Design for Testability (DFT) Using SCAN

By

P.Radhakrishnan,
Senior ASIC-Core Development Engineer,
Toshiba,
1060, Rincon Circle,
San Jose, CA 95132 (USA)

Sep 1999 (Issue-2)

Contents

Introduction 3

Test Design 3

Manufacturing Defect 3

Detecting Stuck-at Faults 5

The Idea of SCAN 5

SCAN Styles 5

Scan Methodology 6

Shift-In 7
Parallel Capture 8
Shift-Out 8
Methodology Summary 8

Fault Coverage 9

Design Approach 9

Asynchronous Resets 10
Embedded Memories 11
Other Issues 11
Timing 11
Chain Length 11
Multiple Clocks 12

DFT Tools 12

Conclusion 12

 Copyright © 1999 by P.Radhakrishnan. All rights reserved

Introduction

In this white paper I am going to address one of the important elements in ASIC design flow. Though we have not discussed anything about the ASIC design flow itself, this topic is almost a parallel concept that enhances many characteristics of a design. This paper is about the "Design for Testabilty" using a most widely used technique called scan chains. We will learn more about this technique in this paper, and by the time you read the conclusion part, you will be familiar with the principles of scan in a chip, the advantages of it and atleast about one of the commonly used scan techniques. So, for time being pretend that you know the ASIC design flow (I'm planning to write about it in the next issue), as we are going to take a close look at the "Test Design" step in the flow. This is the step where we design the scan chain in the chip to improve many things in it.

Test Design

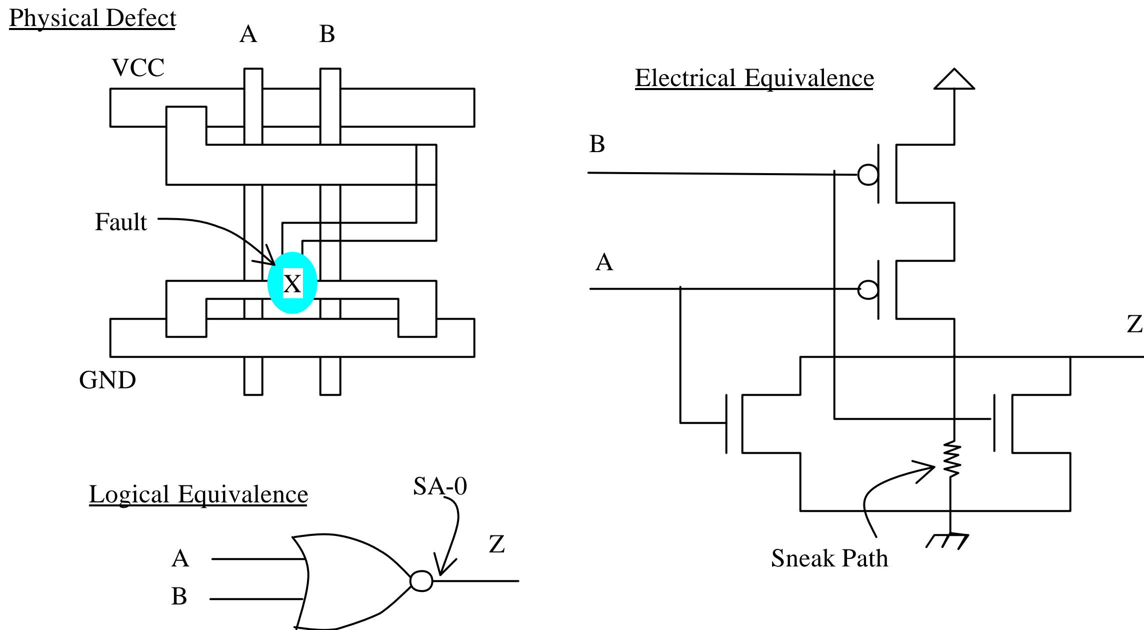
What is test design? Why does one has to do that? To understand this, let us assume that we are designing a chip of your favourite function and that the size of the chip is about 200,000 equivalent logic gates. (This number 200K is known as the gate count of the chip.) After designing the whole chip we will be running the STA (remember this?) to see if the circuit is meeting all the timing requirements. We would also run the functional tests using various test vectors to test the functionality of the device. Once this is done, we are almost sure (except for the one damn corner test case, which we did not think of!) that the chip will function correctly in the field in all the operating conditions. This is the time when you deliver the entire database to the semiconductor vendor who has the manufacturing house to fabricate the chip. When the fabricated chip is soldered on a board in the system, it is expected to do the required function. Now, assume that all the other elements in the board except our device are checked to be correct by some means. Also assume that the board is malfunctioning! (how sad?). Which element would you suspect? Obviously our device must be the culprit! We checked our device thoroughly using STA and using hundreds of test cases, yet it is causing problems! What could have gone wrong and in which step? Manufacturing is a major step in the flow and there is good possibility that something had gone wrong in that process. If so, how do we make sure that the component we get from the foundry is a good one? Here is where the test design plays a major role. Test design process introduces some additional circuit in the design using which one can check the device after manufacturing. This testing is done immediately after the manufacturing of the wafer and also after the die is packaged. Each device in the wafer will be tested in a tester with a set of test vectors that will reveal the health of the device when it is in the wafer. These special test vectors are also generated in the test design phase of the ASIC design flow. By doing this, the probability of shipping a faulty chip to the customer site is minimized. Could you imagine what would happen if a faulty device reaches the customer? (Remember the Pentium bug a few years ago? That was a functional defect, though!)

