Testability of VLSI

Lecture 12
Built-in Self-Test

By Dr. Sanjay Vidhyadharan

sanjayvidhyadharan.in

Built-in Self-Test

Gordon and Nadig described the economic impact of signature analysis and BIST on the first systems that used BIST: two Hewlett-Packard digital voltmeters, one of them called the HP 3455A. Development time and costs rose roughly 1%. There was a 1% increase in parts cost due to added jumpers and extra ROM space required in the electronics for signature analysis, but total factory costs dropped, because of a 5% decrease in other materials costs.

sanjayvidhyadharan.in

Built-in Self-Test

1. There is an extremely high and still increasing logic-to-pin ratio on the chip. This increasingly makes it harder to accurately observe signals on the device, which is essential for testing.

2. VLSI devices are increasingly dense and faster with sub-micron feature sizes.

3. There are increasingly long test-pattern generation and test application times.

4. Prohibitive amounts of test data must be stored in the *automatic test equipment* (ATE.)

5. There is increasing difficulty in performing at-speed (rated clock) testing using external ATE. For clock rates approaching 1 *GHz*, at-speed testing with an ATE is very expensive due to pin inductance and high tester pin costs.

6. Designers are unfamiliar with the gate-level structure of their designs, since logic is now automatically synthesized from the *VHDL* or *Verilog* hardware description languages. This compounds the problem of testability insertion.

7. There is a lack of skilled test engineers.

sanjayvidhyadharan.in

Built-in Self-Test

Complexity. One unfortunate property of large VLSI circuits is that testing cannot be easily partitioned. Even though each part is fully testable and has a test set that gives 100% stuck fault coverage, the cascaded connection of the two parts will often have untestable and redundant hardware and much lower stuck-fault coverage. For design and test development effort, BIST provides a way to hierarchically decompose the electronic system-under-test, so this allows sub-assemblies to be first run through a BIST cycle, and if there are no faults, then boards in the system are run through a BIST cycle.

sanjayvidhyadharan.in

Quality. Typical quality requirements are 98% single stuck-fault coverage or 100% interconnect fault coverage. In huge systems, this is attainable only through *design for testability* (DFT), and BIST is the preferred form of DFT.

Built-in Self-Test

Test Generation Problems. It is difficult to carry a test stimulus involving hundreds of chip inputs through many layers of circuitry to the chip-under-test, and then convey the test result back through the many circuit layers to an observable point. BIST localizes testing, which eliminates these problems.

Test Application Problems. In-circuit testing (ICT) used a *bed of-nails* fixture customized for the PCB-under-test. The bed-of-nails tester applied stimuli to the solder balls on the back of the PCB where the component leads were soldered to the PCB. Power was applied only to the component under test – all others in the PCB were left unpowered. It was effective for chip diagnosis and board wiring tests. However, ICT is not effective unless the PCB is removed from the system, so it is not helpful in system-level diagnosis. Also, *surface-mount technology* (SMT) components are often mounted densely on both sides of the board, and the PCB wire pitch is also too small for accurate probing of the back of the board by the bed of-nails tester. Therefore, ICT is no longer a solution. BIST, however, solves these problems by eliminating expensive ATE.

sanjayvidhyadharan.in

Built-in Self-Test

1. Chip/Board Area Cost vs. Tester Cost:

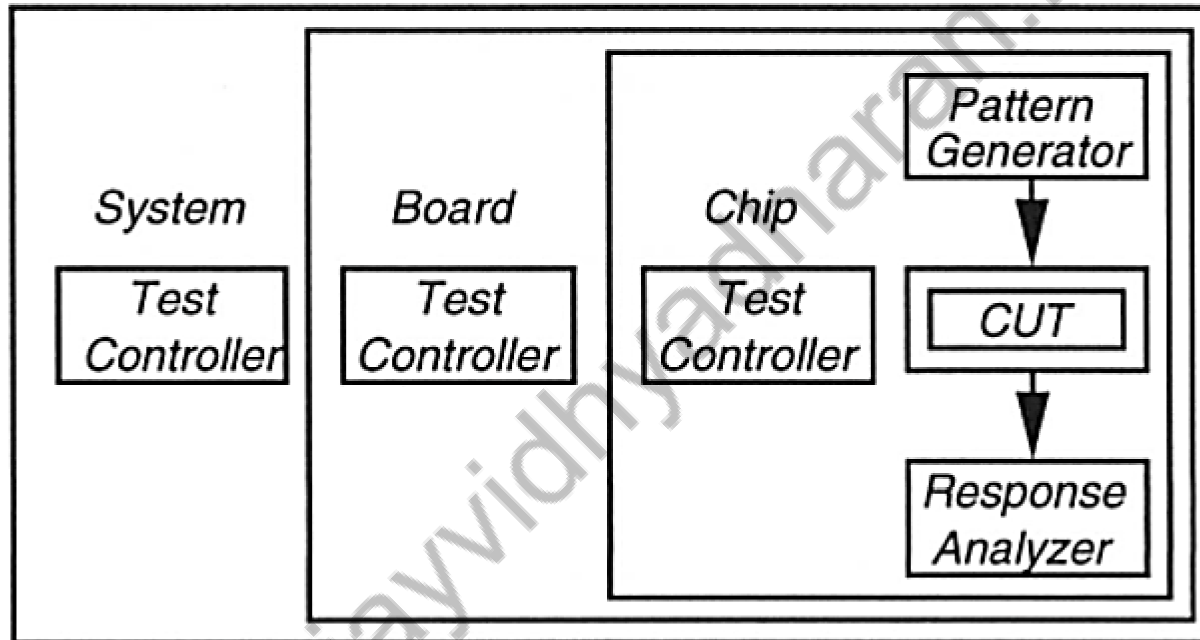
There is a slight cost increase due to BIST in design and test development, because of the added time required to design and add pattern generators, response compacters, and testability hardware. However, is that this is less costly than test development with ATPG.

2. Chip/Board Area Cost vs. System Downtime Cost

Without BIST, maintenance test requires the presence of an expensive ATE at the site of the failing system, and this is a significant cost. With BIST, there is no need for an ATE, so this reduces system test cost. For boards and systems, BIST drastically reduces the diagnosis and repair cost, by quickly determining and indicating which sub-assembly or component is faulty, without the extensive labor and equipment normally required.

