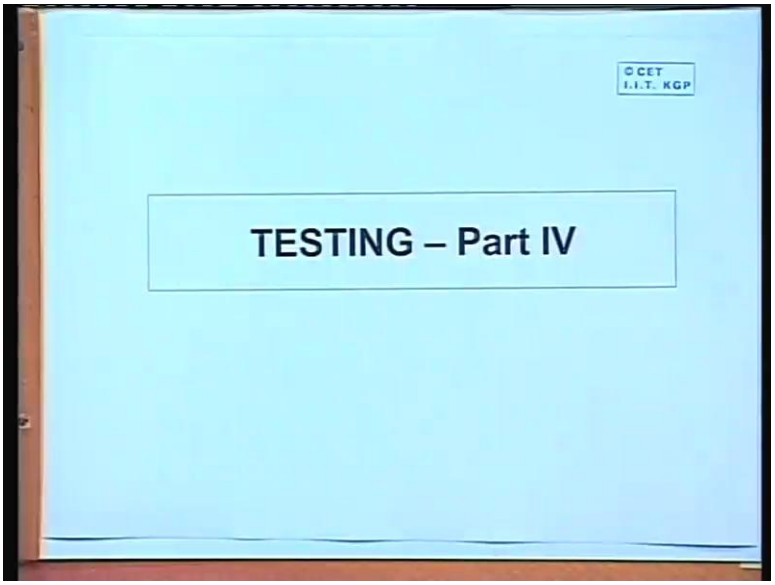
Electronic Design Automation

Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture No #33
Testing (Part – IV)

In this lecture I will be talking about design for testability techniques in the digital circuits which is a very popular technique which people used nowadays in order to add some special testability. You can say sub circuit or some testability hardware inside a given circuit.

(Refer Slide Time: 00:57)

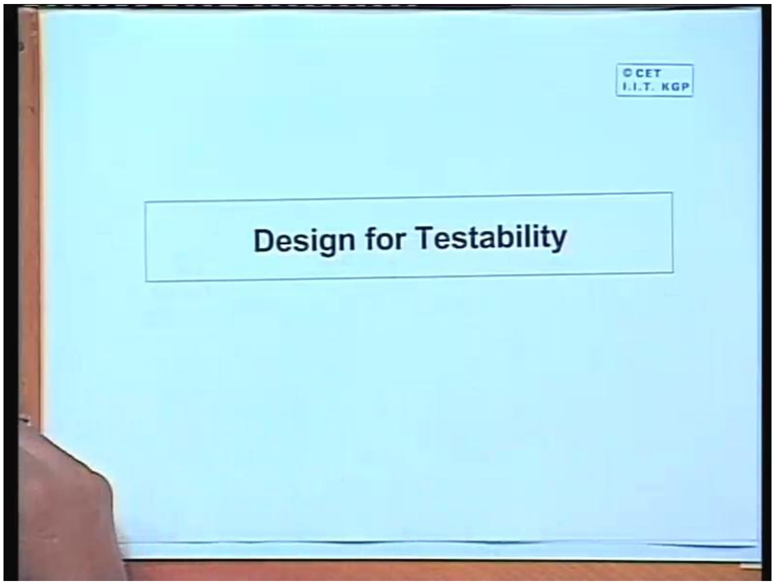


© CET
I.I.T. KGP

TESTING – Part IV

So as to making the problem of testing much simpler for the testing linear.

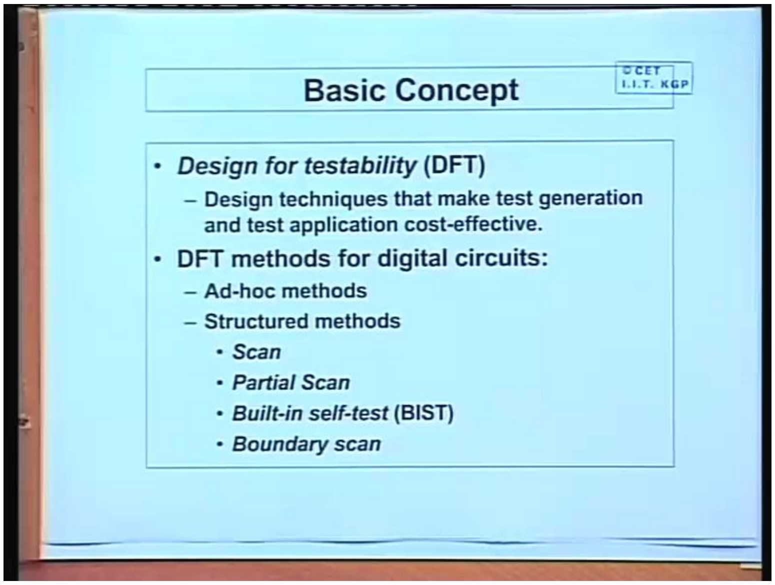
(Refer Slide Time: 01:03)



Design for Testability

So this design for testability is the topic of discussion in this lecture.

(Refer Slide Time: 01:11)



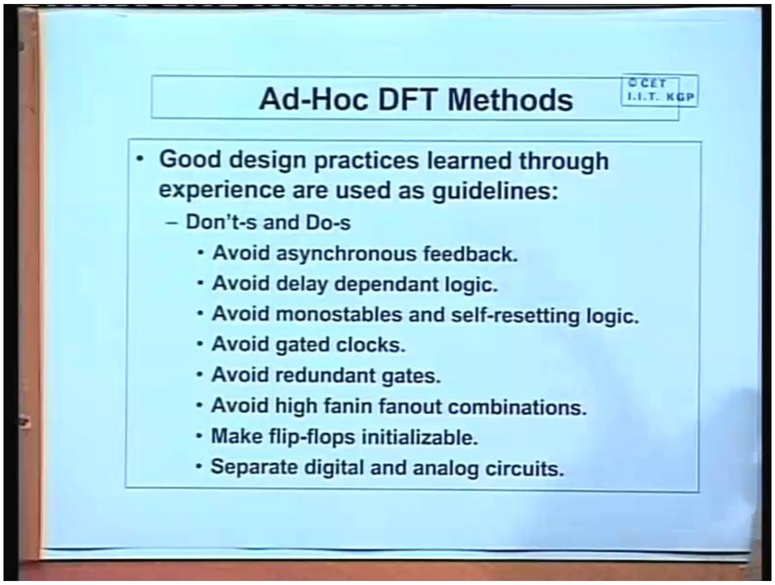
Basic Concept

- *Design for testability* (DFT)
  - Design techniques that make test generation and test application cost-effective.
- DFT methods for digital circuits:
  - Ad-hoc methods
  - Structured methods
    - *Scan*
    - *Partial Scan*
    - *Built-in self-test* (BIST)
    - *Boundary scan*

First let us see what this DFT or design for testability is and what are the different ways in which you can do it. Design for testability is actually a collection of techniques design techniques using which both test generation and test application can be meet more manageable and cost effective okay. So when I say both test generation and test application means the complexity of the test generation process get reduced. And the total time of test application also we try to reduce. Of course these contradictive sometimes we will see the test application time will go up. But in some other scheme again it will go down. Now for digital circuits there are broadly two approaches which people follow for design for testability. The first one is of course a set of techniques which have been laid down by experience designers. These are called Ad-hoc methods.

So designers out of their experience they know that well if I have a circuit structure like this, this becomes difficult for me to test. So you do not use this. So there is a bit set how to do’s and don’ts. That means, you can do this, you don’t do this. There are bit set of Ad-hoc rules which are totally based on design experience okay. These are called these so called Ad-hoc methods. The second and more structural regular method is one where there are some well defined techniques which can be integrated in the CAT tool for synthesis also which results in terms circuits which becomes much easier to test. So there are techniques like scan and partial scan, built in self-test, boundary scan. We will briefly look at these all of these. So first we look at the Ad-hoc methods for DFT and what are the different rules that people have proposed well I am only giving you small set of rules in fact the number rules are pretty large just to give you an idea.

(Refer Slide Time: 03:34)

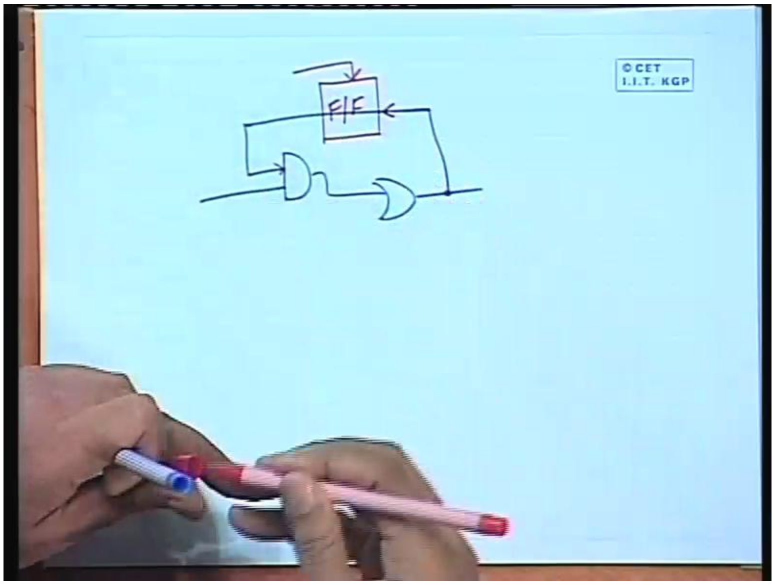


Ad-Hoc DFT Methods

- Good design practices learned through experience are used as guidelines:
  - Don't-s and Do-s
    - Avoid asynchronous feedback.
    - Avoid delay dependant logic.
    - Avoid monostables and self-resetting logic.
    - Avoid gated clocks.
    - Avoid redundant gates.
    - Avoid high fanin fanout combinations.
    - Make flip-flops initializable.
    - Separate digital and analog circuits.

