Memory Built-In Self-Test (MBIST): Advanced Techniques for SoC Design and Verification

ASPDAC 2025

Prashant Seetharaman
Siemens EDA

Agenda

1. Introduction to memory testing
2. MBIST fundamentals
3. MBIST Integration in SoC Design flow
4. Advanced MBIST techniques
5. Emerging trends and future directions
6. Q&A and Discussion

Introduction on memory testing

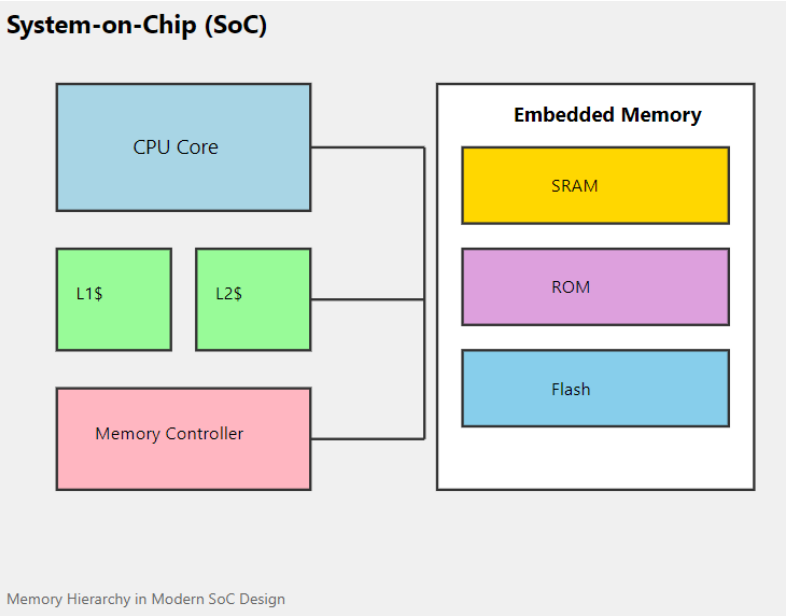
Memory structure and operation

Important to understand the main features of embedded memories that have an impact on their testability and design of MBIST logic.

SRAM is most common case of embedded memory

Two variations include:

- single port and multi port memory.

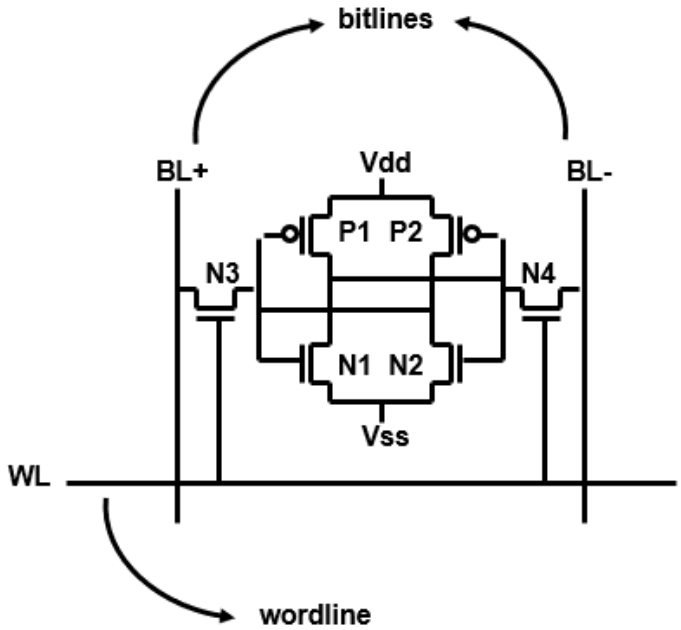


Memory Hierarchy in Modern SoC Design

Key takeaway: Performance considerations b/w access patterns, timing requirements and power optimization

RAM bit cell structure

- RAM cell composed of 6 transistors
- 4 transistors are connected as cross- coupled inverters to form a latch holding the cell value
  - P-channels P1 and P2 are weak as they are only used to stabilize the cell
  - N-channels N1 and N2 are strong as they are used to quickly discharge bitlines during read operations
- Both N- channel access transistors N3 and N4 are turned on when the wordline is active



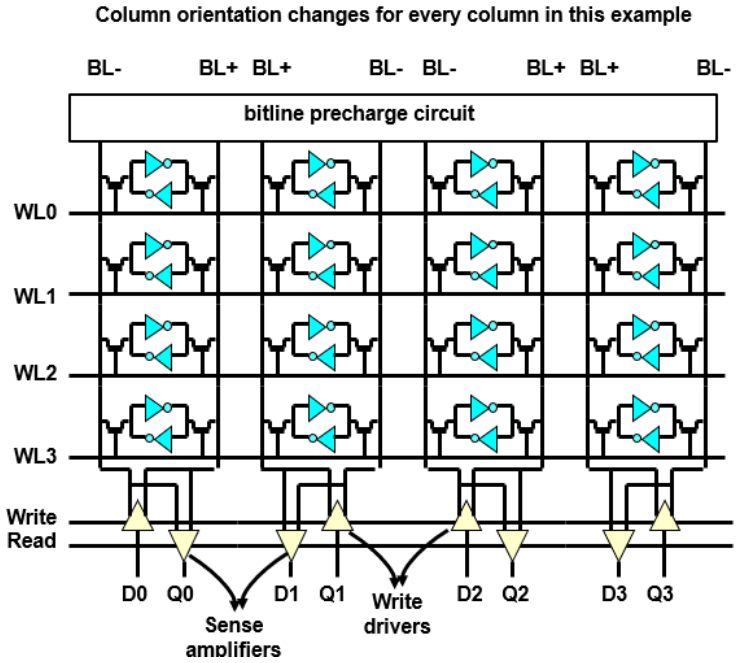
Source: "Digital Integrated Circuits: A Design Perspective" by Jan M. Rabaey et al

Key Takeaway:

By optimizing transistor strengths and ratios, designers achieve a reliable memory cell capable of high-speed operations with minimal noise and power overhead

Array of bit cells

- Data sent to and read from bit cells using differential signaling over bitlines for higher performance
  - Data=1 if voltage of BL+ > BL−
  - Data=0 if voltage of BL+ < BL−
- Write operation
- Read operation
- Sequence of BL+ and BL− have an impact on test data patterns to apply

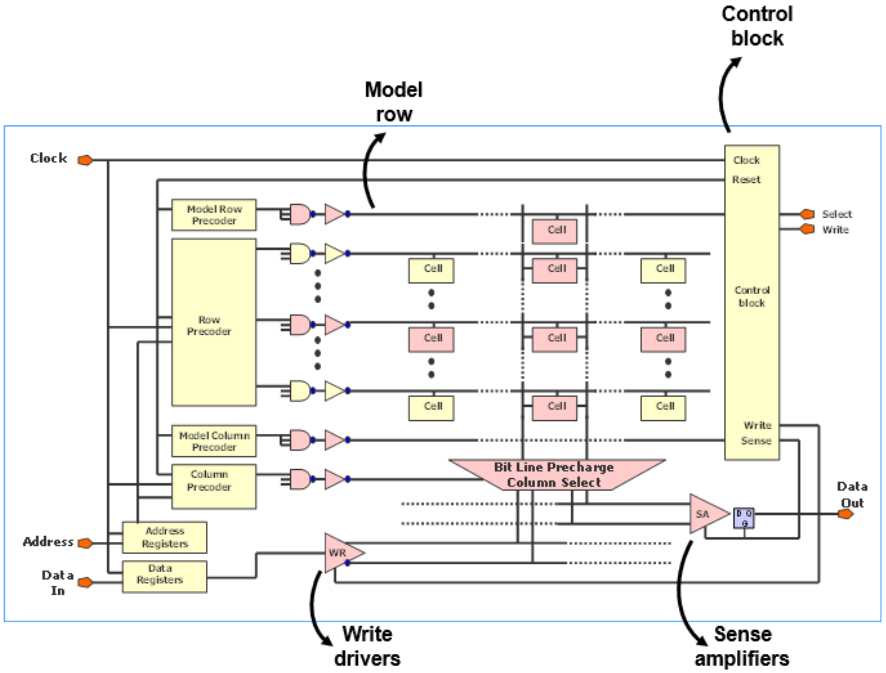


Source: "Digital Integrated Circuits: A Design Perspective" by Jan M. Rabaey et al

Key takeaway: Through techniques like differential signaling, precise precharge, and robust sensing, this design ensures high performance in data storage and retrieval

Complete Random Access Memory (RAM) structure

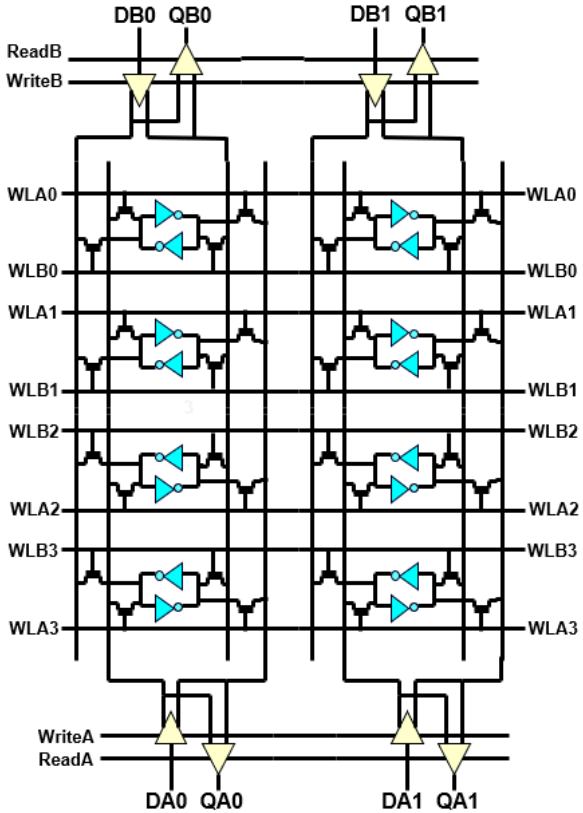
- RAM cells are organized into rows and columns
- Address decoders
- IO circuits
- Control block



