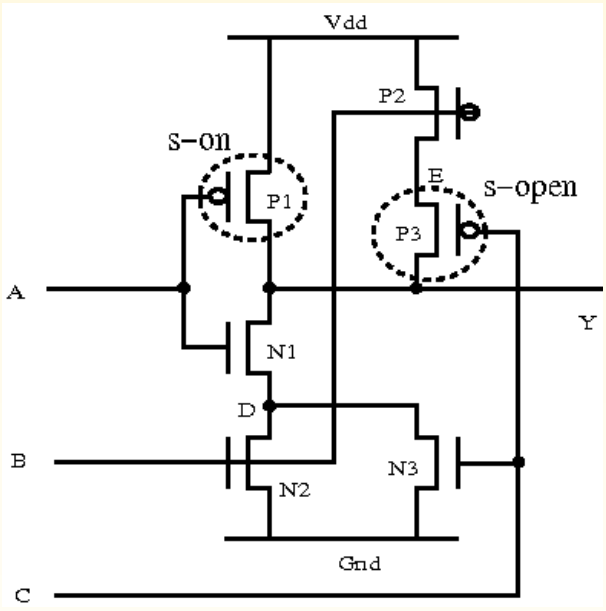
- Boundary scan is accessed through five pins
  - TCK: test clock
  - TMS: test mode select
  - TDI: test data in
  - TDO: test data out
  - TRST*: test reset (optional)
- Chips with internal scan chains can access the chains through boundary scan for unified test strategy

I_{DDQ} Testing

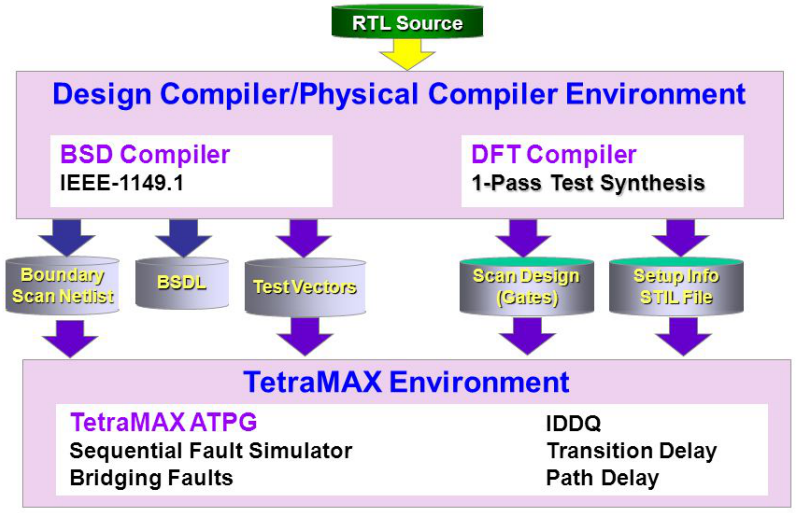
- Measure **quiescent power supply current** of CMOS circuits for selected test vectors
- Example, microprocessor has nanoamps of current with no faults; many defects (shorts, for example) cause much higher currents
- Direct relationship found (Sandia Labs., HP, Phillips) between IDDQ test acceptance rates and quality, reliability of ICs
- Only need to activate site of potential defect (no need to propagate errors)

Leakage current in very large chips may overwhelm the abnormal current due to a fault in one gate

I_{DDQ} Testing, Cont'd



Commercial Tool for DfT Insertion



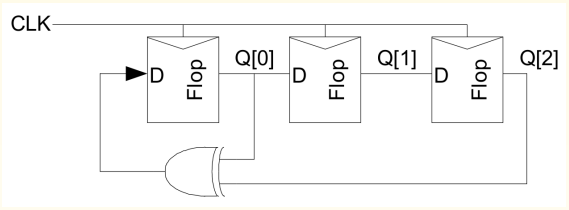
Source: Synopsys, Inc.