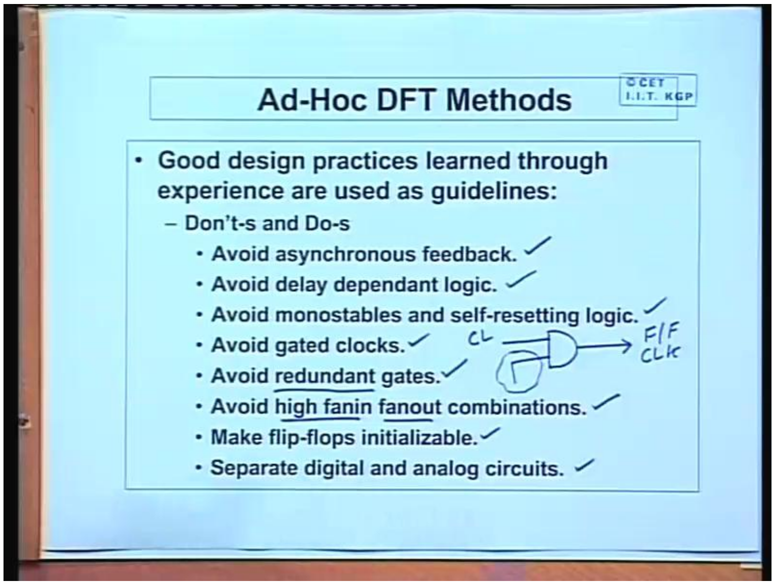
So the Ad-hoc DFT methods these are essentially good design practices, which have been laid down by experience designers who have learned them through design experience over a period of time. Some simple rules are as follows. You avoid an asynchronous feedback. Say avoid asynchronous feedback means.

(Refer Slide Time: 04:06)



It is not good it is not decidable to have a circuit say like this from the output of the OR gate you give a feedback to the input. This is an asynchronous feedback but the designer would not mind if there is there a flip flop in between. If there is a flip flop in between which is controlled by a clock then this is a synchronous feedback. So whenever there is a clock only then the output value will come to the input it will not come continuously okay fine.

(Refer Slide Time: 04:37)



Ad-Hoc DFT Methods

© CET I.I.T. KGP

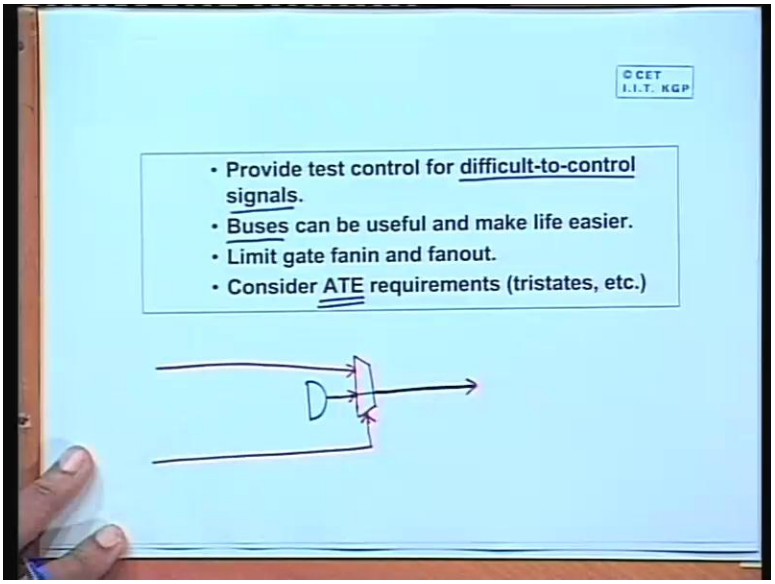
- Good design practices learned through experience are used as guidelines:
  - Don't-s and Do-s
    - Avoid asynchronous feedback. ✓
    - Avoid delay dependant logic. ✓
    - Avoid monostables and self-resetting logic. ✓
    - Avoid gated clocks. ✓
    - Avoid redundant gates. ✓
    - Avoid high fanin fanout combinations. ✓
    - Make flip-flops initializable. ✓
    - Separate digital and analog circuits. ✓

Avoid delay dependent logic. Sometimes we design circuits for the delays of the individual circuit elements are made use of in order to in order to achieve some logic. For example for generating a very narrow pulse equal to the delay of one gate or three gates or so on. So you do not use those things. Monostables multi vibrator and logic which can reset itself to a known state these are not recommended. What designer says that will if you have to reset some logic or some counter or some registers something to known state. They should be a search signal which can be given from outside. Say externally you should able to reset the circuit not using some logic inside it is automatically resetting itself.

Because if there is some problem in the reset logic it will be very difficult for to test okay. Avoid gated clocks. Clocks should goes straight from the from the primary inputs to the respective clock inputs of the flip flops. You never use a structure whether it is the gate that is a clock in one of the input they some logic coming in the other input and the output you are feeding to a flip flop clock. This is something which is which is avoidable. So you do not use gated clocks because if there is any logic a problem again in the logic which is controlling the clock then you will be having a lot of problem in testing circuit, you try to reduce the redundancy as much as possible.

So if you have a circuit realization which is not properly minimized, then it can be shown that there are many faults which can occur and which can prevent the detection of other faults. So if this some redundancy in this circuits your fault coverage can go down significantly. So you try to reduce redundancy do not use high fanin or high fanout. Because these are not good in terms of circuit performance and also reliability. So all the flip flops should initializable externally with relatively less effort. If you have a mix signal kind of a design were both analog and digital circuits along the same chip. You try to separate them out into disjoint areas of the chip. Do not let the analog circuit and digital circuit. The interact with each other closely and result in unpredictable behavior.

(Refer Slide Time: 07:29)



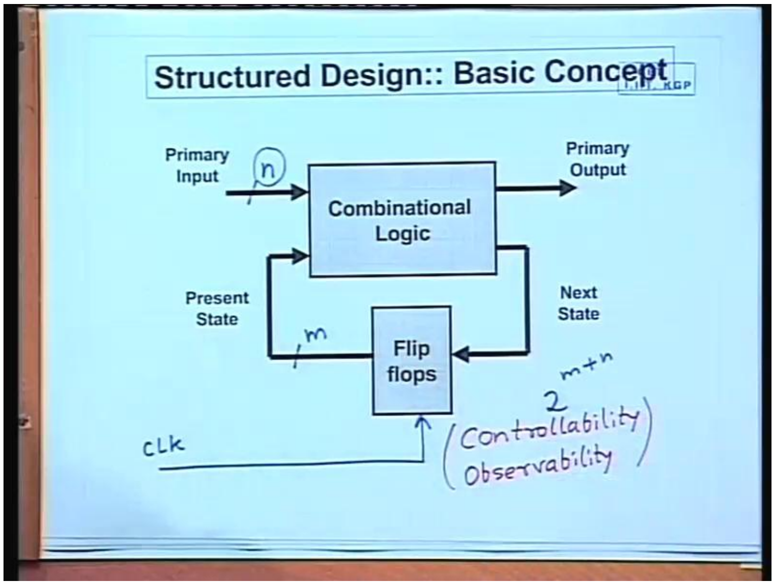
© CET I.I.T. KGP

- Provide test control for difficult-to-control signals.
- Buses can be useful and make life easier.
- Limit gate fanin and fanout.
- Consider ATE requirements (tristates, etc.)

And if you find through analysis that there are some difficult to control signals. Then you can provide some separate test control for them. For example you will set a circuit if you find out that this is a signal which external it is very difficult to test to into control. This is possibly coming from the output of a gate. This was a circuit. So what do you do as a modification is that you introduce a multiplexer. The one input of the multiplexer is coming from the gate other input is coming from the primary input and you also have the control of the multiplexer from the primary input. So by controlling these two lines you can directly control the value of this logic value if the need arises during testing okay. Buses are going in terms of testing.

So it tries to use buses wherever we need. This I have already mention limit fanin and fanout. And ultimately you will be doing the testing using some automated test equipments automated test equipment or ATE. So you will have to look at the requirements of the ATE before you frame the design rules like whether your tester suppose tristates logic or not for example okay. So these are some design rules which you can enforce and the resultant circuit will be will be easier to test. But as you can see these are all Ad-hoc kind of rules. Well even if you follow this rules you cannot really quantify that how much easy it will be to test okay. This is cannot quantify.

(Refer Slide Time: 09:24)



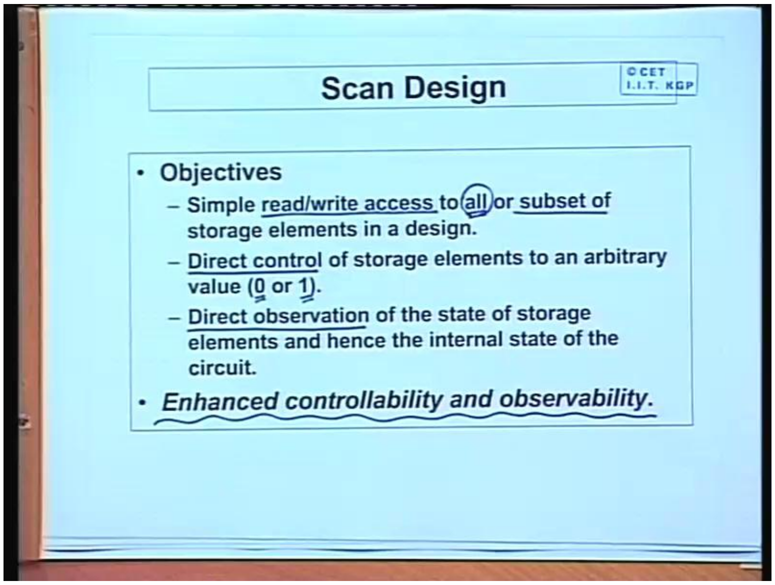
Structured Design:: Basic Concept

Structured design tries to attack the problem from a different view point. See this structured design for testability they are primarily focus or aimed to test sequential circuits. And as you know a sequential circuit can be modeled like this where the combinational portion and the memory portion flip flop portion can be separated out. Of course I am not shown the clock. The clock will be there this will be a clock which will be feeding the flip flops. Primary inputs primary outputs and the state variable this is present state and this is the next state. So when the clock comes, the next state becomes a present state okay. Now because of the presence of the flip flop the problem of testing sequential circuit becomes difficult.

