Module 8

Testing of Embedded System

Lesson 39

Design for Testability

Instructional Objectives

After going through this lesson the student would be able to

- Explain the meaning of the term 'Design for Testability' (DFT)
- Describe some adhoc and some formal methods of incorporating DFT in a system level design
- Explain the scan-chain based method of DFT
- Highlight the advantages and disadvantages of scan-based designs and discuss alternatives

Design for Testability

1. Introduction

The embedded system is an information processing system that consists of hardware and software components. Nowadays, the number of embedded computing systems in areas such as telecommunications, automotive electronics, office automation, and military applications are steadily growing. This market expansion arises from greater memory densities as well as improvements in embeddable processor cores, intellectual-property modules, and sensing technologies. At the same time, these improvements have increased the amount of software needed to manage the hardware components, leading to a higher level of system complexity. Designers can no longer develop high-performance systems from scratch but must use sophisticated system modeling tools.

The increased complexity of embedded systems and the reduced access to internal nodes has made it not only more difficult to diagnose and locate faulty components, but also the functions of embedded components may be difficult to measure. Creating testable designs is key to developing complex hardware and/or software systems that function reliably throughout their operational life. Testability can be defined with respect to a fault. A fault is **testable** if there exists a well-specified procedure (e.g., test pattern generation, evaluation, and application) to expose it, and the procedure is implementable with a reasonable cost using current technologies. Testability of the fault therefore represents the inverse of the cost in detecting the fault. A circuit is *testable with respect to a fault set* when each and every fault in this set is testable.

Design-for-testability techniques improve the controllability and observability of internal nodes, so that embedded functions can be tested. Two basic properties determine the testability of a node: 1) **controllability**, which is a measure of the difficulty of setting internal circuit nodes to 0 or 1 by assigning values to primary inputs (PIs), and 2) **observability**, which is a measure of the difficulty of propagating a node's value to a primary output (PO) [1-3]. A node is said to be testable if it is easily controlled and observed. For sequential circuits, some have added **predictability**, which represents the ability to obtain known output values in response to given input stimuli. The factors affecting predictability include initializability, races, hazards, oscillations, etc. DFT techniques include analog test busses and scan methods. Testability can also be improved with BIST circuitry, where signal generators and analysis circuitry are implemented on chip [1, 3-4]. Without testability, design flaws may escape detection until a

product is in the hands of users; equally, operational failures may prove difficult to detect and diagnose.

Increased embedded system complexity makes thorough assessment of system integrity by testing external black-box behavior almost impossible. System complexity also complicates test equipment and procedures. Design for testability should increase a system's testability, resulting in improved quality while reducing time to market and test costs.

Traditionally, hardware designers and test engineers have focused on proving the correct manufacture of a design and on locating and repairing field failures. They have developed several highly structured and effective *so*lutions to this problem, including scan design and self test. Design verification has been a less formal task, based on the designer's skills. However, designers have found that structured design-for-test features aiding manufacture and repair can significantly simplify design verification. These features reduce verification cycles from weeks to days in some cases.

In contrast, software designers and test engineers have targeted design validation and verification. Unlike hardware, software does not break during field use. Design errors, rather than incorrect replication or wear out, cause operational bugs. Efforts have focused on improving specifications and programming styles rather than on adding explicit test facilities. For example, modular design, structured programming, formal specification, and object orientation have all proven effective in simplifying test.

Although these different approaches are effective when we can cleanly separate a design's hardware and software parts, problems arise when boundaries blur. For example, in the early design stages of a complex system, we must define system level test strategies. Yet, we may not have decided which parts to implement in hardware and which in software. In other cases, software running on general-purpose hardware may initially deliver certain functions that we subsequently move to firmware or hardware to improve performance. Designers must ensure a testable, finished design regardless of implementation decisions. Supporting hardware-software codesign' requires "cotesting" techniques, which draw hardware and software test techniques together into a cohesive whole.

2. Design for Testability Techniques

Design for testability (DFT) refers to those design techniques that make the task of subsequent testing easier. There is definitely no single methodology that solves all embedded system-testing problems. There also is no single DFT technique, which is effective for all kinds of circuits. DFT techniques can largely be divided into two categories, i.e., *ad hoc* techniques and *structured* (systematic) techniques.

DFT methods for digital circuits:

- Ad-hoc methods
- Structured methods:
  - *Scan*
  - *Partial Scan*
  - *Built-in self-test* (discussed in Lesson 34)
  - *Boundary scan* (discussed in Lesson 34)

2.1 Ad-hoc DFT methods

Good design practices learnt through experience are used as guidelines for ad-hoc DFT. Some important guidelines are given below.