Built-In Self Test (BIST)

- Increasing circuit complexity, tester cost
- Interest in techniques which integrate some tester capabilities on the chip
  - Reduce tester costs
  - Test circuits at speed (more thoroughly)
- Approach:
  - Compress test responses into **"signature"**
  - **Pseudo-random (or pseudo-exhaustive) pattern generator (PRG)** on the chip
- **Integrating pattern generation and response evaluation on chip – BIST**

Pseudo-Random Sequence Generator (PRSG)

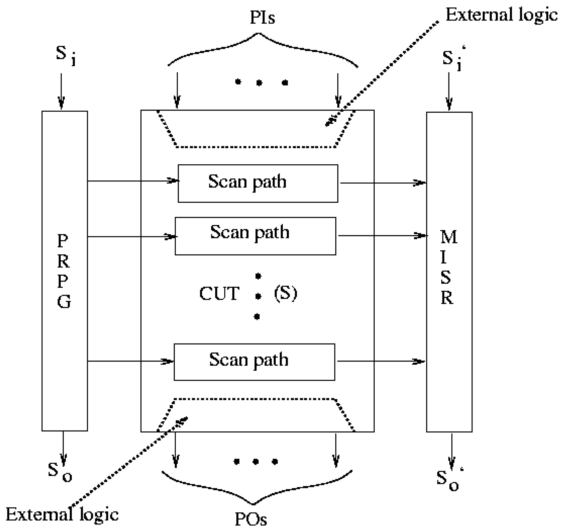
- Linear Feedback Shift Register
  - Shift register with input taken from XOR of state
  - Pseudo-Random Sequence Generator (or Pseudo-Random Pattern Generator (PRPG), or Linear Feedback Shift Register (LFSR))



Step	Q
0	111
1	110
2	101
3	010
4	100
5	001
6	011
7	111

(repeats)

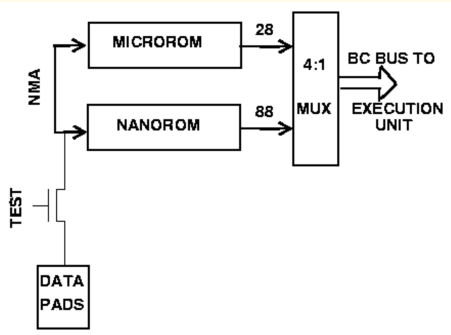
Example of BIST



Technique called STUMPS (from IBM)

Testability Techniques for 68020 ROMs

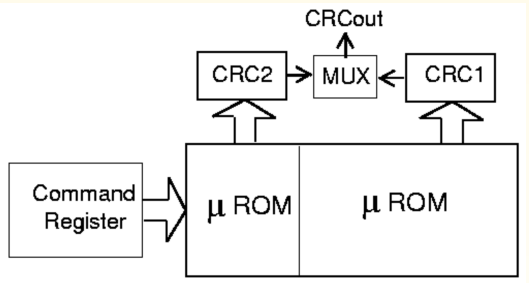
- Used test mode to force next microcode address (NMA) from data pins
- Data pins also control a MUX for both micro and nano ROM outputs, which are moved to the BC bus, into the data section of the execution unit, and to the address bus which can be observed



- Exhaustive testing of the 2K ROM entries
- 32 bits of ROM visible every 2 clocks
- Four passes of tests needed to read the 110 outputs of the two ROMs

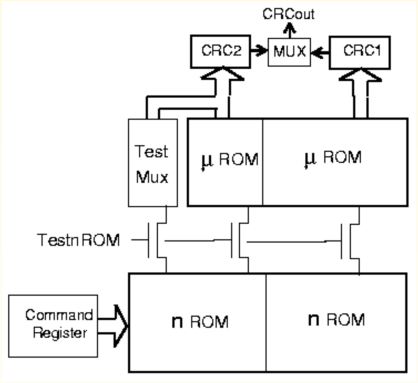
Microrom Test in MC68881 Floating Point Co-processor

- ROM physically split into two sections
- In test mode, ROM is addressed directly through the command register
- Exhaustive addresses fed to ROM in a "ping-pong" fashion (address/address complement)
- Outputs of ROM go to two 16-bit signature registers (using CCITT-16 polynomial $x^{15} + x^{12} + x^5 + 1$)
- Monitor both the quotient and final signature serially on a test pin (Probability of aliasing $2^{-(n+m-1)}$)



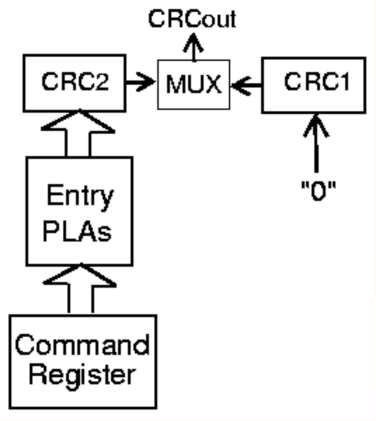
MC68881 NanoROM Test

- NanoROM physically located between the microROM and the execution unit (ECU), and outputs fed to ECU
  - No functional path for nanoROM to access signature registers
- In test mode, nanoROM columns coupled with the microROM columns (with additional columns of nanoROM multiplexed to signature register)