Source: Memory Testing and Built-In Self-Test" in "CMOS VLSI Design: A Circuits and Systems Perspective" by Neil H.E. Weste and David Harris

Multi-port memories

- Basic RAM is single port with read and write capability → 1RW
  - Address input, data input and data output
- Multi-port RAM have several ports, each having an address input, an optional data input and an optional data output
  - Each port can have its own clock
- Typical cases generated by memory compilers are 1R1W and 2RW
  - Two sets of wordlines and bitlines
  - 2RW case illustrated
- Number of possible combinations is large
  - xRyWzRW where x, y and z are integers, including 0
  - Several applications (switches, routers, etc…)



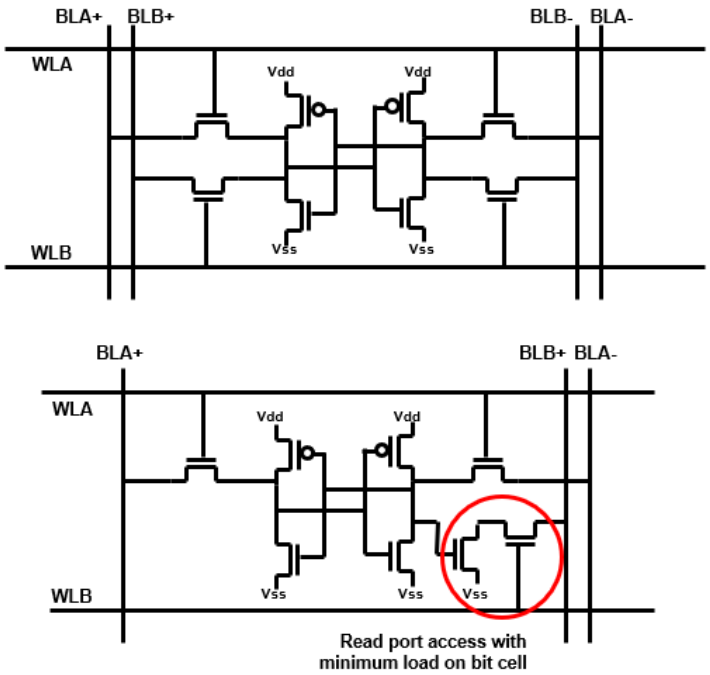
R=Read-only port
W=Write-only port
RW=Read/Write port

Key takeaway: Multi-port memories provide the foundation for parallel data processing, enabling simultaneous and independent operations across multiple ports.

Source: "Digital Integrated Circuits: A Design Perspective" by Jan M. Rabaey et al

Bit cells for multi-port memories

- Bit cells are similar to those for single-port
- Design takes into account that cell can be accessed by more than one port at the same time
  - Access can be direct or indirect
  - Direct if row and column address is the same on both ports
  - Indirect if row address is the same but not column address (more frequent)
  - Implication on test patterns to be generated
- Read-only ports can use a different structure to reduce the load on the bit cell and eliminate the possibility of disturbing its content
  - Especially for large number of read ports



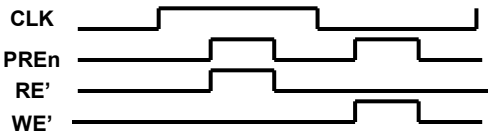
Source: "Digital Integrated Circuits: A Design Perspective" by Jan M. Rabaey et al

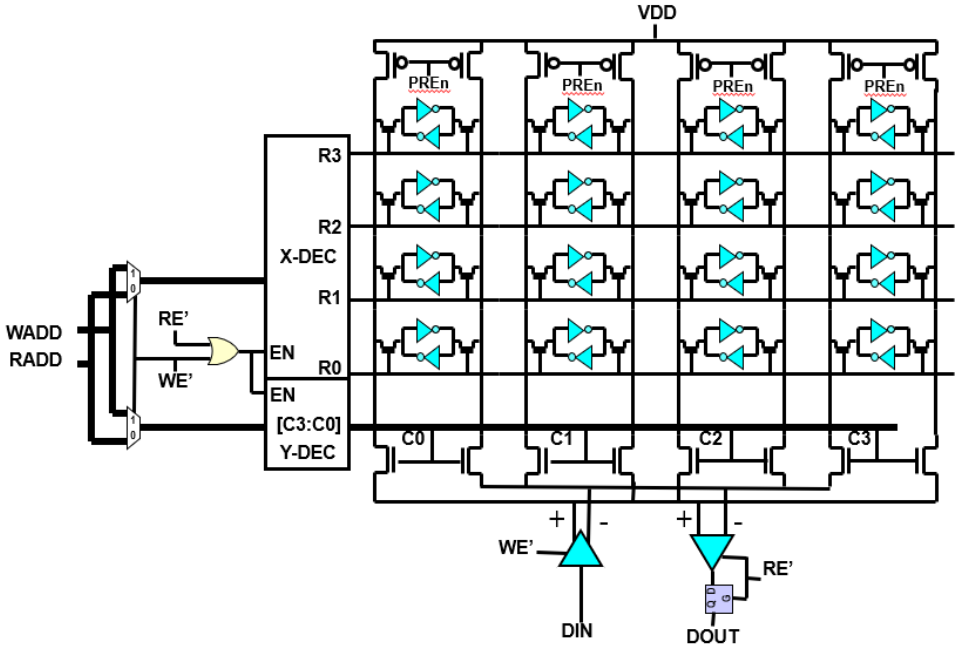
Key takeaway: Optimizations such as single-ended read ports ensure these cells meet the demands of modern high-performance systems while minimizing overhead

Pseudo 2 port memory

Multi-port memory emulation using a single-port memory
- Smaller area
- More complex timing

Two operations performed sequentially within a clock cycle
- Read then write
- 1R1W case shown
- 2RW is similar





Source: "Digital Integrated Circuits: A Design Perspective" by Jan M. Rabaey et al

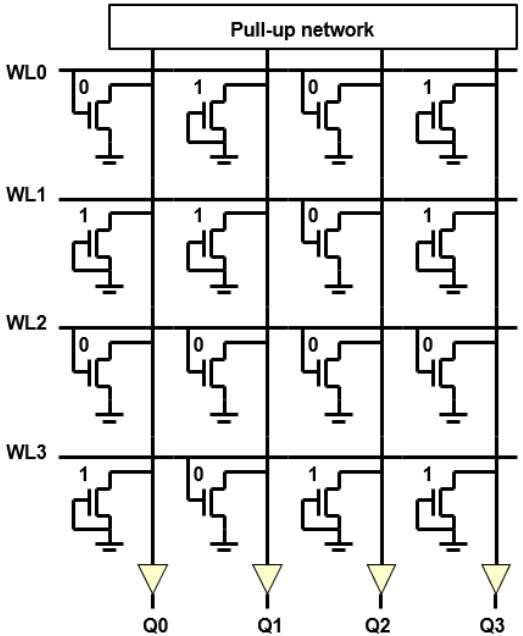
Key takeaway: By emulating multi-port behavior in a single-port design, it strikes a balance between performance, area efficiency, and timing complexity, making it suitable for power-sensitive applications

Read only memory (ROM) structure

ROM structure similar to RAM
- Organized into rows and columns
- Banks and column multiplexing (optional)
- Physical address and data mapping (optional)
- Multiple read ports (optional)

Content is determined at manufacturing time
- Value can be changed late in the manufacturing process if controlled by metal layer

Bit cell is much simpler
- 1 transistor
- 1 bitline



Source: "Digital Integrated Circuits: A Design Perspective" by Jan M. Rabaey et al

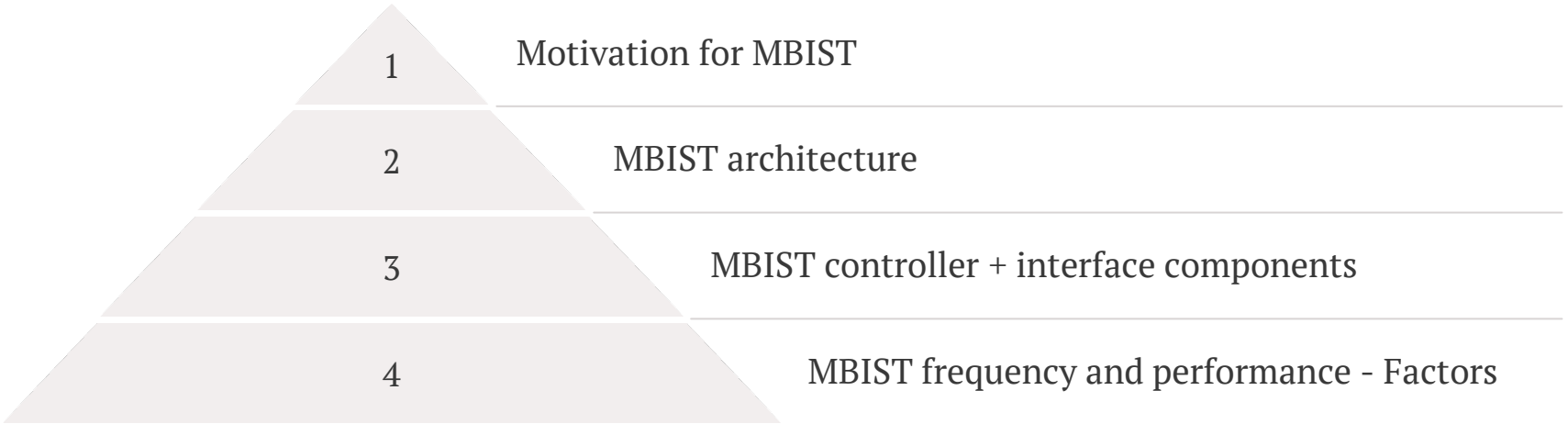
Key takeaway: Its minimalistic design—featuring just one transistor per bit—ensures low area and power consumption, perfectly suited for static data applications

Memory structure and operation summary

- Single and multi-port SRAMs are the most common embedded memories
- All memories are composed of bit cells arranged in rows and columns
- Bit cell structure is specific to each memory type
- Column multiplexing, physical address mapping and physical data mapping can be used to optimize the memory layout, performance and power but they also affect testability
- Architecture of multi-port memories becomes complex due to a mix of R/W/RW ports and time-multiplexed operations
- ROMs have a single bit cell, but their content can be changed late during the manufacturing process