Things to be followed

- Large circuits should be partitioned into smaller sub-circuits to reduce test costs. One of the most important steps in designing a testable chip is to first *partition* the chip in an appropriate way such that for each functional module there is an effective (DFT) technique to test it. Partitioning must be done at every level of the design process, from architecture to circuit, whether testing is considered or not. Partitioning can be functional (according to functional module boundaries) or physical (based on circuit topology). Partitioning can be done by using multiplexers and/or scan chains.
- Test access points must be inserted to enhance controllability & observability of the circuit. Test points include control points (CPs) and observation points (OPs). The CPs are active test points, while the OPs are passive ones. There are also test points, which are both CPs and OPs. Before exercising test through test points that are not PIs and POs, one should investigate into additional requirements on the test points raised by the use of test equipments.
- Circuits (flip-flops) must be easily initializable to enhance predictability. A power-on reset mechanism controllable from primary inputs is the most effective and widely used approach.
- Test control must be provided for difficult-to-control signals.
- Automatic Test Equipment (ATE) requirements such as pin limitation, tri-stating, timing resolution, speed, memory depth, driving capability, analog/mixed-signal support, internal/boundary scan support, etc., should be considered during the design process to avoid delay of the project and unnecessary investment on the equipments.
- Internal oscillators, PLLs and clocks should be disabled during test. To guarantee tester synchronization, internal oscillator and clock generator circuitry should be isolated during the test of the functional circuitry. The internal oscillators and clocks should also be tested separately.
- Analog and digital circuits should be kept physically separate. Analog circuit testing is very much different from digital circuit testing. Testing for analog circuits refers to real measurement, since analog signals are continuous (as opposed to discrete or logic signals in digital circuits). They require different test equipments and different test methodologies. Therefore they should be tested separately.

Things to be avoided

- Asynchronous(unclocked) logic feedback in the circuit must be avoided. A feedback in the combinational logic can give rise to oscillation for certain inputs. Since no clocking is employed, timing is continuous instead of discrete, which makes tester synchronization virtually impossible, and therefore only functional test by application board can be used.

- Monostables and self-resetting logic should be avoided. A monostable (one-shot) multivibrator produces a pulse of constant duration in response to the rising or falling transition of the trigger input. Its pulse duration is usually controlled externally by a resistor and a capacitor (with current technology, they also can be integrated on chip). One-shots are used mainly for 1) pulse shaping, 2) switch-on delays, 3) switch-off delays, 4) signal delays. Since it is not controlled by clocks, synchronization and precise duration control are very difficult, which in turn reduces testability by ATE. Counters and dividers are better candidates for delay control.
- Redundant gates must be avoided.
- High fanin/fanout combinations must be avoided as large fan-in makes the inputs of the gate difficult to observe and makes the gate output difficult to control.
- Gated clocks should be avoided. These degrade the controllability of circuit nodes.

The above guidelines are from experienced practitioners. These are not complete or universal. In fact, there are drawbacks for these methods:

- There is a lack of experts and tools.
- Test generation is often manual
- This method cannot guarantee for high fault coverage.
- It may increase design iterations.
- This is not suitable for large circuits

2.2 Scan Design Approaches for DFT

2.2.1 Objectives of Scan Design

- Scan design is implemented to provide controllability and observability of internal state variables for testing a circuit.
- It is also effective for circuit partitioning.
- A scan design with full controllability and observability turns the sequential test problem into a combinational one.

2.2.2 Scan Design Requirements

- Circuit is designed using pre-specified design rules.
- Test structure (hardware) is added to the verified design.
    - One (or more) *test control* (TC) pin at the primary input is required.
    - Flip-flops are replaced by *scan flip-flops* (SFF) and are connected so that they behave as a shift register in the test mode. The output of one SFF is connected to the input of next SFF. The input of the first flip-flop in the chain is directly connected to an input pin (denoted as SCANIn), and the output of the last flip-flop is directly connected to an output pin (denoted as SCANOUT). In this way, all the flip-flops can be loaded with a known value, and their value can be easily

accessed by shifting out the chain. Figure 39.1 shows a typical circuit after the scan insertion operation.

- Input/output of each scan shift register must be available on PI/PO.

- Combinational ATPG is used to obtain tests for all testable faults in the combinational logic.
- Shift register tests are applied and ATPG tests are converted into scan sequences for use in manufacturing test.