MC68881 Entry PLA Test

- Four entry PLAs, A0, A1, A2 and A3
  - A0 PLA contains the entry point reset vectors and is completely tested functionally
  - A1–A3 tested like the ROMs

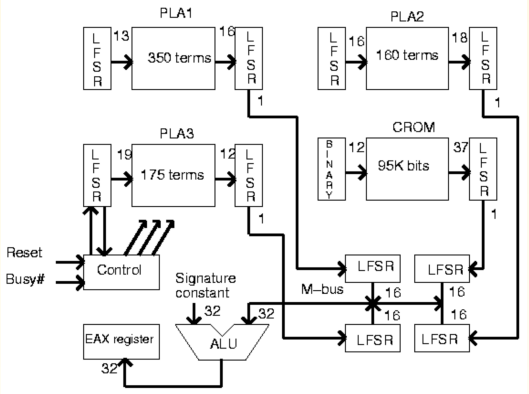


- Command register generates patterns and outputs are routed through a bus to the signature register
- Test patterns generated using PLA test generator

PLA	Inputs	Outputs	Products
A1	6	10	23
A2	13	10	133
A3	5	10	28
Response	25	25	56

Built-in Self Test in the Intel 80386

- Normal PLA inputs disabled during test and output of pseudorandom generator provides exhaustive set of tests to AND-plane input
- CROM tested with binary counter (exhaustive test)
- Responses compressed using multiple-input signature registers

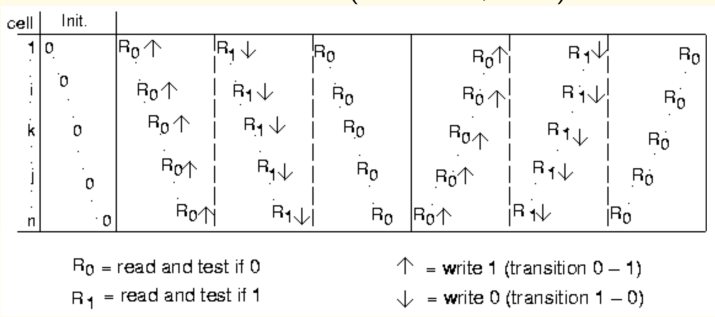


- Test transistors: 2.7%
- Area overhead for BIST: 1.8%
- Transistor sites tested with BIST: 52.5%
- Area tested: 18.6%

Testing Cache Memory Arrays in MC68030

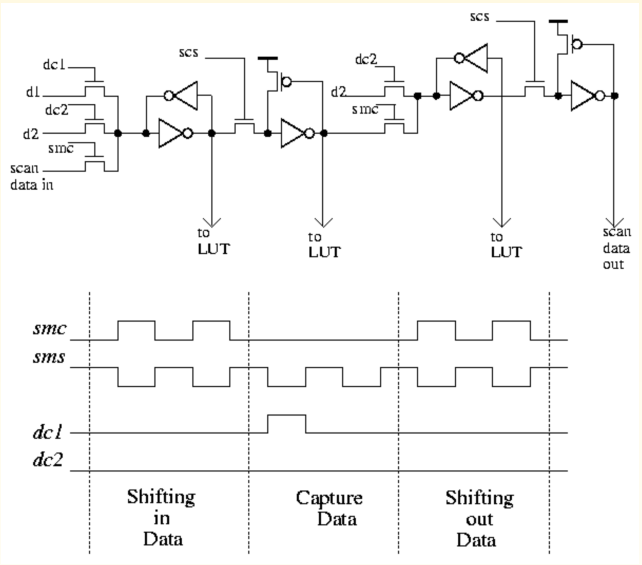
- Cache cell layout design is resistant to both bridging defects and capacitive coupling
  - Most likely bridging defect is between adjacent metal bit lines
- Memory fault model:
  - One or more cells stuck at 0 or 1
  - Coupling between cells

11n March Test (Marinescu, 1982)