MBIST Fundamentals

MBIST Fundamentals



Motivation for MBIST

Access

- MBIST controllers physically close to memories to minimize length of connections
- Low speed serial link used to control MBIST controllers from tester
- Physically impossible to access hundreds or even thousands of memories from pins or embedded CPUs

Timing

- Physical proximity also allows application of at-speed tests
- Not practical to apply tests in the GHz frequency range from tester

Test time

- MBIST controllers can test tens of memories in parallel
- Something not possible with a tester or a CPU based approach

MBIST Architecture

Consists of 3 design objects

BIST Access Port (BAP)

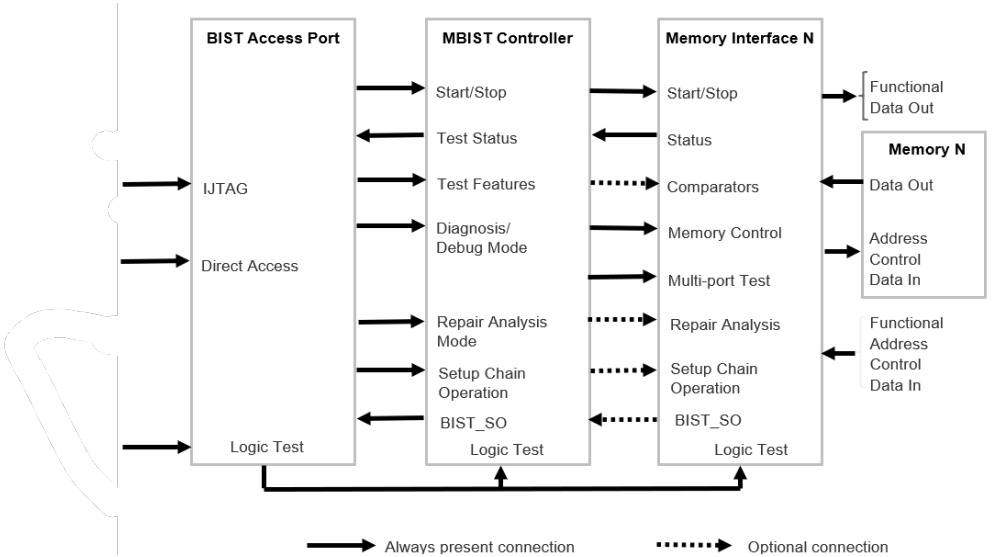
- Provides interface to low-speed serial bus (IJTAG)
- Typically serves several controllers

Memory BIST Controller

- Implements test algorithms
- Typically serves several memories

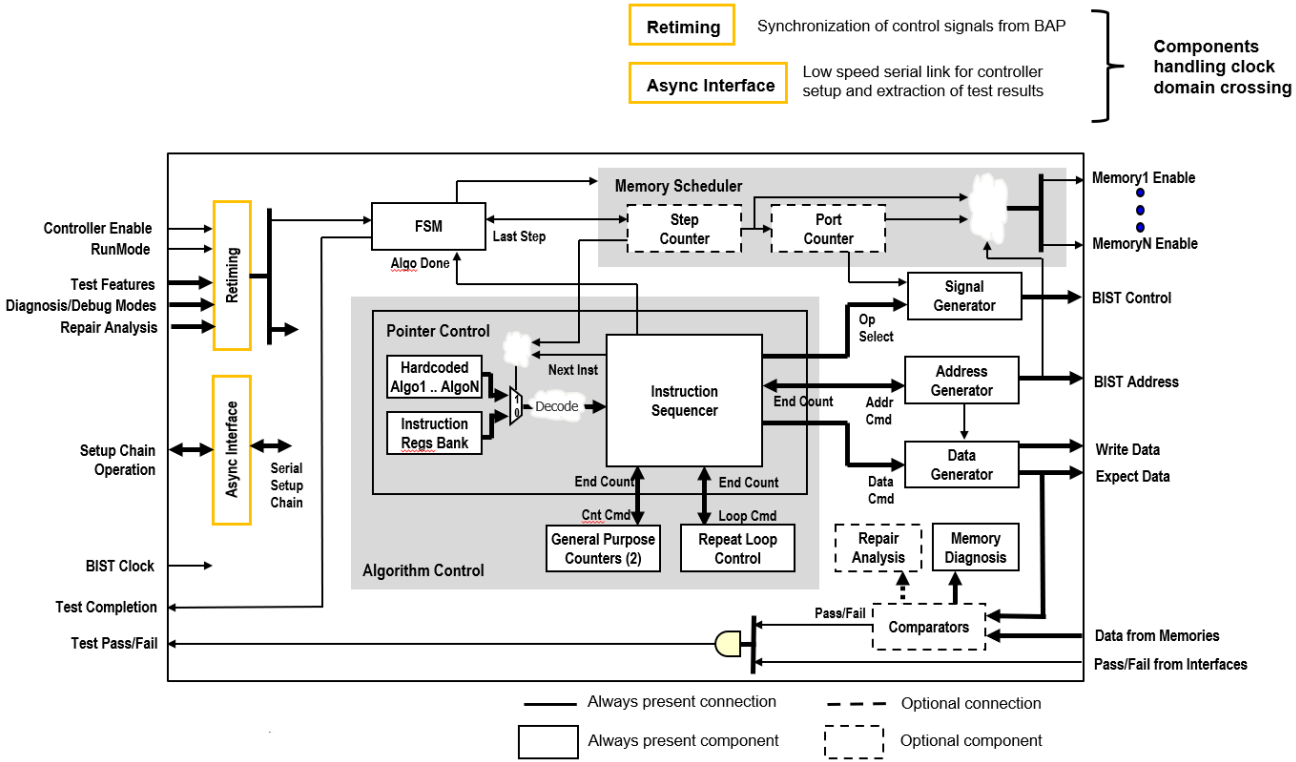
Memory Interface

- Intercepts functional memory inputs for applying algorithm
- Unique to each memory



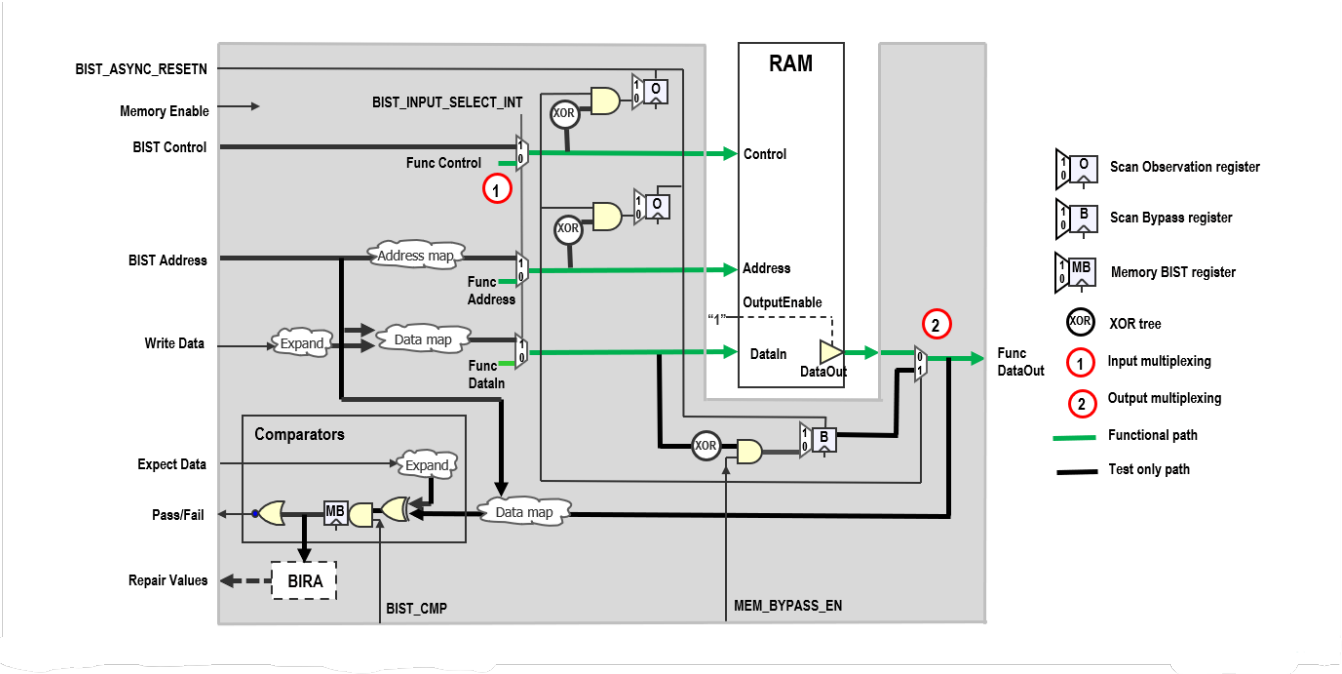
Source: Memory Testing and Built-In Self-Test" in "CMOS VLSI Design: A Circuits and Systems Perspective" by Neil H.E. Weste and David Harris

MBIST Controller components



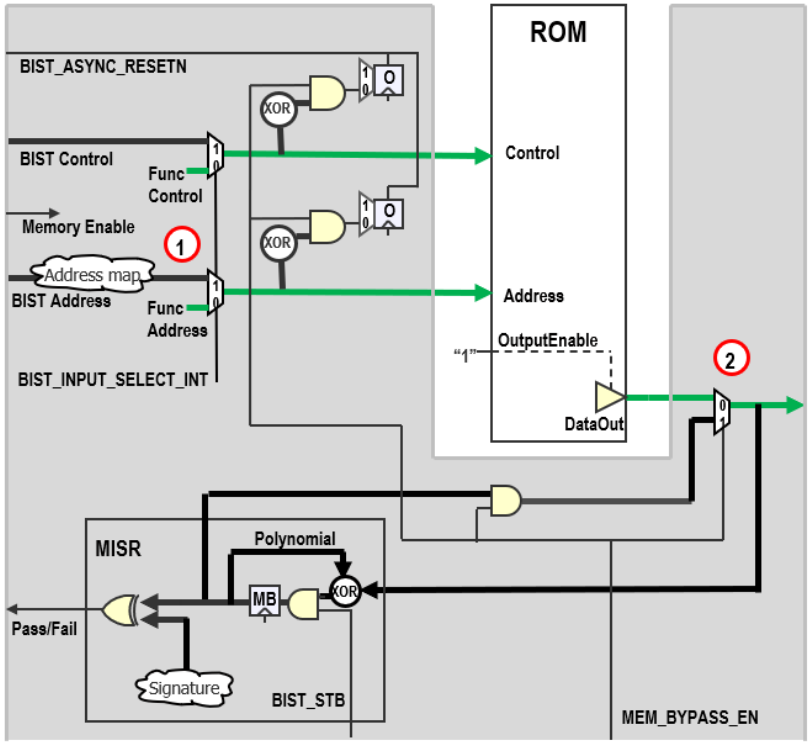
Source: "Memory Testing and Design for Test" by M. Bushnell and V. Agrawal

