

Optimization Techniques for Digital VLSI Design
Dr. Chandan Karfa
Dr. Santosh Biswas
Department of Computer Science & Engineering
Indian Institute of Technology, Guwahati

Module - 04
VLSI Testing
Lecture – 17
Optimization Techniques for Design for Testability

Hello and welcome to the next lecture on the module on VLSI testing. So, at present we are discussing on the optimization techniques for ATPG's which is on basically lecture 3 and 4. So, basically this is topic on which we are considering both optimization on techniques for ATPG as well as we are also trying to study the design for testability.

(Refer Slide Time: 00:39)

Course Plan
Module 4: VLSI Testing
Lecture 1 and 2: Introduction to Digital VLSI Testing, Automatic Test Pattern Generation (ATPG), Design for Testability
Lecture 3 and 4: Optimization Techniques for ATPG
Lecture 5: Optimization Techniques for Design for Testability
Lecture 6: High-level fault modeling and RTL level Testing

So, in the last few lectures on the next lecture on the module on VLSI testing. So, in this lecture we will be mainly looking at optimization techniques for design for testability. So, throughout this module on VLSI testing you might have been observed that we have first discussed basically, what are the different way to generate test patterns; that is the basic idea of testing digital circuits and then you are trying to optimize it so, that we can handle larger circuits, but in all the lectures like when you have talked about scan chain.

So, you have to find or when you have to say that we have to put some kind of a compaction circuit when you are optimizing ATPG's when we are optimizing based on

the terms of test patterns which are generated. So, you are trying to merge them or we are trying to compress them and the outputs, we are trying to compact techniques you are using for the output generation. So, that actually the number of pins which is required for the; a t e is reduced and so, forth.

So, different type of techniques we were looking at. So, that we can try to handle large circuits as we can do efficient testing on them, but on that case we have seen there are two basic issues that one way you are trying to generate good quality test patterns. So, that they can handle very large circuits as well as they can test more than different type of faults and so, forth. But along with that we should also observe that sometimes we are putting some extra kind of circuitry, but testing itself on the device or your all your VLSI circuits.

So, that is for to help in testing we are putting additional circuit for example, very important is your scan chain. So, this scan chain basically with the normal flip-flops we are putting a multiplex for test pattern compression basically we have a compression circuit and also have a compaction circuit at the output. So, additional circuitry we what we put is; basically call the design for testability.

So, in this lecture basically we will try to see; how we can optimize the amount of additional circuits we are going to put for design for testability for example, if I say that to make a circuit device circuit testable.

(Refer Slide Time: 02:36)

DFT based Optimization

- Circuit Dependent
 - Based on functionality of the circuit
- Circuit Independent
 - Optimization of Scan Chains
 - Optimization of Compression, De-compression and compaction circuits

But, see we can put lot of circuits and that of additional circuit. So, that test becomes very easy, but then adding too much circuit will also lead to two type of problems one is the circuit cost will raise, power will raise and at the same time who will test the circuit which is you have put for design for testability. If I have a smaller circuit what designed for testability; then there is how we can assume that; as the circuit is quite small the probability of happening of false in them can be less or it will be easier to test the circuit under designed testable circuit itself.

In other words you are putting a DFT circuit to test help to test your original circuit, but who will test the DFT circuit? If the circuit is small still you can do some kind of testing or assume that the number of faults there will be smaller in number, but if that circuit itself is quite large in nature, then that will not serve the purpose.

So, therefore, this lecture is basically DFT based optimization that what are the optimizations, we can do on the DFT circuit itself that is the key role and one of the key challenge we have. Like for example, I mean I can have instead of having one very long scan chain, I can partition it into 1000 different scan chains. So, scan loading will be faster, but then 1000 extra pin outs has to be brought out for all these scan chains.

So, this way we are thinking how to have a trade off so, that we can have a quite logically simplistic DFT circuit as well as your testing remains easier and that is you are not pushing too much pressure on the DFT circuitry, as well at the same time you are not making the testing of the circuit too difficult. So, we are coming into a balance. So, there are basically if you look there are two ways of handling DFT one is called circuit dependent.

So, for a different very different particular type of circuits you can put different type of inbuilt circuitry to enable them to do the testing. So, it may depend on the functionality of the circuit. So, that is actually a totally design problem we are not going to highlight on this that is given a circuit, given the functionality; I want to put some kind of may be say has some kind of special practice pattern generator inside the chips which can help you to test it. So, but it will be very much dependent on the circuit functionality or the type of circuit being under test.

So, for that it is totally based on design to design and a case to case basis. So, there is no algorithm as such or I can say there is no generalized architecture to do this. So, it has to

be handled in a case to case basis by the designer and the testing engineer targeting that particular design. So, as this not a very generalized way of handling those things. So, we are not going to look at the circuit dependent part; rather we are going to look at this circuit independent part of DFT optimization.

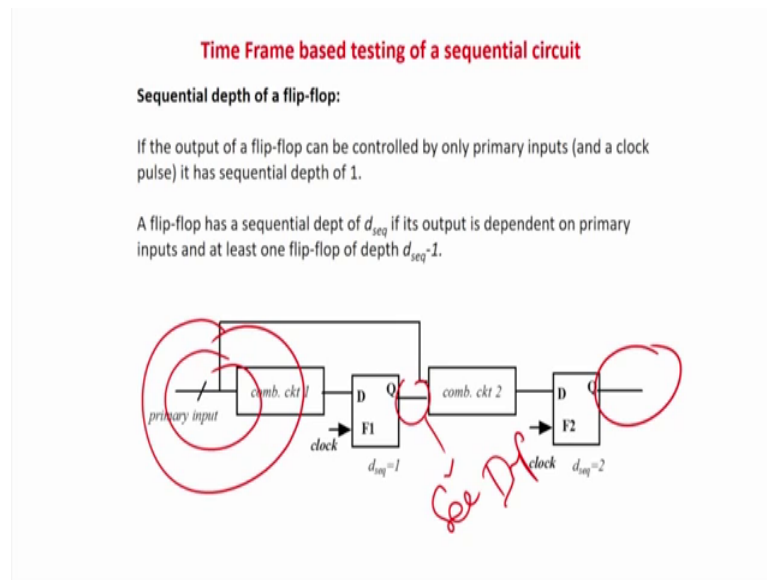
So, what are they? Optimization of scan chains. So, if you have looked at these circuits under test. So, we have seen one of the most important DFT circuitry is the scan chain. So, for every flip-flop we put in make it scan enabled so, that you can make them make a chain out of it. So, that you can very easily rode the test patterns in sequential circuit and get the response out of it.

So, first we will see what are the different ways of optimization of the scan chain? So, may in this lecture mainly we will be focusing on this and how to optimize scan chains you know that is one of the most or say that the ninety nine point nine percent area of your DFT circuit is taken by the scan chain itself. So, we will try to see basically how do you optimize scan business so, that we can load this scan chain faster as well as you do not put too much overhead in terms of the size of scan circuitry; because scan circuitry means you are putting some flip-flops as well as there should not be more pin outs.

Another optimizations are basically compression circuit decompression circuit convection circuit etcetera. So, that is another way that how can you compress the ATP test pattern. So, that your compaction d compaction all those additional circuits you put in which are actually helping in compressing the circuits and compacting the output response also does not become too large, but that is again I mean that is also lot of work has been done in this area also, but that is more related to some kind of floating theory etcetera.

So, in this lecture we are not going to highlight mainly on that mainly will be looking at the scan chain as I told you the irrelevant that almost there is no circuit at present which can be built without the scan chain. So, most test engineers emphasize on optimization of the scans right.

(Refer Slide Time: 06:25)



So, we will while we will go through these lectures will be automatically find; you will be automatically motivated and you will automatically be able to find out that, why scan chain optimization is so important, because scan chains is not only about the flip-flops which you put additionally, but also about the pin outs; one extra scan chain you put one additional pin out has to be there. So, therefore, optimisation of the scan chain has a greater role to pay in optimization compared to other cases like, compaction circuit and de compaction circuits decompression circuits etcetera.

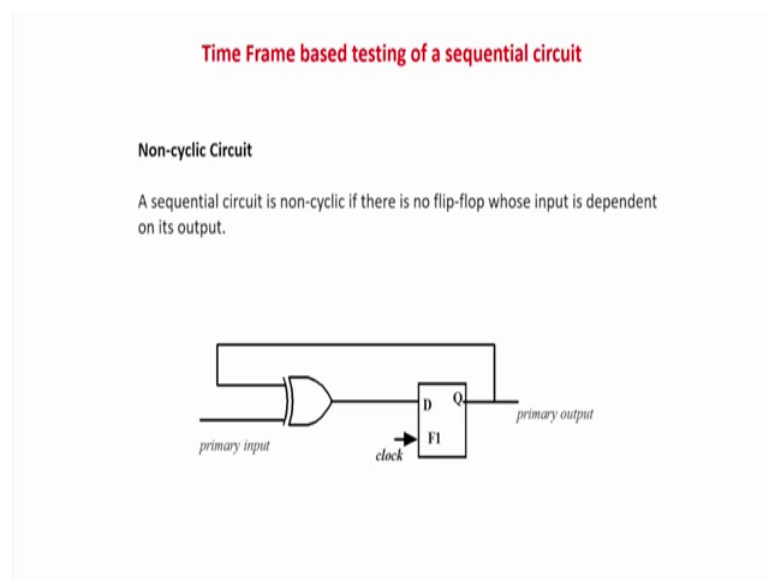
So, first let us look at some definitions. So, first something is called sequential depth of a flip-flop. So, sequential depth of a flip-flop implies that. So, if there is a flip-flop and there is a combinational circuit so this the primary input. So, how many D flip-flops are there in the path between the primary input and this flip-flop. In other words there is a flip-flop like; there is some combinational circuit and there is a primary input. Is there any flip-flop in between this flip-flop and the primary input; if there is none so this will level of one?