sanjayvidhyadharan.in

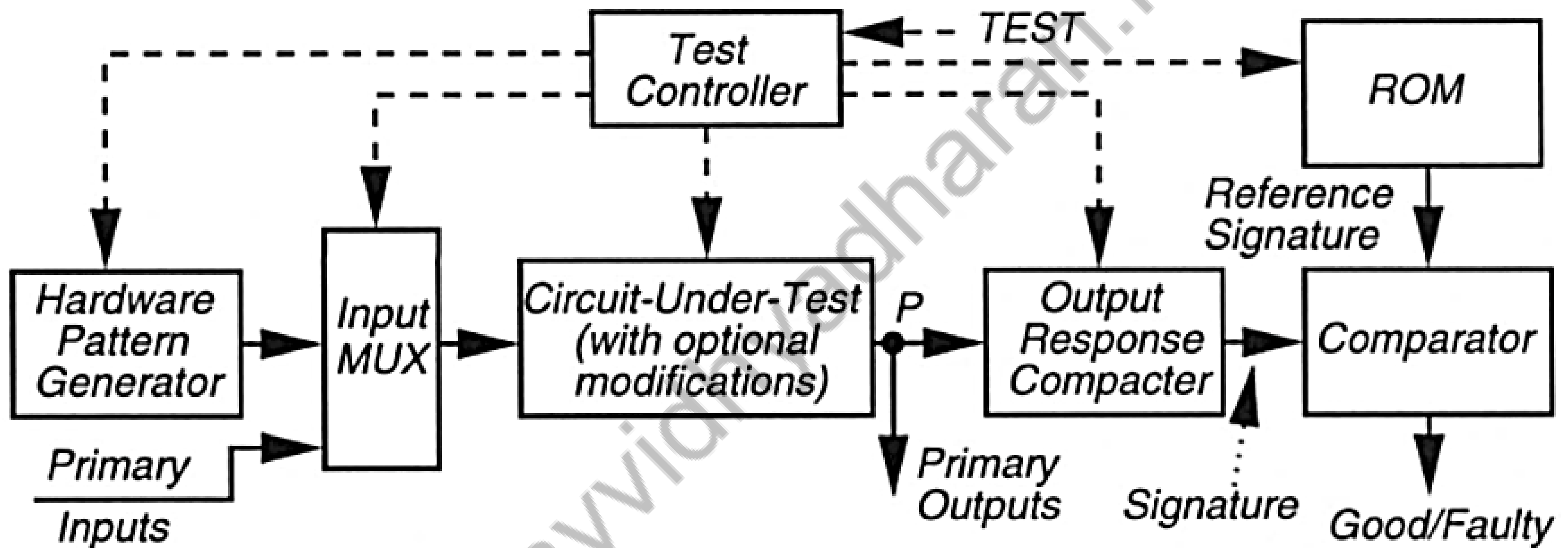
BIST Hierarchy



CUT – *Circuit-Under-Test*

sanjayvidhyadharan.in

BIST Implementation



sanjayvidhyadharan.in

BIST Pattern Generation

1. **ROM :** Expensive in chip area. (Test vectors from ATPG)

2. **Linear feedback shift register (LFSR*)* :** Uses very little hardware and is currently the preferred BIST pattern generation method. (Random Test pattern generation. Coverage may not be not 100%)

3. **Binary Counters :** A binary counter can generate an exhaustive test sequence, but this can use too much test time if the number of inputs is huge. For example, with 64 inputs and the test-pattern generator clocked at 100 *MHz*, this takes 51,240,955.8 hours of test time to generate all patterns. Also, the binary counter requires more hardware than the typical LFSR pattern generator. (exhaustive testing)

4. **Modified Counters:** Modified counters have also been successful as test-pattern generators, but they also require long test sequences.

5. **LFSR and ROM:** LFSR as the primary test mode, and then generate test-patterns with an ATPG program for the faults that are missed by the LFSR sequence.

6. **Cellular Automaton:** In this approach, each pattern generator cell has a few logic gates, a flip-flop, and connections only to neighboring gates. The cell is replicated to produce the cellular automaton.

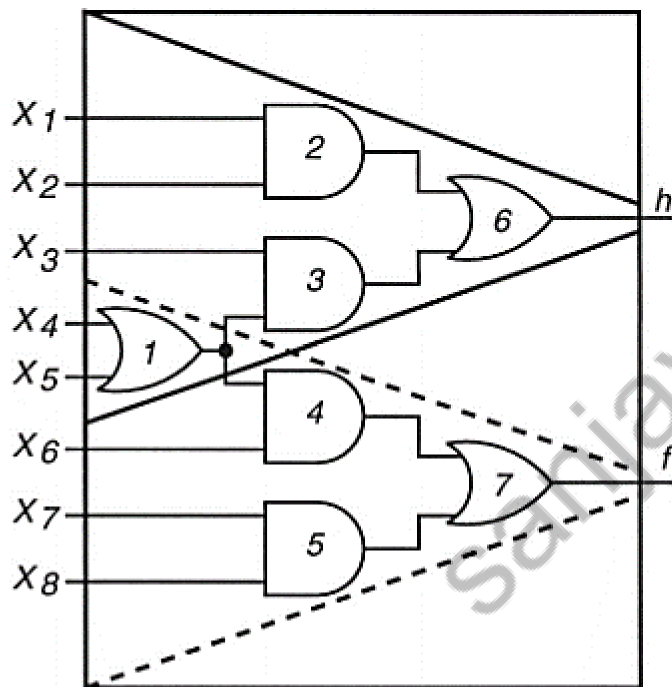
sanjayvidhyadharan.in

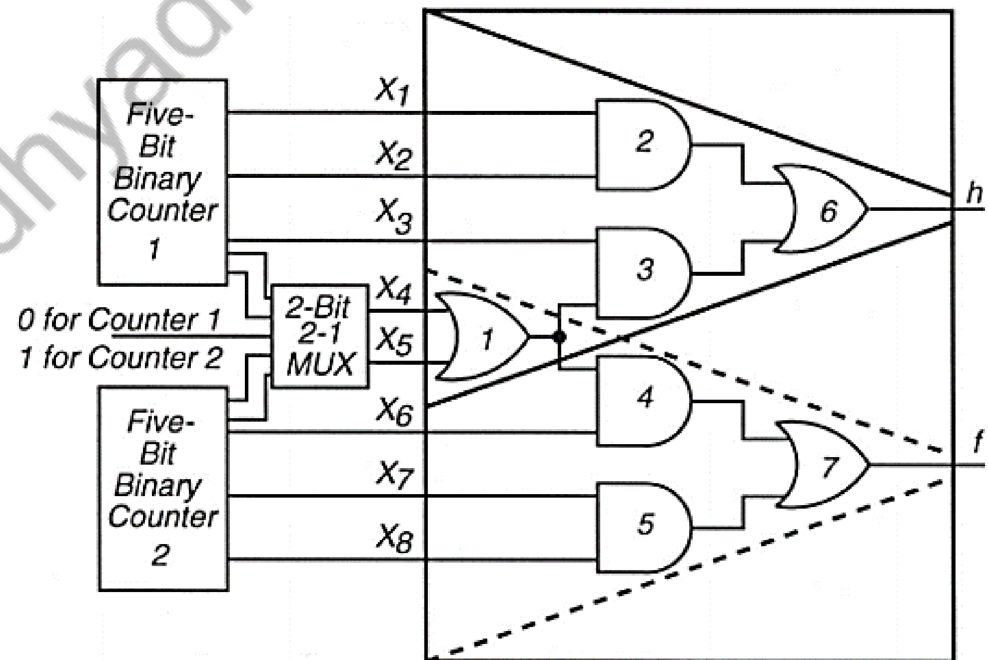
BIST Pattern Generation

Exhaustive Pattern Generation

Total number of test vectors required ? Is it $2^8 = 256$?

1.Partition a large circuit into fanin cones $2^5 + 2^5 = 64$





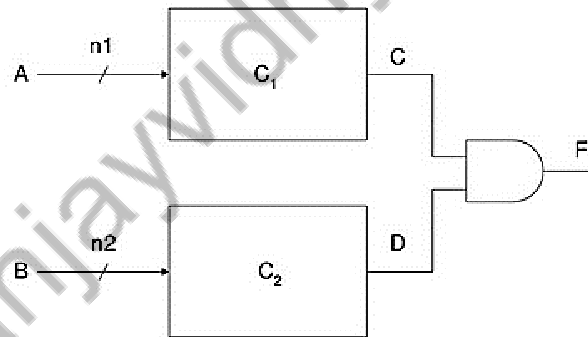
sanjayvidhyadharan.in

BIST Pattern Generation

Exhaustive Pattern Generation

2. **Hardware partitioning** (physical segmentation) in which we add extra circuit logic in order to divide the CUT into smaller subcircuits, each directly controllable and observable. Each of these is tested exhaustively.

3. **Sensitized path segmentation**, in which the circuit is partitioned so that sensitizing paths are set up from PIs to the partition inputs, and then from the partition outputs to the POs. Each partition is tested individually while the remaining partitions are simulated, so that non controlling signals are set in the CUT to sensitize and propagate signals in the partition-under-test.



