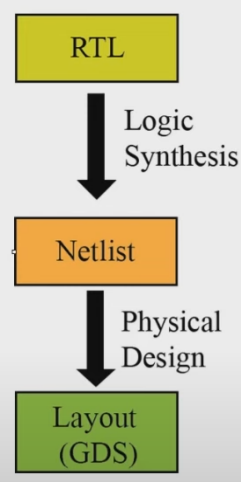
VLSI Design Flow: RTL to GDS

Dr. Sneh Saurabh

Department of Electronics and Communication Engineering
IIIT-Delhi

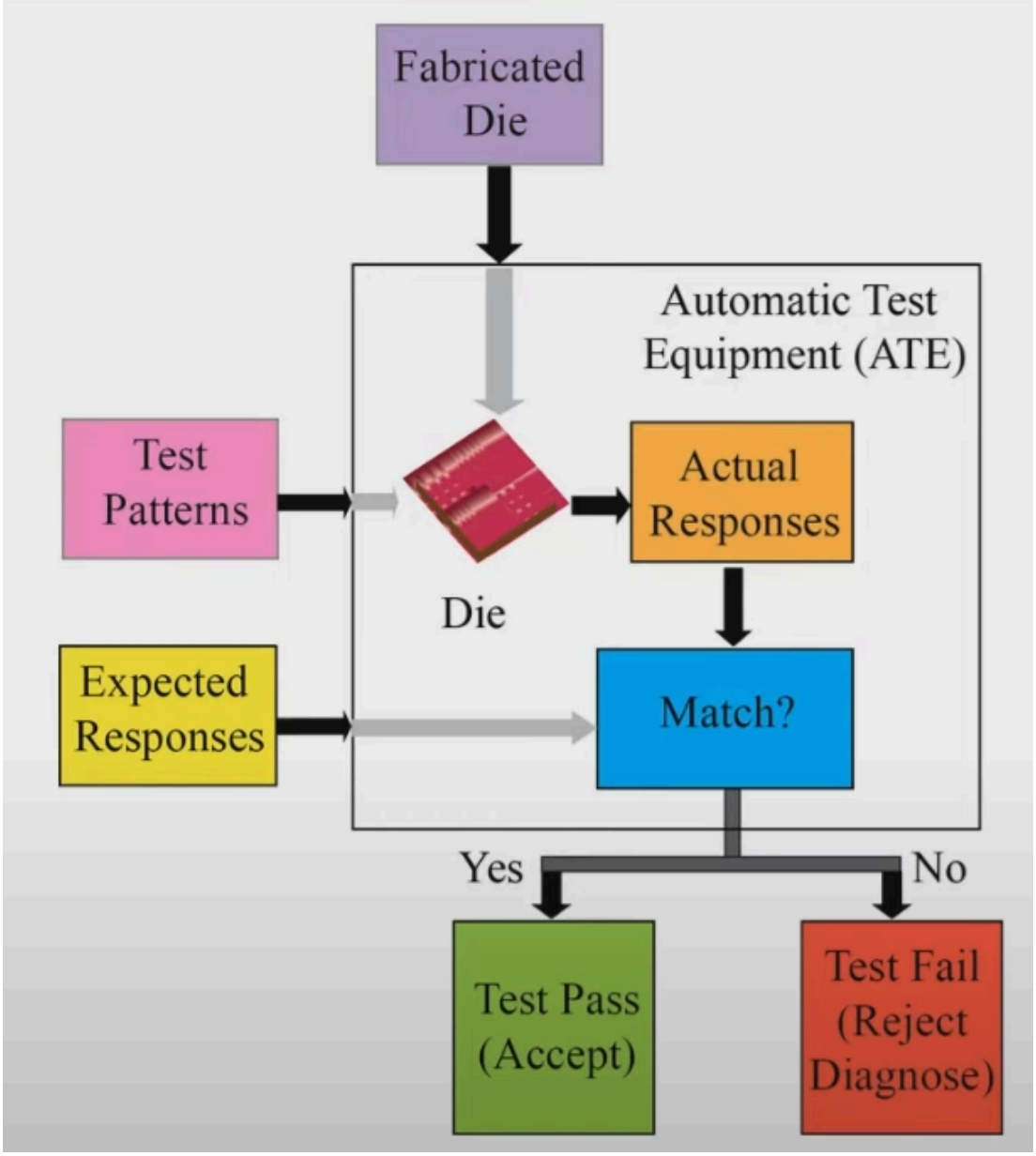
Lecture 39
Basic Concepts of DFT

Hello everybody, welcome to the course VLSI Design Flow RTL to GDS. This is the 31st lecture. In this lecture, we will be discussing some basic concepts related to design for test stability or DFT. Before proceeding further, let us recap what we had discussed while taking an overview of testing.