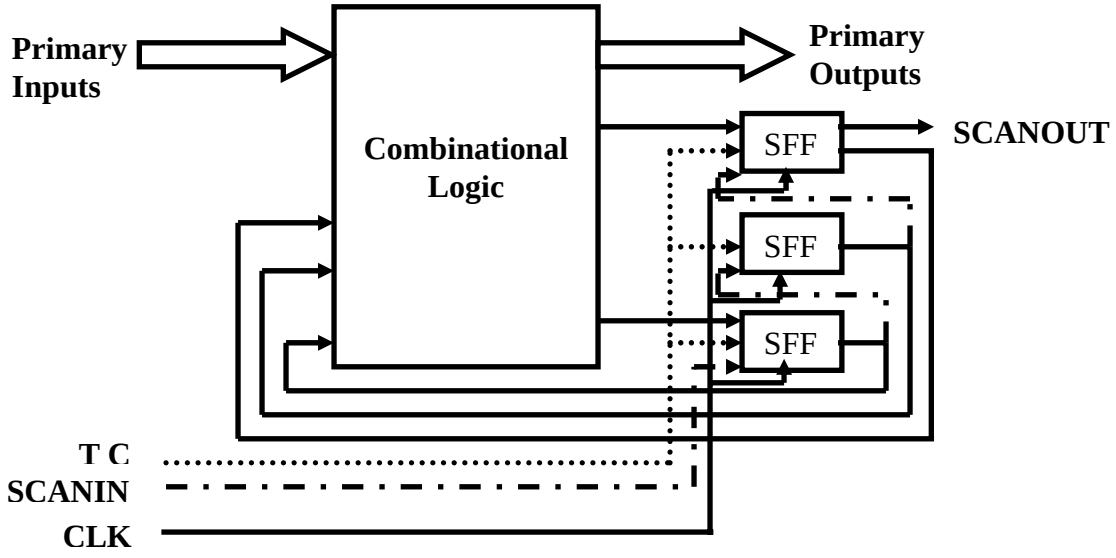


Fig. 39.1 Scan structure to a design

Fig. 39.1 shows a scan structure connected to design. The scan flip-flips (FFs) must be interconnected in a particular way. This approach effectively turns the sequential testing problem into a combinational one and can be fully tested by compact ATPG patterns. Unfortunately, there are two types of overheads associated with this technique that the designers care about very much. These are the hardware overhead (including three extra pins, multiplexers for all FFs, and extra routing area) and performance overhead (including multiplexer delay and FF delay due to extra load).

2.2.3 Scan Design Rules

- Only clocked D-type master-slave flip-flops for all state variables should be used.
- At least one PI pin must be available for test. It is better if more pins are available.
- All clock inputs to flip-flops must be controlled from primary inputs (PIs). There will be no gated clock. This is necessary for FFs to function as a scan register.
- Clocks must not feed data inputs of flip-flops. A violation of this can lead to a race condition in the normal mode.

2.2.4 Scan Overheads

The use of scan design produces two types of overheads. These are area overhead and performance overhead. The scan hardware requires extra area and slows down the signals.

- **IO pin overhead:** At least one primary pin necessary for test.
- **Area overhead:** *Gate overhead* = [4 $n_{sff}/(n_g+10n_{ff})$] x 100%, where n_g = number of combinational gates; n_{ff} = number of flip-flops; n_{sff} = number of scan flip-flops; For full scan number of scan flip-flops is equal to the number of original circuit flip-flops. Example: n_g = 100k gates, n_{ff} = 2k flip-flops, overhead = 6.7%. For more accurate estimation scan wiring and layout area must be taken into consideration.
- **Performance overhead:** The multiplexer of the scan flip-flop adds two gate-delays in combinational path. Fanouts of the flip-flops also increased by 1, which can increase the clock period.

2.3 Scan Variations

There have been many variations of scan as listed below, few of these are discussed here.

- MUXed Scan
- Scan path
- Scan-Hold Flip-Flop
- Serial scan
- Level-Sensitive Scan Design (LSSD)
- Scan set
- Random access scan

2.3.1 MUX Scan

- It was invented at Stanford in 1973 by M. Williams & Angell.
- In this approach a MUX is inserted in front of each FF to be placed in the scan chain.