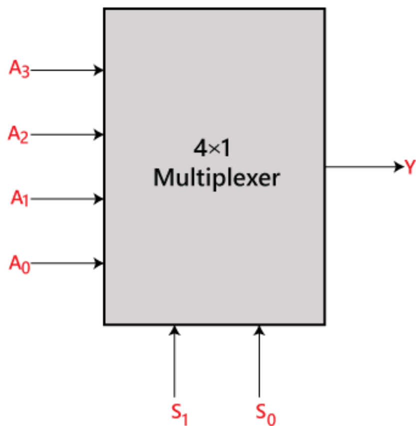
4. **Partial hardware partitioning** , combines *hardware partitioning* for observing signals in the partition-under-test, and *sensitized path segmentation* to control inputs to the partition-under-test

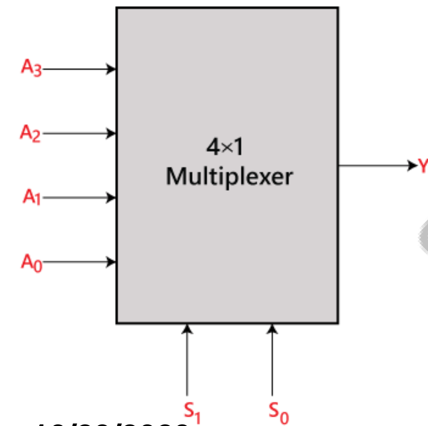
ELECTRICAL ELECTRONICS COMMUNICATION INSTRUMENTATION
sanjayvidhyadharan.in

BIST Pattern Generation

Pseudo-Exhaustive Pattern Generation (8 vector testing)

4 X 1 Mux





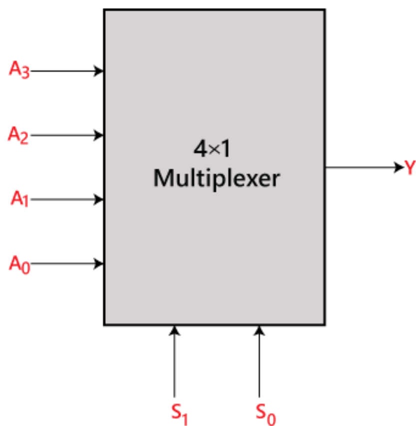
W	X	A_1	B_1	C_1	D_1	A_2	B_2	C_2	D_2	F_1	F_2
0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	1	1
0	1	0	1	0	0	0	1	0	0	1	1
1	0	0	0	1	0	0	0	1	0	1	1
1	1	0	0	0	1	0	0	0	1	1	1

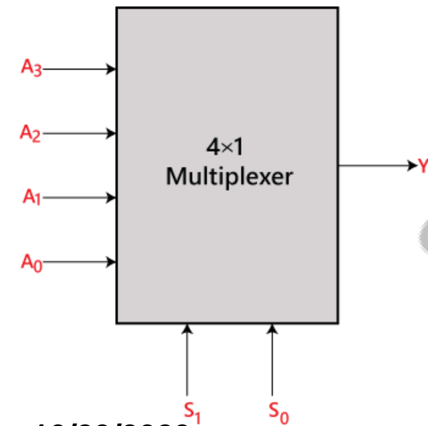
sanjayvidhyadharan.in

BIST Pattern Generation

Pseudo-Exhaustive Pattern Generation (20 vector testing)

4 X 1 Mux



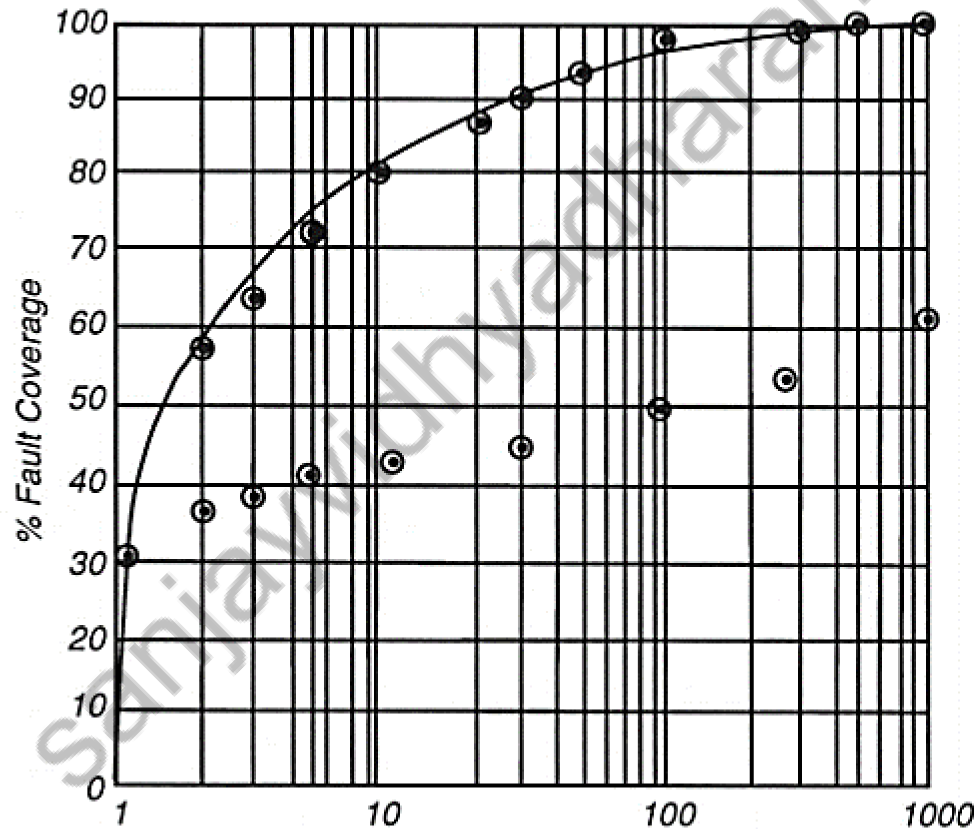


W	X	A_1	B_1	C_1	D_1	A_2	B_2	C_2	D_2	F_1	F_2
0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	1	1
0	1	1	0	0	0	1	0	0	0	0	0
1	0	1	0	0	0	1	0	0	0	0	0
1	1	1	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0	0
0	1	0	1	0	0	0	1	0	0	1	1
1	0	0	1	0	0	0	1	0	0	0	0
1	1	0	1	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	1	0	0	0
0	1	0	0	1	0	0	0	1	0	0	0
1	0	0	0	1	0	0	0	1	0	1	1
1	1	0	0	1	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	0	0	1	0	0
1	0	0	0	0	1	0	0	0	1	0	0
1	1	0	0	0	1	0	0	0	1	1	1

sanjayvidhyadharan.in

BIST Pattern Generation

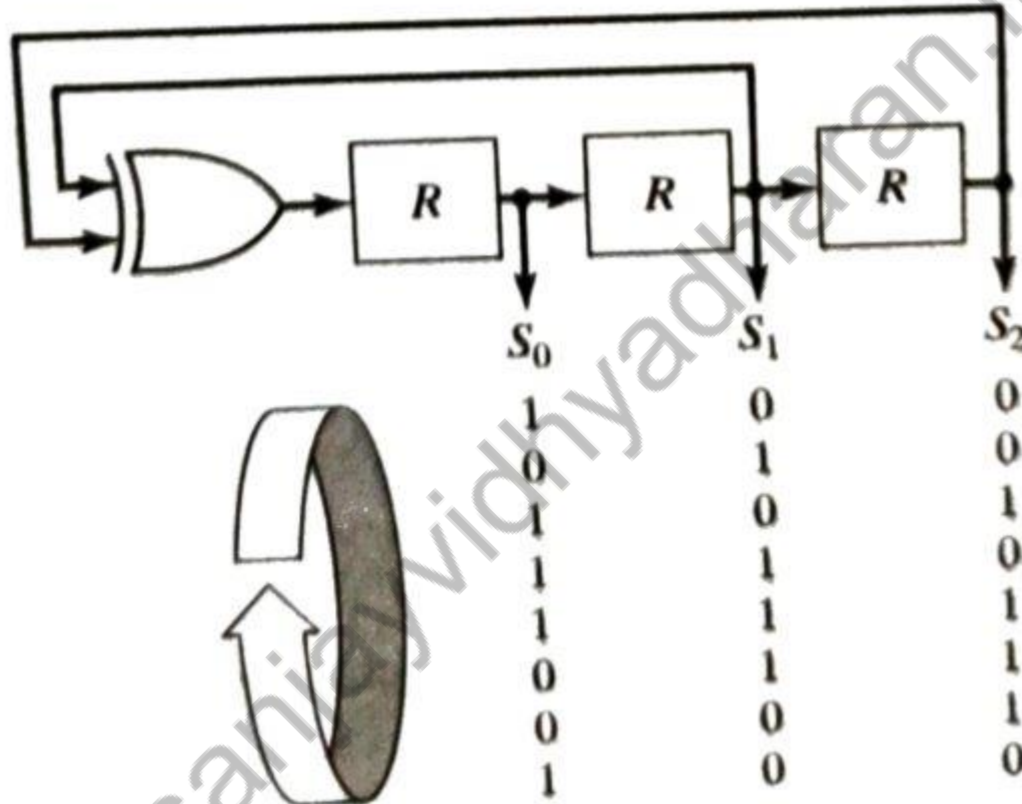
Random-pattern testing and fault coverages



sanjayvidhyadharan.in

BIST Pattern Generation

Linear-feedback shift-register (LFSR)