Manufacturing Defect

These defects are caused during the fabrication of the chip. These could be the result of a poor processing. What are the types of defects you can think of in an electrical circuit? It could be either an open circuit or a short circuit! Similarly in a semiconductor device also there could be shorts or opens between two points, which may cause malfunctioning of devices. These shorts and opens in the transistors manifest as stuck-at-one (ST-1) or stuck-at-zero (ST-0) faults in a gate or a flop in the chip. In addition to these, there can be other effects like a slow transition of the output of a device. These defects cause malfunctioning of the device. As I said above, the defect is caused by poor process, which may be due to Silicon (material) defect, or photolithography defect, or mask contamination or process variation. The following

Copyright © 1999 by P.Radhakrishnan. All rights reserved

diagram shows the physical defect, the electrical equivalence and effect on the gate's function.



The low resistance path cause by the short in the above device makes the output of the gate to be always at "LOW" level, no matter what the input condition is. This is called a stuck-at-zero fault. Similarly there may be a point which is stuck at "HIGH" and never switches to "LOW". This kind of fault is called stuck-at-one fault. The presence of these defects in the chip will cause malfunctioning of the device.

If these kinds of faulty devices are to be found out before it goes into the shipment, there should be a way to test the devices just after manufacturing. Semiconductor companies use testing equipments called "tester" in the plant to test the devices. The silicon wafers (a wafer will contain many devices processed in them) go into the tester. In the tester, a test program will be run at the end of which we can know how many of the devices in that wafer are healthy. The next step is to package those healthy parts into an appropriate package. During this process also something may go wrong. The testers are used on the packaged devices also. Foundries usually keep a record of the wafer yield and the package yeild to measure the quality of their process line. When such healthy devices are shipped to the cutomer, the yeild from the foundary goes high.

Now you know why testing is needed at the manufacturing level. Lets us see how one can do this. When the device is in the die (on the wafer) the IO pads (these are the elements that are connected to the I/O pin of the device through metal bonding) are exposed outside. The tester will have probing pins using which you can apply a signal at the input pins and observe through the output pins. But these stimuli and strobing has to be done in a planned manner so that you will know what to expect from the output pins at a particular instant of time.

 Copyright © 1999 by P.Radhakrishnan. All rights reserved

Detecting Stuck-at Faults

Let us spend a little time in discussing how to find out a stuck at fault using input vectors. Assume that in a big ASIC, a three-input AND gate in one portion of the logic is stuck at zero due to one of the above manufacturing problems. Because of this, the AND gate will not be able to switch to the "HIGH" value and this will cause problem to all the nodes where this AND gate is feeding in. To make sure that there is a SA-0 fault in this AND gate, all we need to do is to apply a set of input signals at the three inputs of the AND gate that will make the output to "HIGH". If the gate were functioning correctly, the output for this set of input signals should be a "one". Otherwise it is faulty! If by some means you can control all the three pins of the AND gate and observe the output of this AND gate using the primary pins of the chip, we can find this SA-0 fault very easily.

The Idea of SCAN

Scan is the idea using which one can control to the inputs of the various gates and flip-flops inside the chip and also observe the outputs from the internal flops in a pre-planned manner. In a chip of the size we have considered, (200K) there will be many flops to generate various functions. You may have about ten thousand flops (I just picked a number) in it. Each of these flops will have a logical cone at the D input and these logical cones might have been built using the primary inputs or from the outputs of other flops or from a mixture of both. Scan works in the following manner. The chip will be initially taken into a special mode, called "scan mode" by setting an input pin to an appropriate value. In this mode all these ten thousand flops are hooked up as a giant shift register, in which the output Q of one flop is connected to the input D of the next flop, whose Q output is connect to the D input of another flop and so on. One of the primary input pins will feed the input of the first flop in this chain in the scan mode. Similarly the output of the very last flop will be taken to a primary output. Using this simple (you will realise that it is not so simple!) setup we can control all the internal nodes and apply a "1" or "0", and observe the effect through the output pin. Don't worry a lot at this point how this is done. You will understand as we go along.