So we had discussed that there are some design tasks which are carried out during the design phase that is while we are transforming our design or implementing our design and taking it from RTL to the final G of final layout during those during the design step or during the design process, we carry out certain design tasks which are basically targeted for testing and these design tasks make testing more cost effective and efficient. So these design tasks which are targeted for testing are grouped together and called as design for test.

Now before proceeding further, let us also recap how testing is done. So the most popular method of doing testing is using automatic test equipment.



So in this testing method what we do is that we take the fabricated die and apply test patterns on those fabricated dies with the help of automatic test equipment and see the actual responses that are obtained for the fabricated die. If the actual responses match the expected responses then we say that the given die has passed the test and this die will be allowed to go to the next go to the customer or go to the packaging and so on.

If there is a mismatch between the actual responses and the expected response then we say that the testing has failed and then we reject that die and that die does not go to the customer or is not processed further. So this is how the testing of our manufacturing test is done. Now for this manufacturing test to be a more testing process to be more efficient

we need to make some changes in our design. We need to insert some logic and test structures into our circuit so that our circuit becomes more testable. So this process of making changes in our design is related to or the design task which makes these changes to our design those are known as design for test or DFT task.

And the other other important step that is carried out during or during the design process is to extract test patterns which need to be applied to the fabricated die and also the expected responses . So these are some of the design tasks which are targeted for testing and in this lecture and in the next three lectures we will be looking at these design for test or these these tasks which are targeted for testing or DFT task . Specifically in this lecture we will be first looking at some basic concepts related to DFT. Now for carrying out testing we use a testing method which is known as structural testing. So let us first understand what structural testing is.

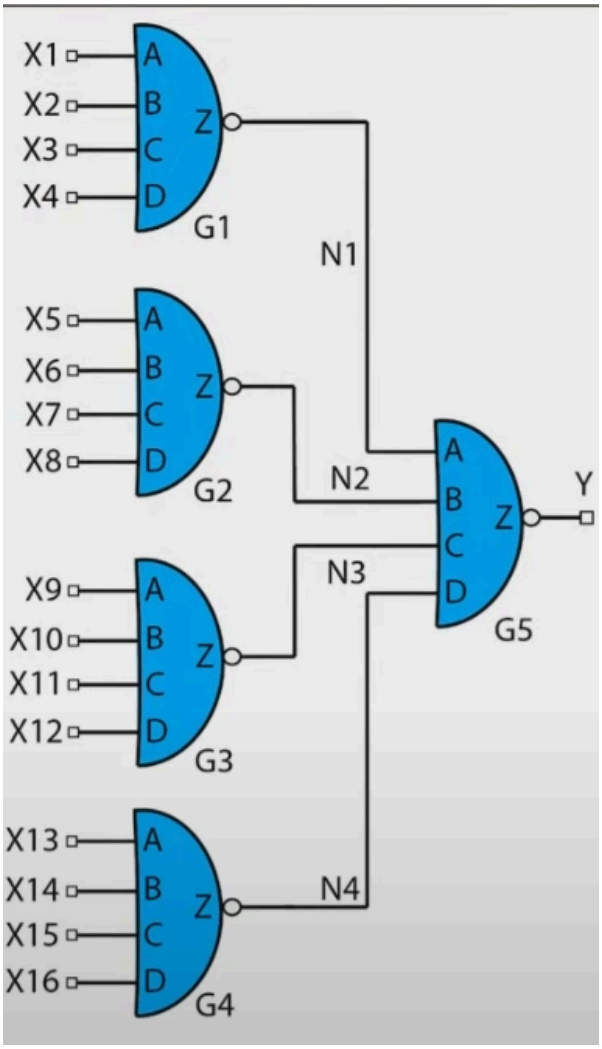
Suppose we have a circuit which implements a Boolean function with n input. Suppose there is a circuit and we have n inputs and Boolean variables and inputs are there and it is producing some output . Now if we want to test this circuit how can we do it? One of the ways can be that we apply test patterns that are the possible values that can come at the inputs of the circuit . So we apply those test patterns for example 0 0 0 0 1 and so on. So if there are n variables then 2 to the power n different combinations are possible and then for each combination we check what is the value y produced by this circuit and compare whether it is as per expectation or not .