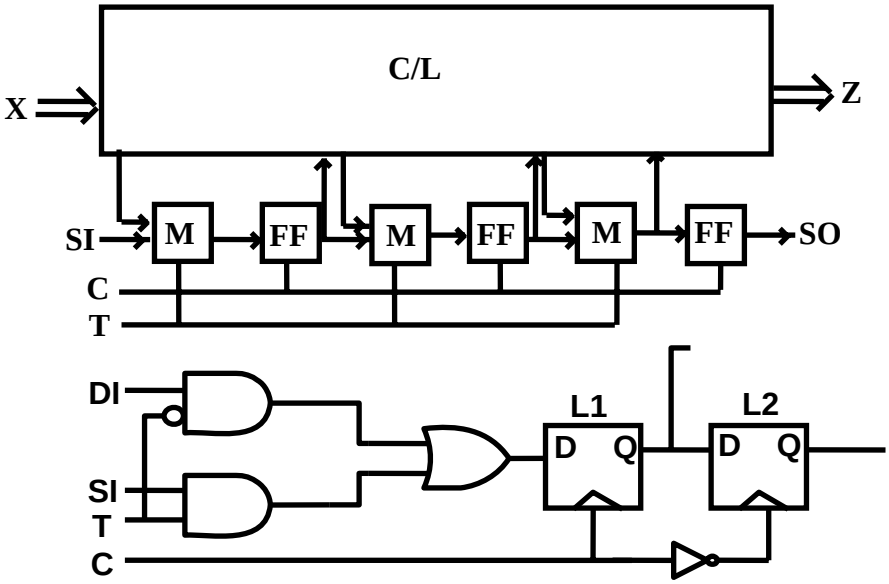


Fig. 39.2 The Shift-Register Modification approach

- Fig. 39.2 shows that when the test mode pin T=0, the circuit is in normal operation mode and when T=1, it is in test mode (or shift-register mode).
- The scan flip-flips (FFs) must be interconnected in a particular way. This approach effectively turns the sequential testing problem into a combinational one and can be fully tested by compact ATPG patterns.
- There are two types of overheads associated with this method. The hardware overhead due to three extra pins, multiplexers for all FFs, and extra routing area. The performance overhead includes multiplexer delay and FF delay due to extra load.

2.3.2 Scan Path

- This approach is also called the Clock Scan Approach.
- It was invented by Kobayashi *et al.* in 1968, and reported by Funatsu *et al.* in 1975, and adopted by NEC.
- In this approach multiplexing is done by two different clocks instead of a MUX.
- It uses two-port raceless D-FFs as shown in Figure 39.3. Each FF consists of two latches operating in a master-slave fashion, and has two clocks (C1 and C2) to control the scan input (SI) and the normal data input (DI) separately.
- The two-port raceless D-FF is controlled in the following way:
  - For normal mode operation C2 = 1 to block SI and C1 = 0 →1 to load DI.
  - For shift register test mode C1 = 1 to block DI and C2 = 0 →1 to load SI.

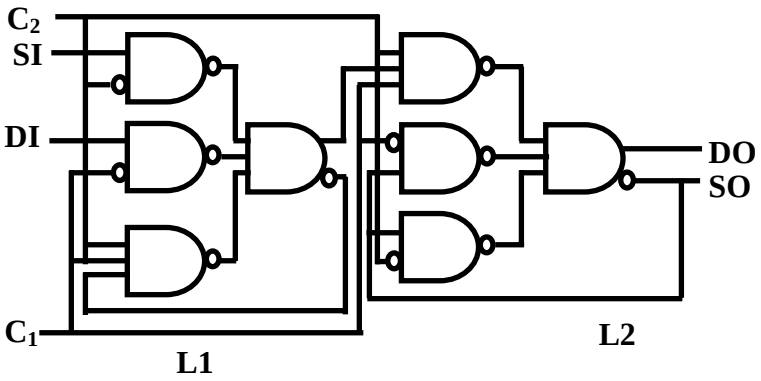


Fig. 39.3 Logic diagram of the two-port raceless D-FF

- This approach gives a lower hardware overhead (due to dense layout) and less performance penalty (due to the removal of the MUX in front of the FF) compared to the MUX Scan Approach. The real figures however depend on the circuit style and technology selected, and on the physical implementation.

2.3.3 Level-Sensitive Scan Design (LSSD)

- This approach was introduced by Eichelberger and T. Williams in 1977 and 1978.
- It is a latch-based design used at IBM.
- It guarantees race-free and hazard-free system operation as well as testing.
- It is insensitive to component timing variations such as rise time, fall time, and delay. It is faster and has a lower hardware complexity than SR modification.
- It uses two latches (one for normal operation and one for scan) and three clocks. Furthermore, to enjoy the luxury of race-free and hazard-free system operation and test, the designer has to follow a set of complicated design rules.
- A logic circuit is *level sensitive* (LS) iff the steady state response to any allowed input change is independent of the delays within the circuit. Also, the response is independent of the order in which the inputs change