Now, again if I zoom here you will see; this another flip-flop another combinational circuit another primary input is here. So, basically between the second flip-flop and the primary input, there is one flip-flop in between in the power. So, therefore, level depth is actually 2. So, what is says; if the output of a flip-flop can be controlled by only primary input, there is the sequential depth of one; it implies that only primary input and a clock.

So, therefore, no flip-flops in between so, that is level one. And if one flip-flop come in the power level 2 we keep on going.

Now, what is the importance of level; because a flip-flop of level 1 can be controlled just by the primary inputs? As you mean that there is no scan or set reset lines; of level 2 how can you control? You have to control the primary inputs as well as you have to also control the flip-flops on depth one. So, to control a flip-flop at level 2 depth you have to control the primary input as well as you have to also have a controlling value of all flip-flops at level one; all implies whichever is lying in that path. So, therefore, that is called the idea of the depth of a flip-flop.

(Refer Slide Time: 08:22)



Next definition; just to keep these things in mind; so that we will be using it to understand, that what is the difficulty of testing a circuit or a flip-flop if the depth is at the higher level?

So, the of course, if all the flip-flops are the level of depth one, then all the primary inputs can be used to control that flip-flop. Of course, if some of the flip-flops are a very higher level depth, then first you have to do control level one, then you have to control level two, level 3 so, forth. So, that finally, the flip-flop at the jth level can be controlled. So, more the depth of a circuit, more difficult is to control those flip-flops to get a proper testing done. So, these are non cyclic circuit one very important thing that the cyclic flip-flop so; that means, what the input of this flip-flop again depends on the output; so

between a sequential circuits is non cyclic, if there is no flip-flop whose input is dependent on its output.

Therefore, this output is dependent on input itself, if such a flip-flop is there, this can may not be controlled. There are many cases they can be controlled, but there is a possibility that if there is a flip-flop which is cyclic it may not be controllable, but in other stories if the circuit is free of cyclism; that if no flip-flop in a circuit is cycling or the cycle is all the features are non cyclic, then they are always controllable. May be there is some people of at 1000 level depth; you may require 9000, 10000 depth; so 9999 clock where as pulses may be required and all the flip-flops from level 1 to 9999 level has to be controlled to control the final flip-flop which depth is level 10000, but you will be able to do it that is guaranteed.

Of course, we have seen that basic idea in this result that; first you control this one will control level 1, then again use the primary outputs, this is already controllable; so you can control this one anyway in that way manner you can do it.

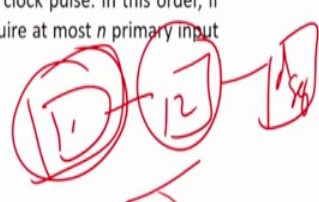
But, it is a cyclic flip-flop somewhere, then that flip-flop may not be controllable. So, there is only way to control such flip-flops either by scan or by set reset. So, that is the basic idea. So, there is a property, which says that the secondary inputs of a cycle free sequential circuit of depth d sequence. Secondary input secondary input means the output of the flip-flops basically something like that. So, if you see; so if I look at here, let me erase this; so secondary input means it is something like say the output is actually fitting the combinational circuit. So, these sometimes people call it as secondary inputs; the output of the flip-flops are called secondary inputs.

(Refer Slide Time: 10:40)

Time Frame based testing of a sequential circuit

Property The secondary inputs of a cycle free sequential circuit of depth d_{seq} can be brought to controllable value in at most d_{seq} primary input patterns and clock pulses.

Proof Idea: The proof is obvious. All flip-flops with $d_{seq}=1$ can be controlled by setting primary inputs and a clock pulse. Now, as all flip-flops with $d_{seq}=1$ have been set, we can control flip-flops with $d_{seq}=2$ by setting primary inputs and a clock pulse. In this order, if there are flip-flops with $d_{seq}=n$, we require at most n primary input patterns and n clock pulses.



So, what it says; the secondary inputs of a cycle free sequential circuit of depth d_{seq} can be brought to controllable value in at most d_{seq} sequence input patterns and clock pulses; that is, how basically if some flip-flop is there at d_{seq} sequence level, so this output can be controlled, and then you have to control all other flip-flops; which are at the level d_{seq} minus 1. So, you require d_{seq} minus 1 plus one patterns to actually control that flip-flop, that is; something like first you like control at the primary inputs then you control level 1, level 2 and keep on doing it. So, you are at the level d_{seq} .

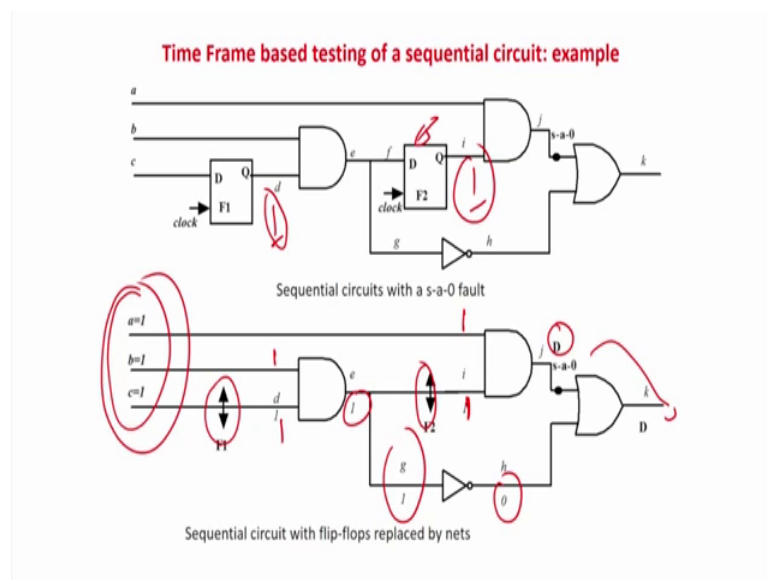
So, that is quite obvious I have shown you from the figure. So, and the proof is also given; the proof is basically obvious all flip-flop flip flop so, with d_{seq} one can be controlled by setting the primary inputs and a clock pulse. Now, level 2 can be controlled by say d_{seq} one thing controllable and the primary input see you keep on doing it and then basically you are going to see that d_{seq} number of primary inputs are required and clock pulses to control a flip-flops at the level d_{seq} right.

So, basically the simple idea is in a block diagram, we can say that is a flip-flop of level 1 this flip-flop at level 2 and this is a flip-flop at level d_{seq} so, ok. So, this one can be controlled in one pattern a primary inputs and a clock; this one will require 1 plus 1, because one pattern set will be required to control this flip-flop and then another primary set will be recording to do it, because there is something the output of this is feeding to

this flip-flop; and if you have to keep on doing it inductively so, that you require a sequence.

So, any flip-flop at the level d sequence can always be controllable by the test patterns whose number is equal to d sequence number of patterns will be maximum liquid d sequence and clock pulse has to be applied for each pattern, because flip-flops keep on moving the data on clock edges. So, that is; what is the idea? So, what it says in a very layman's language, that if there is a system or a circuit with flip-flops whose level is say k and there is no (Refer Time: 12:26) distance. So, in cyclic flops then they can always be controllable; so all the test patterns can be generated and all the test patterns can be put into that circuit and all faults can be tested, but it will take time.

(Refer Slide Time: 12:39)



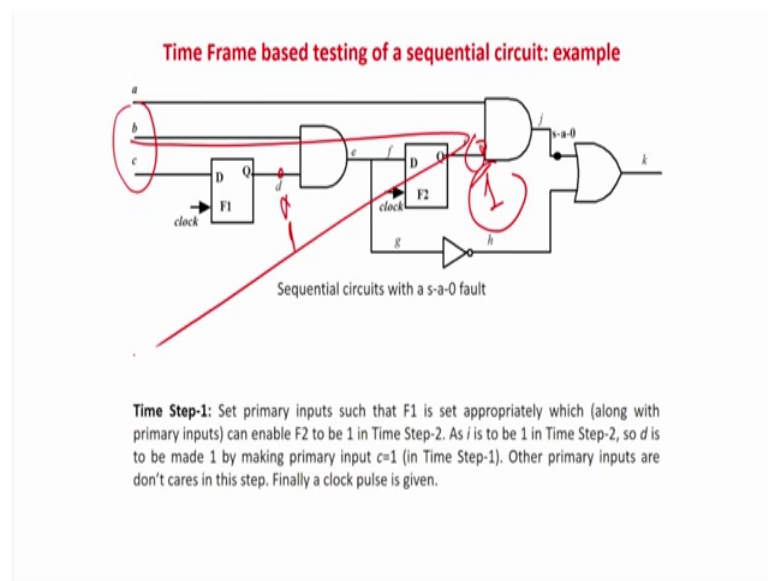
So, if for example, the circuit has some flip-flops with 20000, 30000, 40000 levels, then what we will do? Basically have to keep on pumping the values and everything will take some 10000 or 20000 time right, because some flip-flop develop 10000 so; that means, 10000 primary patterns has to be applied by 10000 different clock pulses. So, it will keep on first it will set flip-flop level 1, level 2, and level 3 and keep on doing. So, a lot of time will be required to pump in the values. So, and that is there.

So, now, if you say that; what if I use a scan chain? Scan chain will solve your problem, but see the time they get will be very high right, I will just give you a very simple example will actually try it I will try to make it scan chain actually solve the problem in a

very indirect manner the idea is that so, as I told you there is two ways of optimization, if you have seen my lectures on ATPG in this in this module so, two things; how difficult is test patterns and how difficult is to apply them.