Because you try to understand if there are n primary inputs for example if there are m number of flip flops. Then the combination circuit can have two to the power m plus n combinations. Now out of these this n inputs you can control directly. But the other m inputs you cannot control directly. So this other m inputs you cannot control directly this lengths the complexity of to this problem that how easy, how difficult will be to control this flip flops. This will dictate the complexity of this total testing problem. Well now suppose if we can device some mechanism, using which we can set these flip flops to any known value externally rather easily. And also we can observe the values of this flip flops. So what we are saying is that we want complete controllability and complete observability of this flip flops.

Now if we can achieve that then you try to understand one thing. If we can achieve that, then the testing problem reduces to the problem of testing the combination logic alone. Because now I can assume that all m plus n inputs are directly controllable because I can set the flip flops to any known value. So as if all m plus n inputs are available to me and similarly all the outputs are also available to me for observation. So if we do this, the problem reduces to the problem of testing combination logic alone. We can forget flip flops for the timing okay. And we can use a combinational test generator which is relatively much less complex for generating the test patterns. This is the basic idea behind the structure design for testability techniques. So one very popular method for this structure design technique is called a scan design or a scan path.

(Refer Slide Time: 12:42)



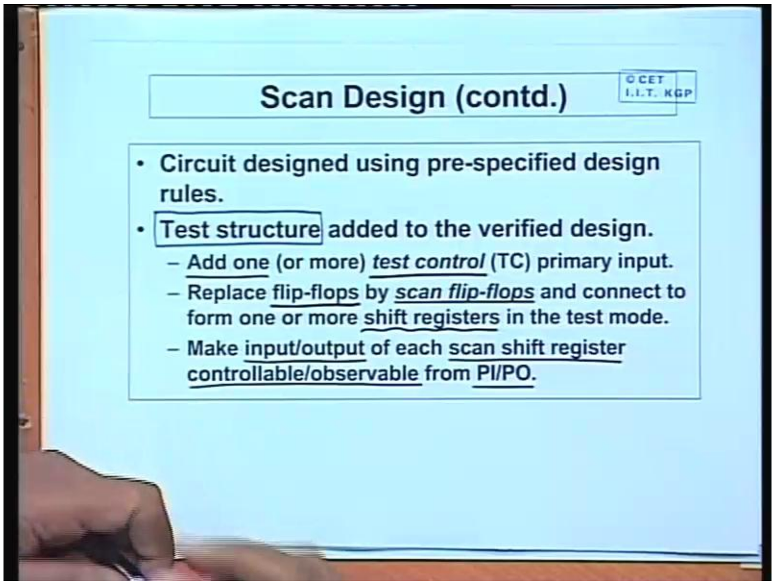
Scan Design

© CET I.I.T. KGP

- Objectives
  - Simple read/write access to all or subset of storage elements in a design.
  - Direct control of storage elements to an arbitrary value (0 or 1).
  - Direct observation of the state of storage elements and hence the internal state of the circuit.
- *Enhanced controllability and observability.*

This scan design says that you have some kind of simple read write access to all or subset of the flip flops. Well for the timing we assume all we shall see later that we can it is slightly relax this restriction. We can have this access only to a subset of this flip flops also. And using this what we can achieve, we can have direct control of the storage elements. We can set them to either logic 0 or to logic 1 as the need arises and the second thing is that we can also observe their values direct control and direct observation. These are the two you can two aspects of scan design. So direct observation of this state of storage elements okay. So essentially we are trying to enhanced the controllability and observability of this storage elements of this circuit. This is the basic it is a basic cracks behind this scan design technique. Now scan design technique also follow some design rules.

(Refer Slide Time: 14:02)



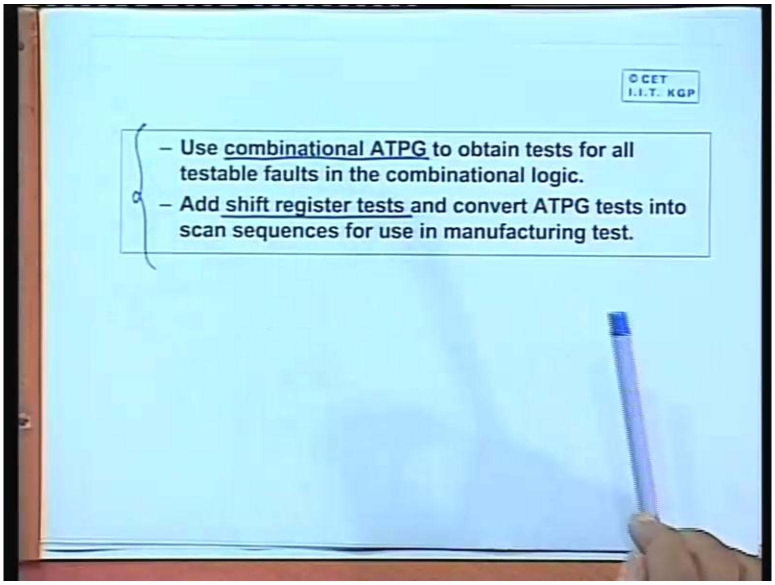
Scan Design (contd.)

- Circuit designed using pre-specified design rules.
- Test structure added to the verified design.
  - Add one (or more) *test control* (TC) primary input.
  - Replace flip-flops by *scan flip-flops* and connect to form one or more shift registers in the test mode.
  - Make input/output of each scan shift register controllable/observable from PI/PO.

But the design rules are targeted towards achieving the goals as I have just mentioned to enhance the controllability and observability of the flip flops. Well and nothing can come free in order to do this we have to add something more to the circuit. You have to add a test structure in terms of some additional hardware and extra pins to the design. So essentially what we need minimum is that you need one extra pin called test control. This extra pin will tell you that whether you are in the normal mode or in the test mode. During testing the test control pin will be high otherwise it will be zero something like that. And you will also to modify the flip flops we will see how you will be replacing each flip flop by so called scan flip flops. And you will be interconnecting them in such a way that during the test mode the flip flops all become long shift register okay. And each of these shift registers which are called scan shift register they can be controllable observable from the primary input output if there inputs and outputs are available.

So what we are doing is that, that all the flip flops that exist in our circuit, we add some extra hardware. So that if the test control is high in the test mode the all become change into a shift register. So now we have a long shift register, we can shift in any data pattern to that shift register and we can load it with any known value. So using this we can achieve complete controllability of the flip flops. Similarly after applying a clock whatever gets slashed in the flip flops we can again configure they must a shift register. And again apply a clock to shift out the bit severely and you can observe them whatever is coming out. So we can also get complete observability okay. So by configuring the flip flops as a shift register help us in achieving complete controllability and observability of the individual flip flops okay.

(Refer Slide Time: 16:42)

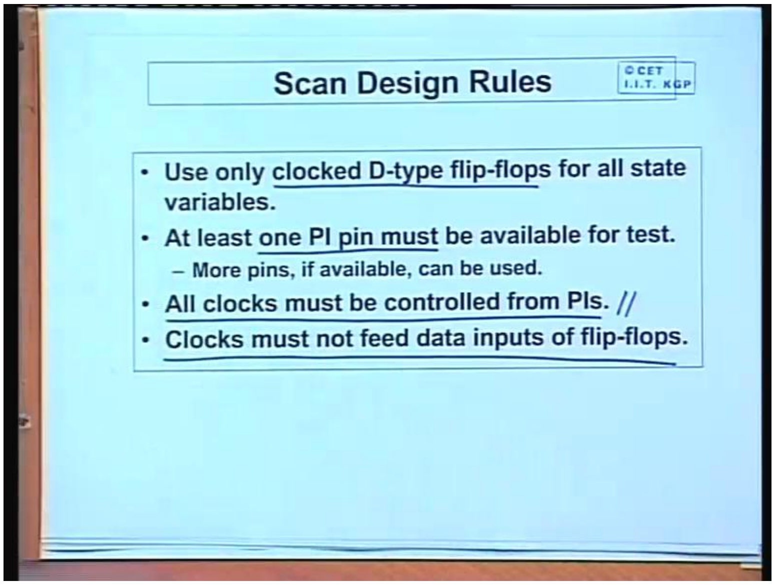


© CET
I.I.T. KGP

- Use combinational ATPG to obtain tests for all testable faults in the combinational logic.
- Add shift register tests and convert ATPG tests into scan sequences for use in manufacturing test.

And once you have done this we have reduce the problem of testing to combination logic testing. So now you can use combinational automated test pattern generator to obtain the tests and some additional test to test the shift registers. This is all we need we need to test the shift register and once we have done this we can use a combinational test patterns to test the combinational logic okay.

(Refer Slide Time: 17:12)



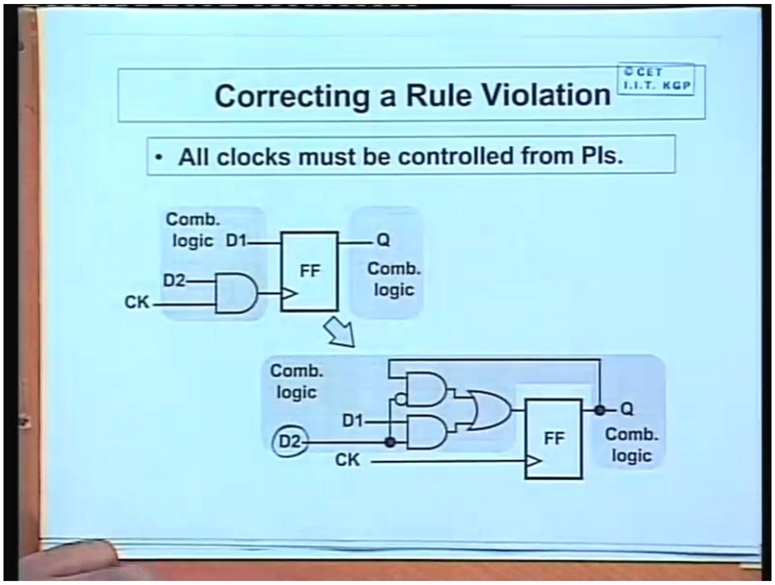
Scan Design Rules

© CET
I.I.T. KGP

- Use only clocked D-type flip-flops for all state variables.
- At least one PI pin must be available for test.
  - More pins, if available, can be used.
- All clocks must be controlled from PIs. //
- Clocks must not feed data inputs of flip-flops.