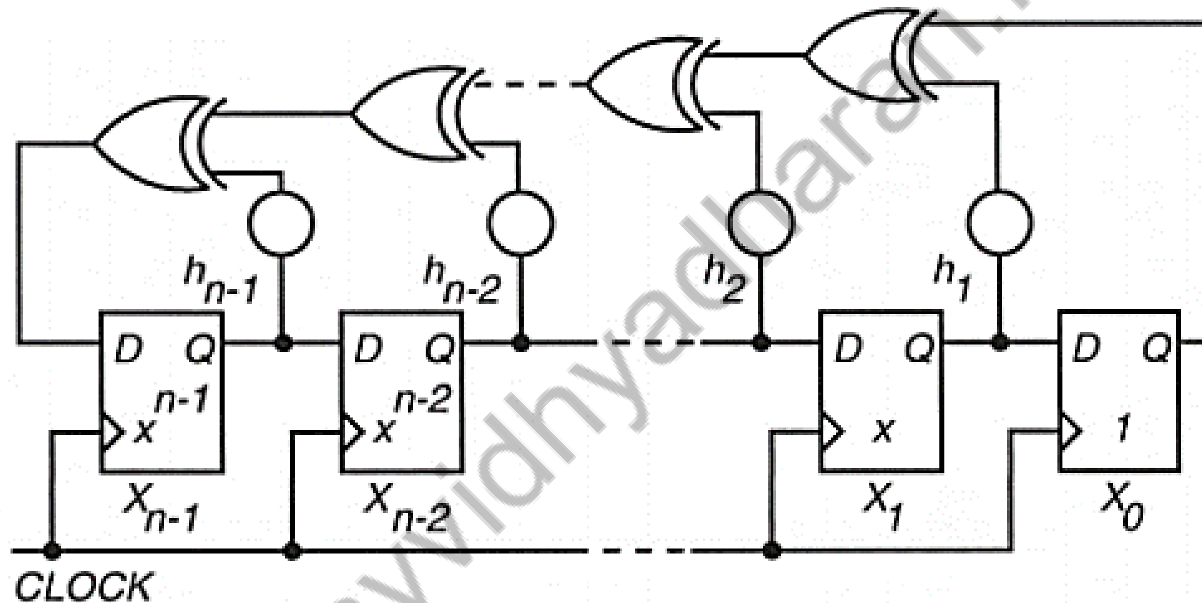


sanjayvidhyadharan.in

Not necessary all 8 possible combinations are generated

Pseudo-Random Pattern Generation

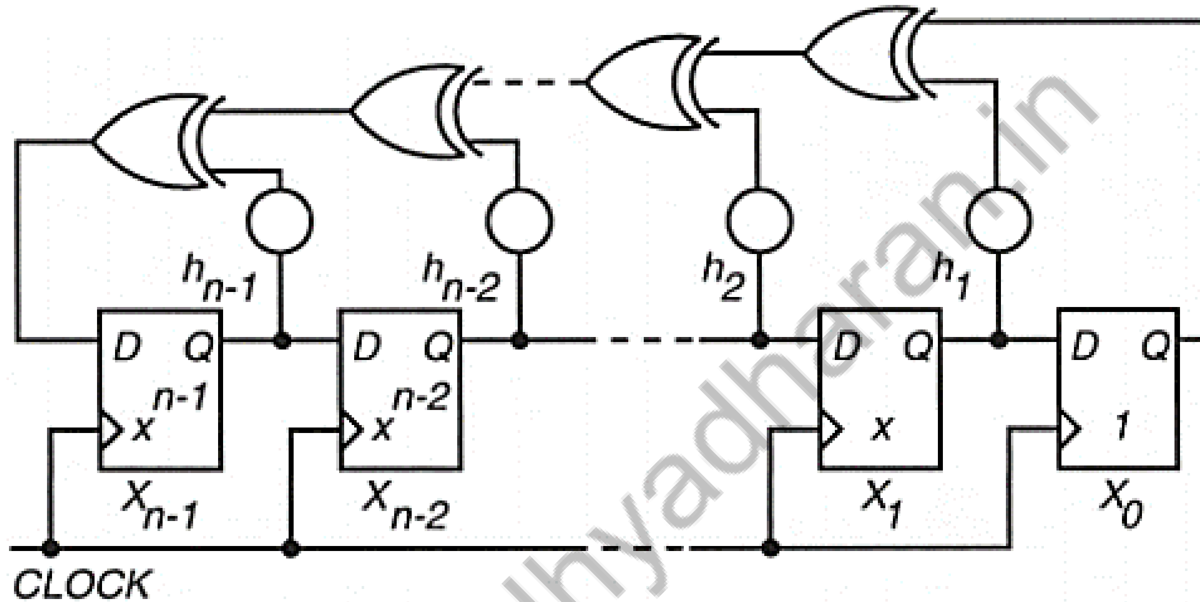
Linear-feedback shift-register (LFSR)



These patterns have all of the desirable properties of random numbers, but are algorithmically generated by the hardware pattern generator and are therefore repeatable, which is essential for BIST. We no longer cover all input combinations, but long test-pattern sequences may still be necessary to attain sufficient fault coverage.

sanjayvidhyadharan.in

Pseudo-Random Pattern Generation



Circuit Initialization

It is very important in random logic BIST to initialize all flip-flops in the circuit when BIST is used with partial scan.In the real hardware, different chips will randomly initialize their flip-flops to different values. Initialization problems can be discovered by setting all flip-flops initially to the X state, running the BIST cycle, and simulating the system in a 3-valued logic simulator. If the MISR or other response compacter finishes the test session with bits in the X state, then initialization is not correct. All such uninitializable flip-flops must then be initialized by adding master set or reset lines to them.

sanjayvidhyadharan.in

BIST Response Compaction

1.Compaction – A method of drastically reducing the number of bits in the original circuit response during testing in which some information is lost. E.g. (Parity checking, Ones counting, where we count the number of ones in the output responses from the circuit.

2. Compression – A method of reducing the number of bits in the original circuit response during testing in which no information is lost, so the original output sequence can be fully regenerated from the compressed sequence. Compression schemes, at present, are impractical for BIST response analysis, because they inadequately reduce the huge volume of data, so we use only compaction schemes.

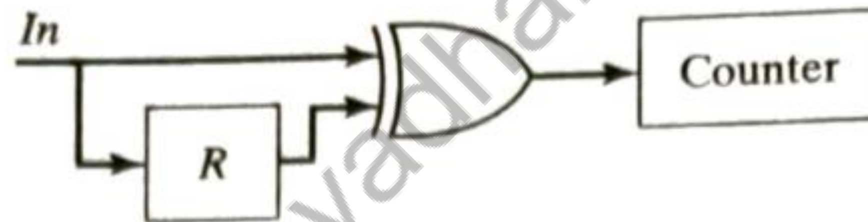
3. Signature – A statistical property of a circuit, usually a number computed for a circuit from its responses during testing, with the property that faults in the circuit usually cause the signature to deviate from that of the good machine.