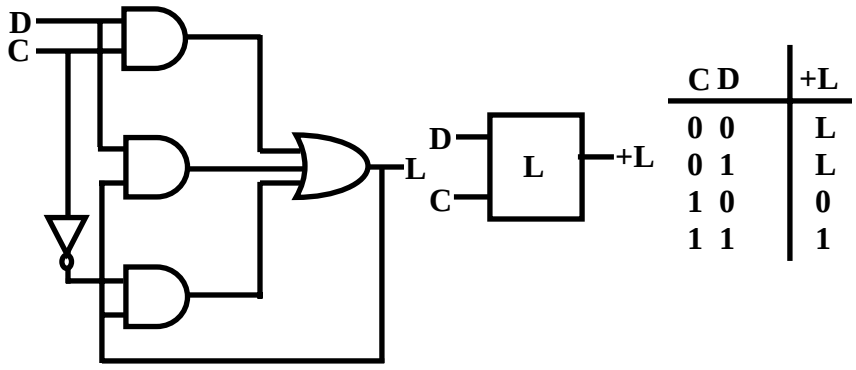


Fig. 39.4 A polarity-hold latch

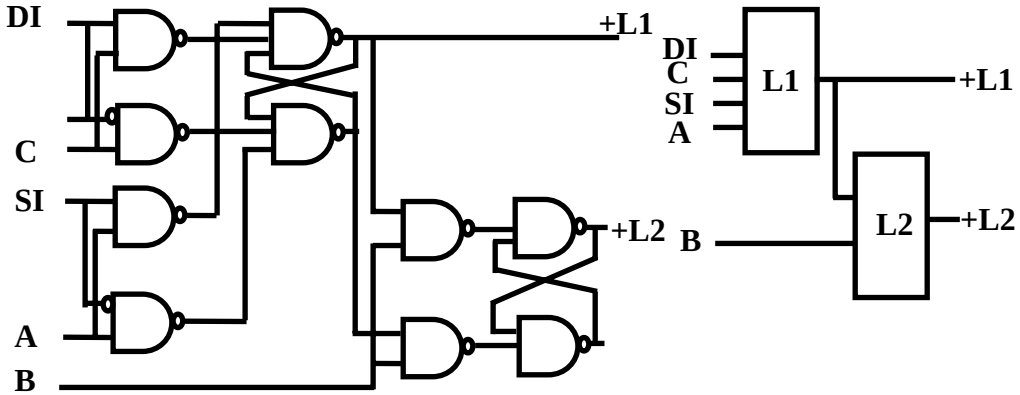


Fig. 39.5 The polarity-hold shift-register latch (SRL)

LSSD requires that the circuit be LS, so we need LS memory elements as defined above. Figure 39.4 shows an LS polarity-hold latch. The correct change of the latch output (L) is not dependent on the rise/fall time of C, but only on C being `1' for a period of time greater than or equal to data propagation and stabilization time. Figure 39.5 shows the polarity-hold shift-register latch (SRL) used in LSSD as the scan cell.

The scan cell is controlled in the following way:

- Normal mode: A=B=0, C=0 $\rightarrow$ 1.
- SR (test) mode: C=0, AB=10$\rightarrow$ 01 to shift SI through L_1 and L_2.

Advantages of LSSD

1. Correct operation independent of AC characteristics is guaranteed.
2. FSM is reduced to combinational logic as far as testing is concerned.
3. Hazards and races are eliminated, which simplifies test generation and fault simulation.

Drawbacks of LSSD

1. Complex design rules are imposed on designers. There is no freedom to vary from the overall schemes. It increases the design complexity and hardware costs (4-20% more hardware and 4 extra pins).
2. Asynchronous designs are not allowed in this approach.
3. Sequential routing of latches can introduce irregular structures.
4. Faults changing combinational function to sequential one may cause trouble, e.g., bridging and CMOS stuck-open faults.
5. Test application becomes a slow process, and normal-speed testing of the entire test sequence is impossible.
6. It is not good for memory intensive designs.

2.3.4 Random Access Scan

- This approach was developed by Fujitsu and was used by Fujitsu, Amdahl, and TI.
- It uses an address decoder. By using address decoder we can select a particular FF and either set it to any desired value or read out its value. Figure 39.6 shows a random access structure and Figure 39.7 shows the RAM cell [1,6-7].