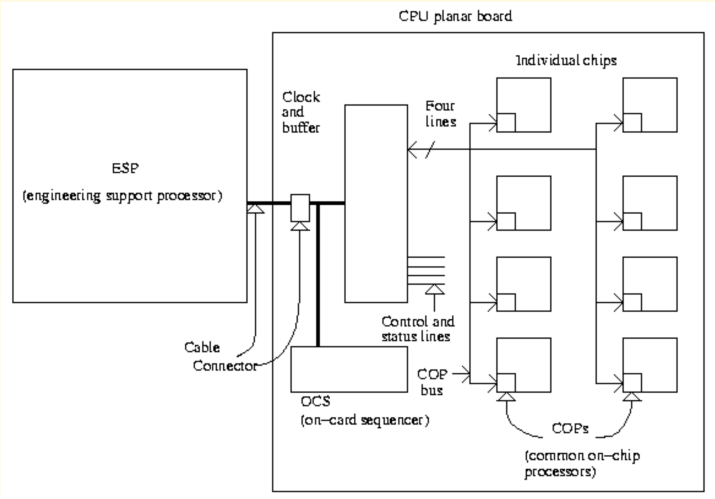
Refresh and data retention tests for the dynamic memory cells

MC68040 Scan Chain and Timing

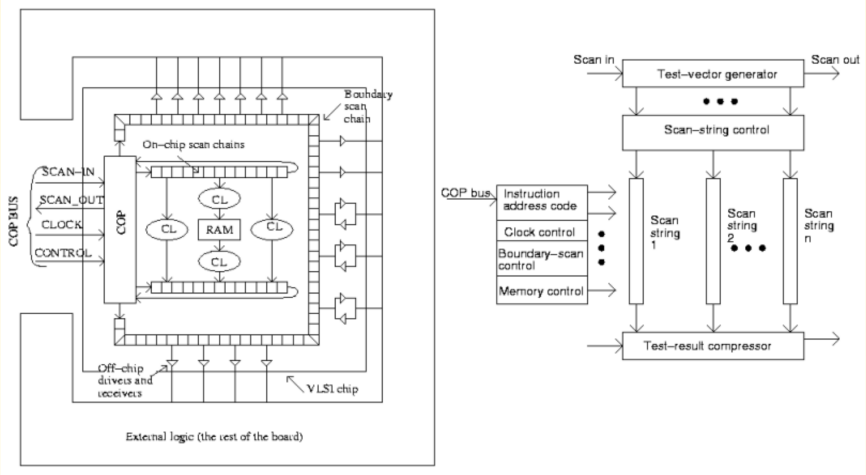


BIST in IBM Risc System/6000

LSSD with pseudo-random BIST



Common on-chip Processor (COP) in IBM RS/6000



Hardware for pseudo-random vector generation and result compression (31-bit LFSR)

BIST in IBM/Motorola Power-PC

- Variety of test techniques applied to the Power-PC 603
  - Full LSSD test of logic
  - BIST of "large" embedded RAMS
  - Functional test of small RAMS
  - I_{DDQ} tests
- BIST for cache and tag RAMs
  - Functional vectors (good for data cache, not instruction cache) and random BIST (size, complexity, test coverage) not applicable
- Use modified march test of Dekker (1988)
- $\log_2 n$ pattern for data
- Overhead: BIST is 2.9% of RAM array, 0.58% of total chip
- Performance impact: less than 100 pS due to extra MUX input leg
- All four RAMs tested in parallel, 2.5 mS at 80 MHz

Issues with Built-In Self Test

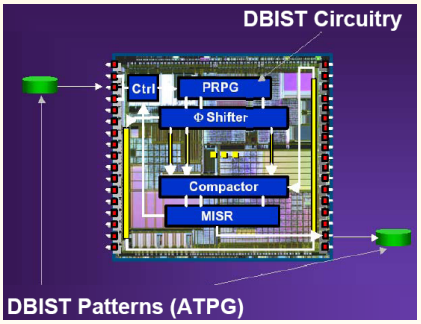
Technique can run test sequences at operating frequencies and capture results within the chip

- Pseudorandom pattern generators, signature analyzers

Can also use weighted random patterns or deterministic patterns

Example (Synopsys): Deterministic Logic BIST

Problems:
Hardware overhead
Test power
Non-functional modes during test



Industry Issues with Processor Testing

Concern with detecting real defects

- Small delay defects due to process variations, power droops and capacitive coupling
- Cause a shift in the speed of the part