There are some terminologies I wish to introduce at this point. We use the pin that is connected to the input of the first flop in the chain to send a series of patterns (1's and 0's) into the chain. Let us call this pin as the "scan_in" pin. Similarly the output pin where the shifted pattern comes out is called the "scan_out". The input pin that we use to enable the scan mode can be called as "scan_enable". There will be one input pin using which we should apply the clock signal to do the shifting of patterns through the giant shift register. We will call that input pin as "clk". With these four pins we can do amazing things to improve the testability of a design.

SCAN Styles

There are three types of scan styles that are commonly talked about in the industry. They are,

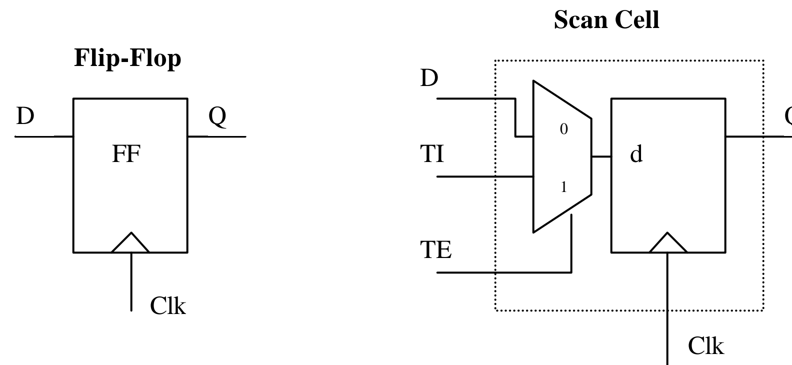
1. Multiplexed Scan Style
2. Dual Clock Scan Style and
3. LSSD (Level Sensitive Scan Design) style

Eventhough all the above three methods achieve the same goal, there are preferences among the designers and foundries to follow one of the above three methods. The most commonly used method is the first one (MUXScan style) and this is a simpler method too. Most of the tools and ASIC vendors support this method of scan design. The second method is also not a very difficult one. The difference here is that there will be a separate clock for the scan mode,

Copyright © 1999 by P.Radhakrishnan. All rights reserved

apart from the normal functional mode clock. These two methods are more suitable for flip-flop based designs. The third method is more suitable for latch based designs and the ASIC vendor has to support this scan design using special types of cells particularly designed for this method. These scan cells occupy almost double the size of the normal flop cell and hence there is a huge area impact on the chip. This method is not very widely used. IBM supports this method in their design flow due to some special reasons.

In this document we will be discussing only about the most commonly used scan design, the muxscan design. In this flow, each of the flops in the logic will be replaced by an equivalent scan cell. The scan cell differs from the normal flop by the fact that it has a multiplexer in front of the input pin "D". The figure below shows the normal flop and an equivalent scan flop.