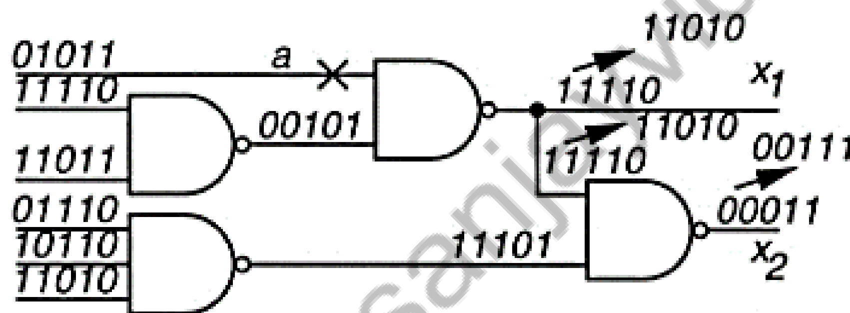
sanjayvidhyadharan.in

BIST Logic Circuits

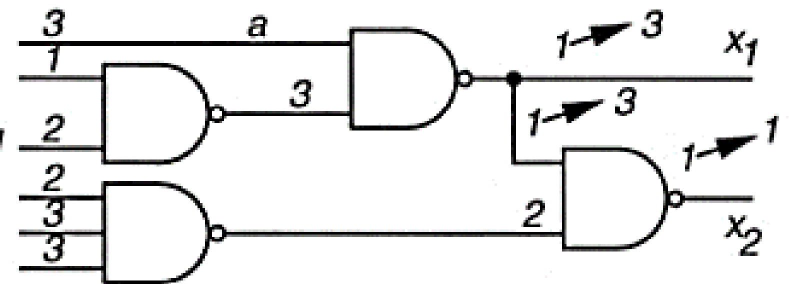
Single Bit signature register (MISR)

Transition Count Response Compaction





(a) Logic simulation of good machine and fault a stuck-at-1.

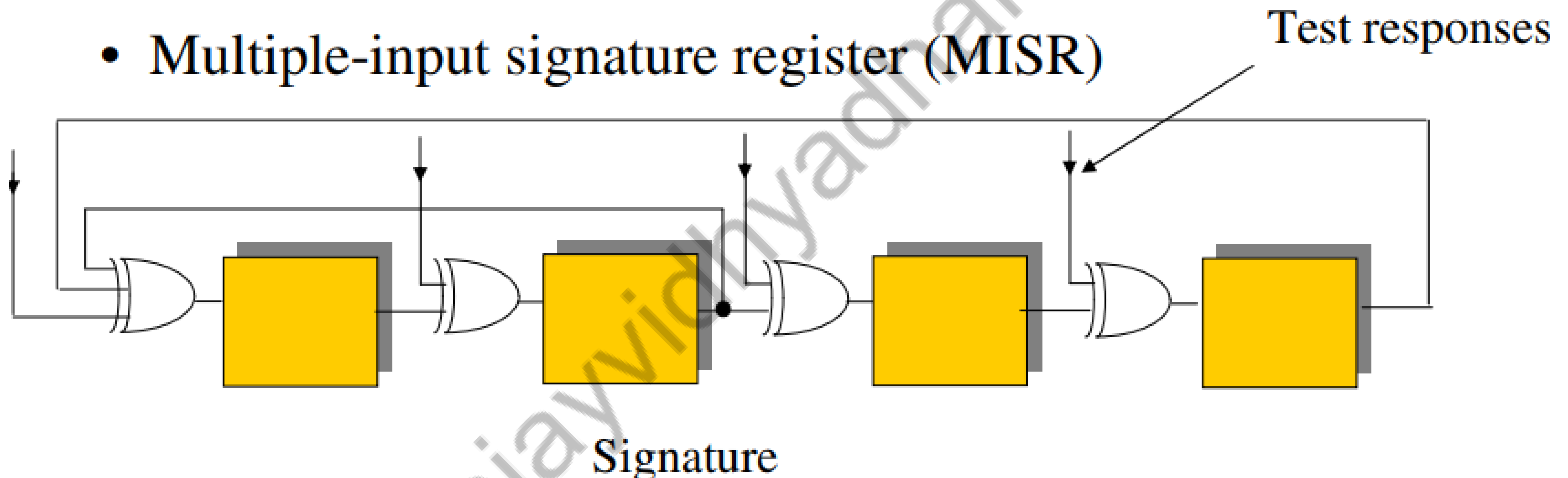


(b) Transition counts of good and failing machines.

sanjayvidhyadharan.in

BIST Logic Circuits

Multiple-input signature register (MISR)



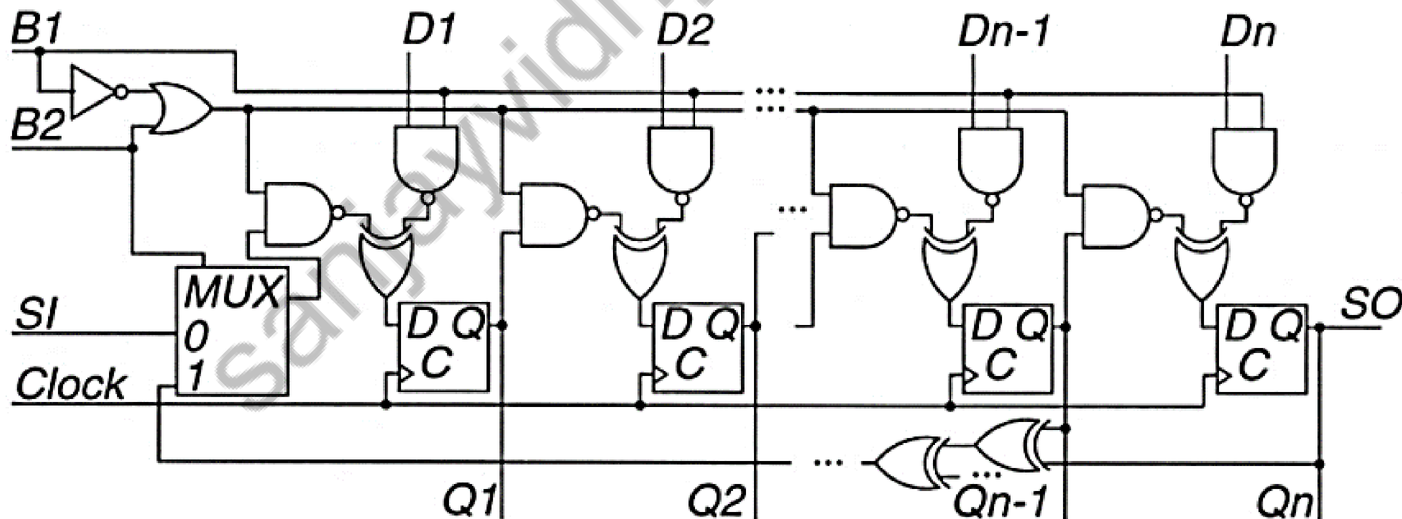
sanjayvidhyadharan.in

BILBO

A BILBO is a bank of D flip-flops in the CUT that has test hardware added to make it behave in one of four modes:

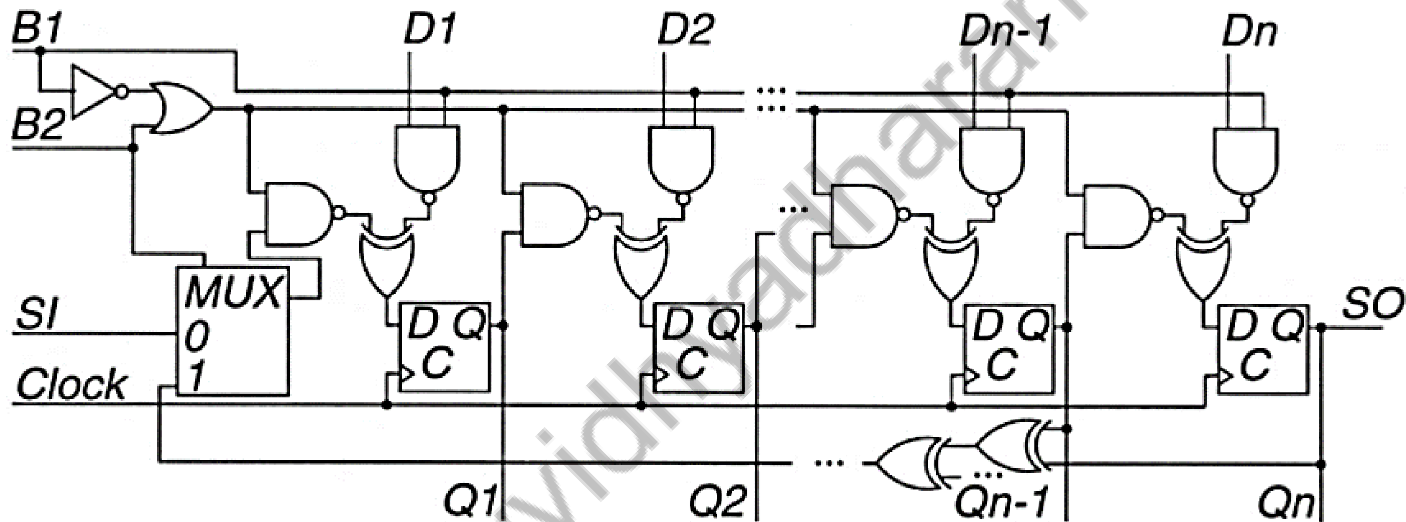
- As ordinary D flip-flops.
- As a *linear feedback shift register* (LFSR) hardware pattern generator.
- As an LFSR configured to compact a circuit response.
- As a scan chain

sanjayvidhyadharan.in



BILBO

- Built-in Logic Block Observer
  - Combine scan with PRSG & signature analysis

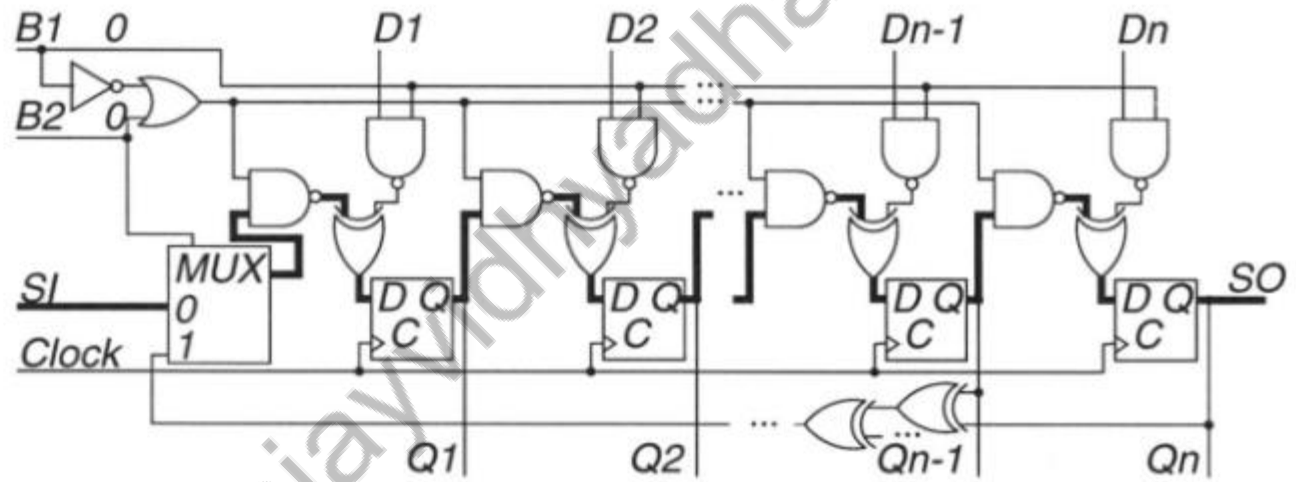


$B1$	$B2$	Mode	$B1$	$B2$	Mode
0	0	Serial scan chain	1	0	Normal D flip-flop
0	1	LFSR pattern generator	1	1	MISR response compacter

Sanjayvidhyadharan.in

BILBO

Example BILBO in serial scan mode

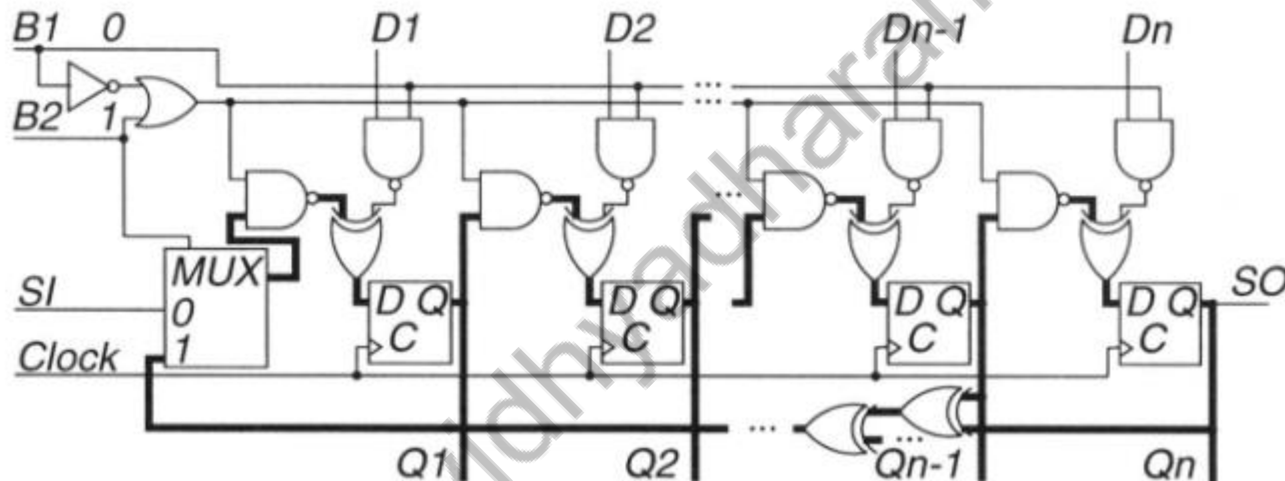


$B1$	$B2$	Mode	$B1$	$B2$	Mode
0	0	Serial scan chain	1	0	Normal D flip-flop
0	1	LFSR pattern generator	1	1	MISR response compacter

sanjayvidhyadharan.in

BILBO

Example BILBO in LFSR mode.

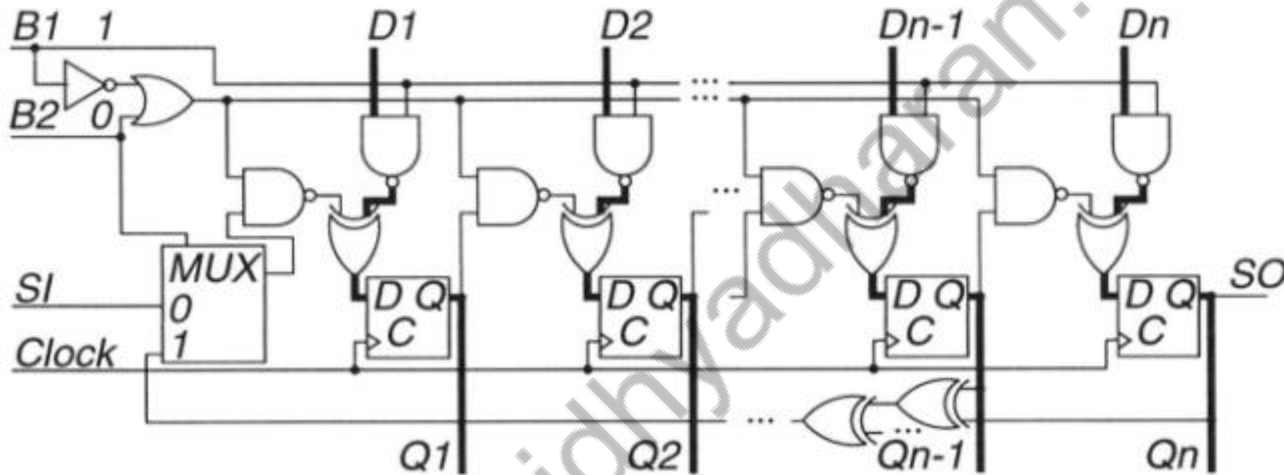


$B1$	$B2$	Mode	$B1$	$B2$	Mode
0	0	Serial scan chain	1	0	Normal D flip-flop
0	1	LFSR pattern generator	1	1	MISR response compacter

sanjayvidhyadharan.in

BILBO

Example BILBO in normal D flip-flop mode.