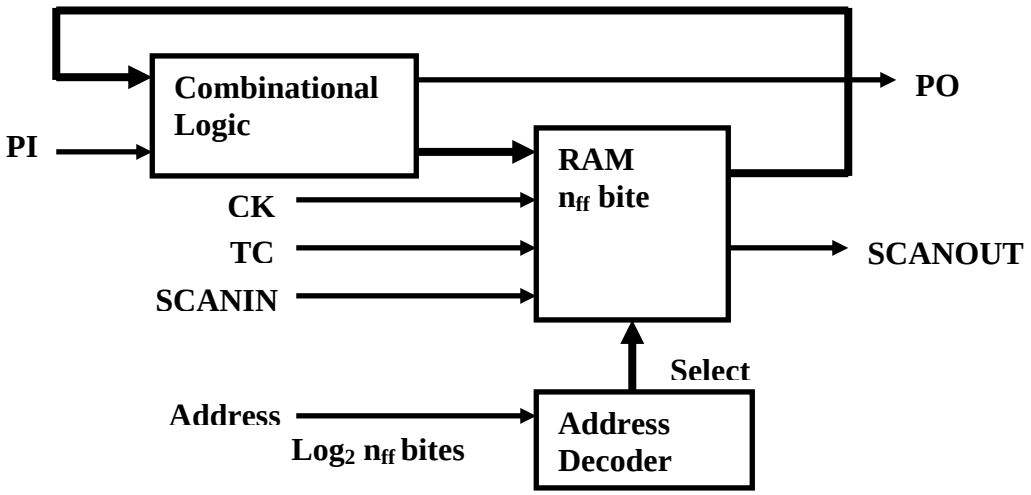


Fig. 39.6 The Random Access structure

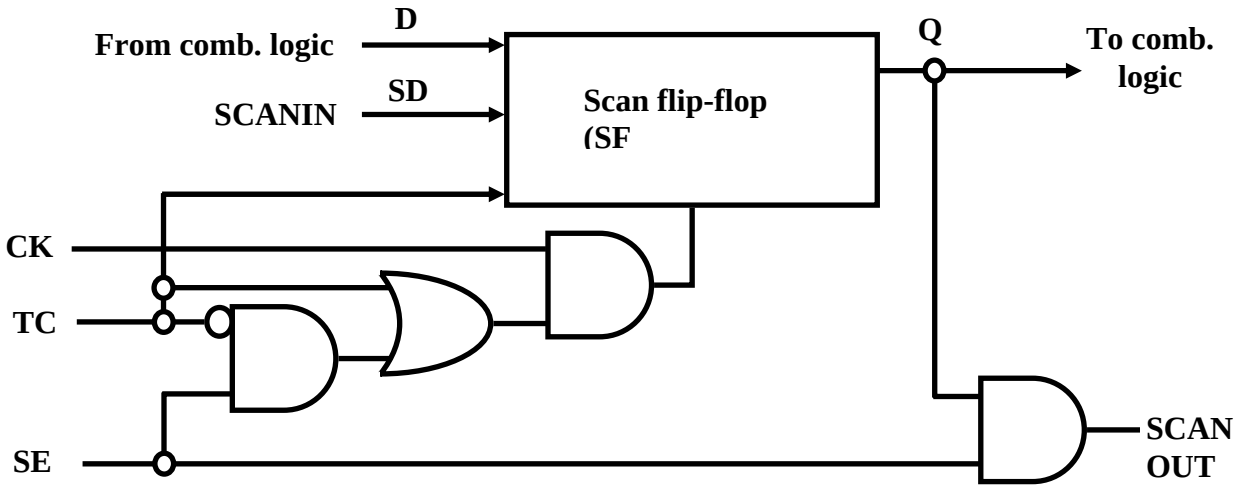


Fig. 39.7 The RAM cell

- The difference between this approach and the previous ones is that the state vector can now be accessed in a random sequence. Since neighboring patterns can be arranged so that they differ in only a few bits, and only a few response bits need to be observed, the test application time can be reduced.
- In this approach test length is reduced.
- This approach provides the ability to `watch' a node in normal operation mode, which is impossible with previous scan methods.
- This is suitable for delay and embedded memory testing.
- The major disadvantage of the approach is high hardware overhead due to address decoder, gates added to SFF, address register, extra pins and routing

2.3.5 Scan-Hold Flip-Flop

- Special type of scan flip-flop with an additional latch designed for low power testing application.
- It was proposed by DasGupta *et al* [5]. Figure 39.8 shows a hold latch cascaded with the SFF.
- The control input HOLD keeps the output steady at previous state of flip-flop.
- For HOLD = 0, the latch holds its state and for HOLD = 1, the hold latch becomes transparent.
- For normal mode operation, TC = HOLD =1 and for scan mode, TC = 1 and Hold = 0.
- Hardware overhead increases by about 30% due to extra hardware the hold latch.
- This approach reduces power dissipation and isolate asynchronous part during scan.
- It is suitable for delay test [8].