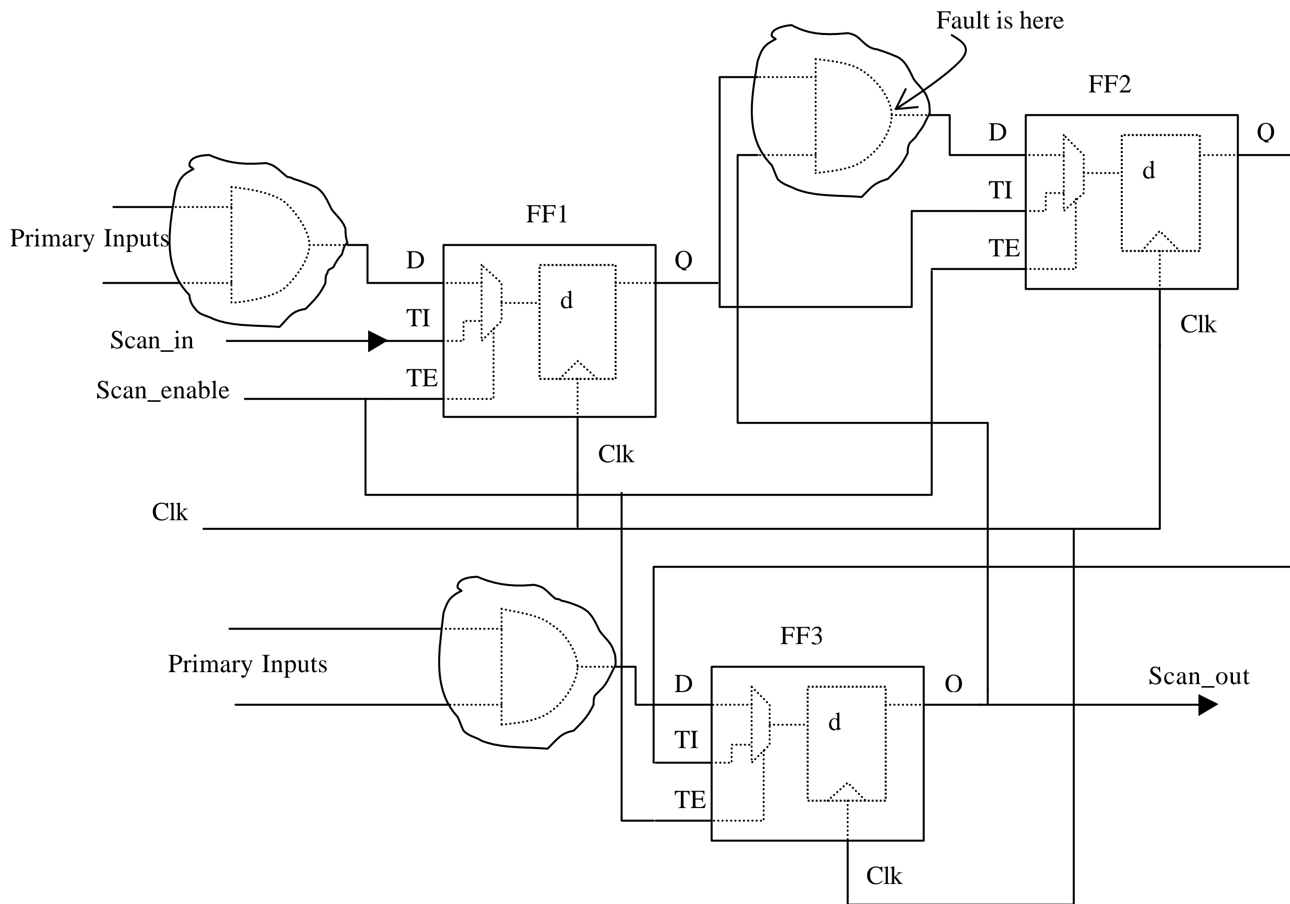


The multiplexer has two inputs and a select line. During the normal mode the "TE" (test enable) pin of the scan flop will be "zero" and the MUX selects the normal mode input to the "D" pin of the actual flop. During the test mode, the "TE" pin will be driven to "one" for applying the input vectors. When the "TE" pin is "one" the test input (TI) is selected by the MUX and applied to the input pin "D" of the flop. With this arrangement of the flop with a multiplexer, we can hook up a scan chain to test various portions of the die.

Scan Methodology

In this section let us see how the scan works to achieve a high fault finding capability. Let us consider the circuit shown in the following figure to demonstrate how the scan chain functions. I have shown only three flops in this circuit for simplicity. There are seven primary inputs (including the scan control and clock pins) in the circuit. During the normal mode, the "scan_enable" pin of the chip will be "zero". So the muxes in the scan cells will select the output from the combinatorial cloud and apply at the "**d**" input of the flops. The logic can be taken into the test mode by making the "scan_enable" pin to "one". During this phase, the "TI" input of the cell is selected by the mux and applied to the input of the flop. So the outputs from the combinatorial clouds are isolated. Thus in this mode, the first flops gets its input from the "scan_in" pin and the output of this flop is connected to the input of the next flop "FF2". Since this flop is also in the scan mode, the "TI" pin gets connected to the input of flop. Similarly the output of the second flop gets connected to the input of the third flop. The output of the third flop is brought out as the "scan_out" where the signals are monitored for expected results.

 Copyright © 1999 by P.Radhakrishnan. All rights reserved



The whole scan testing takes place in three phases. These three phases are explained in the section below.

Shift-In

How do you think such an arrangement of three flops in a chain helps to find the SA-1 or ST-0 faults in a circuit? Again for simplicity reasons, let us assume a two-input AND gate as the equivalent of the combinatorial cloud connecting the flops. Let us assume that the cloud feeding the flop-2 is having a SA-0 fault. The method to isolate this SA-0 fault is to apply "11" at the input of the AND gate. But in this circuit both the input pins of the AND gate are fed from internal logic. We do not have direct access to these pins using the primary inputs. This means we do not have direct controlability to these input pins. However there is a way to control these inputs of that AND gate. In this circuit the two input pins of the AND gate are fed from the output of the flop-1 and flop-3 respectively. So, by some means if we can make the outputs of these two flops to a "one", we have applied the input vector that detects the SA-0 fault at the output of the AND gate.