MBIST Interface components (RAM)



Source: "VLSI Test Principles and Architectures: Design for Testability" by Wang, Wu, and Wen

MBIST Interface components (ROM)



Source: "VLSI Test Principles and Architectures: Design for Testability" by Wang, Wu, and Wen

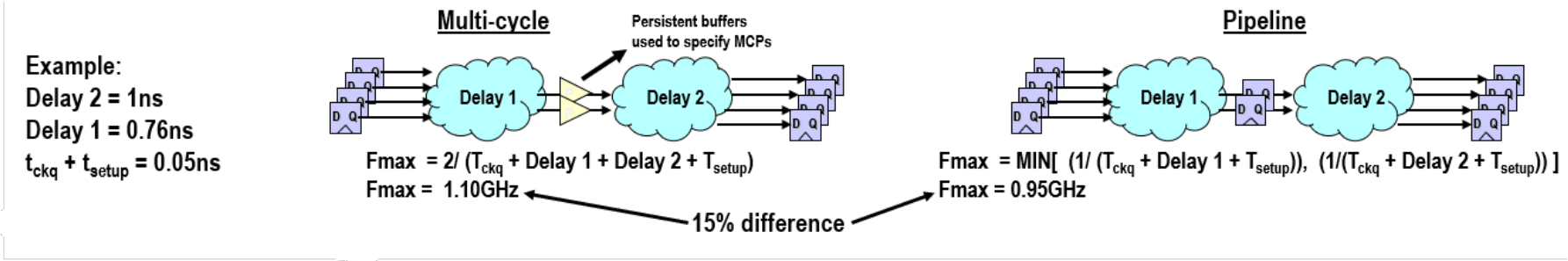
Factors influencing MBIST frequency

- Technology and design environment
  - Cell library
  - Operating conditions
  - Synthesis tool and options
- Memory configuration
  - Number of memories per controller
  - Range of address bus and data bus width
  - Complexity and/or differences in physical data and address map
  - Type and granularity of repair
  - Memory placement relative to controller
- Algorithm features
  - Number of algorithms and instructions per algorithm
  - Address register segmentation
  - Data pattern inversion based on address
  - Comparison of address registers

MBIST logic is typically implemented as “soft” IP synthesized together with the functional logic.
→Difficult to predict the maximum achievable frequency given the large number of factors affecting performance

How to improve MBIST performance

- MBIST controller logic can be very complex, and options must be provided to enable operation at very high frequency (GHz range)
- Pipelining is required for high fan-out signals or signals transmitted at a large distance from the controller potentially toggling at the maximum clock rate
- Multi-cycle operations provide an advantage over pipelining for parts of the logic with large combinational depth not toggling at the maximum clock rate
  - Not possible to balance logic depth before and after pipeline flops due to large number of configurations
  - Pipeline flops introduce additional delays



Source: "TMBIST User's Manual"

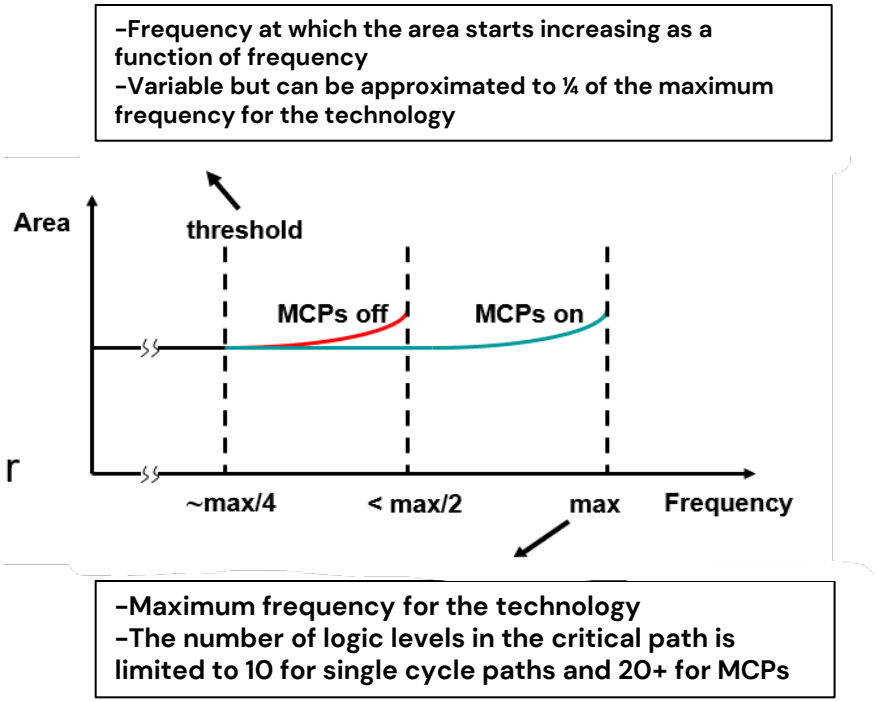
How to improve MBIST performance (contd.)

- Using multi-cycle paths (MCPs) is also useful to reduce area at high frequency

- Area increases when approaching the maximum frequency of the MBIST circuit

- Due to buffering of signals and resizing of cells necessary to meet timing

- MCPs allow to keep the area constant over a wider range of frequencies while improving the performance



MBIST summary

- Memory BIST is the only practical method of testing a large number of memories due to access, timing and test time considerations
- The MBIST logic consists of 3 main design objects: access port, controller and memory interface
- Interoperability of MBIST and ATPG scan is important
  - Coverage/diagnosis of MBIST logic itself
  - Controllability/observability of functional logic around the memory
- Use of pipelining and multi-cycle paths improves performance, but it is still difficult to predict the maximum frequency when implemented as “soft” IP

MBIST Integration in SoC Design flow

Design Integration

- Chip test architecture
- Test planning
- Test access for manufacturing test
- Test access for in-system test
- IP cores with pre-defined test bus interfaces
- Testing off-chip memories
  - PCB, 2.5D, 3D

Chip test architecture

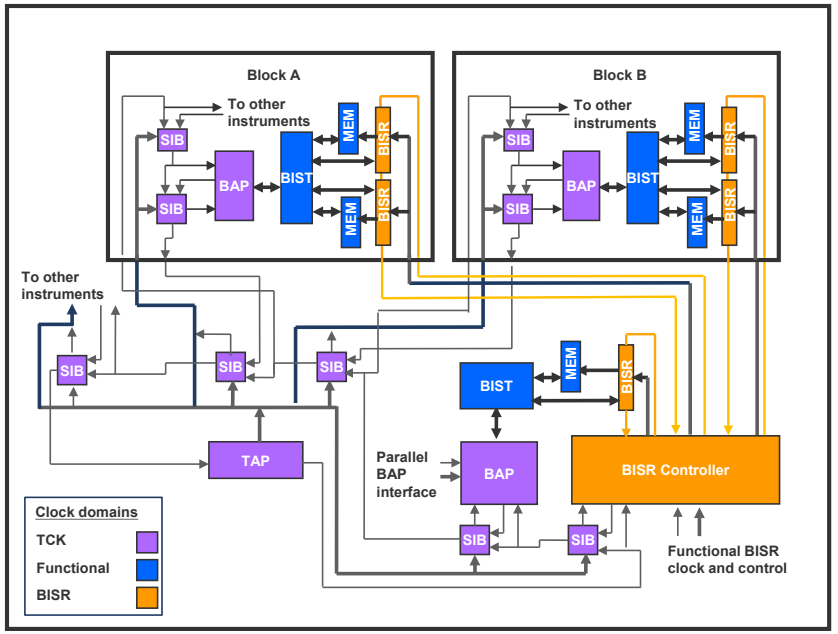
1. Large majority of designs are hierarchical, and block based
   - Divide-and-conquer approach beneficial to both design and test

2. IEEE 1687 is most appropriate to implement test architecture

- Easy integration of all test and non-test instruments
- Easy to include/exclude blocks during verification and actual test using segment insertion bit (SIB)

3. MBIST also coupled to Built-In Self-Repair (BISR) system

- BISR has additional functional clock and control IOs to provide a low latency interface for system power-up



Source: “TMBIST User’s Manual”

Key takeaway: A hierarchical, block-based test architecture with integrated BIST and BISR enhances scalability, reliability, and test efficiency, making it essential for modern SoC designs

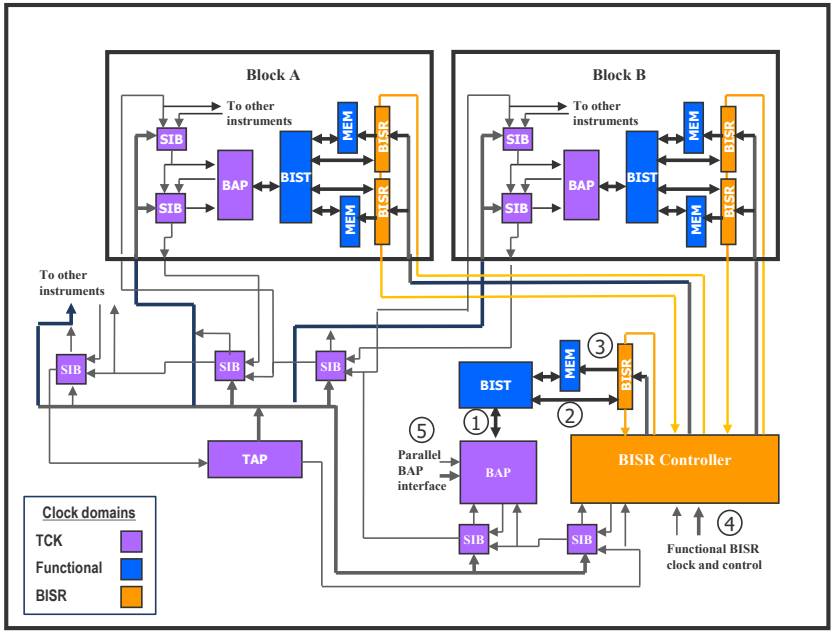
Memory BIST and BISR timing interfaces

Several cross-domain interfaces

1. BAP←→BIST
2. BISR registers ←→BIST
3. BISR registers → Memories
4. BISR controller ←→ Functional
5. BAP ←→ Functional (not shown)