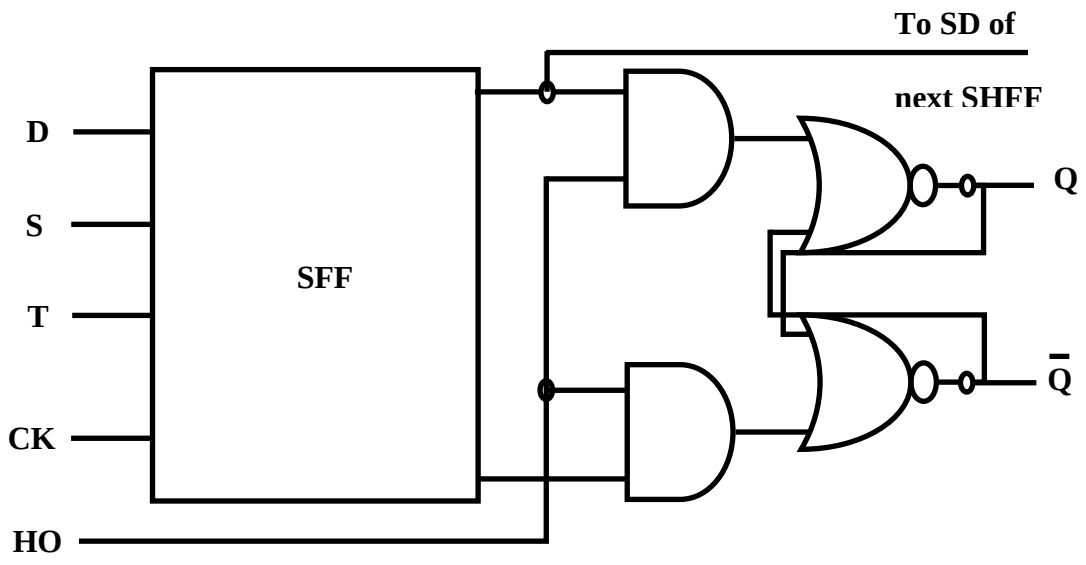


Fig. 39.8 Scan-hold flip-flop (SHFF)

Partial Scan Design

- In this approach only a subset of flip-flops is scanned. The main objectives of this approach are to minimize the area overhead and scan sequence length. It would be possible to achieve required fault coverage
- In this approach sequential ATPG is used to generate test patterns. Sequential ATPG has number of difficulties such as poor initializability, poor controllability and observability of the state variables etc. Number of gates, number of FFs and sequential depth give little idea regarding testability and presence of cycles makes testing difficult. Therefore sequential circuit must be simplified in such a way so that test generation becomes easier.
- Removal of selected flip-flops from scan improves performance and allows limited scan design rule violations.
- It also allows automation in scan flip-flop selection and test generation
- Figure 39.9 shows a design using partial scan architecture [1].
- Sequential depth is calculated as the maximum number of FFs encountered from PI line to PO line.

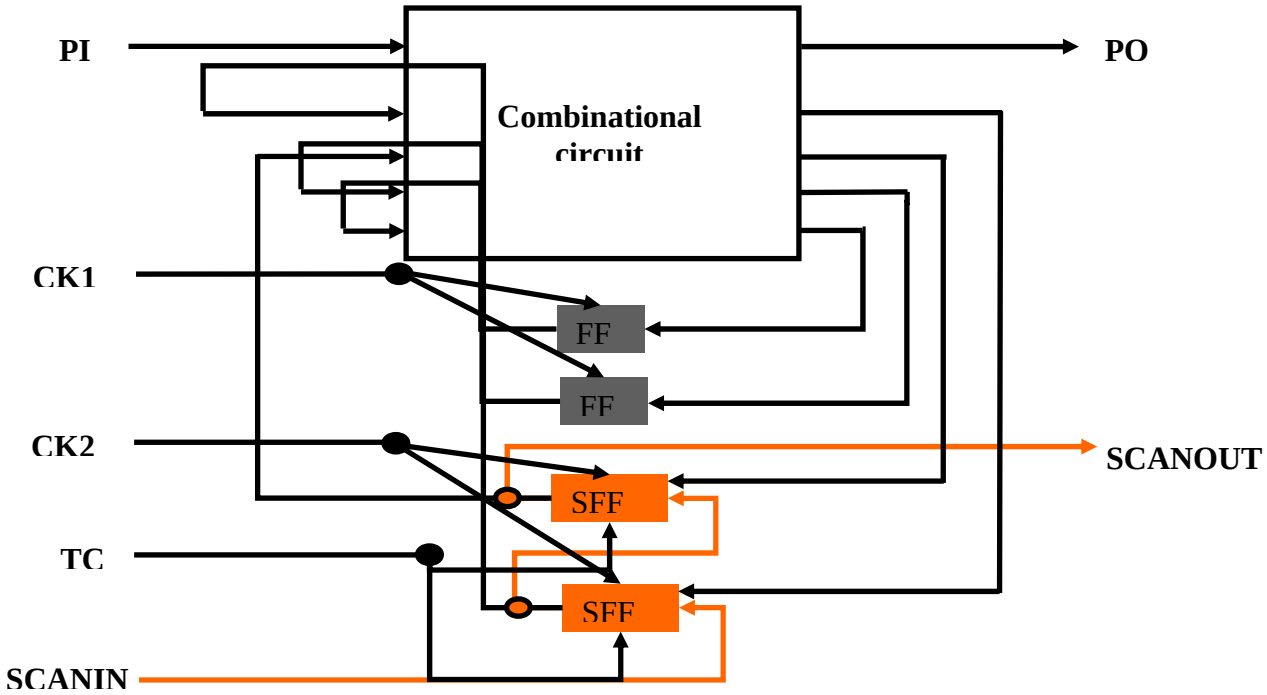


Fig. 39.9 Design using partial scan structure

Things to be followed for a partial scan method

- A minimum set of flip-flops must be selected, removal of which would eliminate all cycles.
- Break only the long cycles to keep overhead low.
- All cycles other than self-lops should be removed.