Now there some scan design rules some of this have already mentioned it says that you only use clocked D flip flops. And at least one primary input pin the test control must be available extra. More pins if it is available you can used your advantage but minimum one pin must be available. Third no gated clocks all clocks must be directly controlled from primary input and clocks must not feed the data inputs of the flip flops okay. This is also not allowed. Clocks must only feed the clock inputs of the flip flop nowhere else. Well there are ways to enforce this rules. For example let us take one example say the third one all clocks must be control from primary input. Suppose in a design say like this.

(Refer Slide Time: 18:13)



Correcting a Rule Violation

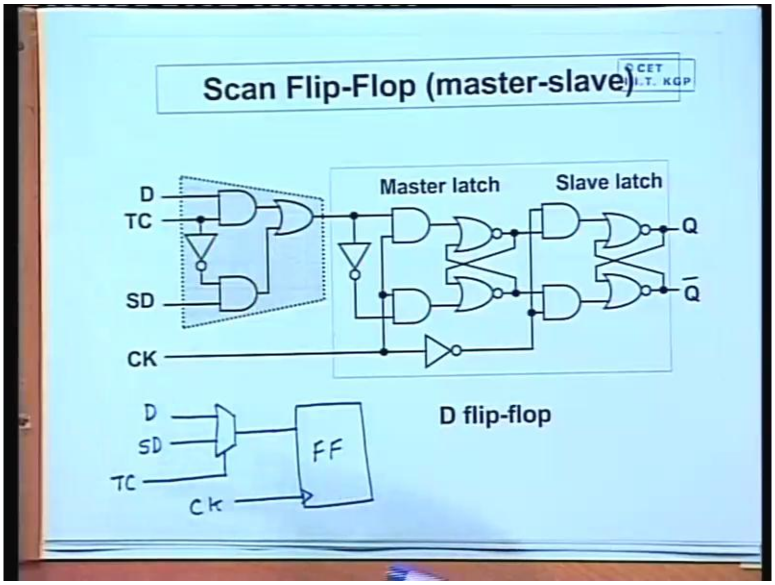
© CET I.I.T. KGP

- All clocks must be controlled from PIs.

We had a circuit a portion of the circuit where this design rule violation was there like the clocks were not directly control from primary input that means here you had a gated clock. These a flip flop clock some other data due to coming from somewhere they were ended and then the clock was generated. But the rules says the clock should come directly to the gate inputs okay. So this design means that if D2 is 0, okay the clock will not come and the flip flop will return its state. So I can modify this circuit like this in order to eliminate this design rule violation you see what I have done. Now this clock is coming straight to the clock input okay. Here this is nothing but a simple multiplexer this is two line two one line multiplexer where D 2 is acting as a select line. If D 2 is 1, D 1 is selected.

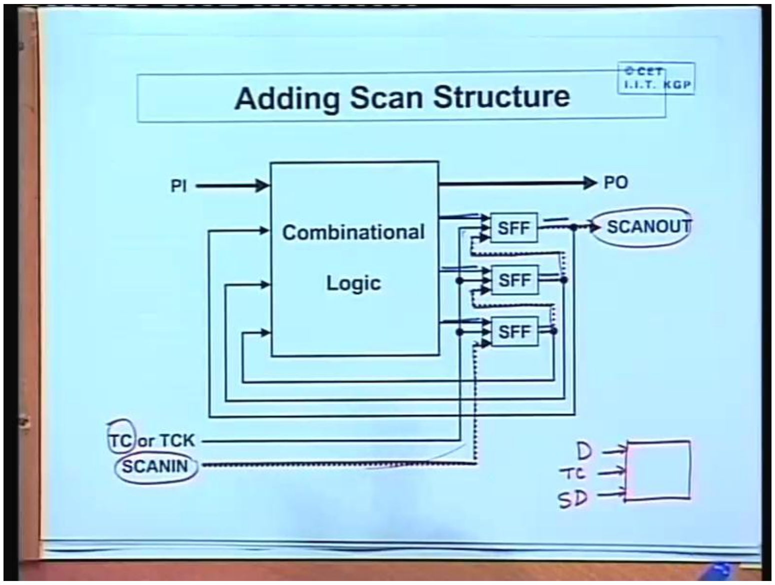
This is the normal behavior of the circuit D 2 is one then whenever clock comes D 1 will going exactly that the happens. So D 2 is 1 if clock comes D 1 will going. But if D 2 is 0 here the clock was not coming at all. But here you cannot stop the clock from coming clock are directly connected. So here what you have done that the output whatever is there that same value will come. So the value of the flip flop output will not change the same value will again be stored once more if D 2 is 0. So by making a small change we have eliminating the design rule violation. And this is also not an asynchronous feedback because feedback is taken from the output of a flip flop okay. So this is perfectly allowed. Well and the second thing I mentioned that in scan design all the flip flops must be converted into scan flip flops okay. So let us see how this scan flip flop looks like.

(Refer Slide Time: 20:25)



So scan flip flop looks like this. You can forget this portion for the time being the rest is a conventional master slave latch. In the master stage the slave stage and the clock is coming the master is clock by the clock high and after inverter slave this latched. But this is the scan flip flop actually what you are doing is that you have the flip flop. This was the flip flop clock was there clock is still coming. But the data input which is there your connecting a multiplexer out here. There are two inputs to the multiplexer one is the normal data input one is the scan data. SD is the scan data and the test control input is controlling the multiplexer. This is actually functionally what is happening. So the test control is deciding whether the normal data input will going or this scan data input will go okay. Now let us see that how this kind of a structure can be configured so that in the test mode the flip flops behave as a shift register okay. Let us see.

(Refer Slide Time: 22:01)

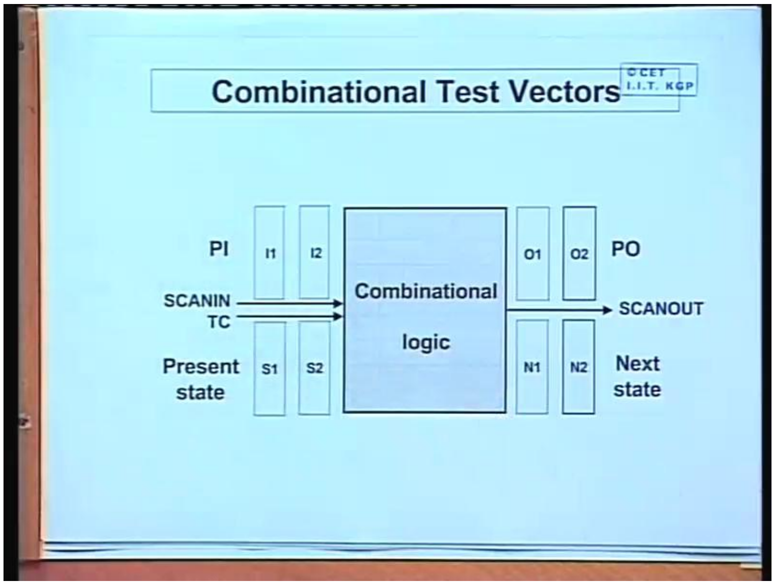


So the modification is done like this. Normally we had a combinational logic some flip flops with feedback. Now this flip flops had been converted into scan flip flops. Now scan flip flops in this diagram there are three inputs the inputs are in this order. The top input is the normal data input D the second input is the test control and the third input is the scan data okay. Now you look at this scan data pins I shown it by doted I am just highlighting this path. Suppose test control is high. So instead of D going in now scan data will going you just look at the previous diagram. If TC is high SD will going okay. So this is SD input this will go into output get latched and the output of this is feeding the SD of the next flip flop output of this is feeding the next SD and in this way.

So you see that through this SD is flip flops are change into a shift register. So whenever this TC search the circuit to the test mode the flip flops are in the form of shift register there is a SCANIN there is a SCANOUT. Through this you can feed data and you can observe. Now SCANIN and SCANOUT can be some of the primary input and primary output pins also. If you have extra available pins you can use them as a dedicated way. But TC must be a separate pin okay. So and if TC is 0 which means circuit is operating in the normal mode then the D input whichever was there the normal input. This will be going out, this will be going out, this will be going out and the same value will be fed back. This is the normal mode of operation okay.

So in this circuit you try to observe that for applying each test pattern what we have to do? For applying each test pattern I will have to first shift in the data bits into the shift register. Say in this example there a three state variables the three flip flops. So I need three clock pulses for doing that after that I will be applying the primary input and apply another clock set the circuit to the normal mode and apply a clock pulse. So after applying the clock pulse I can see the primary output and these outputs will get latched into the flip flop. Then again I set the circuit to the test mode and I again apply three clock pulses. So that the next state which is coming out will be observed in the primary output. So in this way I can control and observe okay.

(Refer Slide Time: 25:31)

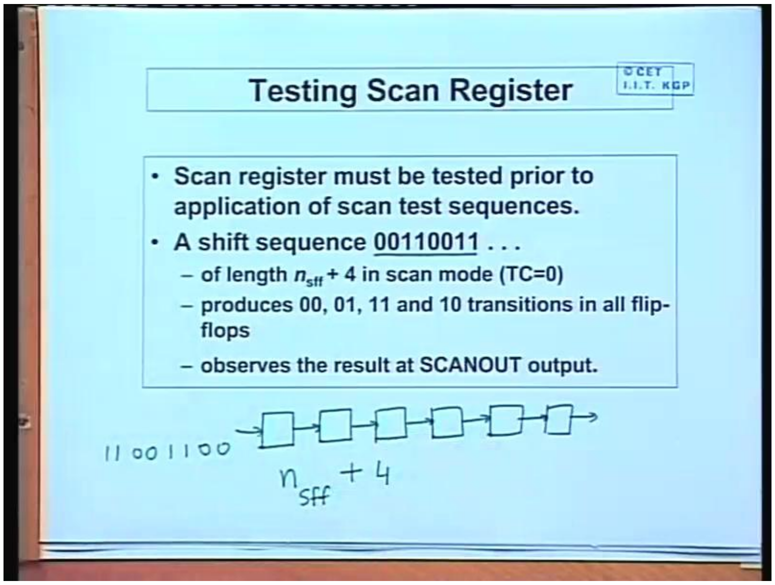


Combinational Test Vectors

Diagrammatically this diagram shows in the access of time what happens is the combinational logic. These are the primary input you are applying. These are the primary outputs which are going this I 1, I 2, O 1, O 2 are in sequence and this present state S 1, S 2 and the next state N 1, N 2, these are shifted in and shifted out. So you need additional clock pulses for feeding the present state and observing the next state okay. So scan in and test control are the control inputs.