Three methods used for cross-domain transfers

1. Low speed serial link between controllers
   i) 2-edge clocking with TCK
   ii) Asynchronous interface
2. Protocol based transfer
   i) Source register holds when destination register captures
3. Synchronizers
   i) Used for few control signals when other two methods not applicable



Source: "TMBIST User's Manual"

Test planning (controller assignment rules)

Several partitioning rules used to assign memories to a test controller (Cx rules) and determine memories that can be tested in parallel within a controller (Px rules)
- Partitioning done within a physical block/layout region

C1: memories of a same type (SRAM, ROM, DRAM,…)
Technically possible to use a same controller for different types but it adds complexity to both the hardware and test flow due to differences in algorithms used.

C2: memories of a same clock domain
Facilitates at-speed test and timing closure

C3: memories physically close to each other
Memory placement can be obtained from standard DEF (Design Exchange Format) file
User can control maximum distance between controller and memory

C4: memories part of a same power domain
Not all domains are guaranteed to be powered especially for system level test
Information can be obtained from standard UPF (Unified Power Format) or CPF (Common Power Format) file

Test planning (parallel group assignment rules)

Px rules determine memories that can be tested in parallel within a controller
- Even if all memories in a controller are of a same type, they might have different requirements

P1: Memories must use a common set of algorithms and operations
- E.g., slight difference in operation set for RAMs with synchronous vs asynchronous read port

P2: Memories must use the same pipelining options
Necessary for the data read from memories to arrive at the same time at the comparators
Otherwise, it complicates diagnosis significantly and makes repair sharing impossible

P3: Memories should not exceed the maximum average power allowed

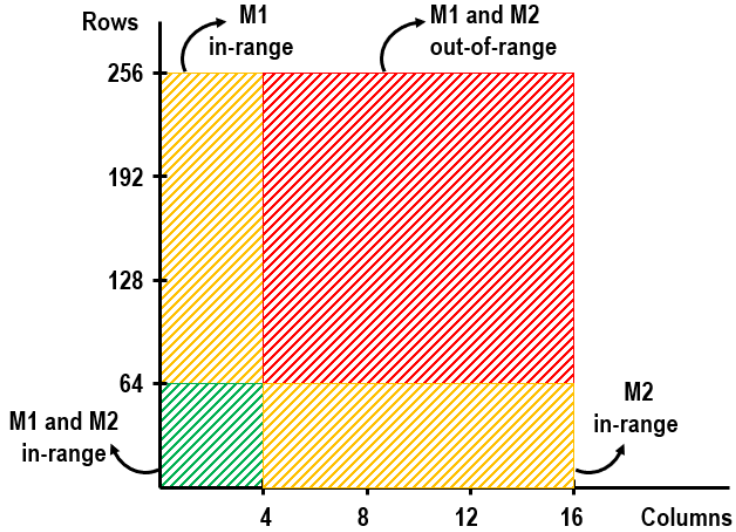
P4: Groups should be formed to minimize test time
- It is not always faster to test memories in parallel
- See example on next page

Optimizing parallel group assignments for test time

Memories for which both the number of rows and columns differ may end up being tested serially

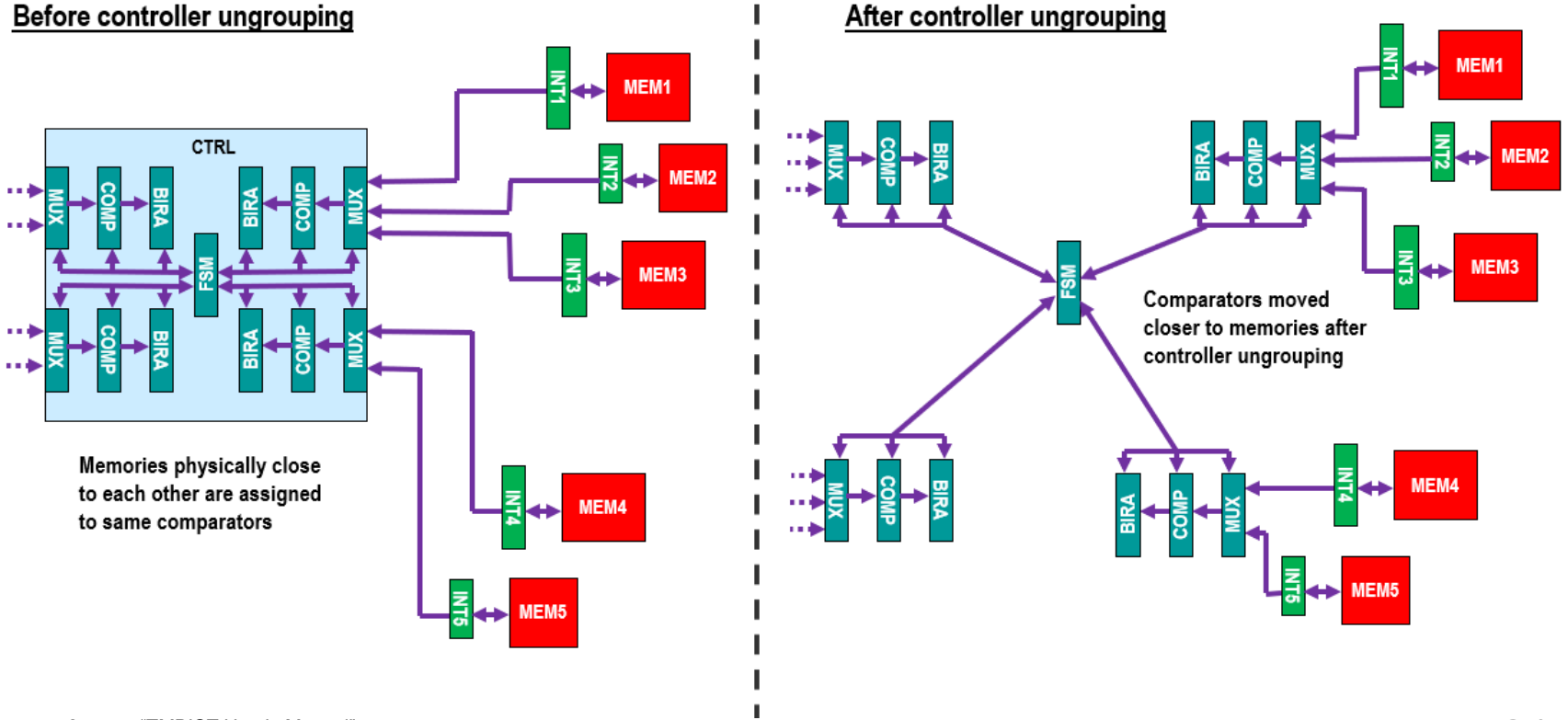
Example: two 1Kx16 memories
- M1: 256 rows x 4 columns
- M2: 64 rows x 16 columns
- Aspect ratio is roughly the same

Controller must use an address counter with 256 rows and 16 columns
- Address space multiplied by 4
- Memories operated out-of-range most of the time
- More advantageous to test memories separately
- Test time cut in half



Source: "TMBIST User's Manual"

Optimization of comparators assignment based on placement



Source: "TMBIST User's Manual"

Tiling architecture

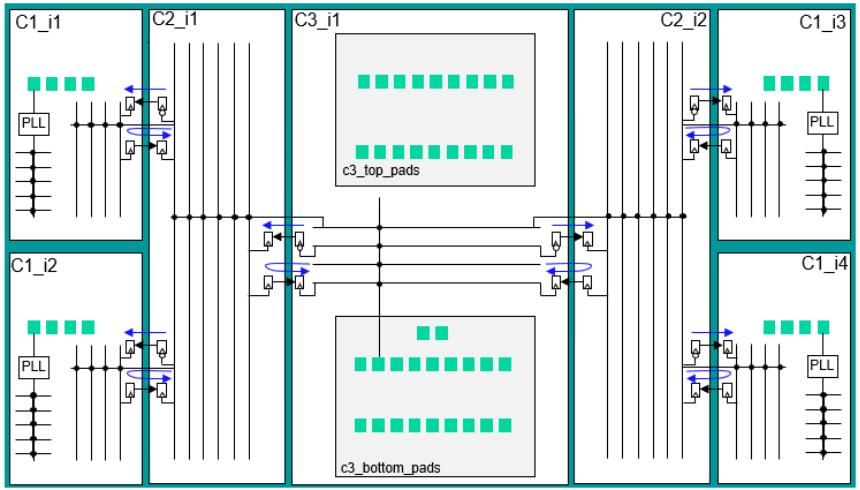
Design style becoming more prevalent in industry

Example chip with 3 unique tiles
C1 and C2 mirrored on both sides of C3

IO pads mostly located in C3
RX/TX pads in each C1 instance

Functional clock tree localized within each tile

Clock sent from C3 to outer tiles
Forward timing interface outward
Loop timing interface on the returned data