Scan chain do not actually help in reducing the test application time, it only reduces the test pattern generation time it is a same (Refer Time: 12:48) factor effect, because test patterns are generated only for ones at the soft level and they have to be applied at all the circuits. So, test pattern application that is very very important compared test pattern generation, but on the other hands still if you circuit is very very large you will also like to optimize at the level of test pattern generation also, how as you have already discussed scan chain basically will tell you this point may be I want one this point may be I require 0 and so, forth.

(Refer Slide Time: 14:01)



Scan chain will directly tell you that I can do it you need not put any kind of mathematics or backtracking or sensitize propagate and justify to find out what are the values here; because if you remember if I do not have a scan chain, then if I want say for example, one here then you have to calculate on what values I have to give up the primary input so that I can get a one; then again what I have to do here so that I get a one. So, lot of sensitize propagate justify. So, all this graph theory traversal has to be done to find out how I can make one use this using this circuit which actually controls it, but if you are using a scan chain, so I can directly make it as a one (Refer Time: 15:34).

So, test pattern generation time did not do any kind of backtrack etcetera, directly it can say that directly I can make this pattern as one. So, test pattern generation time is highly optimized it scan chain, but test application time is not solved, because if there is a scan chain whose length is say 10000; that means, basically if you for example, I can say quote unquote, if some of the flip-flops are in the depth of level 1000.

So, this scan chain length also be 1000 because; that means, this last flip-flop is dependent on in the from primary input to the primary output of this primer sorry there is a flip-flop say whose depth is 1000; that means, if you take a path to the primary inputs, then there will be 9999 flip-flops which will be in the path. So, if you make a scan chain this scan chain length also will be of the level of say will be 10000 and also you will require 10000 clock pulses to feed it.

So, scan chain basically does not save your test application type it only help to generate test patterns quite faster, because you need not do all kind of backtracking that what values will be required to put this value at one and what at this gate what values will be required, because flip-flop will be directly controllable. So, therefore, now when we will try to see that what we can. So, the actually just a minute time frame testing. So, whatever path where we are saying that without use of scan chain still you can test, if there is no cycles.

So, therefore, we have to calculate what will be the value, then we will have to check for the primary input values. So, that the values can be propagated to the flip-flops and so forth; that is sensitizing and justifying and all this if I do the time is called time frame expansion approach, because the 100 flip-flop or a 10000 flip-flop will be set at the 10000 clock pulse of the 10000 time, but before that all the time frames has to be set up from 0 to 9999.

So, that the final test pattern can be applied at the 10000 e t region so, the time is called time frame expansion. In the other one you know is the scan chain base. So, time frame base expansion, it should takes lot of time for test pattern generation, but time up test pattern application time is same for both scan as well as your normal circuit without a scan where there is no cycles the idea is very simple. So, if scan also if there is a flip-flop, if there is a circuit bit left there were level 10000 depth this scan length will be 10000 as well as the normal circuit without of scan also will be having some flip-flops

are the level of depth 10000. So, you require test patterns and clock 10000 clock pulses to set it and instead of scan also it is the same story around the line.

So, generally code all code say that tests scan change may not be directly is helping you to solve the test pattern application time it only helps to generally helps to solve or minimize the time value required to generate the test patterns.

We will now let us take some examples so these are circuit as you can see and we have a fault at the output this one. So, now, there is no scan; so there are no flip-flops no scan. So, we are assuming that there is no scan chain over here. So, how do you do the test pattern generation by the time frame expansion method? Generally, what we do we first actually replace these flip-flops with some dash we assume that the lines are continuous, then will go for a ATPG.

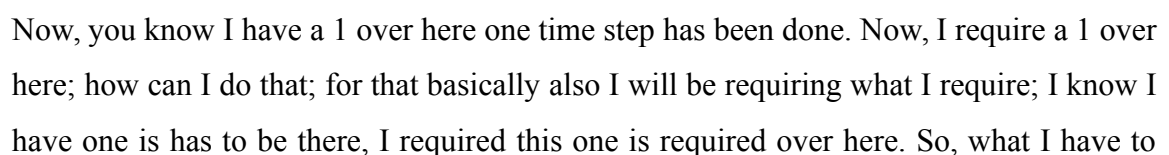
So, what will be the ATPG here stuck at 0. So, you have to apply a one normal case and fault it is 0. So, that is what basically normal one for this sensitization, that is going to give you the value of d over here, then I have propagated the value of at the output. So, I am propagating. So, sensitization is done propagation is done.

Now, if I has a (Refer Time: 17:45) or gate from justifying so d is a or gate. So, I will have a 0 over here, this one not gate 1; of course, this is a and gate so I require both 1, 1 here. So, I will one over here. So, anyway this is a 1 the one required fortunately there is no conflict. So, I come over here; so I require out this again the output of this and gate e equal to 1. So, I will be replacing a 1 over here 1 over and your job is done. So, basically this is the case. So, this is what is required to test it.

Now, what it says that this flip-flop this output also has to be a 1 and this also be a 1. So, I am doing the ATPG. Now you see this done the ATPG is done so if this scan flip-flop non scan flip-flop this ATPG has to be done. So, now, what it says that; the output of the flip-flop this has to be a 1 and this has to be a 1. So, it is the scan chain what you will do we will do nothing we will just say there was scan business. So, you can directly set them to 1, 1.

So, your ATPG is done, but if this scan is not there then what you have to do, then I have to think that how can I make d equal to 1, and how can I make I equal to 1, they has to be done in two time steps, because these this flip-flop that level 2 and this flip-flop is that

(Refer Slide Time: 19:21)



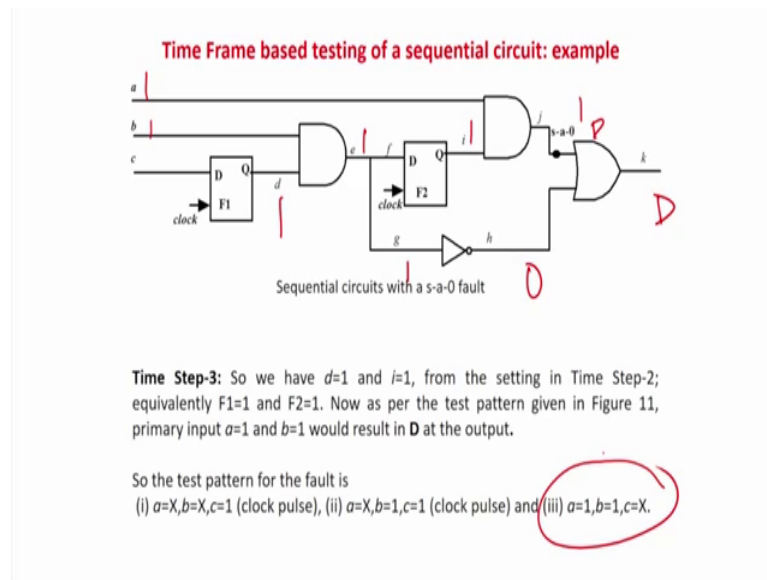
do; and also if you remember that next stage also it has to be a 1. So, what I have to do 1 has to come from here this 1 will be used to set it this is required.

So, this 1 will be helping to set it and, but when this 1 will go away from here I also still require D to be a 1, because final testing I require D has to be a 1. So, what I will do? Again, I will put a 1 over here in the second time step; I will require a 1 over here. So, assume if understand that some ATPG tool is telling you, because you will be applying it, but from the ATPG tool is actually telling you what to do when. So, first it has told you that basically you have to apply a 1 and you have to apply a block so 1 it will be there.

Next, what it will tell you; you put a 1 over here, apply a clock pulse; also you have put D equal to 1 a can be X at this time, why; because this 1 and 1 you apply, because this is one already one has been said you apply this one at the second time step, these are one and get then you again apply a clock pulse, this 1 will be actually reflected over here and this 1 will be replacing any one at the second time step.

So, you can see lot of extra computation is being done, which is telling you that first stage you apply X X 1 ATPG is telling you next stage you apply 1, 1 and a second clock pulse, then basically you will be having a condition of; so free stage ATPG it has become. First step basically it will tell you; what to apply have not been a scan chain business no more ATPG has to do anything automatic test pattern generation will say this scan is there, directly you said this flip-flop as 1 and 1 and 2, but we are assuming the at present there is no scan chain, so we have two more extra step the ATPG is telling you that first you have to do this and second time step you have to do it and once it is done you are going to have this value, then you will be able to test this circuit.

(Refer Slide Time: 22:04)



Now, what I have to put over here? I have to put a 1 over here, so 1, 1 is 1. So, the answer is D. Now propagation already we know D is 0, already we know that it is a 1. So, it has been done and b also has to be a 1, because I require D to be 1.

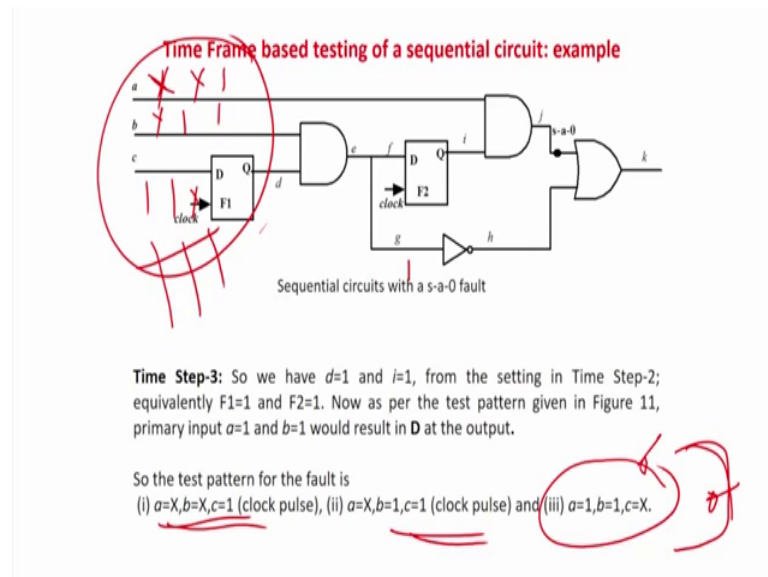
So, if you look at now what; now you look there are three patterns which has to be applied. So, I am zooming it out. So, that this is the final pattern which will test your circuit. So, first pattern is this one a X, b X, c 1; second stage is a X, b 1, c 1, clock pulse and this is the final one a 1, b 1, c I do not require and your circuit will be tested. In fact, this is the only test pattern which is required to test your circuit, basically which will be done if you are having a circuit without a with a scan.