So now let us try to estimate that how many clock pulses we need for doing this whole testing okay well. Now I told you earlier that before we actually start testing the circuit with the combinational test patterns you will have to test the shift register okay. So first let us see that how we can test the shift register. Because shift register testing is very important because unless you are shear of the shift register is working correctly you cannot really start testing the combinational logic. So there is standard ways of testing a shift register.

(Refer Slide Time: 26:48)



Testing Scan Register

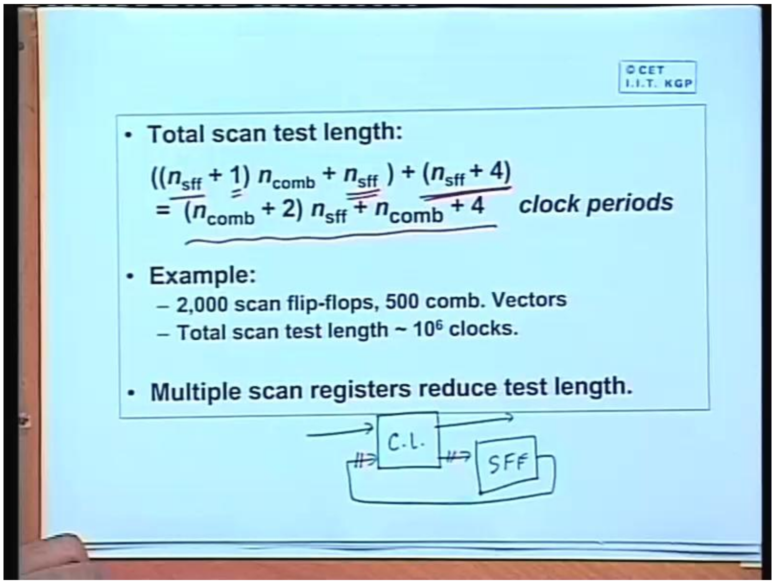
© CET I.I.T. KGP

- Scan register must be tested prior to application of scan test sequences.
- A shift sequence 00110011 . . .
  - of length $n_{sff} + 4$ in scan mode (TC=0)
  - produces 00, 01, 11 and 10 transitions in all flip-flops
  - observes the result at SCANOUT output.

For people normally do is that if you have. Let us take an example. Suppose you have a shift register with 6 stages, six stage shift register you want to test it. Now you apply a continuous shift sequence of 00110011to it, you apply 00110011 and all the bits you apply they will have to come out of this. Now how many bits will be applying? If there are n sff number of flip flops. Here 6, it will be this plus 4. So the idea behind applying this particular pattern is that if you look at this pattern then each flip flop in terms of the clock cycles will be assuming values 0, then 0, then 1, then 1, then again 0.

So if you look at the present value in the next value they are they undergoing all possible transitions 0 to 0, 0 to 1, 1 to 1 and 1 to 0. So actually I am testing all possible transitions in each individual flip flop and also whatever I am feeding is coming out at the other side without any change without any modification. Then only I can say with some confidence that the flip flop is working correctly. Now if you have six flip flops you will need n sff number clock cycle to feed the data into this and four more clock cycles to bring them out. So you need so many clock cycles to actually test the flip flop well here I am assuming TC equal to 0 is the test mode okay. So test control 0 means the test mode there you need n sff plus 4, so many clock cycles. Now regarding testing okay.

(Refer Slide Time: 28:58)



©CET
I.I.T. KGP

- Total scan test length:

  $((n_{sff} + 1)\, n_{comb} + n_{sff}) + (n_{sff} + 4)$
  $= (n_{comb} + 2)\, n_{sff} + n_{comb} + 4$ *clock periods*

- Example:
  - 2,000 scan flip-flops, 500 comb. Vectors
  - Total scan test length ~ 10^6 clocks.

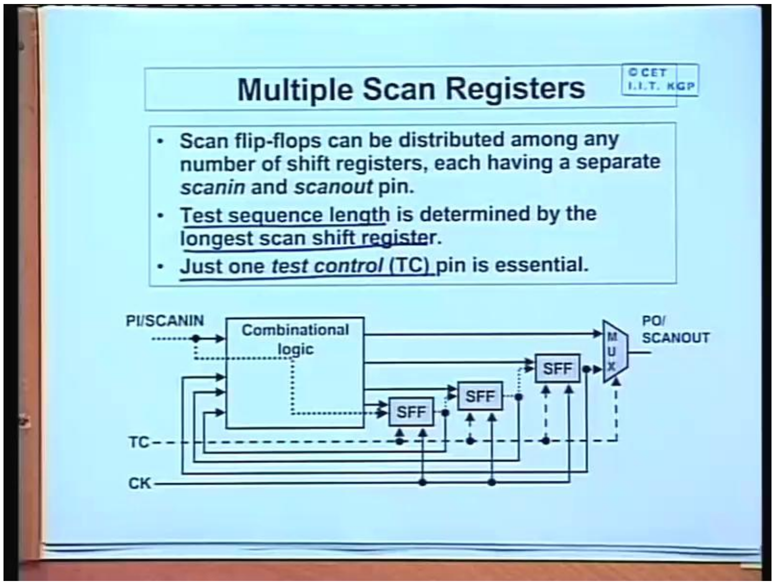
- Multiple scan registers reduce test length.

Now you try to recall what I said earlier. You have the combinational logic. You have this scan flip flops. So you have a structure like this. Now for applying each test pattern what do you do? You apply n sff clock pulses to feed this registers okay. So you need n sff clock pulses to load this register so that these inputs are no applied. Then at the same time we can apply a primary input and then we will have to apply one more clock pulse this one. So that the output value new get latched into this flip flops. So if n comb denote the number of combinational test patterns, then n sff plus 1 into n comb this is the total number of clock pulses you need to apply the test patterns.

And everywhere feeding the data into the sff at the same time you can go on observing the data for the previous cycle. So when you are feeding data for the present test vector you can also observe the response that was generated from the previous test vector all together okay. So this is the number of clock pulse you will be requiring total. And this n sff extra term is coming due to that that additional feeding and coming out. Because for the last pattern for the last test pattern there is no next. So for that you will have to you will have to explicitly shift them out.

So you will need an extra n sff for the last one. So this is the total number of clock cycle you need to test the combination logic and this I have mention for the shift register. So if you simplify this it comes to an expression like this. So just for an example if you have 2000 such flip flops and 500 combinations vectors in the total scan test length becomes of the order of 10 to the power 6 clocks. Now in order to reduce the test time instead of changing all the flip flops into a single shift register. You can change them into multiple shift register so that you can feed data and observe response on to the multiple shift register parallel. So you are total test testing time will drastically go down okay.

(Refer Slide Time: 31:58)



Multiple Scan Registers

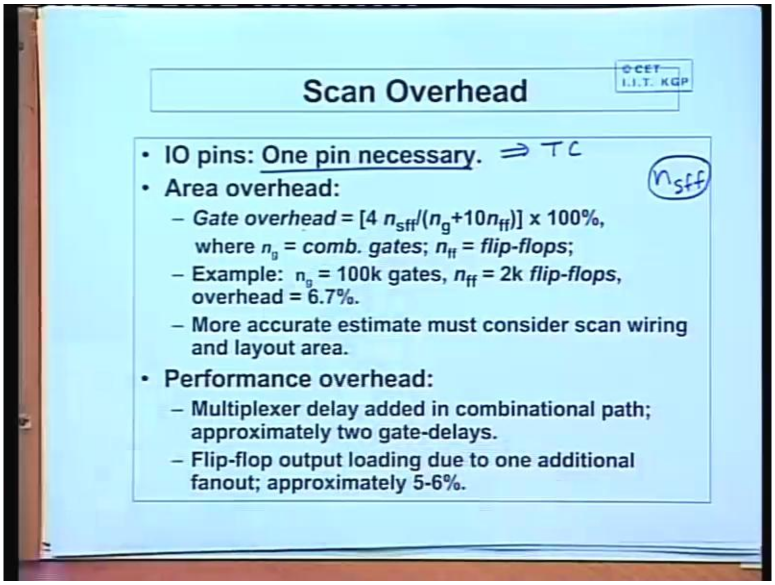
© CET I.I.T. KGP

- Scan flip-flops can be distributed among any number of shift registers, each having a separate *scanin* and *scanout* pin.
- Test sequence length is determined by the longest scan shift register.
- Just one *test control* (TC) pin is essential.

So if you want to have multiple scan register kind of a scheme you can do something like this. Well of course here I have shown for only one scan one register but you can you can extend it to more. Say the first thing is that these are the scan flip flops. This is a test control feeding. And instead of having a separate scan in line you have one of the primary input lines acting as this scan in. So this line will straight away come to here and similarly the output of the shift register instead of having a separate scan out line you can have a multiplexor. Control by this TC where the output will be sometimes the one of the primary outputs sometimes this shift register output. So this is one way you can have scan in and scan out without any additional input or output pins. For the input you do not need any hardware. But for the output you need one multiplexor, now if you have several primary inputs and primary outputs pins that are disposal you can have several such shift registers in parallel.

You can have ten such then you will be utilizing 10 primary inputs and 10 primary outputs of course you will need. In that case so many number of multiplexers 10 number of multiplexers 1 powers can change okay. But even if you multiple scan register the point note is that only one test control is sufficient. And if you are unable to split the shift registers exactly equally then the test sequence length will be determined by the longest scan shift register. Because you will have to scan all of them only then you can apply the next primary input okay. So the longest scan change will determine the total time finally okay. Now this as scan path is a very good method in the sense that you can have complete you can say controllability and observability of the memory elements and you can have combinational testing scheme for testing a sequential circuit it is very good. But as I as I mentioned earlier nothing comes free you have to invest in extra hardware in order to achieve this. Now let us try to see what kind of extra hardware we need in this case. This is so called scan overhead.