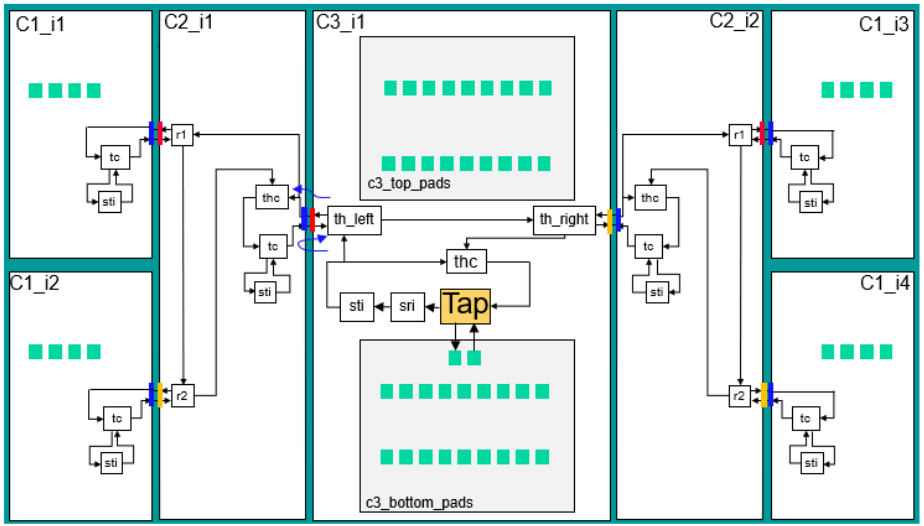


Source: “IJTAG User’s Manual”

Tiling architecture including test (IJTAG, MemoryBIST)

The flow is similar to the hierarchical flow

IJTAG clocking localized to each tile
- TCK clock sent outward from central tile
- Forward timing in outward paths
- Loop timing on the return paths



Source: “IJTAG User’s Manual”

IP cores with pre-defined test bus interfaces

CPU/GPU IP cores are common examples

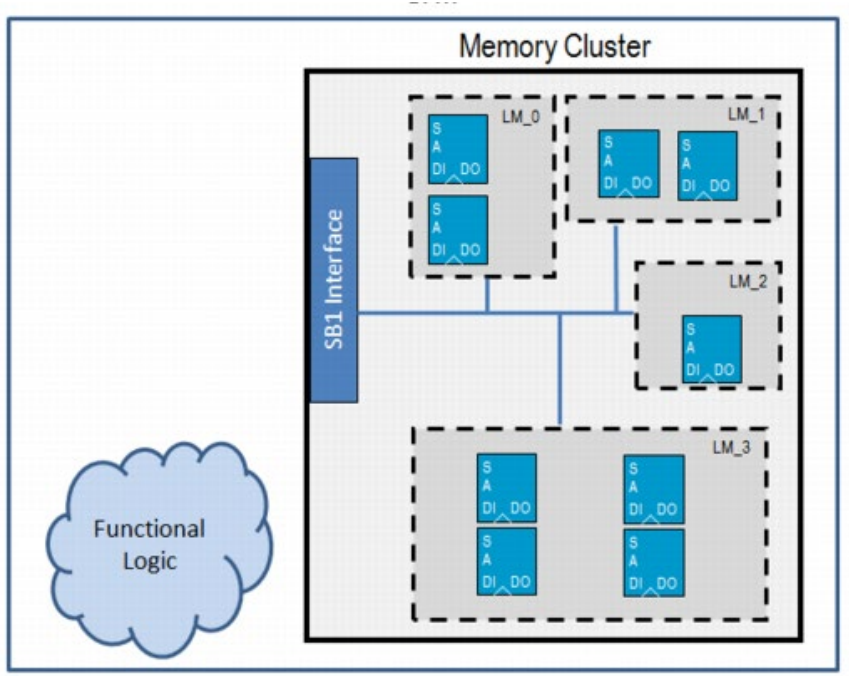
Memories all share a same test bus

- Logical memory groups are serially tested

"RTL defines logical memories which can be implemented with different combinations of physical memories by core integrator"

Advantages

- IP provider can minimize impact of MBIST on core performance
- "Perfect" at-speed test can be achieved by accessing memories through functional registers



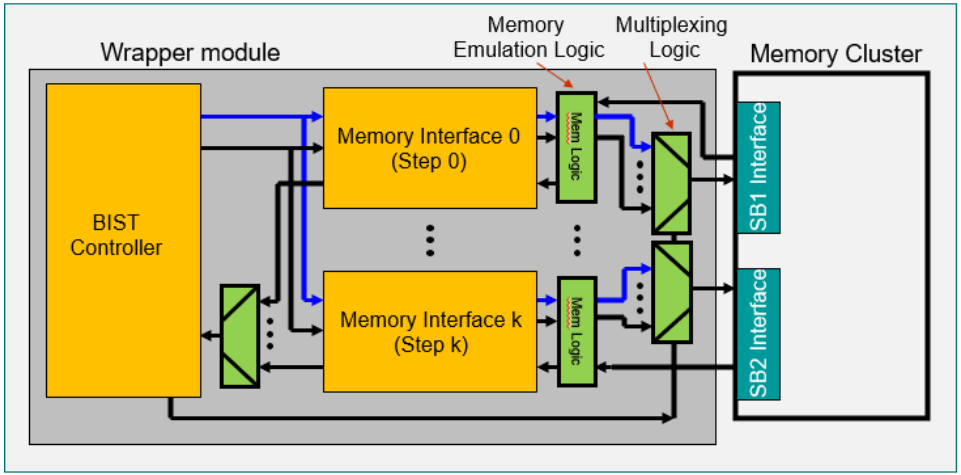
Source: "TMBIST User's Manual"

Memory BIST for IP cores with shared test bus

I. The hardware generated for shared bus interfaces is similar to the hardware generated for standard memories

II. The memory emulation modules and the multiplexing logic
provide virtual access to all the memories

III. The interfacing logic external to the cluster can be minimized given the serial test schedule



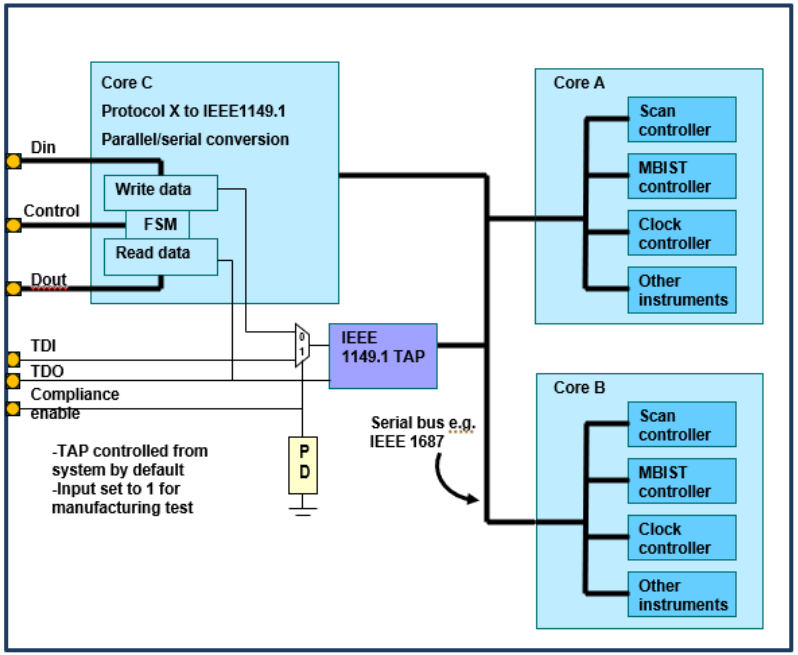
Source: “TMBIST User’s Manual”

Access to memory BIST at system level – method 1

a. In high reliability systems, it is becoming more frequent to run memory BIST upon chip power-up or during maintenance

b. It is usually not practical to get access to test circuitry, like MBIST, through the TAP interface used during manufacturing

c. Automation exists to interface a system bus (e.g., APB) to a TAP and gain full access to the test circuitry including detailed diagnosis



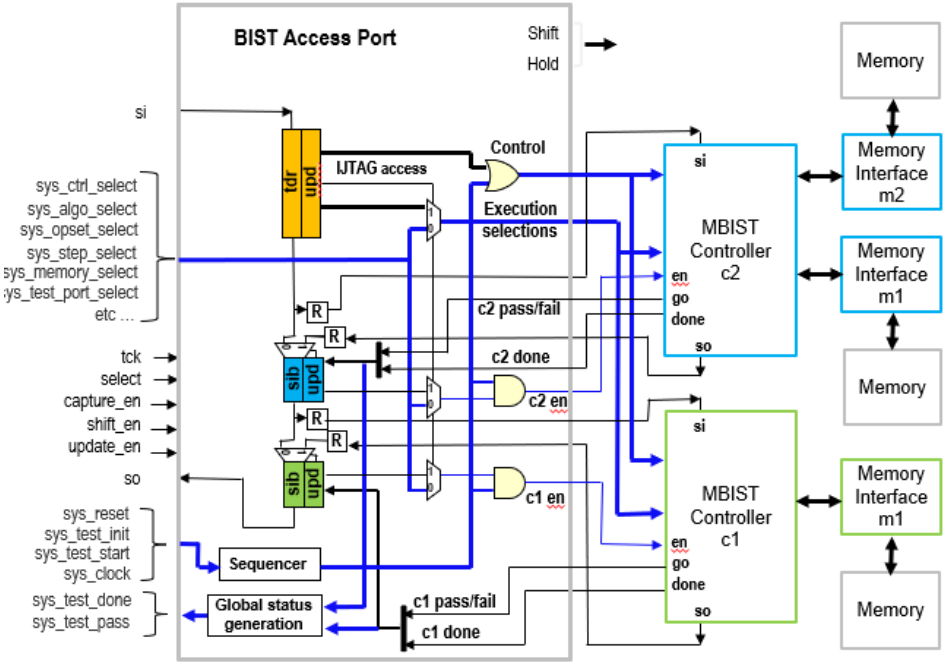
Source: "IST User's Manual"

Access to memory BIST at system level – method 2

a. The advanced BAP enables memory tests through system signals connected to the BAP direct access interface

b. The direct access interface supports a low-latency protocol to configure the MBIST controllers

c. The parallel interface between the BAP and the controllers offers a broader selection of test options



Source: “TMBIST User’s Manual”

Verification checklist for memory test logic

All verification steps used for functional logic apply:

- ✓ **Memory test view verification**
- ✓ **Sign-off verification**
- ✓ **Timing verification**
- ✓ **Logic equivalence check**
- ✓ **Clock/reset domain crossing**