So, if you are if you just take this, raw circuit, run for ATPG it will give you some pattern, and that is a single pattern will be told by the ATPG. So, your job is done assuming a as the scan is there; if you are not assuming a scan, so two port extra computation has to be done again I am saying our code our computation has been done which will tell you what to apply. So, finally, ATPG is a three step ATPG in this case. So, first pattern, second pattern and third pattern.

So, this ATPG is going to tell you and you have to do it. So, what we will do first you will take your circuit, and the test now you are we are saying test application. Now, I am changing the mode. So, this tool has ATPG tool has told you that these are patterns to be applied computed. So, with scan only one computation is required, because we know that

very easily I can set this flip-flop and do not require any computation for that, but now no no flip-flop is scan elaborate, so two port test pattern path it has told because there is no scan; so it is called time frame expansion method.

(Refer Slide Time: 23:45)



So, it has told me so first I have to put a X, a X then I have to apply a 1. Then next stage I have update one next stage again it is an X, but next stage it is an X I have to put a 1, 1 and a clock pulse and third stage I have to put 1, 1 and I do not require it. So, three patterns I have to put and as my test will be done.

So, this application, but what will apply basically is computation. So, it is a three part computation has been done by ATPG tube and just told me that this pattern has to be a plan, now I am happening. So, one thing I think is very clear to you at present that even a scan chain. So, I do not require a three stage ATPG to do it. So, you have to appreciate the fact that if there are 10000 flip-flops in the dependency list or the depth is 10000. So, this can. So, the ATPG will also have a 10000 step ATPG in the worst case. So, it will be ATPG test pattern generation will be very very complicated, but once say I have done that ATPG it has told me that these three patterns are required now I applied; so I am applying first this second this and this and your job is tested.

Now, again assuming that there is a scan; so how will you scan it? So, I have to put a 1 over here I have to put a 1 over here and then I have to put some 1, 1 or some value for testing. So, scan chain will also take two time steps: first I have to set this and then I

have to set this. So, test pattern application time is same both for scan as well as for a normal mode of circuit, because I have to set these two flops to 1,1 to make the values requirement of two to testing.

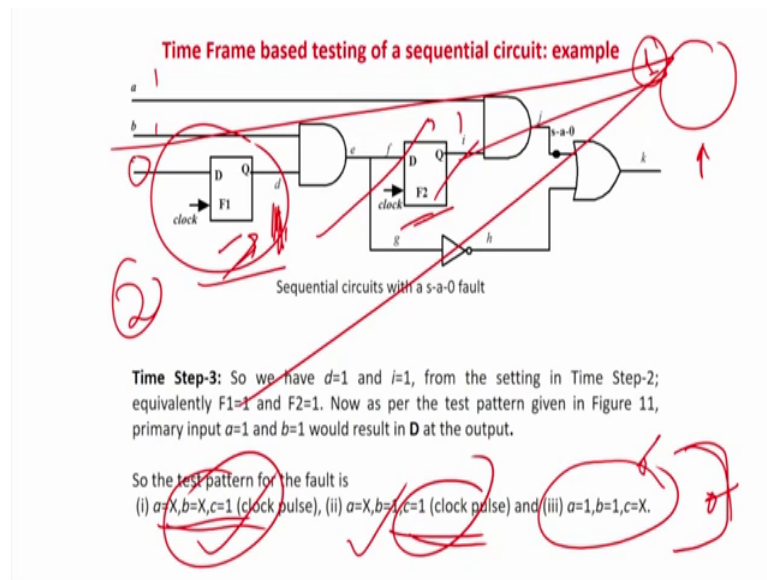
So, from this example it is clear that time they test the circuit using scan is same as the one without a scan, because in this case these two flip-flops are there I have to set it is also required two clock edges, and in case of a without a scan also I just required three patterns to do it. So, scan chain basically as I illustrated in the example is not that way saving your test application time; only they say saving your test pattern generation time, because in this case this two extra time frame patterns has to be generated, which is actually not good. So, if you have a very big circuit you will have some 10000 flip-flops in your line. So, test pattern generation also will kill you.

Now, important question arises. So, if your scan chain is not saving time for test application, because as I told you test pattern generation optimization is also important, but time of test pattern application is more important; why? Why more important, because test pattern has to be generated only once, but test pattern application has to be done for all circuits. So, it is very very important to think, how can we reduce the time; time to apply test patterns to all the circuit, how the test pattern number has to be reduced any way; that already we have seen in test pattern compaction etcetera we are not going it to right now.

So, now, the question is if the test pattern application time is not saved by the flip-flops then can you do something, because anyway test patterns are basically scan chains are taking more area over it because of your flops. So, what can I do something like this; some of it a flip-flop, actually I will make scan of course, which are very down the depth in line, because you will require a lot of ATPG computation to find out because as we have seen here. So, the level of this flip-flop depth is 2.

So, $2 \times \dots \times$ patterns we are required. So, can I do something like some of the flip-flops are very down the line in depth? So, those I will put scan, but some of the flip-flops which are very easy like; this one I really do not require any computation it is a primary input directly connected to get a one I just require a 1. So, ATPG that all to find out how I can get a one is very very straight forward. So, ATPG will not take much time to generate that.

(Refer Slide Time: 27:05)



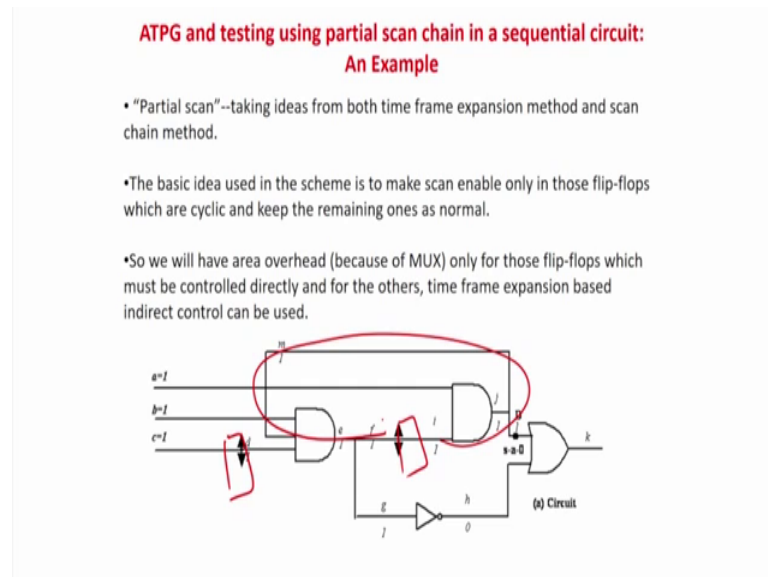
But, say there is a flip-flop very long down the line and I want to have a one in this. So, the test pattern generation tool will take lot of time to tell you, how I can get a 1 in this flip-flop and what are the values to be set down the line? So, such distance flip-flops or difficult flip-flops can be scan enabled, but such flip-flops which are very near to the primary inputs, which gone a very easily directly controllable and the test pattern to be applied does not matter much can we cannot can we can eliminate, first scan flip-flops such flip-flops from the scan enable; that means, flip-flops which are very down the line will be scan enable flip-flops.

We are very near to the primary inputs, which can be very easily directly controlled will not have any scan in that. So, this one will actually make a trade off difficult flop difficult to be controlled yes; you will keep scan, because anyway that is going to take some extra pins as well as you have to appreciate the fact that they will also be a lot of multiplexing arrangements for that, but flip-flops will be near the primary inputs very easily can be controlled we will not connect them to scan. So, we will save in the number of pins as well as we will save in the number of your basically multiplexers addition has to be put.

So, therefore, there is a some concept called partial scan chain. So, partial scan chain says that difficult to test patterns difficult fops down the line will be scan enabled easier ones to be set and reset will not be scan enable which will set using a time frame

expansion method; that is, what is called the concept of partial scan chain and which you are optimization technique for scan business. So, this whatever; I have told you in orally is written in this slide you can find read this and find out the actual interpretation of my top.