(Refer Slide Time: 34:42)



Scan Overhead

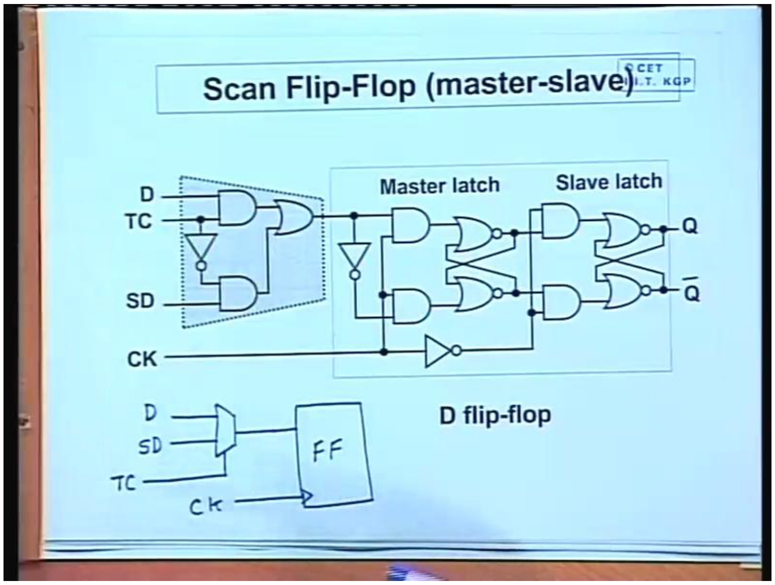
©CET
I.I.T. KGP

- IO pins: One pin necessary. $\Rightarrow$ TC
- Area overhead:
  - *Gate overhead* = [4 n_{sff}/(n_g+10n_{ff})] x 100%, where n_g = *comb. gates*; n_{ff} = *flip-flops*;
  - Example: n_g = 100k gates, n_{ff} = 2k *flip-flops*, overhead = 6.7%.
  - More accurate estimate must consider scan wiring and layout area.
- Performance overhead:
  - Multiplexer delay added in combinational path; approximately two gate-delays.
  - Flip-flop output loading due to one additional fanout; approximately 5-6%.

n_{sff}

Now scan overhead says that one extra IO pin is necessary. This is the test control pin in terms of area overhead. Well you observe that you have n sff, so many numbers of flip flops and you have converted all of this flip flops into scan flip flops in order to implement scan path. Now when you are converting a flip flop in the scan flip flop if you recall you are adding a multiplexer in the input stage which means four additional gates. So if you just look at the diagram once more I have the diagram yes.

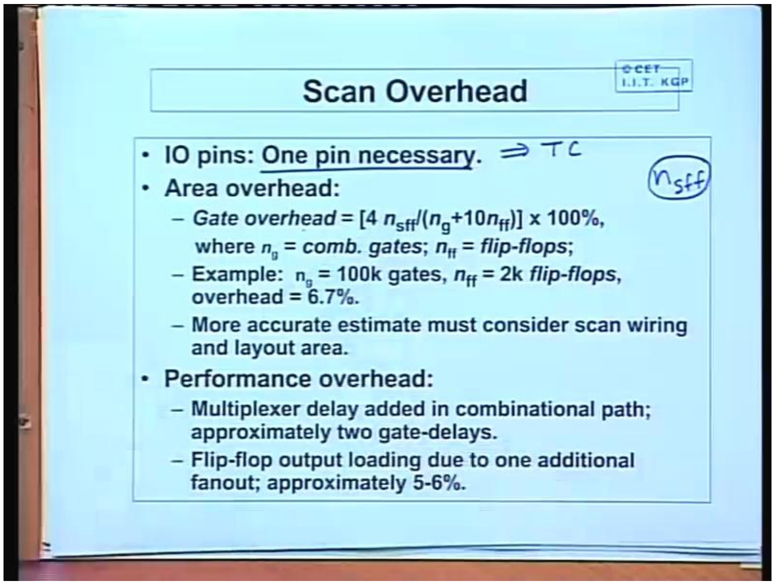
(Refer Slide Time: 35:47)



Scan Flip-Flop (master-slave)

So here the original flip flops circuit. Without this required 10 gates okay. To make it a scan flip flop you need 4 additional gates okay. This something remembers.

(Refer Slide Time: 36:03)



Scan Overhead

- IO pins: One pin necessary. ⇒ TC
- Area overhead:
  - *Gate overhead* = [4 $n_{sff}/(n_g+10n_{ff})$] x 100%, where n_g = *comb. gates*; n_{ff} = *flip-flops*;
  - Example: n_g = 100k gates, n_{ff} = 2k *flip-flops*, overhead = 6.7%.
  - More accurate estimate must consider scan wiring and layout area.
- Performance overhead:
  - Multiplexer delay added in combinational path; approximately two gate-delays.
  - Flip-flop output loading due to one additional fanout; approximately 5-6%.

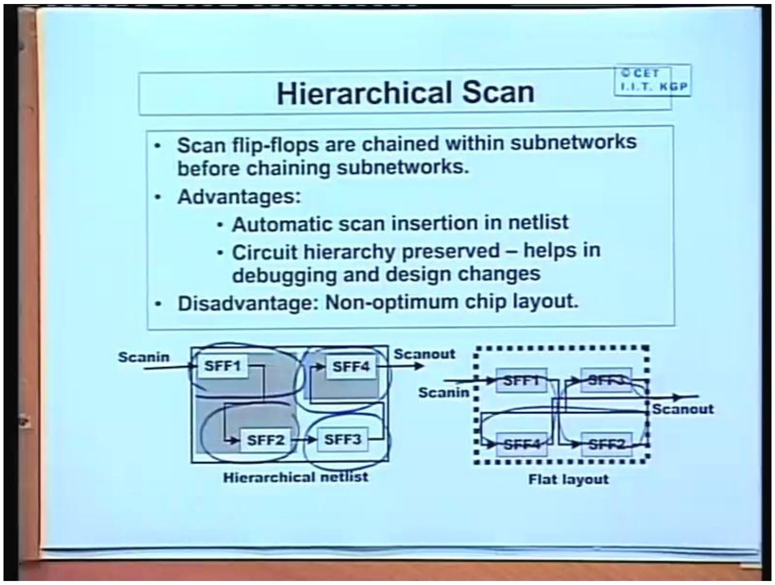
n_{sff}

So with this input you can calculate the gate overhead like this where n_g is the number of gates in the combinational logic. So earlier without scan the number of gates was n_g plus 10 into number of flip flops. Because each flip flops has 10 gates. But now after converting into scan flip flop you need 4 into n sff, this many additional gates. Each flip flop because 4 gates. So taken as a percentage this can be defined as the gate overhead just an example. If you have 100k gates in the combinational logic and 2k flip flops then the overhead comes to be comes to around "6.7" percent. But actually these are rough estimate you have to make other estimates as well like I have ignore those multiplexors in the outputs. If you are using the primary outputs to observe this scan outs you need to consider these multiplexors. And also there are lots of additional words you have to lay out.

Now in order to configure the shift registers you need so many additional wires. So your total routing area will also increase the total shift area will also increase that we have not taken in the account here okay. This is of course the area overhead but you also include some kind of a performance overhead here. We see the multiplexor delay will be added in the combinational path approximately to get delays. Flip flop output loading because the fan out of the flip flop is also now increasing. So earlier the flip flop for straight away going out. Now the flip flop output is now feeding the input of the next flip flop. So when you increase the fan out of a gate in most technology the capacity load of the output increases. So instead of one you are not driving two gates so the capacitor load becomes double. So the charging discharging time of the output note becomes approximately double and the gates becomes slower.

So in that sense this speed of the circuit drops down. An addition at the input of this scan flip flops you have an multiplexor. So some additional logic delay. At the output primary output you are having a multiplexor some addition logic delay. Now if some of these lie in the critical path of this circuit this will mean that the total speed of the circuit will go down. Of course the clock frequency has to be drop down slightly because the flip flops are now working slightly slower because of the additional multiplexing logic at the input okay. So it has been found that the performance overhead is again to the tune of 5 to 6 percent. So you have getting a number of advantages fine. But you are also paying some price you are paying for more area you are paying for a slower clock. So if you are trying to build a very high performance circuit, then you will have to think about something else also. Just a pure scan path approach may not be very desirable in that case because you are making a circuit slower okay.

(Refer Slide Time: 39:44)



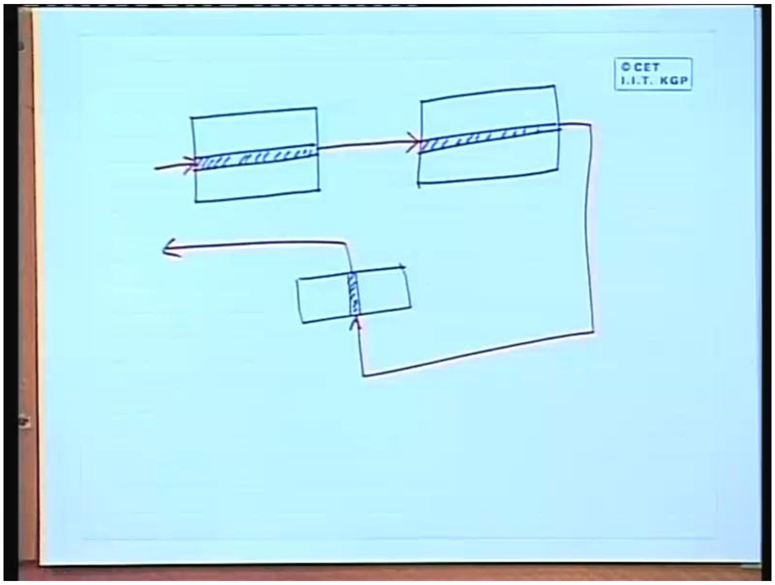
Hierarchical Scan

©CET I.I.T. KGP

- Scan flip-flops are chained within subnetworks before chaining subnetworks.
- Advantages:
  - Automatic scan insertion in netlist
  - Circuit hierarchy preserved – helps in debugging and design changes
- Disadvantage: Non-optimum chip layout.