Problems with logic BIST (same issues with scan AC tests)

- Overheads on chip
- False paths tested
- Test operating conditions different from normal operating modes

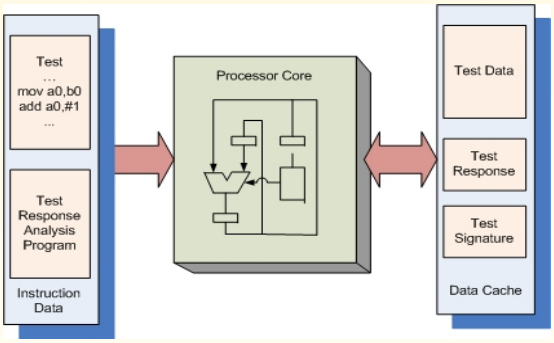
Software-Based (Native-Mode) Self Test for Processors

- Why not use functional capabilities of processors to replace BIST hardware?
  - No additional hardware
- Reduce test costs by using low-cost testers
- Increase coverage of delay defects and increase yield by testing native
- No issues with excessive power consumption during test

Developed at University of Texas (Int'l Test Conference 1998)

Application to processors at Intel (Int'l Test Conference 2002)

Software Based Self Test



Advantages

- ✓ Minimized DFT circuitry
- ✓ Reduced external tester performance
- ✓ Excessive test power and over-testing eliminated

Intel Functional BIST

Functional Random Instruction Testing at Speed (FRITS) applied to Itanium processor family and Pentium 4 line (ITC 2002)

Tests (kernels) are instruction sequences

- Kernels loaded into cache and executed in real time during test application
- They generate and execute pseudo-random or directed sequences of machine code

On Pentium-4, FRITS

- added 5% unique coverage to manual tests
- screened 10% – 15% of chips which passed wafer sort/package tests, but failed system tests
- enabled low-cost testers: 40% increase in defect screening on structural tester
- Kernels execute 20 loops in $\approx$ 8 mSecs

Are Random Tests Sufficient?

Intel implementation involved code in the cache which generated random instruction sequences

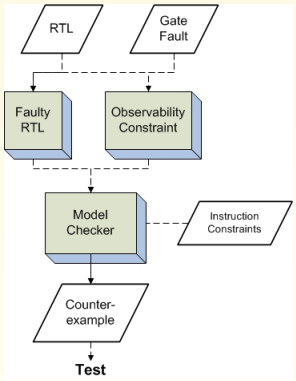
Interest in generating instructions targeting faults

Possible to generate instruction sequences which will test for an internal stuck-at fault in a module

In order to deal with defects in DSM technologies, **need to target small delay defects**

Automatically generate instruction sequences which will target small delay defects in an internal module

RT Level Test Generation for Hard-to Detect Faults



Overview

- Map gate level stuck-at fault to RTL
- Capture the propagation constraints as an LTL property
- Generate a witness for the LTL property using Bounded Model Checking
- All required constraints available in RTL
- Use SMT based Bounded Model Checking
- Scaling with cone-of-influence reduction

RTL Test Generation for Hard-to-Detect Faults

Experimental Setup

- OR1200 RISC processor was DUT
- EBMC Model checker / Boolector SMT solver
- Bound of pipleine depth + 1
- Focused on hard to detect faults in control logic
- Commercial ATPG to seive out easy to detect stuck-at faults
- 78% Fault coverage by commercial ATPG

RT Level Test Generation for Hard-to Detect Faults

Experimental Results using SMT Solver

Module	ATPG FC(%)	Flts.	SAT based method			Naive Observability Method		
			FC(%)	# TO	T(sec)	FC(%)	# TO	T(sec)
if	80.35	328	84.11	310	96.18	88.49	161	95.13
ctrl	63.21	832	65.97	817	83.12	97.15	59	69.72
oprmux	73.66	378	76.09	354	95.49	98.26	6	57.46
sprs	89.59	393	90.85	381	93.71	93.78	57	90.27
freeze	82.94	17	99.14	2	64.41	100	0	43.51
rf	78.59	7444	80.50	7268	97.57	90.21	463	69.83
except	72.69	1263	73.48	1209	98.63	92.79	128	96.19
Overall	78.05	10655	79.17	10343	96.23	93.86	874	76.11

FC(%) : Fault Coverage in %
 # TO : # of Timed Out faults
T(sec) : Average Time for generating a test for a fault in seconds