Applying the fault detecting test vectors to the node in the design is the first step in the scan cycle. In our case, if the circuit is in the scan mode (this means the "scan_enable" pin is set to "1") and if we shift a pattern of "101" using the "clk", at the end of three clocks we will have "1" at the output of FF1, "0" at FF2 and "1" at FF3. After these three shifts we have accomplished applying the test vector that makes our troublesome AND gate's output to a "1". This phase of the scan cycle is called "shift-in". So, in the shift-in phase the input vector that will detect a fault will be applied by a series of shifting of a particular pattern. At this time do not wonder how these patterns are identified. There are advanced tools available in the market, that are smart enough to find the patterns to detect SA faults in all the nodes of the circuit! The ability to apply a set of inputs so as to force a node to "1" or a "0 is called

Copyright © 1999 by P.Radhakrishnan. All rights reserved

controlability. So, when we say that we have good controlability to a combinatorial could, it means that we can make any node inside that combinatorial cloud to a “1” or “0”. This is very essential for the scan logic to function. Many people call this phase as “Scan Shift” also.

Parallel Capture

The second phase in the scan cycle is called “parallel-capture”. After successfully shifting in the fault detecting pattern we expect a particular value at the output of the logic that is under investigation. In our case, we expect a “1” from the AND gate if it is functioning properly. We should find out a way to access the output of the AND gate to see if it is actually a “1”. Since this node is internal to the chip we have no direct access to this point. But we have the flops and the scan chain in place. After the proper value has been shifted into the chip using the scan chain, the circuit is taken into the parallel capture phase by making the “scan_enable” pin to “0”. This is equivalent to the normal mode of operation. So, during this period, the “D” input of the flop gets its input from the output of the combinatorial cloud. In our case, the “D” input of the flop FF2 gets its input from the output of the AND gate we are trying to observe. If we switch the clock input once to apply an active edge to the flop, we can capture the value in the input “D” of the flop to the output “Q” of the flop. All we have done in this phase is

1. made the “scan_enable” pin to “0” and
2. applied one clock pulse at the “clk” pin

Many people in the industry used to mention this phase as “parallel measure and capture” as it really makes sense to call it that way. I am purposely avoiding a tiny bit of detail about the parallel measure at this point.

Shift-Out

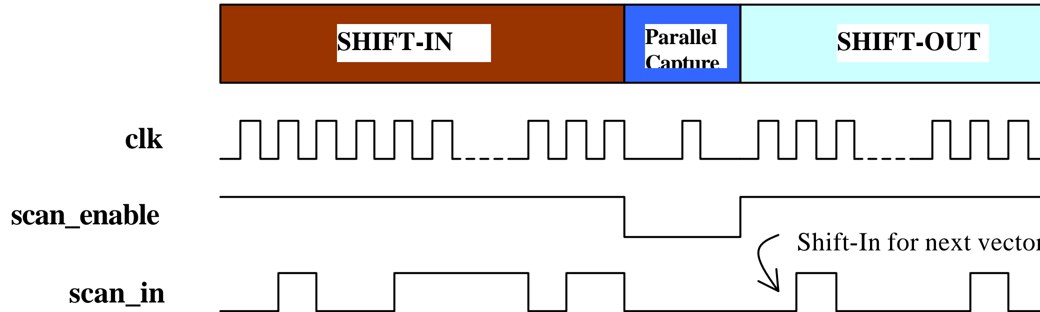
This is the third phase in the scan mode operations. Since we have already captured the value of the output of AND gate into the “Q” pin of the flop it is enough if we device a method to observe the output of this flop (FF2) instead of the AND gate’s output directly. (Notice that the output of this flop is also not directly accessible from any of the primary output pins of the chip). This can be done using the scan chain again. All we need to do is to go to the shift register mode and keep on shifting the bits by applying clock pulses, until the output from FF2 pops out of the “scan_out” pin. The following steps accomplish this. Take the circuit back into the scan mode by applying a “1” to the “scan_enable” pin. Then apply one more clock pulse to shift the value captured in the “Q” pin of FF2 into the “Q” pin of FF3. (In this case it so happened that just one shifting brought the observable value to the output pin). Since the output of FF3 is directly connected to the primary output pin, we can observe the output value straight away.