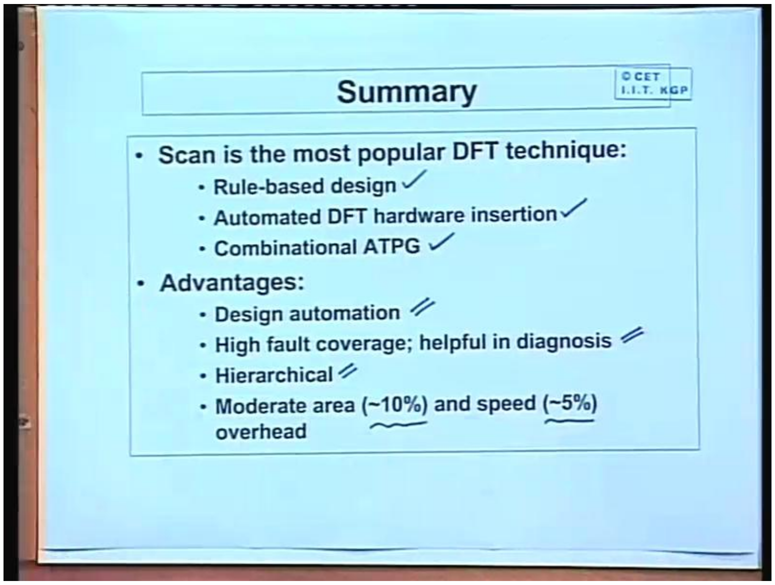
So the way this scan path is actually implemented in CAT tools is that you can do it hierarchically. Like well instead of considering the whole circuit as a whole and totality and we incorporate scan and do it, you can do it in a hierarchical fashion. Like suppose in this block you have a hierarchical netlist like during you are say this circuit partitioning and placement you had four different blocks. So I am not showing the different blocks I am only showing the flip flops. So there are different blocks in a circuit where there flip flops. So without loss of generating I will show only one flip flop in one region. But it can be more, say this is one region, this is one region, this is one region and this is one region and this is how they can be connected. Now if you had a flat layout, then all this flip flops should be connected in a single scan chain like this, okay like this. There will be connected in a long scan chain. But here what I am saying is that well here also same thing is true. But instead of considering the whole circuit as a whole you do it hierarchically. When you are developing this small module what I am saying is that.

(Refer Slide Time: 41:20)



When you are developing a small module you develop this module with scan chain. This is a scan in here shown it like this. You develop another module this also will have a scan chain. This another module this will also have smaller scan chain. Now this modules you are developing hierarchically. Now when you put all of them together on the surface of the silicon and try to interconnect them you can also integrate this scan paths. So now this will be yours scan in and this will be your scan out. So instead of considering the whole circuit together and doing it you can do it individually, this helps you in saving time in testing in the sense that see the test generation and faults simulation algorithms they are pretty complex. They are dated time which is more than of the order of n square where n is the number of gates. So you can break the whole circuit into number smaller circuit and applies these two these smaller sub circuits then the total time that you require will be much less in general okay. So that is why this kind of a divider and concur approach helps in reducing the total time that you need for testing okay. So just you summarize this scan path approach.

(Refer Slide Time: 42:59)



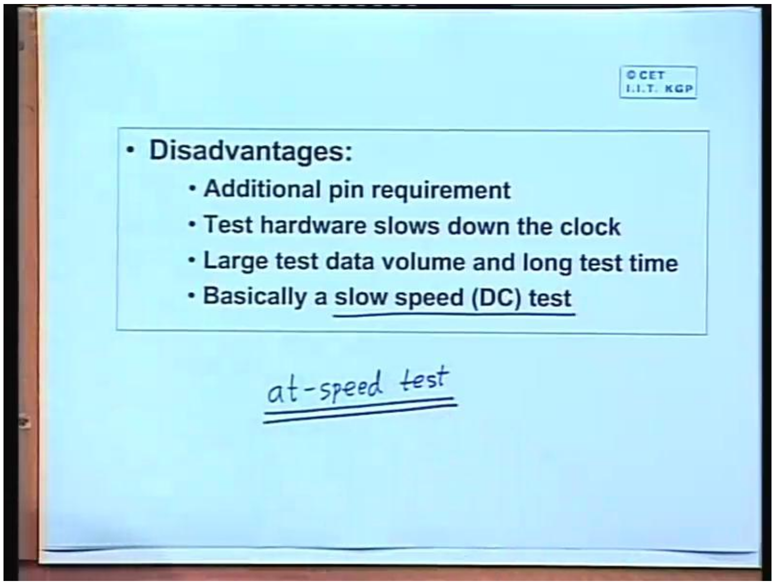
Summary

© CET I.I.T. KGP

- Scan is the most popular DFT technique:
  - Rule-based design ✓
  - Automated DFT hardware insertion ✓
  - Combinational ATPG ✓
- Advantages:
  - Design automation
  - High fault coverage; helpful in diagnosis
  - Hierarchical
  - Moderate area (~10%) and speed (~5%) overhead

Scan is the most popular design for testing technique which people used nowadays. It is a rule based design there are the set of ways define rules. Now since the rules are so simple and systematic they are easily intergratable. Intergratable there can be easily integrated into the commercial CAT tools. Most of the commercial CAT tools have an option to insert scan path automatically okay you can do this and if you do this, if you have a full scan which means all the flip flops are in the scan path you can use combinational automated test pattern generator for generating a test patterns. So the advantages are obvious you can have better design automation. You can have high fault coverage because the combinational ATPG works much better as compare to sequential. You can have hierarchical approach if you want moderate area and speed overhead which can be tolerated in most cases other than extremely high performance design for you want to want increase this speed by all means other than those applications. For most of the applications this overhead can be acceptable okay. There are disadvantages also as you are seen.

(Refer Slide Time: 44:25)



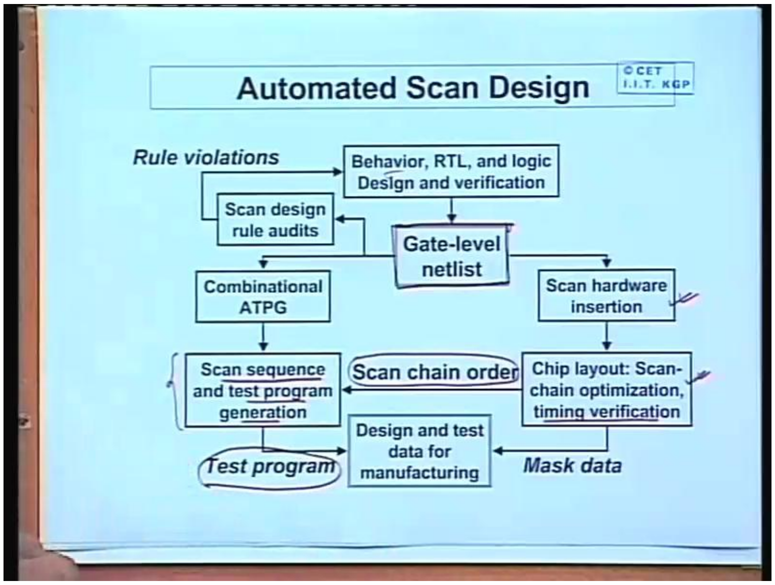
© CET
I.I.T. KGP

- Disadvantages:
  - Additional pin requirement
  - Test hardware slows down the clock
  - Large test data volume and long test time
  - Basically a slow speed (DC) test

at-speed test

We need additional pins we need additional test hardware which can slow down the clock. Large test data volume because for applying each individual test we have to apply. So many clocks and so many shifts. So the total testing time is increased and this is a slow speed DC test. See this is called slow speed test to distinguish this from another kind of a testing which is called at speed testing. At speed test is a mechanism well you are testing the circuit at the clock frequency at which it is intended to work. But here since for each application of the pattern we need to shift the data in the shift register. We are not making the circuit work at full speed okay. So if we could apply the input patterns one by one apply clock at the maximum speed that could be the at speed testing. But here we are applying the test pattern at a much slower clock rate as compare to the intended speed of operation. Because the controllability and observability is making the process much slower okay.

(Refer Slide Time: 46:00)

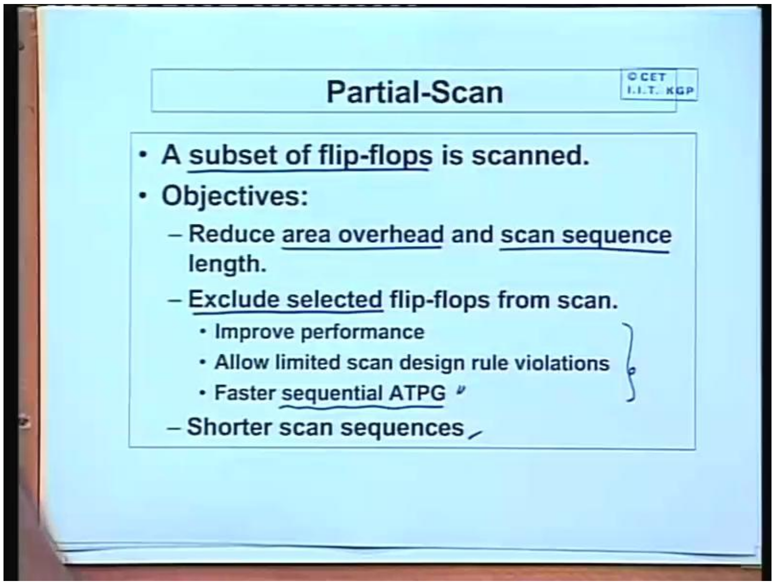

Automated Scan Design

© CET I.I.T. KGP
Rule violations
Behavior, RTL, and logic Design and verification
Scan design rule audits
Gate-level netlist
Combinational ATPG
Scan hardware insertion
Scan sequence and test program generation
Scan chain order
Chip layout: Scan-chain optimization, timing verification
Design and test data for manufacturing
Test program
Mask data

So I am just showing you overall picture see you have the gate level netlist here. So what we normally do in a typical VLSI design flow. We have a get list we have a get netlist here which you have obtained through synthesis. So using some tool from the gate level net list you can carry outs scan hardware insertion. So some of the flip flops will be made scan flip flops and these scan paths will be made. So after this insertion you can go to the chip layout you can optimize this scan chains and you can also see that whether the timing requirements are satisfied okay. Now in order to do this scan chain optimization a process called scan chain ordering is done where not going the details of this. So some sort of an ordering of scan chain maybe required.