Experimental Results, Structural Observability

Module	FC(%)	# TO	T(sec)
if	98.17	25	23.14
ctrl	99.21	8	21.16
oprmuxes	100	0	19.33
sprs	97.53	12	18.39
freeze	100	0	10.48
rf	98.37	172	22.85
except	97.63	69	38.14
Overall	98.87	454	24.23

FC(%) : Fault Coverage in %
 # Faults : # of Undetected Collapsed Faults
 # TO : # of Timed Out faults
T(sec) : Average Time for generating a test for a fault in seconds

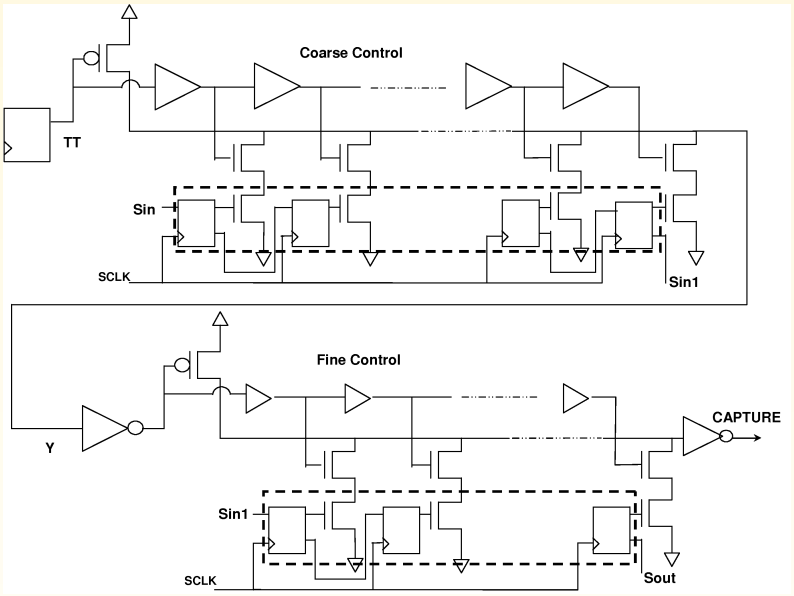
Summary of Results

- Functional fault coverage of ≈99% for OR1200 processor
- SMT based approach was 4x faster than SAT

Detecting Delay Defects After Manufacturing

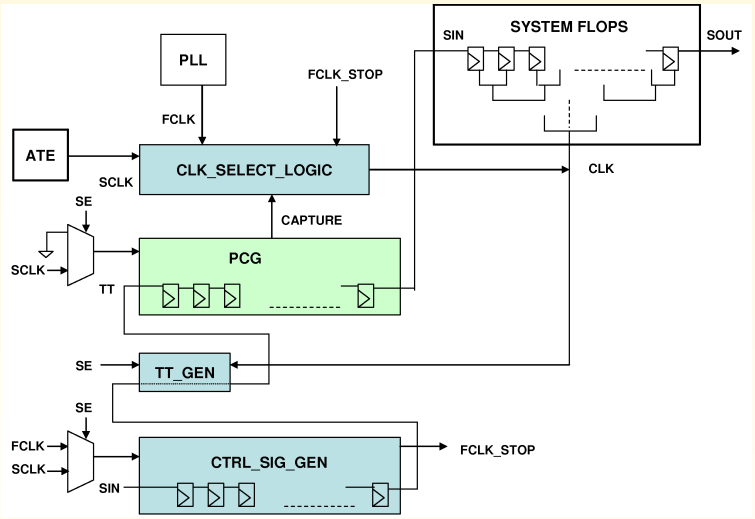
- Necessary to detect "small-delay defects"
- Delay defects which don't affect performance soon after manufacture could be reliability hazards
- Need Path Delay Tests to ensure that defective parts are screened out
  - Difficult to apply two-pattern tests in scan mode
  - Requires precise control of clocks for high-speed circuits
  - If capture clocks are based on the system clock, there is no information on the slack for the path
- **Solution: On-chip programmable capture mechanism**
  - Ability to capture faster than at-speed

Delay Lines for On-Chip Programmable Capture



Source: R. Tayade, *et al.*

System for On-Chip Programmable Capture



Source: R. Tayade, *et al.*

ECE Department, University of Texas at Austin | Lecture 20. Design for Testability | Jacob Abraham, November 5, 2020 38 / 38