So this method is known as functional testing and this is the same and this kind of thing we have already looked at when we discussed functional verification of our RTL . Now this technique will become infeasible if we have a very large number of Boolean variables for example say if the n is 50 . If n is 50 then the number of test patterns that we need to apply will be 2 to the power 50 and then we are off in a limit and to apply so many test patterns it will need a lot of time and therefore the cost of testing will increase . So the cost of testing depends on how much time we spend for a given die on the AT equipment or automatic test equipment. If there are a large number of test patterns that we need to apply then we need to keep that a given die for a longer duration on the AT and therefore the test time will increase and therefore the cost of testing will also increase and therefore when n is large then we cannot afford to do functional tests.

So in that case what we do is that we follow another testing paradigm which is known as structural testing. Now what do we do in structural testing? In structural testing we test the components that implement a logic function. So we go deeper into this circuit C and there will be logic gates inside this which will be used to realize this Boolean function. So we will be actually testing these components which are there inside our

circuit individually . So test the components that implement a logic function rather than testing the input output functionality .

So that this testing paradigm is known as structural testing. In structural testing we are not not testing the input output behavior of our circuit rather we test each individual component in our circuit and if the individual components are found to be good then we say that our circuit is passing the manufacturing test ok. So the the the paradigm of structural testing is widely employed in in in manufacturing test in VLSI in in in the semiconductor industry and why why it is it is very popular the reason is that it reduces the number of test patterns that is required to to ensure the our or perform testing efficiently is comparatively much lesser for structural testing than with the than with the with the functional testing. Therefore the structural testing paradigm is widely employed in the semiconductor industry. So now let us look into the functional differences between functional testing and structural testing in more detail.



Suppose we are given this circuit which is shown here so in this circuit there are say 4 or 5 NAND gates G1, G2, G3, G4, G5 5 NAND gates. And there are 1 x 1 to x 16 there are 16 input ports . Now if we want to do functional testing of a functional testing of this circuit then how many test vectors will be required. So since there are 16 input, input ports will require 2 to the power 16 or 65,536 input combinations . So this many test vectors will be required if we do functional testing or the input output if we are trying to compare the input output behavior of the expected response and the circuit which is manufactured .

Now if we do structural testing how many test patterns will be needed . Now to test one of the gates, suppose we want to test G1 G1 gate. This is a NAND gate so this has got 4 inputs . Now if we want to just test this NAND gate how many test vectors will be required. In this case since the input number of inputs is 4 for the NAND gate then we will need one and only 2 to power 4 or 16 input combinations . And since there are 5 NAND gates it will require 16 into 5 that is 80 test test test patterns to test individual these components .

So we have come down from 65000 or around 65000 to 80. There is a drastic reduction in the number of test patterns required . So in that sense structural testing is very efficient. It requires very less number of test patterns but there are certain assumptions involved here and what are those assumptions. So if we are carrying out structural testing we need to observe the output pins of all the components . So if we want to test whether G1 is functioning correctly or it is giving this the NAND gate G1 is correct or has been manufactured correctly then we need to observe the value of this output . So when we apply a test pattern X1 at X1, X2, X3, X4 as say 111 we expect that this G1 gate will produce an output of 0 since it is a NAND .

But do we have a way to see this signal or observe this signal? The answer is no, because when we apply a test pattern to our circuit we have access to only the input ports and the output ports . So only these points are accessible for the test for the test equipment and at this point the internal nodes or internal nets of the design or the or a circuit are not accessible to the automatic test equipment and therefore we cannot observe these nets N1, N2, N3 and N4 . So assuming this may not be correct, assuming that we can observe the output pins of all the internal components of our circuit may not be correct . Now we can. The other thing that we have assumed in structural testing is that we can write any value at the input pins of all the components. For example if we suppose that we are trying to test this G5 then we are saying that we are free and we can easily apply a test pattern that is 1111 at the inputs of the gate G5 .