Methodology Summary

Let us summarize all the points we mentioned in the above three sections. Scan operation takes place in three phases. In the first phase (Shift-in) a sequence of patterns of 1’s and 0’s is applied through the “scan_in” pin using a determined number of clock cycles. This ensures that a particular pattern of inputs is applied to the internal gates to make its output to a known state. The Automatic Test Pattern Generation (ATPG) tools will analyze the entire circuit and sequence the appropriate test patterns with the exact number of clocks to reach a particular node inside the logic cloud. Once this is accomplished, in the second phase of scan operation, the value in the node that is being tested is captured into the following flop using one clock pulse. Remember that the “scan_enable” pin has to be set to “0” to make the normal mode input to reach the “D” inputs of the flop. After this step the chip enters into the scan mode again and a definite number of clock pulses are applied to shift the value in the “Q” pin of the flop to a primary output pin , which is the “scan_out” pin. The number of clock pulses that are to be applied for shifting the value depends on how deep the capturing flop lies in the circuit.

Copyright © 1999 by P.Radhakrishnan. All rights reserved

This is also determined by the ATPG tool and at the appropriate clock pulse the output from the "scan_out" pin is compared against an expected value. If it does not matche with the expected value, then we know that there is some problem in the element that drives the node under test. The whole cycle is repeated for each and every node where a fault can occur.



In a realistic design there may be thousands of flops connected into a huge scan chain. Assume that there are 10 thousand flops in a design. During the shift-in phase, we may end up in shifting patterns of 10K bits long using 10K clock pulses (this will be the worst case scenario). Similarly after the "parallel capture" is completed, we may have to apply about 10K clock pulses to shift the captured value out of the chip for doing a comparison. This whole process has to be repeated for all the internal nodes that are controlable using the scan chain. This will be a time consuming process and the time spent on the tester is very expensive as these testers are extremely costly. The ATPG tools will conserve some time by shifting in the next pattern while the shift out of the previous cycle happens.

Fault Coverage

The fault coverage of a chip is expressed in percentage. This number is the ratio of the number of nodes that are tested for SA faults against the total number of faults that can be detected in the chip. A high value of fault coverage like 97% is a good sign for the semiconductor vendor. This is because when the device is in the wafer and also after the packaging, the health of device can be determined to a greater accuracy. Most of the foundries have it as a policy to go for high fault coverage, close to 95%. This number is used as an indicator of the quality of the parts delivered from a foundry.

From what we discussed so far in this paper, to achieve high fault coverage using the scan methodology, we need to have very good controlability and also observability of various nodes inside the chip. This will become possible only if the designer thinks about the problems that can reduce the controlling and observing capabilities while he designs the chip. Design practices that are un-friendly to scan can lead to drastic reduction of the fault coverage. We will see some of these issues in the following sections.

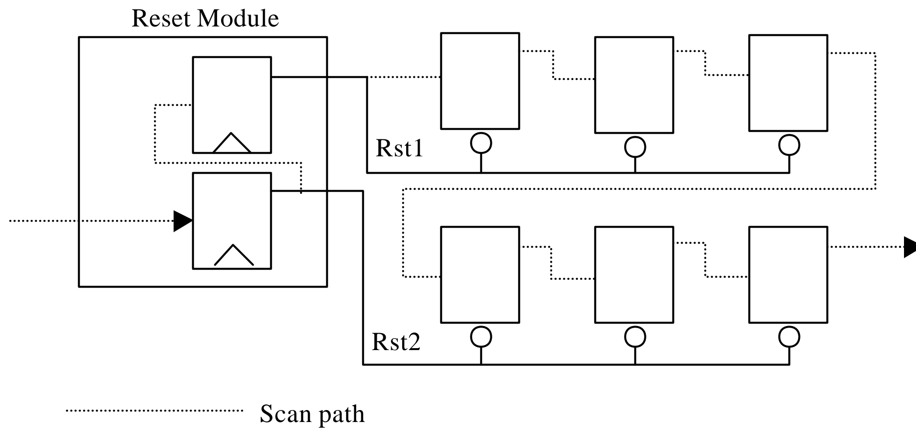
Design Approach

There are many points a designer has to bare in mind while he/she is doing a chip to make it friendly to DFT (Design For Testability) techniques. Let us take some of these issues and discuss to some extent.

