Manufacturing & System-Level Use of BIST - Organization

- Manufacturing Use
  - Collapsed Vectors & ATE
  - Burn-In Testing & Other Issues

- System-Level Use
  - Requirements
    - Input Isolation & Test Controller
    - Design Verification
  - Effective Fault Coverage & Diagnosis
  - Multiple BIST Sessions

- General BIST Architectures
- BIST Design Process
- Clock Enable Problem
Manufacturing Testing

- Collapsed vectors for Automatic Test Equipment (ATE)
  - ATE have limited vector storage for high speed inputs
    - Many clock cycles can be described in 1 vector
      - Other inputs must be held constant
      - Outputs cannot be monitored
  - Once BIST is initiated, only clock cycles applied
    - 100K transistor chip tested in only 300 vectors with BIST
      - 32,000 clock cycles, most for BIST sequence
  - BIST sequence consists of
    - Short vector sequence to initiate BIST
    - 1 vector for clock cycle burst
    - Short vector sequence to retrieve results
Manufacturing Testing

- For long BIST sequences
  - Intermediate signatures for early detection of many faulty chips
    - Reduces fault simulation time
    - Reduces test time and cost
    - Fault profiling helps determine optimal points
  - Intermediate signatures allow changing input values
    - Improves fault coverage during manufacturing testing
    - Controlled values to input isolation circuitry
    - If BIST used only for manufacturing test, no input isolation circuitry is needed

- Low-cost ATE
  - Low-speed interface
  - High-speed I/O

FIGURE 6.1 Using low-speed ATE for high-speed testing with BIST.
Manufacturing Testing

- Burn-in testing - stress testing to detect infant mortality
  - BIST produces high data activity
    - Good for burn-in test
    - Stress due to power dissipation from data activity along with elevated voltage and temperature
  - Design TPG to continue running after BIST sequence is complete
    - Make the BIST initialization and initiation simple
      - BIST results typically don’t need to be read during burn-in
        - Complete testing of device performed after burn-in
      - Be careful to avoid exceeding package thermal consideration
- Manufacturing testing under control of ATE reduces BIST considerations when compared to system-level use
System-Level Testing

- Requirements for system-level use of BIST
  - Ability activate/deactivate BIST from system software
  - Ability to retrieve BIST results from system software
  - Isolation of system inputs to the CUT
    - Isolation of data between asynchronous timing regions in CUT
  - Complete initialization of BIST circuitry and CUT
  - Ability to control fixed-length output response compaction (for non-comparison-based ORAs)
    - Hold BIST results until read by system software
  - Ability to prevent CUT outputs form causing undesirable effects during BIST sequence
    - Example, interrupts that cannot be masked
Purpose of Test Controller in BIST

- Isolate system data from CUT (typically via MUXs)
  - ensures reproducible test results at system level testing
- Initialize CUT, TPG, ORA
  - seed registers (control registers that are static during BIST)
  - initialization sequence length = sequential depth of CUT
    - assuming no global reset/presets
- Control length of BIST sequence
  - keep track of the number of test patterns
  - ensures reproducible test results at system level testing
- Hold BIST results until read by system
  - ensures reproducible test results at system level testing
Input Isolation

- Necessary to prevent reproducible results from unknown system data
- Performance penalty
  - Additional set-up time at primary inputs
  - Additional delay between asynchronous timing regions
- Types of input isolation
  - Blocking gates
    - OR forces logic 1 to CUT, AND forces logic 0
  - Multiplexers
    - Can supply test vectors from TPG
Design Verification of BIST

- Begin logic simulation with un-initialized circuit
  - BIST and CUT
- Control absolute minimum number of inputs to initiate BIST
- Apply undefined logic values ($U$s) to all other inputs
- Initiate BIST sequence as early in simulation as possible
  - To avoid inadvertent initialization
  - Initiate in as system-like manner as possible
- Monitor ORA outputs during simulation to look for $U$s
  - Improper BIST and/or CUT initialization
  - Improper input isolation
- Run simulation well past end of BIST sequence
  - Ensure that BIST results are held in ORA
Reuse of System-Resources

- Multiple BIST sessions typically required when reusing
  - Control registers for TPGs
  - Status registers for ORAs
  - Need to apply different control values during each BIST session
- Often provides processor interface for accessing BIST
- Reduces area overhead and performance penalties
Centralized vs. Separate BIST Architectures

Centralized BIST

- Reduces TPG & ORA area overhead
  - Good for multiple identical CUTs
- Requires distribution system (denoted $D$ below)

Separate BIST

- Good for mix of regular structures & general logic
Test-per-Clock vs. Test-per-Scan BIST

Test-per-Clock
- Vector applied each clock cycle
  - shorter test time
  - at-speed test possible
  - higher fault & defect coverage
  - larger area overhead

Test-per-Scan
- Vector applied each scan cycle
  - simple control logic
  - separate resources may be shared
  - longer test time
BIST Interface to Boundary Scan

BIST Resource Interface Controller (BRIC)

- Developed by *Bell Labs*
- Mapped as a 2-bit Boundary Scan user test register
- Uses Boundary Scan to:
  - Activate the BIST test
  - Report the test results
- Uniform interface for macrocells
  - BIST for regular structures
    - RAMs, ROMs, FIFOs, etc.
  - BIST – BIST start
  - BFC – BIST Flag Check
  - BC – BIST Complete
  - BF – BIST Flag

C. Stroud 9/09
Manufacturing & System-Level Use
BIST Design Process

- Analyze the logic and partition the circuit
  - Unstructured (“Random”) logic
  - Regular structures (RAM, ROM, Register file, FIFO, etc.)
- Select sub-circuit BIST techniques
  - Select TPG type (LFSR, counter, or FSM)
  - Select ORA type (comparison or compaction based)
    - Reduce area overhead by using existing registers for TPG & ORA
- Select architectural features for BIST elements
  - Embedded within the CUT or separate
  - Centralized (shared) or distributed
- Design BIST controller
- Maybe integrate with boundary-scan
  - BIST controlled from the boundary-scan bus
Factors Influencing BIST Architecture Selection

- Test application time
  - Degree of test parallelism
- Fault coverage
- Cost
  - Area overhead
  - Additional design time
- Performance degradation
- Complexity of replaceable (repairable) units
- Packaging & power dissipation
- Repair strategy
  - Both factory & field
- BIST hierarchy
BIST Summary

**Advantages**

- Short test application time & at-speed testing
- Vertical testability – same for all levels of testing: device-system
  - May be applied hierarchically from device through system
- High fault & defect coverage
- Provides embedded ATE ⇒ reduces need for expensive external ATE
- Reduced system diagnostic test development
  - 50% of BIST applications at *Bell Labs* initiated by diagnosticians
- Growing support by CAD vendors
BIST Summary (cont.)

**Disadvantages**

- Area overhead typically 10-25%
  - all costs of full scan in many BIST approaches plus more
    - Extra logic for TPG & ORA
    - BIST controller
- Fault simulation may be lengthy
  - But necessary to determine fault coverage
- Additional risk to project
  - Must design BIST & system function
    - Longer design time if no CAD automation available
    - Both must work for chip to ship
The Clock Enable Problem

- Clock enables are commonly used
  - Single-clock, single-edge, synchronous designs
  - Allow larger propagation delay in combinational logic
    - FFs A&B get new data every clock cycle
    - FFs C&D get new data every N clock cycles
- If pseudo-random patterns are applied to clock enable
  - clocking at system frequency causes good chips to fail
  - clocking at reduced clock rate will not test at-speed