3. Conclusions

Accessibility to internal nodes in a complex circuitry is becoming a greater problem and thus it is essential that a designer must consider how the IC will be tested and extra structures will be incorporated in the design. Scan design has been the backbone of design for testability in the industry for a long time. Design automation tools are available for scan insertion into a circuit which then generate test patterns. Overhead increases due to the scan insertion in a circuit. In ASIC design 10 to 15 % scan overhead is generally accepted.

References

[1] M. L. Bushnell and V. D Agarwal, “Essentials of Electronic Testing” Kluwer academic Publishers, Norwell, MA, 2000.

[2] M. Abramovici, M.A. Breuer, and A.D. Friedman, “Digital Systems Testing and Testable Design”, *IEEE Press* 1990.

[3] V.D. Agrawal, C.R. Kime, and K.K. Saluja, “ATutorial on Built-In Self-Test, Part 1: Principles,” *IEEE Design and Test of Computers,*Vol. 10, No. 1, Mar. 1993, pp. 73-82.

[4] V.D. Agrawal, C.R. Kime, and K.K. Saluja, “ATutorial on Built-In Self-Test, Part 2: Applications,” *IEEE Design and Test of Computers*, Vol. 10, No. 2, June 1993, pp. 69-77.

[5] S. DasGupta, R. G. Walther, and T. W. Williams, “ An Enhencement to LSSD and Some Applications of LSSD in Reliability,” in Proc. Of the International Fault-Tolerant Computing Symposium.

[6] B. R. Wilkins, Testing Digital Circuits, An Introduction, Berkshire, UK: Van Nostrand Reinhold, 1986[RAM].

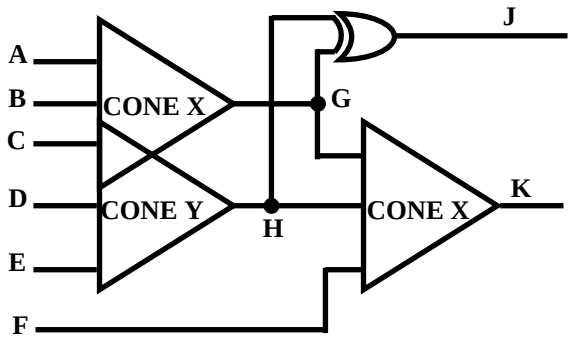
[7] T.W.Williams, editor, VLSI Testing. Amsterdam, The Netherlands: North-Holand, 1986 [RAM].

[8] A.Krstic and K-T. Cheng, Delay Fault Testing for VLSI Circuits. Boston: Kluwer Academic Publishers, 1998.

Review Questions

1. What is Design-for-Testability (DFT)? What are the different kinds of DFT techniques used for digital circuit testing?
2. What are the things that must be followed for ad-hoc testing? Describe drawbacks of ad-hoc testing.
3. Describe a full scan structure implemented in a digital design. What are the scan overheads?
4. Suppose that your chip has 100,000 gates and 2,000 flip-flops. A combinational ATPG produced 500 vectors to fully test the logic. A single scan-chain design will require about 106 clock cycles for testing. Find the scan test length if 10 scan chains are implemented. Given that the circuit has 10 PIs and 10 POs, and only one extra pin can be added for test, how much more gate overhead will be needed for the new design?
5. For a circuit with 100000 gates and 2000 flip-flops connected in a single chain, what will be the gate overhead for a scan design where scan-hold flip-flops are used?
6. Calculate the syndromes for the carry and sum outputs of a full adder cell. Determine whether there is any single stuck fault on any input for which one of the outputs is syndrome-untestable. If there is, suggest an implementation possibly with added inputs, which makes the cell syndrome-testable.
7. Describe the operation of a level-sensitive scan design implemented in a digital design. What are design rules to be followed to make the design race-free and hazard-free? What are the advantages and disadvantages of LSSD?

8. Consider the random-access scan architecture. How would you organize the test data to minimize the total test time? Describe a simple heuristic for ordering these data.
9. Make a comparison of different scan variations in terms of scan overhead.
10. Consider the combinational circuit below which has been portioned into 3 cones (two CONE X's and one CONE Y) and one Exclusive-OR gate.



For those two cones, we have the following information.

- CONE X has a structure which can be tested 100% by using the following 4 vectors and its output is also specified.

A / G	B / H	C / F	OUTPUT
0	0	1	0
0	1	1	0
1	1	0	1
1	0	0	1

- CONE Y has a structure which can be tested 100% by using the following 4 vectors and its output is also specified.

C	D	E	OUTPUT
0	0	1	0
0	1	0	1
1	0	1	1
1	1	1	0

Derive a smallest test set to test this circuit so that each partition is applied the required 4 test vectors. Also, the XOR gate should be exhaustively tested.

Fill in the blank entries below. (You may not add additional vectors).

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**J**	**K**
0	0	1	1			0			
0	1	1				0			
1	1	0	1			1			
1	0	0				1			