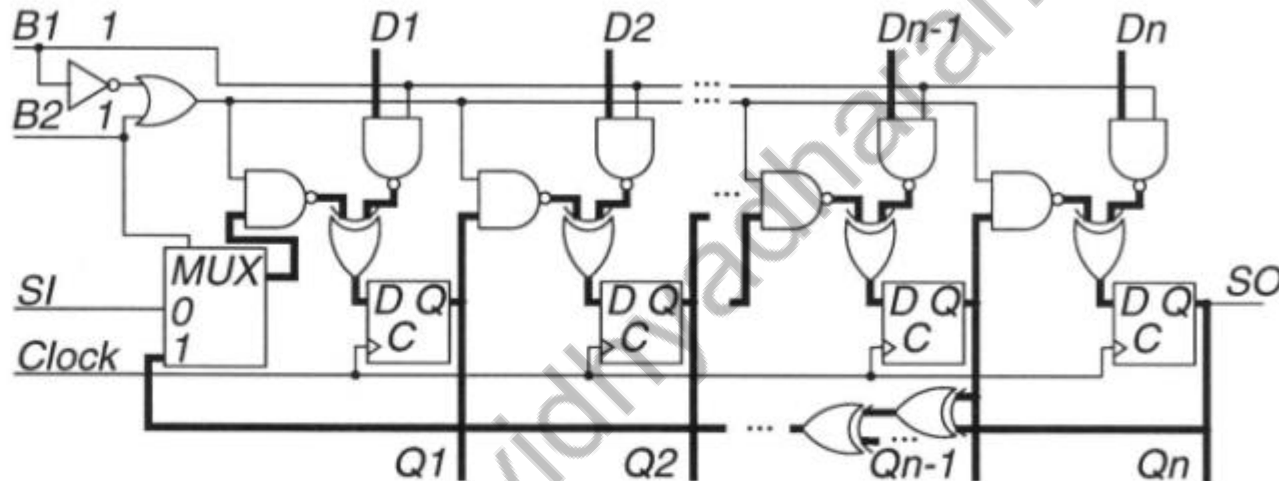


$B1$	$B2$	Mode	$B1$	$B2$	Mode
0	0	Serial scan chain	1	0	Normal D flip-flop
0	1	LFSR pattern generator	1	1	MISR response compacter

sanjayvidhyadharan.in

BILBO

Example BILBO in MISR mode.



$B1$	$B2$	Mode	$B1$	$B2$	Mode
0	0	Serial scan chain	1	0	Normal D flip-flop
0	1	LFSR pattern generator	1	1	MISR response compacter

sanjayvidhyadharan.in

Test Point Insertion

When random-logic BIST is used in a circuit, often not all of the faults are detected, either because of aliasing or because the test-pattern set from the hardware pattern generator is not rich enough to excite all faults.

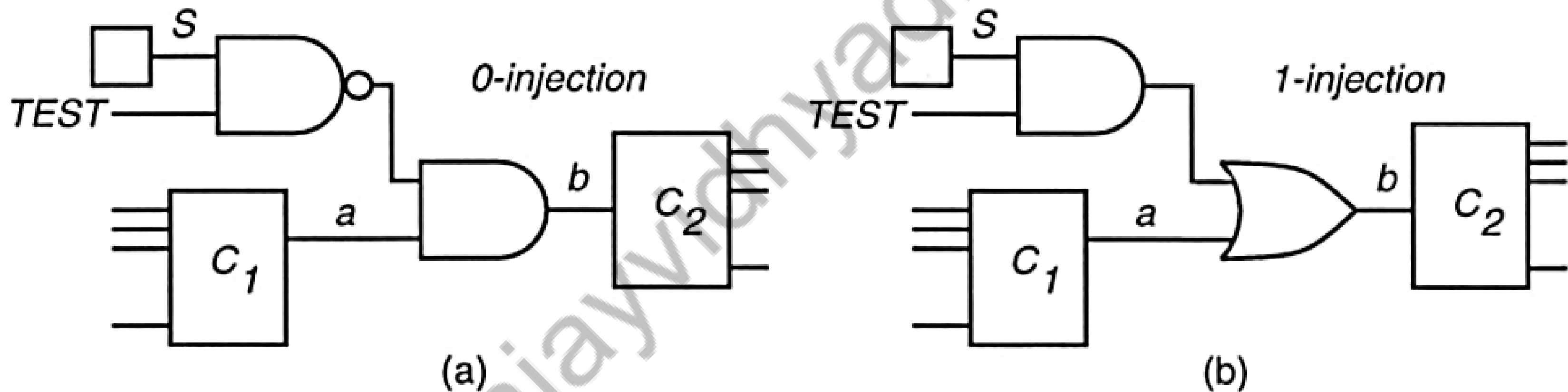


Figure 15.36: Control points to force 0 and 1.

sanjayvidhyadharan.in

Memory BIST

Most memory BIST schemes exploit the parallelism within the memory device to achieve a massive reduction in test time (and therefore cost.) This is done by a test mode where more than one memory cell is accessed with each address, usually by accessing the entire row of cells on a word line for a single read or write operation.

However, the parallel mechanism makes it difficult to test for memory coupling faults between cells in the same row, so it may not be appropriate.

Random, or pseudo-random memory BIST is not generally used , because the march tests achieve higher fault coverages with shorter pattern sequences than random or pseudo-random memory tests.

sanjayvidhyadharan.in

Memory BIST

Concurrent BIST – A memory test mechanism where the memory can be tested concurrently with normal system operation.

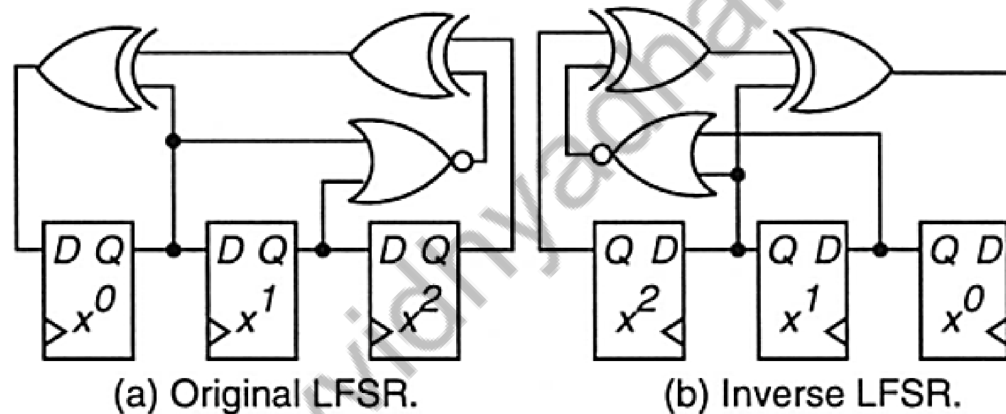
Non-Concurrent BIST – A memory test mechanism that requires interruption of the normal system function in order to perform the testing. The original memory contents are lost.