So depending on the scan chain order you know generate something called scan sequence and test program generator. These are the actual patterns which will be told to the automatic test equipment. So the ATE will blindly apply these test patterns to the circuit okay. This in order to do this you also need the combinational test pattern generator which have been generated by a combinational ATPG. So this total pattern which you find out from here this is apply to the actual chips which you fabricate and this will you test program. So these are the roughly the you can center flow. And in the gate level net list if you see that if there are any scan design violations you can go back and make some changes and again come back okay. So this is the whole picture. Now I told you that it is not mandatory to have all the flip flops in a circuit present in the scan change. You can have something called partial scan also.

(Refer Slide Time: 48:21)



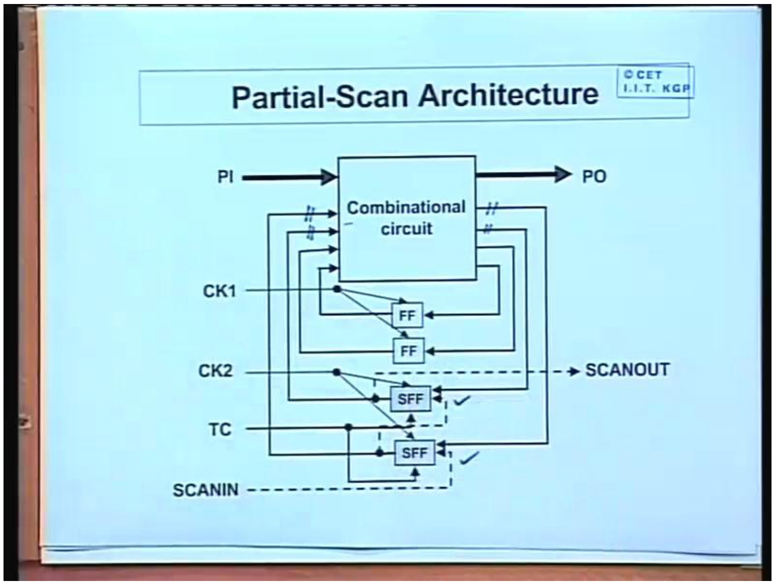
Partial-Scan

©CET I.I.T. KGP

- A subset of flip-flops is scanned.
- Objectives:
  - Reduce area overhead and scan sequence length.
  - Exclude selected flip-flops from scan.
    - Improve performance
    - Allow limited scan design rule violations
    - Faster sequential ATPG
  - Shorter scan sequences

Now let us see why this is required what is the motivation behind having partial scan. Well obviously partial scan means a subset of the flip flops are scanned. And some of the objectives which are listed here or also obvious it follows from this. Since we are not converting all of the flip flops into scan flip flops, so area overhead is reduced and scan sequence length is also reduced. Some of the selected flip flops are excluded from scan. There are ways to do this we will talk about this. This can improve performance as compare to full scan design and you can have faster sequential ATPG. So I will just come to this point in the next slide. Shortest scan sequence I already mentioned.

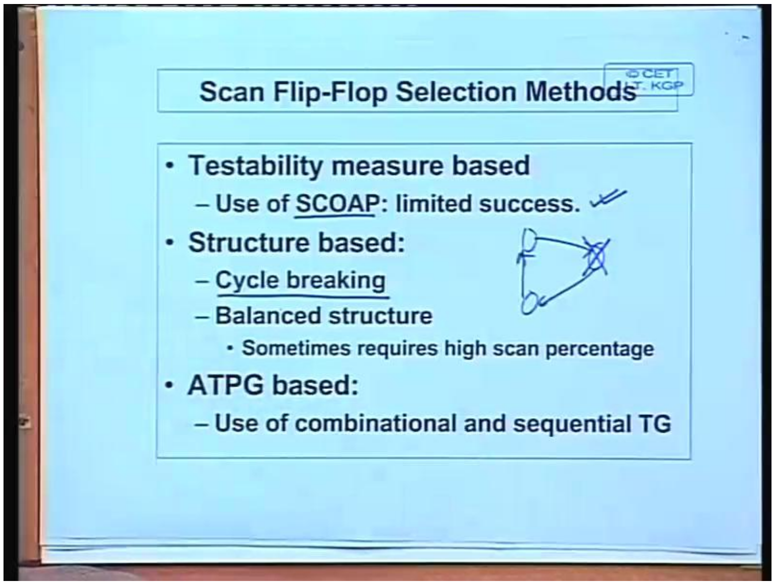
(Refer Slide Time: 49:22)



So the partial scan architecture will look like this. We have a combinational logic there are the flip flops. Some of the flip flops you convert into scan some you do not. So for these two flip flops this and this you can treat that is part of the primary inputs. Similarly these two outputs, but for the other two they still remain conventional sequential circuit. So if you have partial scan like this one thing is true that by changing them into a shift register you are unable to have complete controllability and observability of all the flip flops. This means that you cannot use a pure combinational circuit test generator to generate the patterns. You need sequential test pattern generator to generate the test patterns.

But one thing I told you that sequential test pattern generator is more complex it takes much more time to generate test patterns as compare to combinational. But from experience we know that what are the kinds of scenarios which can make the sequential test pattern generator more difficult. So here idea is that we try to identify or classify the flip flops as which of those are more difficult with respect to sequential ATPG which of those are not so difficult. So the difficult once we convert into scan the not so difficult once will leave as it is. So idea is we have a sequential circuit the flip flops alright. But with respect to sequential ATPG that will be not much of a problem because the problem cases we have eliminated okay. So there are number of different techniques which have been proposed for selecting these scan flip flops.

(Refer Slide Time: 51:31)

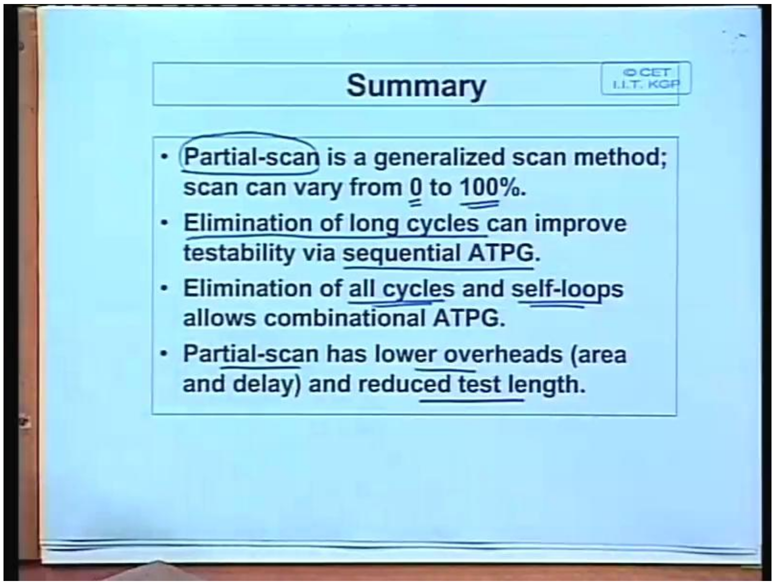


Scan Flip-Flop Selection Methods

- Testability measure based
  - Use of SCOAP: limited success.
- Structure based:
  - Cycle breaking
  - Balanced structure
    - Sometimes requires high scan percentage
- ATPG based:
  - Use of combinational and sequential TG

The first approach is base function testability measures well. There as several standard techniques one popular method is SCOAP. Here you can statically analysis the circuit and try to find out the controllability and observability metric of each individual line. So you can find out which of the flip flops are more difficult to control and observe. So you can select those as scan flip flops. You can have some structure based approach. Like you convert the circuit into a graph structure where the notes will be the flip flops and edges will be paths from the flip flop output to the input. You try to break the cycles there. If there are three flip flops which you see the line in a cycle you make one of them as scan slip flop. So this flip flop is no longer there in the conventional sense. So you have broken the cycle. So if you break the cycle then cycles in the flip flops are not there which are more difficult for sequential ATPG. Secondly you try to a some kind of a balanced structure that means a number of flip flops that can lie in cascade you limit that into the same value in all cases. So these help in sequential ATPG. So combinational sequential destination you can do.

(Refer Slide Time: 53:01)



Summary

©CET
I.I.T. KGP

- Partial-scan is a generalized scan method; scan can vary from 0 to 100%.
- Elimination of long cycles can improve testability via sequential ATPG.
- Elimination of all cycles and self-loops allows combinational ATPG.
- Partial-scan has lower overheads (area and delay) and reduced test length.

So to summarize partial scan is the technique which is more practical you can say because if your circuit has ten thousand flip flop, there is no point in converting all 10000 of them it will scan flip flops. Moreover you just try to think in a design you can have many registers I have a 32 bit register. I can a 20 such registers. So 20 into 32 straight away I have a 600 flip flops. Shall I change them into flip flops? Well the registers need not be put as part of scan chains. So this is the flexibility which the partial scan method works. But you can select which are the flip flops you need to be include included as part of this scan chain. So depending on the circuit the number of flip flops you need to scan can vary from none up to all. And elimination of long cycles in the flip flop paths I told you this improves the testability via sequential ATPG. You eliminate all cycles eliminate all self loops if you do this.

Then you can allow combinational ATPG also. You need not convert all flip flops into scan path but you eliminates cycles in the flip flops flip flop paths. There are ways in which if you can break this cycles and self loops you can use combinational ATPG for testing. And obviously partial scan has much more overheads and less number of test pattern that we need to be applied for testing. So today we talked about some of the design for testability techniques which people use normally. In our next class we could be talking about another technique which is also use pretty widely that is called built in self-test. Which means, given a chip we introduce some additional circuit inside the chip itself. So there the chip can be tested on its own. It is a self-test kind of the think okay. So this will be discussed in our next class. Thank you.