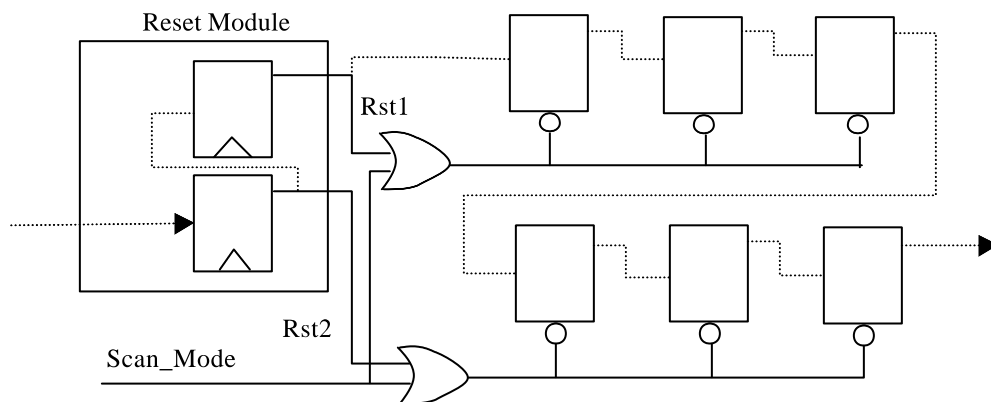
Copyright © 1999 by P.Radhakrishnan. All rights reserved

Asynchronous Resets

One of the common problems that affects the controlability is the usage of asynchronous resets for the flops in the design. Using asynchronous reset is the simple way to reset a flop. In most of the cases the reset will be generated synchronous to a clock from a module and the output of this module is connected to all the async clear pins of the flops. By doing this eventhough the clearing at the flop is done using the async clear pin, the reset event is synchronous to the clock using which it is generated. This approach solves many timing issues. When you hook up the scan chain for this chip, those flops in the reset module will also get connected in the chain of flops. Can you guess what would happen if these flops are also hooked in the scan chain? When the scan pattern is shifted in through the “scan_in” pin, whenever the output of the flop that generates the reset becomes a “zero”, all the flops that are connected to this net will get cleared (assuming an active low reset). This will disrupt the scan pattern that is being shifted into the chain. This scenario is depicted in the following diagram.



It can be seen from the above diagram that the flops outside the reset module can get get a reset when the corresponding flop’s output in the reset module changes to “zero”. This circuit has uncontrolable resets when the chip is in the scan mode. Due to this the scan can not function properly and would lead to very poor fault coverage. The solution to get rid of this problem is to provide some way of controlling the reset pins of the flop during the scan mode. If we can propagate a “one” to all these clear pins of the flop during the scan mode, we can shift the scan patterns without any problem. The introduction of an OR gate with a control signal solves this problem. This is shown below.



Introduction of a top level pin to the chip and an OR gate will solve our problem. By making the “Scan_Mode” pin a “HIGH” during the complete scan testing duration, we can gain control of the reset signals that are applied to the rest of the circuit. During the normal mode

 Copyright © 1999 by P.Radhakrishnan. All rights reserved

operation of the circuit, this pin will be set to “LOW” and the Rst1 and Rst2 signals will have effect on the clear pins of the flops.

Embedded Memories

Another common problem in high density ASIC is the embedded RAMs in the design. These day’s ASICs are moving in the direction of System-on-a-Chip (SoC) to integrate a complete system’s functionality inside a single chip. Such high integration demand introduction of memory elements like the RAMs and DRAMs inside the chip itself. These memories can vary from a small form factor (256x8) to very high form factors (e.g. 32Kx32). These memories are normally provided by the silicon vendor as single unit with its access ports. The design of such memories are done at the transistor level and hence we will not have any access into the memory cells. This means, we can not hook the logic inside the memory cells into the scan chain. For the scan logic these memories are assumed to be “black boxes”. (There are other methods like BIST (Built-in Self Test) to check the health of memories).

In a design with memories, the logic around the memory element will have the scan related problems. The logic that feeds into the memory (the address generator, and the controll signal generating logic, etc.) will have observability problem. This is because there are no suitable capturing flops after these logic to capture a value onto the scan chain. These logic drive straight into the memory element. Similarly the logic cloud at the output of the memory will have controlability problem as this portion is directly driven from the memory.

One way to solve the issue is to connect a flop to each and every pin of the RAM around the periphery so as to provide controlability and observability to the logic around the memory.

Other Issues

Timing

To implement a scan design in a successful manner in an ASIC, one has to take care of various other issues that may arise in the logic. Let us touch upon some of those issues. Let us take the timing issue. In the multiplexed scan style the flop has a MUX in front of the actual flop. This MUX will take some time to propagate its input to the ouput. This will cause a small amount of additional delay to the signals that are going to the D input of the flop. This additional delay will reduce the “setup margin” by a small extent. Similarly at the output of the flops, we get an additional load due to the fact that the Q output is connected to the TI input of the next flop in the scan chain. This will also introduce a delay in the “clock to Q” output timing of the flop. These effects are very small in quantity but has to be considered while doing the timing analysis. Otherwise the logic may not function at an expected clock frequency.

