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*

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

Ad Hoc Testable Design Techniques

One way to increase the testability is to make nodes more accessible at some cost by physically inserting more access circuits to the original design. Listed below are some of the ad hoc testable design techniques.

Partition-and-Mux Technique :-

Since the sequence of many serial gates, functional blocks, or large circuits are difficult to test, such circuits can be partitioned and multiplexors (muxes) can be inserted such that some of the primary inputs can be fed to partitioned parts through multiplexers with accessible control signals. With this design technique, the number of accessible nodes can be increased and the number of test patterns can be reduced. A case in point would be the 32-bit counter. Dividing this counter into two 16-bit parts would reduce the testing time in principle by a factor of 2_{15}. However, circuit partitioning and addition of multiplexers may increase the chip area and circuit delay. This practice is not unique and is similar to the divide-and-conquer approach to large, complex problems. Figure 1 illustrates this method.

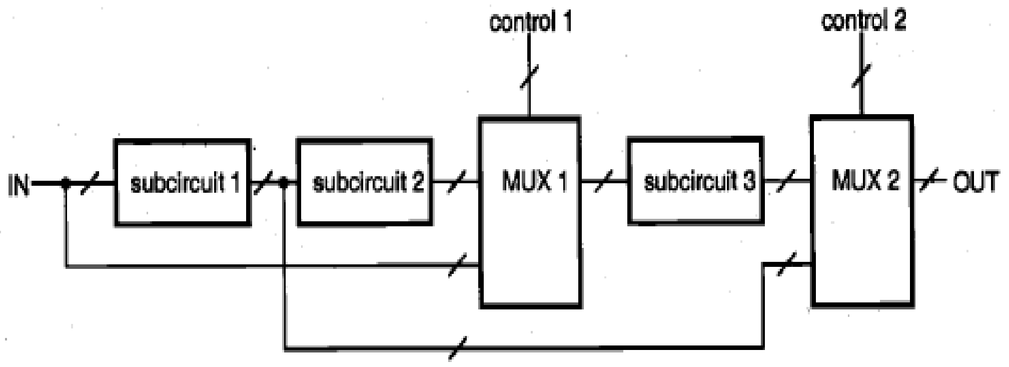

Fig 5.3.1 : Partition-and-mux method for large circuits

[Source: R.Jacob Baker, Harry W.LI., David E.Boyee, ―CMOS Circuit Design, Layout and Simulation]

Initialize Sequential Circuit :-

When the sequential circuit is powered up, its initial state can be a random, unknown state. In this case, it is not possible to start the test sequence correctly. The state of a sequential circuit can be brought to a known state through initialization. In many designs, the initialization can be easily done by connecting asynchronous preset or clear-input signals from primary or controllable inputs to flip-flops or latches.

Disable Internal Oscillators and Clocks :-

To avoid synchronization problems during testing, internal oscillators and clocks should be disabled. For example, rather than connecting the circuit directly to the on-chip oscillator, the clock signal can be ORed with a disabling signal followed by an insertion of a testing signal as shown in Fig. 2.

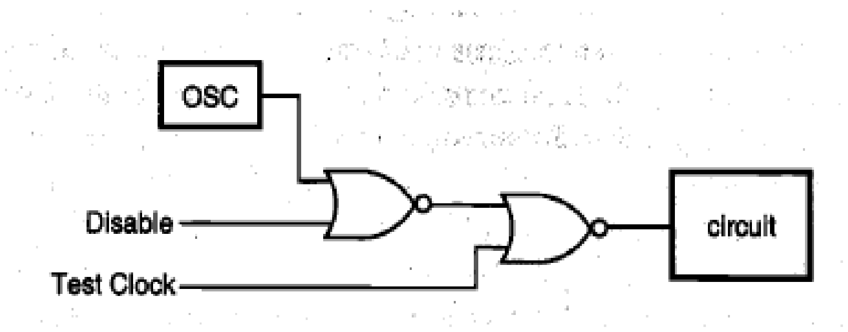

Fig 5.3.2 : Avoid synchronization problems-via disabling of the oscillator

[Source: R.Jacob Baker, Harry W.LI., David E.Boyee, —CMOS Circuit Design, Layout and Simulation]

Avoid Asynchronous Logic and Redundant Logic :-

The enhancement of testability requires serious tradeoffs. The speed of an asynchronous logic circuit can be faster than that of the synchronous logic circuit counterpart. However, the design and test of an asynchronous logic circuit are more difficult than for a synchronous logic circuit, and its state transition times are difficult to predict. Also, the operation of an asynchronous logic circuit is sensitive to input test patterns, often causing race problems and hazards of having momentary signal values opposite to the expected values. Sometimes, designed-in logic redundancy is used to mask a static hazard condition for reliability. However, the redundant node cannot be observed since the primary output value cannot be made dependent on the value of the redundant node. Hence, certain faults on the redundant node cannot be tested or detected. Figure 3 shows that the bottom NAND2 gate is redundant and the stuck-at- fault on its output line cannot be detected. If a fault is undetectable, the associated line or gate can be removed without changing the logic function.

$$
\begin{aligned}
F &= AB + BC + \bar{A}C \\
&= AB + \bar{A}C
\end{aligned}
$$

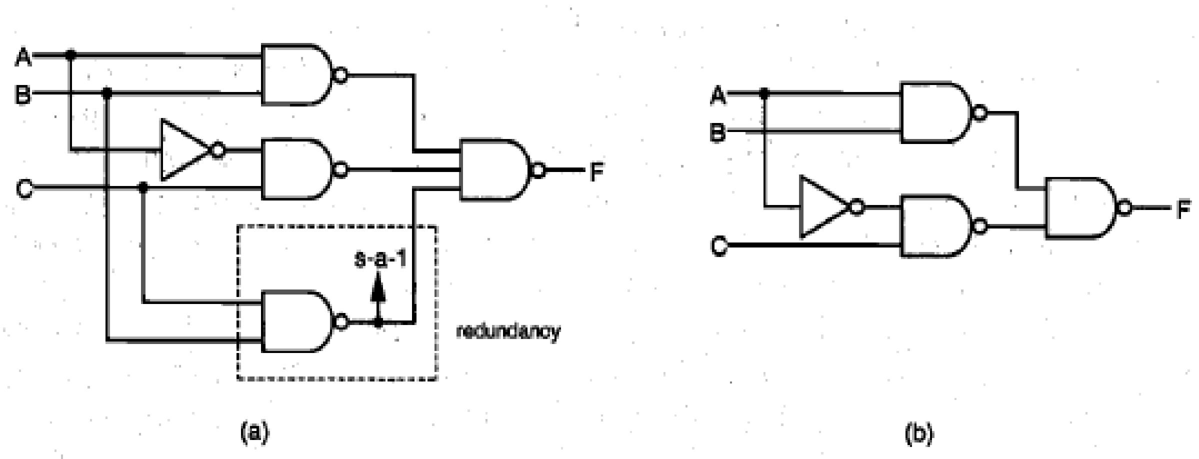

Figure 5.3.3 : (a) A redundant logic gate example. (b) Equivalent gate with redundancy remove

[Source: R.Jacob Baker, Harry W.LI., David E.Boyee, ―CMOS Circuit Design, Layout and Simulation]