Design Integration summary

- ✓ **Memory BIST and repair implementation must be well integrated with the functional design and verification flow**
- ✓ **Several partitioning rules are used to assign memories to a test controller (Cx rules) and determine memories that can be tested in parallel within a controller (Px rules)**
- ✓ **Tiling architecture becoming more prevalent in industry**
- ✓ **CPU/GPU IP commonly use a shared bus architecture**
- ✓ **Reuse of memory BIST and repair functions in-system becoming more frequent**

Advanced MBIST Techniques

Advanced MBIST Techniques

- Functional debug
- Dynamic RAM (DRAM)
- DRAM BIST hardware
- Content Addressable Memory (CAM)

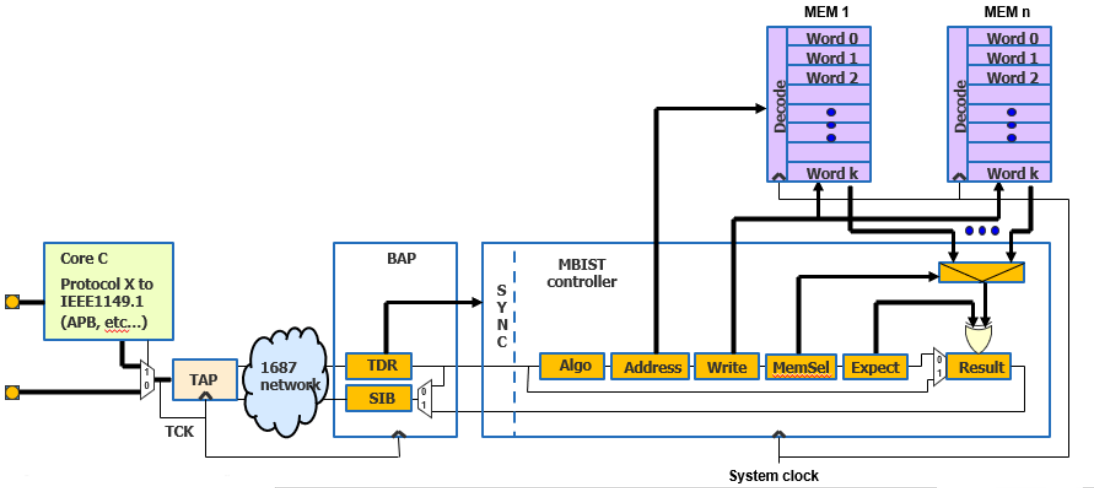
  - Basic CAM structure
  - CAM bit cell structure + defect detection
  - CAM custom interface

Functional debug

MBIST logic can be leveraged to debug systems
- E.g. Read memory locations after a system crash
- E.g. Write memory locations to "patch" memory content and resume system operation

Controller features to optimize access
- Memory selection
- Initial address selection
- Address auto-increment
- Direct access to read result

Access might have to be Unrestricted for security reasons



Source: "TMBIST User's Manual"

Key takeaway: By leveraging MBIST for functional debugging, engineers can efficiently diagnose, isolate, and resolve memory-related faults in complex systems

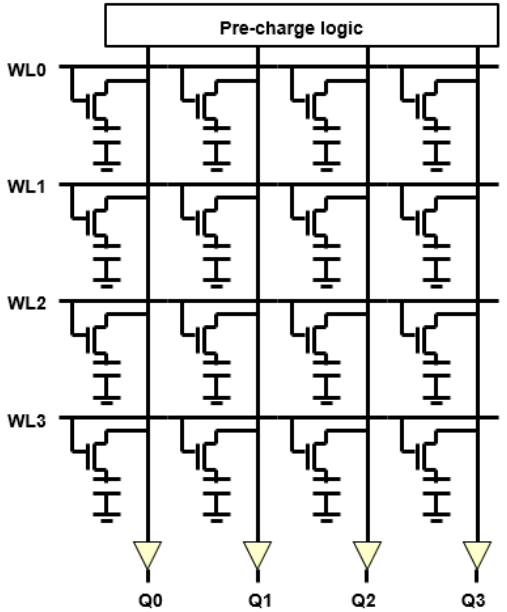
Dynamic RAM (DRAM)

Bit cell consists of a single transistor and capacitor

Capacitor is charged or discharged to store a value of 1 or 0

Retention time of capacitor is limited, and periodic refresh is required

Test algorithms different from SRAM
- Large address space
- Need to limit algorithm complexity for test time considerations
- Burst read/write modes of operation
- Inefficient to perform Read-Modify-Write on a single address
- Failure modes limited to memory cell aging and defective connections to/from memory which is typically off-chip
- Additional operations required for pre-charge and activation of banks



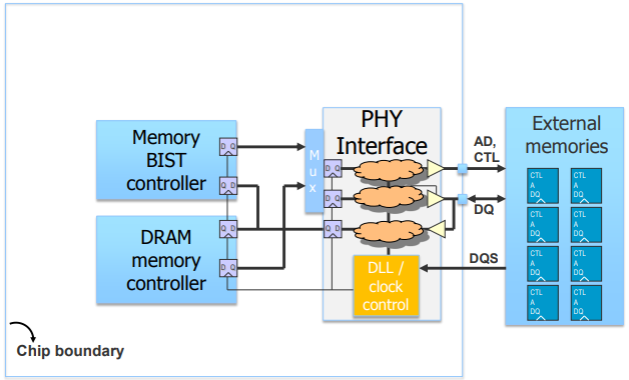
Source: "Digital Integrated Circuits: A Design Perspective" by Jan M. Rabaey et al

Key takeaway: DRAM testing demands unique algorithms and procedures to account for its reliance on capacitors, burst operations, and pre-charge logic

DRAM BIST hardware

Several standard DRAM interfaces used in industry
- DDRx and LPDDRx family, WIDE-IO, HBM, etc…
- Access performed through PHY (Physical layer interface)

Memories might come from different manufacturers
- Internal structure might vary slightly
- E.g., address/data scrambling
- Difficult to apply same quality of test than during manufacturing of the memories
- Information about internal structure not always available
- Requires run-time programmable scrambling



Source: "TMBIST User's Manual"

Key takeaway: DRAM BIST hardware provides a robust framework for testing external memory interfaces

Emerging trends and future directions

Emerging trends and future directions

- Non-destructive memory test
- Memories with ECC
- Non-volatile memories (NVMs)
  - STT-MRAM
- BIST for MRAM

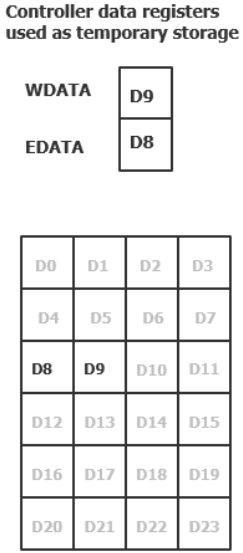
Non-destructive memory test

Also known as transparent MBIST
- Testing memory without loosing its initial content
- System can resume operation after test

Used to detect memory aging faults in system

Possible to take advantage of system idle time to perform some testing and reduce latency of fault detection

- Algorithm performs short sequence of operations on two (or more) memory locations
- Restores initial content if modified during test
- Operations performed at-speed
- Relatively large area cost for wide memories as two full-size data registers are required to store locations under test



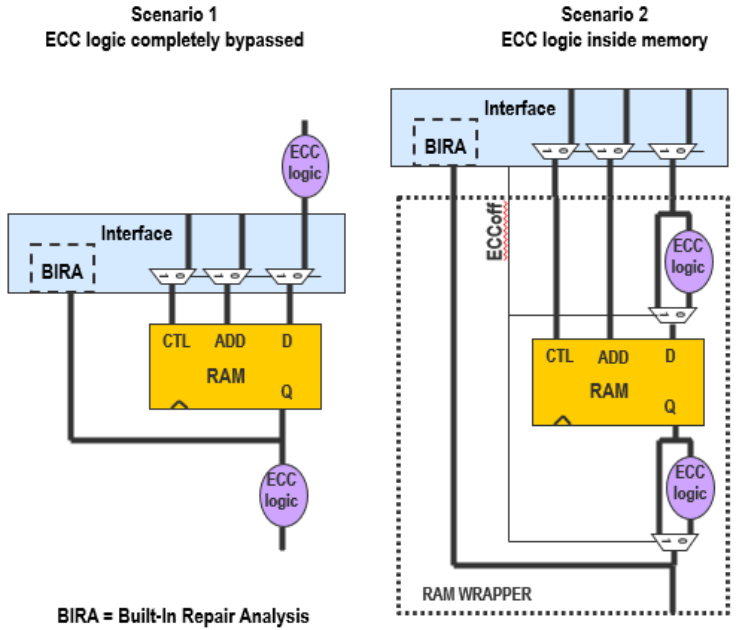
Key takeaway: Non-destructive memory testing combines fault detection with data preservation

Testing memories with ECC

4 cases to address for memories with Error Correction Code (ECC) logic

Case 1: ECC exclusively used to fix soft/intermittent errors when system is in operation
- Completely transparent to MBIST (Scenario 1)
- Input provided to turn off ECC (Scenario 2)
- ECC always bypassed to avoid test escapes

Case 2: Masking of check bits when ECC is enabled (scenario 2 only)
- Can be useful during design verification
- Allows partial functional test of ECC logic
- Recommended to use scan for complete structural test of ECC logic



Source: "TMBIST User's Manual"

Key takeaway: Testing memories with ECC involves balancing fault detection with error correction

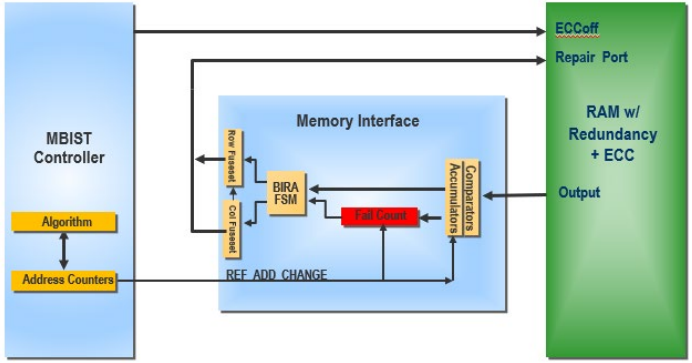
ECC-assisted repair

ECC can be used as complement to spare rows/columns to fix hard errors

BIRA modified to tolerate a number of errors in a word that is less than or equal to the maximum error correction capability
- Number can be different during manufacturing and in system
- Beware of test escapes!