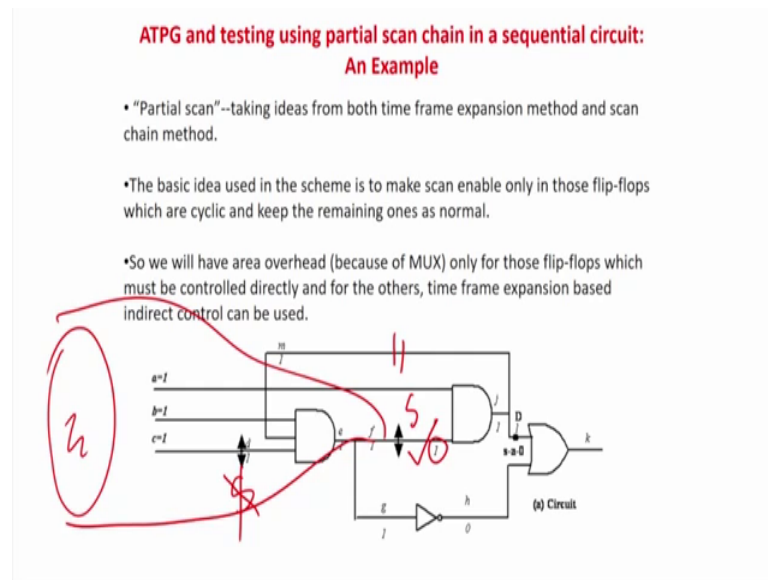
(Refer Slide Time: 28:36)



But, I will give you an example, again we always it is easy to illustrate with example some circuit is there and also of course, one thing I forgot; if there are any cyclic points. So, if there is cyclic point of course, their flip-flop has to be scan enabled. So, if flip-flops are very down the line difficult to control and also flip-flops which are cycling, because cyclic flip-flop anyway cannot be controlled by any means? So, there you put a cap.

So, that it can be directly controlled. So, in this case basically we have taken a circuit and there are flip-flops these lines are flip-flops I have not drawn over here. So, there are 2 flip flops. So, if you look at this circuit this is a circuit, where there is a cyclic business the output of this flip-flop is dependent on it is input. So, these are cyclic flop so; obviously, this there has to be a scan for this you cannot see. So, this should be a scan flop.

(Refer Slide Time: 29:23)

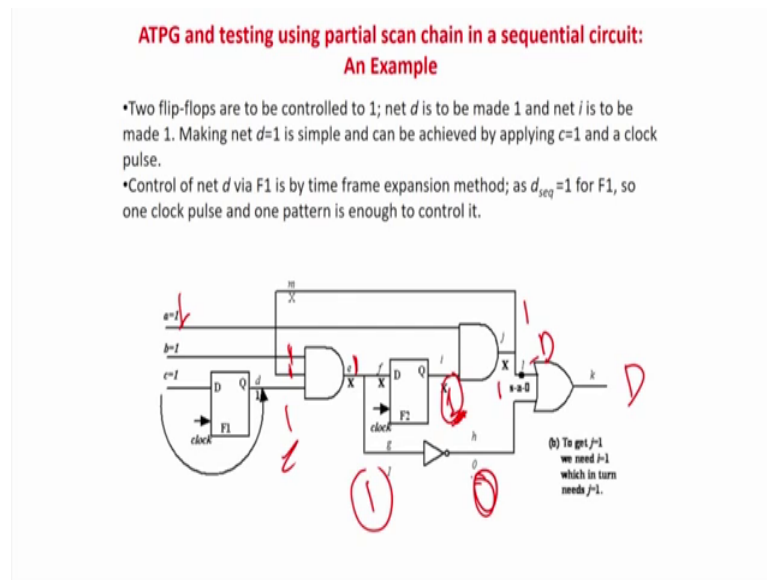


But, this flip-flop if you see directly connected to the primary input very easily it can be controlled. So, whatever value you require can be directly obtained from this there is no requirement for any computation to find out the value directly you can see whatever in c will be reflected at t . So, do I require a scan for this flip-flop no. So, a partial scan I will make this a scan enabled and this has not make a scan enabled of course, it is a cyclic flop you have to have a scan, but also there can be a story that these flip-flop is around depth of some 20000 very difficult to control.

So, ATPG will also take long time to find out a value to be controllable. So, we assume that is not cyclic scan, but a lot of 10, 20000 gates are there and say that I require a 1. So, for the ATPG to calculate or the ATPG 2 to calculate that one I should do what values I should have here to get a one here it will take long time to compute these values at the inputs. So, that resource also I can make scan.

So, to rule of thumb flops which are near the primary inputs do not make scan enable flip flops, which are cyclic or down the line in depth make them scan. So, that is actually called a partial scan we will take an example this example will now elaborate. So, for example, stuck at 0 has to be tested over here. So, I have to put a 1 stuck at 0.

(Refer Slide Time: 30:36)



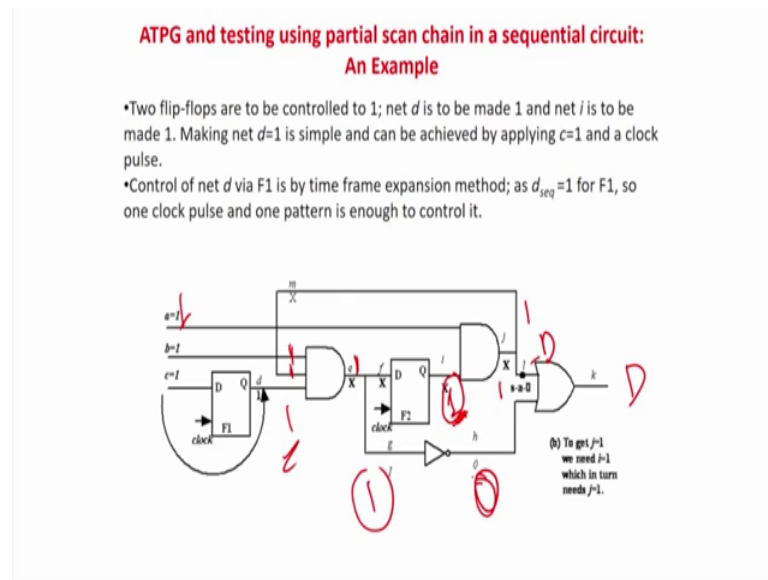
So, I have to put a 1 over here. So, the answer is D propagation will be D or gate the answer will be a 0 over here the answer will be a 1 right; then basically this is a 1nd gate. So, all these values have to be a 1, because I require a 1 over here and basically I require a 1 over here to test it. So, a has to be a 1 and of course, this p f has to be a 1. So, basically you require a 1 at this flip-flop you require a 1 at this flip flop.

So, very simply you do it you will get the idea. Now, basically you see that this is a cyclic flop I require a basically I require a 1 over here and this point basically is 1; this is 2 these are stuck at fault. So, now, as the cyclic fault cyclic flip-flop I cannot do this control. So, what I will do I will make it as a scan business. So, directly I will set it to 1, because whatever value I set over here is again dependent on this see I require a 1 over here and from this is like m actually required to be a 1 to make this output as 1, but again this 1 is again dependent on this is a loop very simple that this story is a loop on this.

So, to avoid this loop so, what I will do? I will actually make this F2 as a scan flip flop. So, I can directly set it as a one. So, what the arises; it is the one. So, basically and this is also a 1 a is also 1; this is directly set as 1. So, your output is going to be a 1. So, now, in this case it is a 1, 1 and this has to be a 1. So, you will also get a 1 and this 0 can be updated.

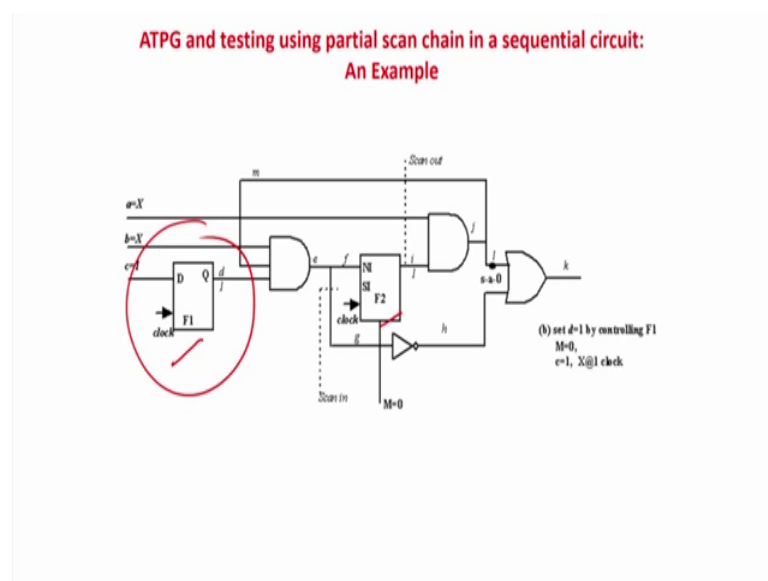
So, what I will do is that; I will make F2 as a scan flop and this I will not make as a scan flop.

(Refer Slide Time: 32:06)



So, again this one is illustrated over here. So, this is actually I am make this is a scan flop. So, now, I will make it as a scan enable. So, you can see over here I zoom it. So, this is a scan flop for this scan e scan out. So, I have made this value as a 1. So, this can set has already been done as a 1. Now, next is this is already a 1 has been set by scan. So, you can see this is the scan enable has been done. So, this 1 has been set you can see this output is being decoupled. So, this is scan has been enabled and this value has been set with the clock pulse.