Transparent Testing – A memory test mechanism that requires interruption of the normal system function for testing. The original memory contents are preserved in the memory after testing is finished.

sanjayvidhyadharan.in

Memory BIST

Memory BIST requires an *address generator* or stepper (often an LFSR) and a *data generator.* An LFSR is better for march test BIST than a binary counter, because it uses substantially less area, and can easily be made self-testable . Furthermore, the LFSR can be adjusted to provide the all-zero pattern and the forward and exact reverse



LFSRs that count up/down in inverse order.

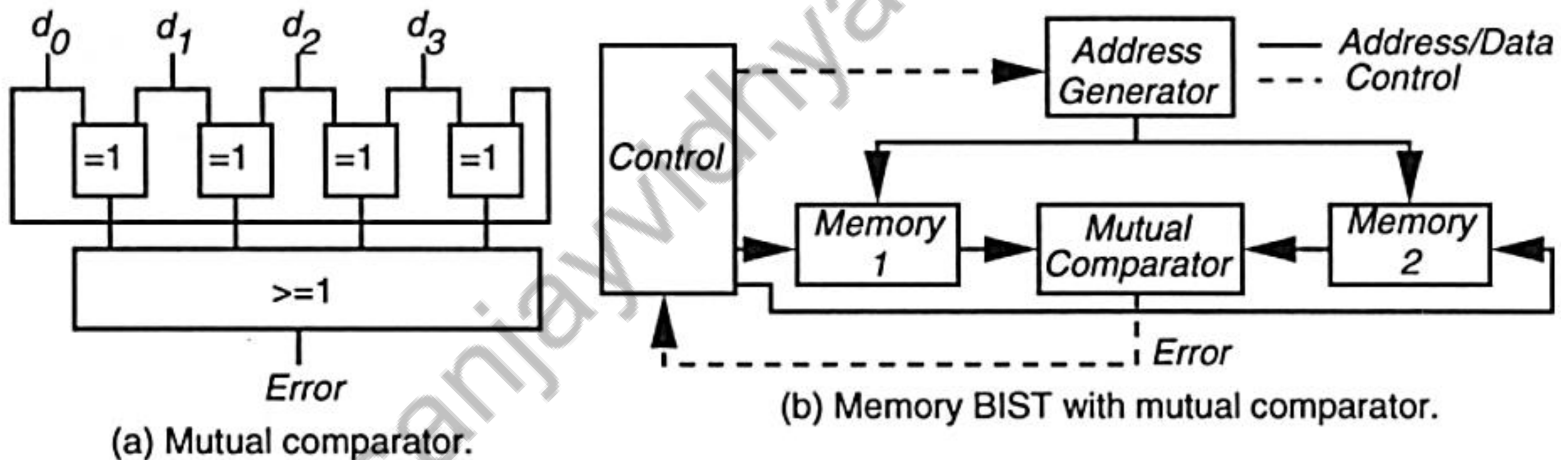
(a) LFSR generates the sequence $1 \to 0 \to 4 \to 6 \to 7 \to 3 \to 5 \to 2$ when initialized to 1.
(b) LFSR generates the sequence $1 \to 2 \to 5 \to 3 \to 7 \to 6 \to 4 \to 0$ when initialized to 1.

(c) These two LFSRs can be combined into a single LFSR, by adding a few additional logic gates.

sanjayvidhyadharan.in

Memory BIST

The *mutual comparator* is useful in memory BIST when the memory system has multiple arrays. We test two or more arrays (in this case 4) simultaneously, by applying the same test commands and addresses to all 4 arrays. The mutual comparator asserts the *Error* signal when one of through disagrees with the other data coming out of the memory arrays. The comparator eliminates the need to generate the good machine response, and implicitly assumes that only a minority of the memory array outputs are incorrect at any given time.



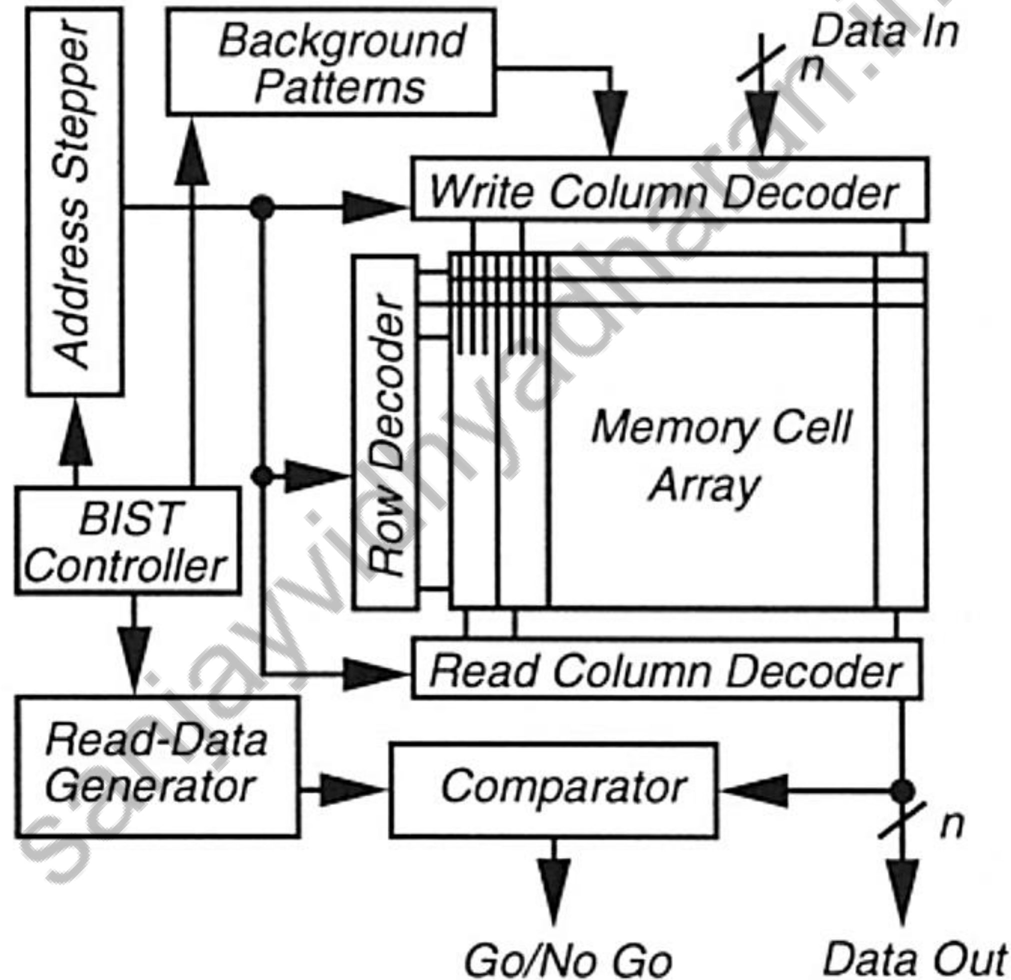
(a) Mutual comparator.

(b) Memory BIST with mutual comparator.

sanjayvidhyadharan.in

Memory BIST

March Test SRAM BIST



Parallel memory BIST

References

1. "Essentials of Electronic Testing, for Digital, Memory and Mixed-Signal VLSI Circuits", Michael L. Bushnell and Vishwani D. Agrawal,–Kluwer Academic Publishers (2000).

2. Video lectures by Professor James Chien-Mo Li
Lab. of Dependable Systems Graduate Institute of Electronics Engineering
National Taiwan University
https://www.youtube.com/watch?v=yfcoKOUV5DM&list=PLvd8d-SyI7hjk_Ci0zpTqImAtpEjdK5JF&index=1

3. NPTEL Lectures
https://www.youtube.com/watch?v=M8VEEaYwlQ&list=PLbMVogVj5nJTClnafWQ9FK2nt3cGG8kCF&index=31

sanjayvidhyadharan.in

Thankyou

sanjayvidhyadharan.in