Case 3: ECC assumed to fix hard errors during power-on self-test (POST)
- Applicable to SRAM to improve system availability
- Repair done as usual during manufacturing

Case 4: ECC assumed to fix hard errors during manufacturing
- Required for Non-Volatile Memories (NVMs) such as MRAM



Source: “TMBIST User’s Manual”

Non-Volatile Memories (NVMs)

- Several new memory types are emerging
  -Most promising ones are Magnetoresistive RAMs (MRAMs) and Resistive RAMs (ReRAMs)
- NVMs come with new defects requiring new fault models and test algorithms
- NVMs also require calibration (trimming) of reference voltage or current for both read and write operations
- ECC logic often required on top of conventional row/column repair to improve yield

Key takeaway: Non-volatile memories are revolutionizing data storage by providing persistent, reliable, and scalable solutions

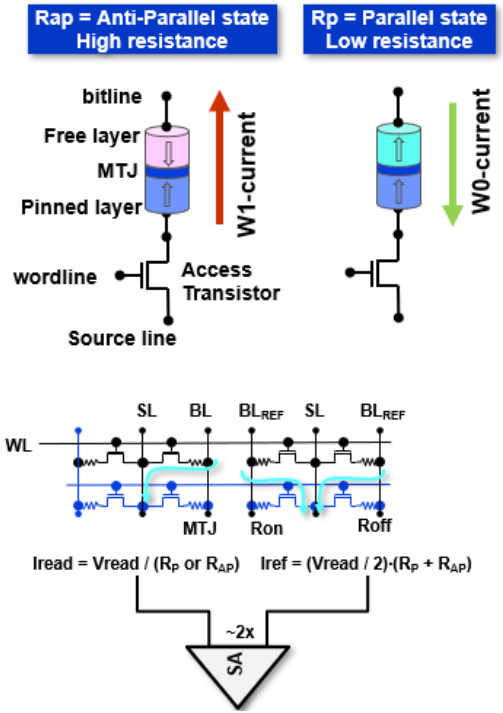
STT-MRAM cell structure and operation

Magnetoresistive RAM (MRAM) is non-volatile
- i.e., retains its state even if power is turned off
- Most common style is Spin Transfer Torque (STT)
- Most promising emerging NVM

Bit cell structure composed of single access transistor and Magnetic Tunnel Junction (MTJ)

Bit cell value determined by magnetic polarity of MTJ's free layer
- Anti-parallel = 1 (high resistance state)
- Parallel = 0 (low resistance state)

Read performed by sensing differential voltage or current level of selected bitline w.r.t. reference bitline



Key takeaway: Its innovative use of spin-transfer torque for switching represents a major leap forward in memory design, making it a strong candidate for future applications in both consumer and industrial markets

BIST for MRAM

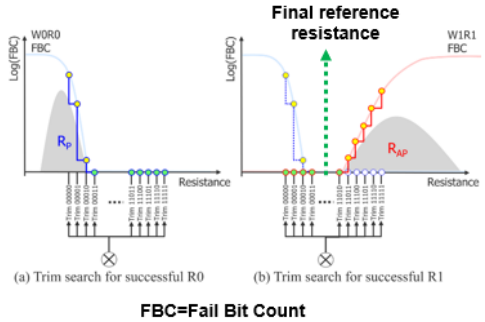
MRAM adds two major requirements to BIST in order to improve yield

Use ECC to fix manufacturing defects that can't be fixed with conventional repair

- ECC typically used in combination with row repair
- Column repair might also be used to repair column logic failures even though repairable with ECC only
  - e.g., failures in bitline, write driver or sense amplifier

Trimming of read/write reference voltage/current

- Resistance of cells (Rp/Rap) varies significantly within a same memory due to process shift, array configuration, temperature, etc…
- MRAM has relatively small read/write windows which require accurate trimming based on test results
- Read reference resistance needs to be set between distribution tails of highest Rp and lowest Rap values



Case studies and publications

Siemens link

ETS'20

ITC'20

ETS'21

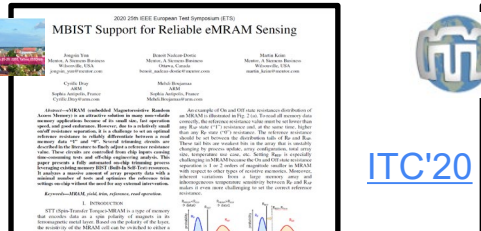
ETS'21

Vghc'21

VTS'22

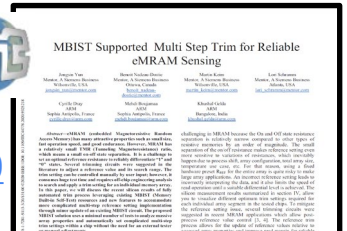
DATE'23

ITC'23



2020 25th IEEE European Test Symposium (ETS)

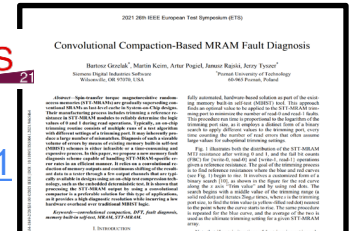
MBIST Support for Reliable eMRAM Sensing



MBIST Supported Multi Step Trim for Reliable eMRAM Sensing

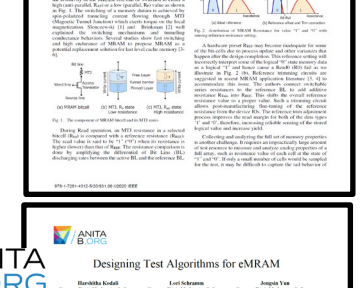
2021 26th IEEE European Test Symposium (ETS)

MBIST-supported Trim Adjustment to Compensate Thermal Behavior of MRAM

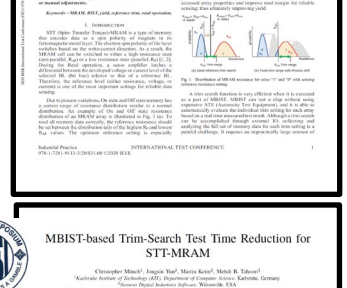


2021 26th IEEE European Test Symposium (ETS)

Convolutional Compaction-Based MBIST Fault Diagnosis



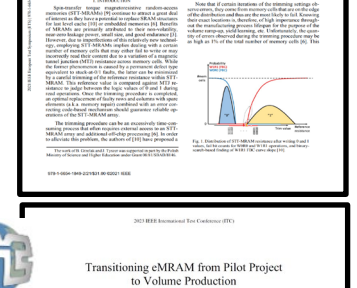
Designing Test Algorithms for eMRAM



MBIST-based Trim-Search Test Time Reduction for STT-MRAM

2022 Design, Automation & Test in Europe Conference (DATE 2022)

Smart Hammering: A practical method of pinhole detection in MRAM memories

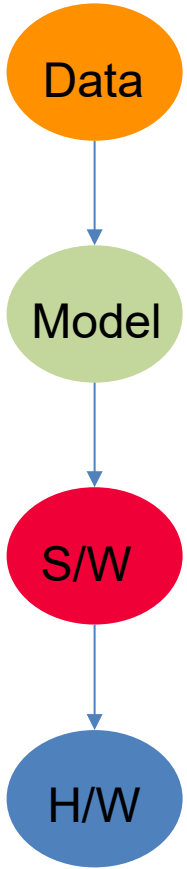


2023 IEEE International Test Conference (ITC)

Transitioning eMRAM from Pilot Project to Volume Production

Scaling in AI

1. Scaling data, model size, and compute consistently leads to new capabilities
2. Scaling comes in all forms, from data all the way to hardware
3. Inference and test time compute is a new frontier for scaling
4. Constitutional AI (Harmlessness vs Evasiveness)



[3] Constitutional AI : Harmlesness from AI Feedback
Bai et al., Anthropic

Source: LAD presentation by Azalia M.

AI in Electronic design automation (EDA) at Siemens

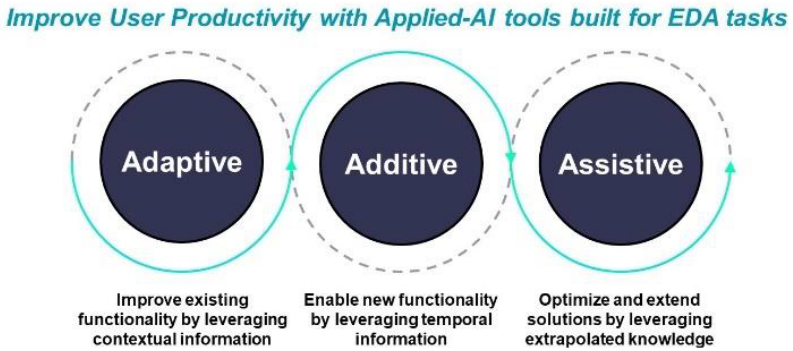
At present

1. Machine learning has led to intelligent problem solving – significantly improved scan diagnosis and yield analysis.

2. Decision making using limited memory
3. Self-learning capabilities

Future focus

1. Layout pattern systematics for improved yield.
2. Knowledge acceleration to guide users and automate flows
3. Training ATPG on circuit structures for optimization



Source: Using AI for advanced in DFT

Key takeaway: AI in EDA is not just about enhancing testing processes—it's about transforming the entire design and manufacturing lifecycle

Tutorial Takeaways

- Importance of MBIST in modern SoC design and its advantages over traditional testing methods
- Integrate MBIST into SoC design flow, from RTL insertion to physical design
- Apply advanced MBIST techniques to optimize fault coverage, test time and area overhead
- Efficient future directions with AI in memory testing

Thanks!

Acknowledgements:
I would like to thank Dr. Benoit Nadeau-Dostie, Ron Press, Albert Au, Artur Pogiel and Sebastian Bromberek for feedback and helpful pointers.

Q&A and Discussion