(Refer Slide Time: 32:35)

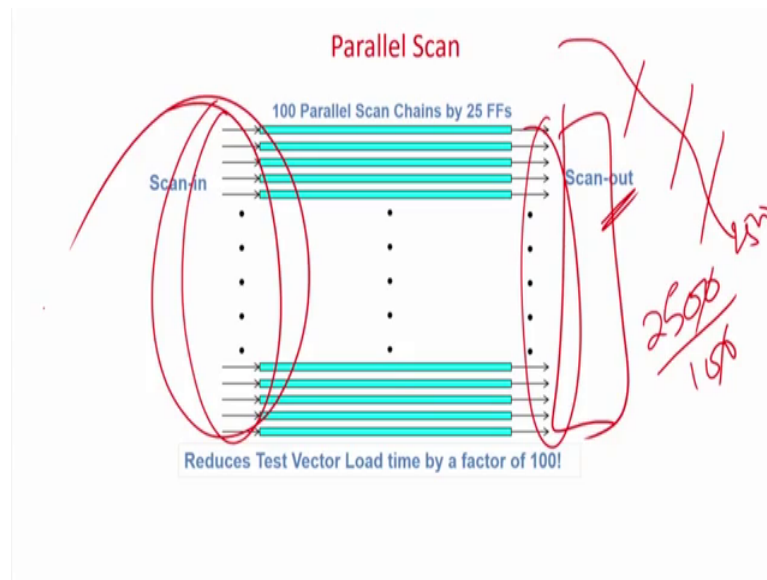


Next this you can zoom over here and see this can has been disconnected. So, the value is 1 over here and, then you put a c equal to 1 and you apply your clock pulse. So, now, interestingly this enabled by scan and c equal to 1, our D will appear over here. So, this part is actually non scan this part is they are must know time frame expansion method. So, see even if they do not be in a double scan, if even if you have made this and this as scan enable till two clock peers would have been required to set this as a 1 and this as a 1, but what I am doing is that I am not setting it as a scan enabled, because is very easy to know that if I require a 1, a c can be set as a 1.

So, ATPG or two does not require any complexity. So, I do not make F1 as can enable and I save something on, what do I called your DFT cost, because I there is no multiplexer over here, but here there is a multiplexer over here. So, next is the; what that has been done. So, already one has been set I put D equal to 1 and now my job is done. So, you can just have a look at it. So, after that I require a equal to 1, b equal to 1. So, if you do that basically this is 1. So, this is also 1. So, you are going to get the value 1 and this is a 1 inverter 0. So, you are getting a 0 over here and already you see a equal to 1. So, which is going to this gate and basically j i equal to 1 already set by the scan flop. So, this is one and a 1 is 1 D and the value is propagated and your job is done.

So, basically there are three steps in this one set the flip-flop F2 by scan to be one set flip sorry set flip-flop to a scan mode make it one set flip-flop one as non scan mode using a time frame expansion approach c equal to 1 you get D equal to 1 and finally, apply the pattern 1, 1 and we are going to test this one advantage that this pattern this flop is not scan enabled. So, you are setting on the multiplexing cost and these have to be made scan enabled, because it is a cyclic flop also F2 could have been a flip-flop which is very down the line of depth some 10000 to 20000. Then the same approaches which I which we will do.

(Refer Slide Time: 34:38)



So, basically that is, what is the idea; that is one way of optimizing this scan chain, which is very very popular is partial scan. Select reception of flip-flops are easy to control, do not make them scan enable. Select another set of flip-flops which are very difficult to control or cyclic make them scan enabled. So, there should be a trade off. So, the trade off means we are saving in the number of pins; as well as we are all setting on the number of ATPG computation time. So, this trade off actually is an optimization criteria, you have to decide that, which to be set scan mode which set to be non scan mode and there is an optimization.

So, that is why; frankly saying that all the flip-flops will be scanned enabled in a very very large on a complex circuit we put a lot of extra multiplexers and utility up your area and see, if you say that in a very very weak circuit, I do not want to put anything in scan will actually kill you in the test pattern generation time. So, there should be a trade off for optimization.

Next, problem again we actually told you that scan chain does not help you in reducing the test pattern application time, because if a scan chain has 10000 flops 10000 time clock pulse required to be fill it, how we can optimize it? There is something called parallel scan. So, what do you do? So, for example, say they say for example, there is 2500 is the chain length say there is a chain of flops whose depth is 2500 scan it you require 2500.

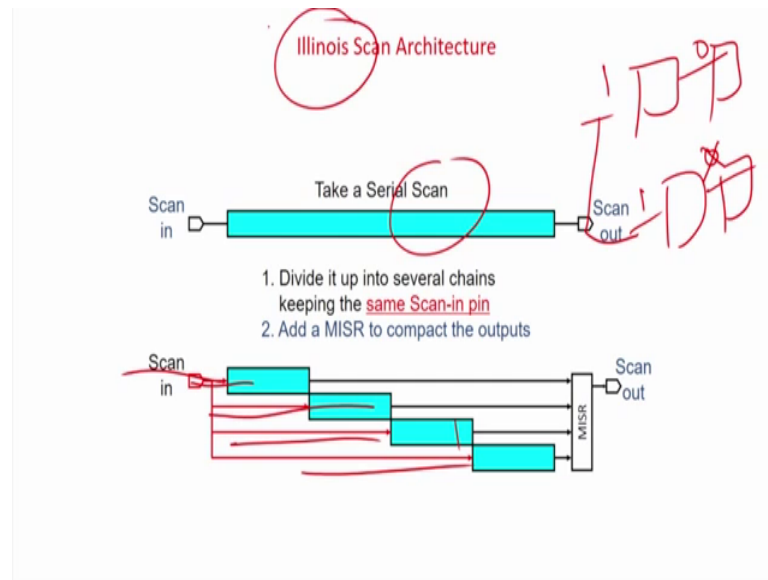
So, what you can do you make parallel you cut it. So, all the flip-flops have been paralyzied into 100 different parallel scan chains. So, instead of one long scan chain I have partitioned. So, now; obviously, these 100 parallel chains can be data can be sent in parallelly. So, what you are saving; so instead of 2500 clock pulses to do it; you will just now require by divide by 100, 25 clock periods you will be able to set up all the flops; obviously, long chain you cut it you feeding parallel. So, that amount of speed up you will get, what is the cost you will pay in the number of inputs p.

So, in one scan chain you would have only 3 pins scan in, scan out and scan control here in this case you will have 200 extra pins, because for each of these scan in and sometimes actually somebody put a compaction here, because this does seem very much interested to load the circuit output response a compact in an and insert it does not exactly talking about scan out, but whatever you do parallel scan load means at least 100 extra pins you have to require to load them parallelly.

So, as I already told you many times that having many extra pins loads, this is a very difficult to have been a packaged circuit as well as that much pin you have to have to have in your automatic test pattern generator machine. So, that is what is it is reason test pattern compression. We have already seen we try to reduce the test patterns by this week; so that is the number of pins of your ATP automatic test equipment reduces.

So, again I told you there is lot of trade off. So, for very large circuits or circuits which have a very long scan chain we will try to parallelize that is, but how much depends on what are the amounts of time you will save and what is the extra amount of p? Now, you can spare if you have lot of pin out to spare you make very large number of parallel scans. So, you can load faster, but of course, your pin outs will be higher. So, there is another way of optimizing optimization to find out basically, what is the number of extra pins you have and what is the how much speed you require to load?

(Refer Slide Time: 37:41)



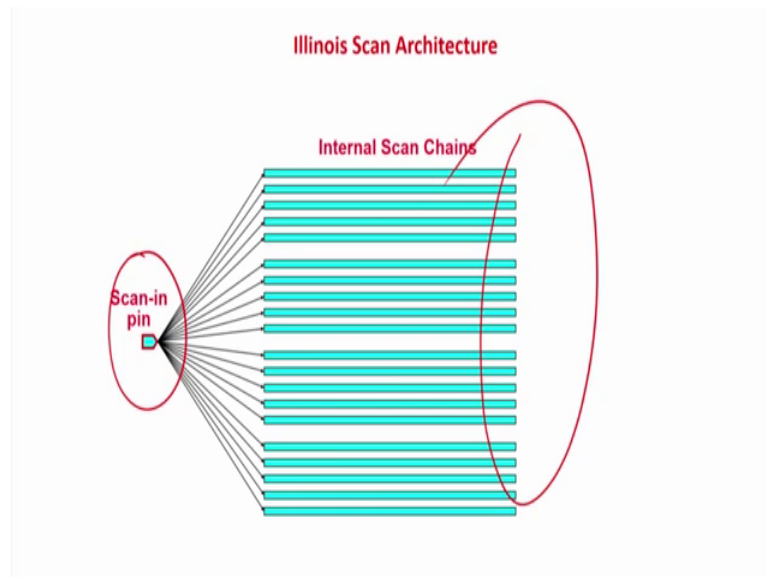
Of course, there is we do not talk about external pin outs is 100, because of the fact that they do the self compaction interesting something like, say that this is a big scan span can chain over there I partition into 1, 2, 3 and 4; I am away partition. So, I require 1, 2, 3, 4 pin outs to do it, but can I or reorient or organize this circuit in such a fashion. So, that I will have a single pin load out of the circuits means can I shuffle out all these can selves in such a fashion.

So, that and I partition it in very nice scan chain may not be of equal length. So, that the same values can be given in all the time say for example, I have 2 scan flops here I make up 2 scan lines for example, I require 1, 0 over here and at the same time for equal 1, 0 over here, then I can have a single scan in p, because I am suffering this can change in such a fashion.

So, that here also I require 1, 0 here also I require 1, 0 of course, it can also be 1 X dependent. So, again compatible vectors are coming into picture. So, can I do something like that can I reshuffle these scan in such a fashion and can I put these scan chain cells can parallelise the scan chain scan chain is a fashion in such a way. So, that same values can go in this, then what it will help you it will speed up the application time of course, the parallelism is there and you do not require a number of scan pins problem is pin out is very very difficult to have this.