Chain Length

One other common problem is to end up with too many flops in a scan chain. If there are about 10 thousand flops in a design and if they are all hooked in a single scan chain, we will end up in spending a lot of time in the tester in testing the chip. The ATPG tool will generate the test vectors that will test all the stuck-at faults in the design and we need to run all those test cases when the device is in the tester in the form of wafer or in the form of packaged device. This will cost a lot of money as the usgae of the tester will cost a lot. Moreover there will be schedule impacts also to introduce the chip in the market. There are few methods to reduce the impact of large testing time.

One method is to split the scan chain into two or three chains and run the test vectors simultaneously through all the three chains at the same time. By doing this, the time spent in the tester is reduced. While doing this split, we have to be careful to split the chain in such a

Copyright © 1999 by P.Radhakrishnan. All rights reserved

way that there are almost equal number of flops in each of the chains. Could you imagine what would happen if they are grossly differnent? The longer the scan chain, more will be the time spent in shifting the scan patterns. If the 10K flops are split like 100, 2000 and 7900, it will not be very advantageous. In this case the chain with 7900 flop will take longer time to get tested, whereas the smallest chain would have gotten tested very early. To avoid such problems we can make a split which is somewhat equal. (e.g. 3300, 3300 and 3400). Most of the tools handle this automatically once we specify the configurations.

Multiple Clocks

Splitting the scan chain can be done based on other factors also. In many cases splitting may not be that easy as said above. Current day's SoC type of chips will have interface to many types of devices (e.g. memory controllers, modems, display, and so on). So various portions in the chip will be functioning at different speeds to interface to all these devices. This calls for internal logic to operate in different clock frequencies. Such a design is called as multi-clock design. The scan chain design in such a chip is more complicated. When those flops that function using two different clocks are hooked in the scan chain next to each other, the value propagation in the scan chain depends on the phase difference between the two clocks. If the two clocks do not have any timing relation between the two, data launched from one flop can not be captured using the other flop reliably. This is a classic synchronization problem. There are ways to handle this. The simplest way is to introduce a "lock-up" latch in the path. This topic is not addressed in this paper and we will discuss this at some other point in one of the following issues.

Similar problems are posed by designs which have flops operating on both positive and the negative edges of the clock. Such cases also have to be treated very carefully when doing the scan chain design.

DFT Tools

There are a handful of tools available in the industry for doing the DFT implementation. "TestCompiler" from Synopsys, Inc. is one such tool that is being used in the ASIC industry very widely. There are other test-design tools provided by the same company and by other companies also. In the test design phase of the ASIC design, designers use one such tool to read in the entire design database (this database is provided in the form of netlist[1]). There are suitable commands provided to the tool to set the scan style, scan input and output pins, to insert the scan elements in the place of ordinary flops and to stitch the scan path as we discussed in the earlier sections. Many of the tools will also have the ATPG engine, which can generate the test patterns that are to be used in the tester for testing after manufacturing. Discussing too many details about the tools may lead to confusion due to the unfamiliarity towards these tools. So I'm avoiding them.

Conclusion

Design for Testability (DFT) is one of the important factors to be considered in the ASIC design. Foundries use this method to improve the reliability of the product they ship. The semiconductor vendor will have a policy to have a minimum level of fault coverage for the device he delivers. Often times, achieving a 90% fault coverage may not be so difficult, but to increase this to still higher value will be a much difficult effort. The number of test vectors necessary to improve the fault coverage beyond a particular value will increase exponentially.

[1] Netlist is the list of nets describing the interconnectivity of the entire design using various gates and flops.

Copyright © 1999 by P.Radhakrishnan. All rights reserved

It used to be a compromise between the number of test vectors (hence the time spent in the tester) and the fault coverage. I would say that achieving a 95% fault coverage is a very good situation. But there are foundries with still tougher standards.

I guess we have gone thorough the principles of the scan design and discussed about the benefits of DFT. We also discussed about the problems that may arise because of un-friendly design practices that may reduce the fault coverage. I wish to conclude this paper here. Look forward for more details of the ASIC design flow in the next issue.

Trademarks note:

Synopsys and “Test Compiler” are registered trademarks of Synopsys Inc., Mountain View, CA, USA

Copyright © 1999 by P.Radhakrishnan. All rights reserved