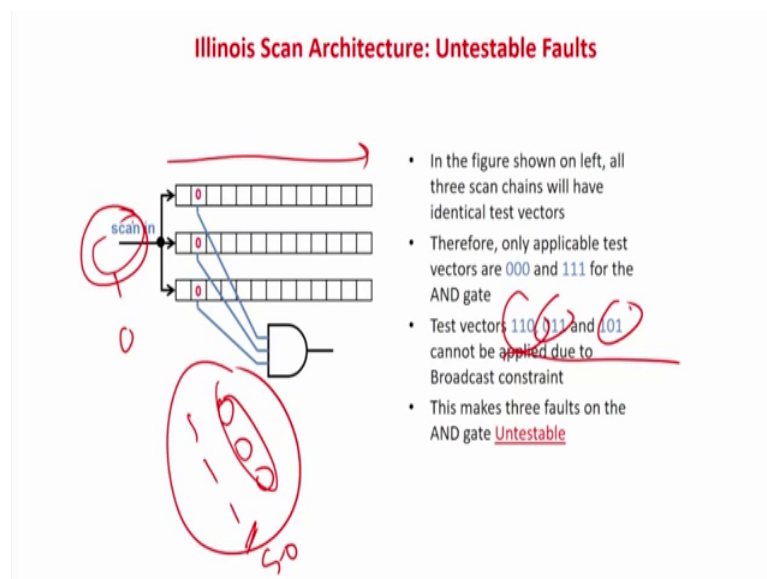
More pin outs very difficult in a packet at the same time a t e will have lot of pins. So, that is actually called the Illinois scan architecture which is one of the most hounding discovery for scan chain or DFT optimization that, what do you do?

(Refer Slide Time: 39:13)



You have tried to find out this is the most generalized good there will be a single scan and internal scan chains I arranged in such a fashion.

(Refer Slide Time: 39:33)



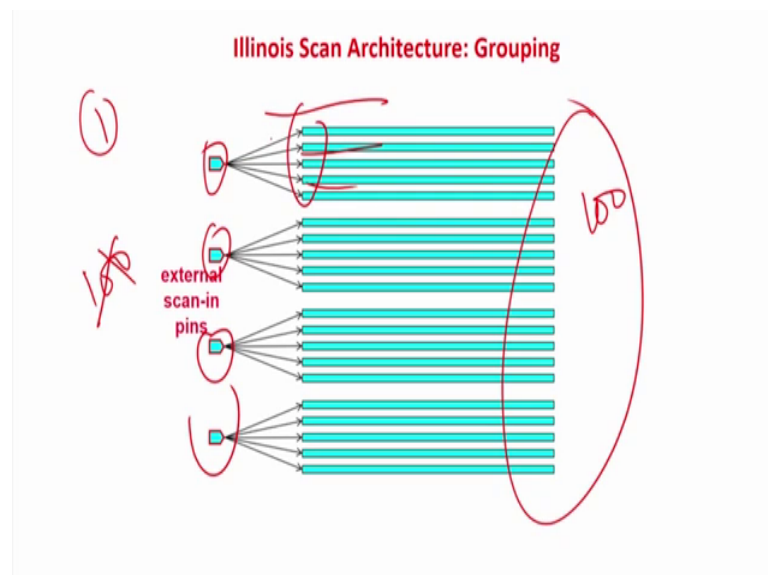
So, that the same test values can go in all the scan chain and your job is done, but this is actually our dream which can never be true, because I will give you an example say

some exam for example, this is some paralysation you have done this and gate and same values goes in, because I have only one scan you always want to reduce the scan things you spin out more pin outs more problem more a t e pins, but you see then only I can have two patterns that you can apply all zeros and all ones of course, because same input either input can one or the input can be 0.

So, this and gate can have only two that test patterns 1, 1, 1 and. So, many faults will remain untestable of course, because as we know so, 1, 1, 1 will test for all stuck at zero faults, but 0, 0, 0 will test only of the few fault few test factors like this and this are actually required to test all other flops.

So, many other faults which will not be done any way that you can easily visit the first lecture on testing and you will find out. So, the idea here is that as a single thing. So, I can either put 1, 1, 1 in this gate or 0, 0 in this gate, but other patterns will not be able to applicable. So, some faults will not be able to be tested and we do not require that situation we are actually having very less pin very good scan chains are loaded talent, but still I do not want to compromise on test. So, what I mean. So, this is basically knows Illinois architecture.

(Refer Slide Time: 40:43)



So, again we trade off in between that is one in intelligent grouping that of in among the all these caches will reshuffle as we make some groups and in those groups you can at least put the same test patterns. So, these call actually Illinois scan intelligent grouping

architecture. So, that the number of pins is not say there are 100 scan chains, here the number of pins will not be 100 and also it will not be 1, 1 is 1 extreme and 100 is another extreme will have something 10 or 12 in between.

So, that now the scan chains are grouped. So, that in each group you can put the same data and no faults will remain not tested like, that is actually concept of scan compatibility has come.

(Refer Slide Time: 41:18)

Illinois Scan Architecture: Intelligent Grouping

- Compatibility between two scan cells
 - Two inputs (scan cells) are compatible if and only if no fault becomes untestable as a result of tying the two cells to a single input
- Compatibility between two scan chains
 - Two chains are compatible if and only if every pair of scan-cells that receive the same broadcast value are compatible
 - Determination of all pairwise compatibilities is computationally very expensive

It has very similar concept to test pattern compatibility; that is two inputs scan cells are compatible in a if and only if, no faults become unstable as a result of trying the two cells are single input; that is a scan sailor type and they will always get the same input. So, there should not be any fault that is non testable. Similarly, this cell concept you can also move it to scan chain and you are going to get the same thing, that is two strands sense as compatible such that; if you put they can always get only the similar pair same pattern will go through this can check, then if they cannot have different input patterns, but still no fault should be untested then they are actually called compatible.

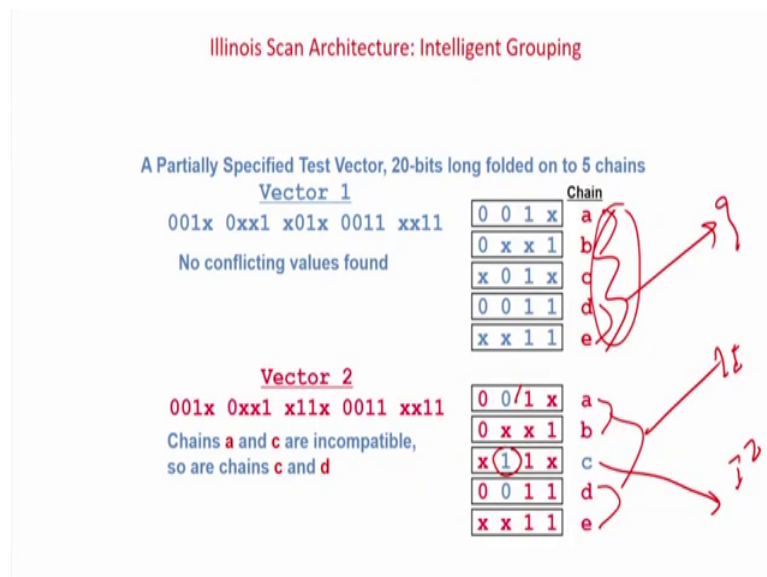
So, determinations of all possible compatible scan chains are extremely difficult problems. So, there are a lot of heuristics to solve it problem, anyway that you can you are free to read it as I am already telling this course actually directs towards optimization. So, we cannot tell you about all the heuristics how to solve it; they are all available in the research papers; so the idea I think is now clear to you, that somehow we

have to reshuffle these scan cells sometimes you have to make some groups such that scan chains which are compatible are in a single group.

So, now, we will not have as simple as a single p and take a long time to load the circuit also will not have 100 pin outs have is a very infeasible situation and you are loading all this scan chain back you will have these are something in between.

Now, interestingly will have nice grouping, so that in those groups at least you can send this scan chain parallel using a single pin not true, there is something called a what I told you is very similar to compatible patterns or compatible ATPG which you have already seen in the last lectures that if two test patterns are compatible then they can be much similar idea.

(Refer Slide Time: 42:40)



So, basically there are two test vectors like this and the second vector is this. So, you can see if you look say example I am or partitioning into 5 scan is this a, b, c, d and e. So, you can always see that a, b and d. So, they are all compatible chains you can just see that is a 0, 0 and basically this is also 0 x, 0 x, 0 x.

So, this is 1, 1, 1, 1 so, you can easily see that a, b, d and e compatible chains; that means, I can have a single input for this whole thing second club it as a 1, because there is no conflict, but if you look at c; I think if you look at c there is a. So, in this case no conflicts sorry in this first case this everything is compatible this is a full compatible

chain. So, I can connect it you can see there is no conflicting pattern. So, all a, b, c, d can be supplied from a single chain, but here if you look at in this case if you see a and b there is no compatibility problem c and d also there is no compatibility problem.

So, these two I connect angle, but only problem with this pin c. So, in this case if you see this is 1 and the other are 0. So, these are non compatibility. So, I have to supply scan chain c with a separate p and a, b, d and e I can club and I can put data through a single chain single input. So, this is input one and this is input two, but in the first example all these cans are compatible the single input will actually solve the purpose. So, this concept is very similar that basically, now again this is not as simple as that you know always your scan chain not be of equal length.

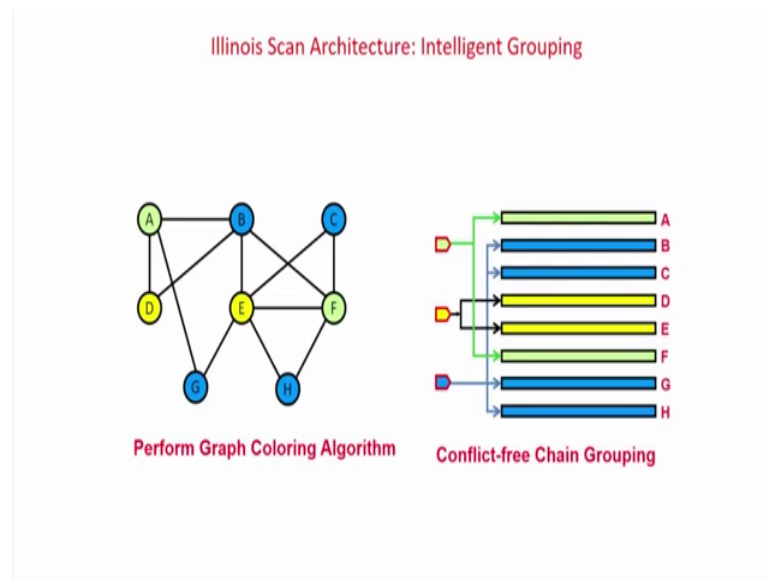
So, slightly up and down all these a lot of tradeoffs actually I am not going to go about it, but how do I exactly select the scan length how do you partition whether in file or in terms of 6 or 7. So, there are lot of tradeoffs to find out that; what will be the part, because means one group can have length of 5 and scan chain length 5, the second group chain have scan chain of length ten the third group can have scan chain of 200 each.

So, how exactly partition and what is the most optimal we are partitioning. So, that the number of pin outs is minimized as well as there is no fault remain untestable and we can solve this by the compatibility purpose this is an n p complete very tough computational problem and you cannot find an exact algorithm to do it will take exponential amount of time. So, there are lot of heuristics to do it.

So, one of the parameters how many parallels can; will have how many groups will require in each group which scan cells will be there and what will be the length of the scans in each group. So, lot parameters are there idea is that you should minimize the scan pins and no falsehood remains untestable this is possible if and only if your inputs to a group are compatible that is what is the idea.

So, that is the problem lot of heuristics us there you can read it, but also you can find out the most; worst kind of example will be you try all possible combinations. So, that is very difficult to solve anyway should not think about it, but in this slide I think I have tried to give you the idea, what is the problem and what the solution would look like very simply you can easily map to a very well known called a glass colouring problem.

(Refer Slide Time: 45:43)



So, basically you think that A, B, C, D are all these scan groups or scan chains basically, in this case there are 8 scan chains, sorry; the 8 scan chain. So, 8 scan chain will be a basically your node and I have a point in between A and B. So, there is an edge between A and B, if they are they are conflicting. So, A and B are conflicting so I have an edge. So, that A and B cannot be supplied patterns you know by a single pin, because their conflicting patterns.

Now, you colour your graph. So, of course, if you see that if you have an edge you cannot have the same colour; that means, you cannot have a same pin to supply it; you do it with minimum colour and for each colour you will have a single pin like A and F they are basically compatible test patterns, because there is no edge in between. So, you solve with the graph colouring problem with n p complete problem, then you can find out how many minimum graphs; minimum colours are required to do it, along with that you can make group accordingly.

But again very big circuit the graph size will be very large and we also know that rough colouring is an n p complete problem. So, a lot of heuristics are there you can use the few districts to solve this problem and then you can find out, what is the basically your groups and each group will see what will be number of groups and each group will have a single pin and this optimization problem can be solved.

So, that is actually called the Illinois scan architecture intelligent groups. So, the problem is very simple to design very simple to map, but solving will take a long time, because in such big case of big service the graph itself will be large. So, people follow different type of heuristics to solve the problem.

So, now so this is about basically your flip-flops scan chains partial scan and of course, with Illinois scan architecture you can also have a partial can business just making the problem more interesting and more practical, but anyway first we have seen scan, then you have seen partial scan, then you have seen how scans be paralyzed, and then you also seen how it can be optimized the; in terms of number of input pins as well as you should not have too many input and output pins for your scan still you should not you should have parallelism and the same time you should not make this faults untestable.

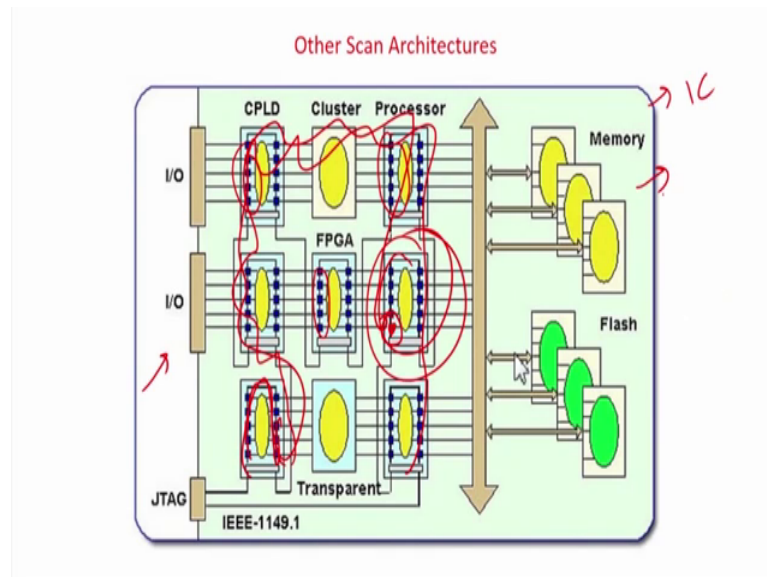
So, taking everything is an optimization problem, which can be solved in this manner as a graph colouring problem. So, this mapping I have shown you. So, this is one way of optimizing the scan and as I told you that most thing most people talk about DFT scan actually plays the role of more than 99 percent of area or you can say algorithms or your space complexity or the main heart of DFT is scan chains. So, therefore, lot of research has been devoted to find out how we can optimize scan chains as well as reduce the test application time and do not make any taste bad they are false untestable. So, the Illinois scan architecture is one of the most happening algorithm or architecture for that, which we have mainly discussed in this architecture in this lecture partial scan and Illinois scan.

So, apart from them before I finish the lecture there are different type of other scans also has been proposed and I am not going to them, but you should know the terminologies, which is actually called the boundary scan I will give you an idea. So, basically this is achieved so you can apply all the input output patterns, but only thing is you have found out that the internals of the flops cannot be controlled, because they are quite inside the circuit and it is very difficult to control so, we are using scan.

But they something called boundary scan say for example, there is a big like you have seen your motherboard. So, in this case you see 1, 2, 3, 4 so, many processors. So, many I c's are there. So, this pins basically I can say control this thing I can control this mean I can control if this chip is separate as a block, but now say all the chips has been put in your motherboard and some faults develop later; how do you test it. So, in that case there

is something, because you cannot put this chip out and test it you cannot do that. So, rather now you think there is something called a boundary scan concept of boundary scan which is what IEEE 1149.1 standard.

(Refer Slide Time: 49:11)



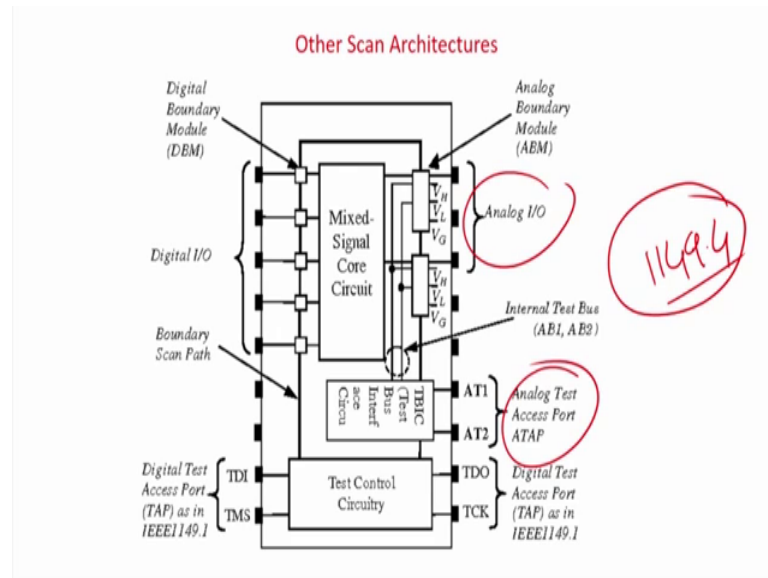
In which case all the input pins of this of all the chips are also having a boundary scan. So, what is the boundary scan is similar to a normal scan architecture? But in fact, this is talking about the chips and a system board in the board you have put all the chips. Now, say I want to access this it is very easy to access, if this chip can be brought out and put in the resistor, but once has been fabricated tested and put in a board after that some problem happens how do you detect it.

So, how do you control it? So, there is a scan chain, so then it is called boundary scan. So, whatever values you have to put you have to pass by this and this value will be applied. So, they similar till now instead of a single circuit the whole concept is a boot. So, instead of a flip-flop the circuit has become your flop. So, this is a board and these are your chips.

So, there is this concept of scan chain called a boundary scan it is very synonymous to what we know about a normal scan chain. So, if this is not a board, but an IC and then therefore, you can think this is this point is a flip-flop you can only control it from, but you cannot directly control. So, basically use a scan chain, but when the same thing is mapped to a board and lot of IC's are there we actually call it a boundary scan and we

call 1149.1 just the terminologies we are not going into the depth, but because the sake of completeness whenever we talk about scan and DFT is a very important standards. So, the sign just telling you the need; now if some recent ones have come out which is called 1149.4. So, in this case what happens? So, in this case is the digital circuit. So, you can do not require any kind of analogue things.

(Refer Slide Time: 50:40)



But, if there is something called to configure a, d, c is there. So, all those testing of analogue components are there. So, there is something called analogue pins. So, analogue tests pins. So, if there is some error it has some analogue compatibility. So, that scan architecture is called 1149.4 architecture so, these are some terminologies.

So, therefore, it will also have some analogue components to be tested. So, then and then after that there is some standard called p 1500 and so, forth. So, this scan architecture is also continuing not only in the cheap level, but also in the board level. So, basically with this come to the end of this lecture in which we have talked about mainly optimization of DFT circuits in terms of scan chip mainly looking at partial scan noise scan that is what is the core idea and next lecture we are going to see some more concepts on test optimization.

Thank you.