But the nets N1, N2, N3, N4 are internal nets that are not directly accessible to the AT

test equipment and therefore we can make the assumption that we can write any value at the internal nodes of the circuit that is not correct . And the third assumption that we have made in a structural testing is that there can be problems in the integration of the components which we are not considering. For example this net was broken at the G1 gate, this component and G2, G3, G4, G5 all these components may be working perfectly fine. But the interconnection between them may not be working properly and that also we need to test during manufacturing tests. So these are some of the assumptions that we made while deriving that only 16 input 16 patterns are required for structural tests.

So though the thought function though the structural testing requires very less time or number or significantly less number of test patterns compared to functional testing but it makes assumptions about accessing internal nets and also the integration of the components . So now DFT techniques or the or the design task that we carried out which are targeted for testing those make these tasks make changes in our design those tasks make changes in our design such that the above these three assumptions becomes more or less valid for our circuit . So the goal of DFT techniques is to make structural testing more effective by ensuring that these assumptions hold . We can easily control the internal signals, easily observe the internal signals and also verify the interconnections between the components . So now how these DFT techniques ensure that these assumptions are valid we will see in today's lecture and in the subsequent lectures.

Now the manufacturing test is done with the help of fault models . So we design test methods or test patterns or extract test patterns by assuming a fault model for our circuit. Now what are these fault models? So fault models represent a defect using a logical or electrical model . So let us take an example: suppose we had an AND gate, we had two inputs A and B and it was producing an output set. Now for this AND gate we made a layout and with this A line was running B line was running and also a ground line was running .

Now because of the random variation in the processes some conducting test particles came and it shorted the line B and ground . So this is a defect. This is an example of a defect. Now our circuit has got a defect and it is a fatal defect. It may be a fatal defect because it is shorting the line B and the ground . So this defect can be represented using a logical or electrical model that is what the fault model is. So how can we represent this defect using a logical or electrical model? We can represent it using say making this ground B B line stuck to the ground line .

So we say that B is taking a constant value of 0 because it is shorted to ground. So this is a representation of a defect and this is known as a fault. We have said that we are

representing the defect. Defect is a physical phenomenon . This is a physical phenomenon which is observed on the on the on the on the layout or on the on the die and we are representing that defect you you in our in a circuit model using a fault model .

So this is what a fault model is. Now why do we use a fault model? What is the purpose of using the fault model? So fault models allow us to analyze the impact of defects using logic or circuit analysis techniques. For example if we said that we we we have modeled the defect of the short circuiting line B and ground using this circuit using this fault then we can analyze the impact of this fault by saying that this B line is equal to 0 and since this is connected to an AND gate the output Z will also become 0 and then this fault will propagate and so on. So we can apply the circuit analysis techniques once we abstract out the impact of defects to our circuit. We represent the defect in the circuit and then use the circuit analysis techniques to carry out further design tasks .

So it allows deriving test patterns algorithmically for detecting a given fault . Now once we have got the defect in the form of a circuit now we can use it in our design task for example extracting the test pattern . Now suppose this B B line was grounded B was grounded or B was stuck to 0 . Now if we apply a test pattern to say A is equal to 1 and B is equal to 1 for a good circuit we expect that Z is equal to 1 . But if this line B was stuck to 0 Z will produce a value of or the value that will be produced at Z will be 0 .

So we can say that A is equal to 1 B and is equal to 1 is a test pattern . Now once we model the defect in a circuit we can apply various algorithms to extract test patterns and how to extract these test patterns and what algorithm will be seen in the subsequent lectures. And so we can also use the fault model to assess the quality of testing meaning that if we say that defect is causing this this this fault was and the manifestation of defect is some form of logical logical defect . Now whether our test patterns are catching those conditions or circuit conditions or not we can measure it using circuit analysis techniques. And based on that we can say whether our quality of testing is good or not.

So the fault model allows us to have a quantitative quantitative assessment of testing or effectiveness of testing . If we are able to cover most of the faults in our design you in our testing we say that the testing is good. If many or many faults are not detected meaning many possible faults for a given fault model if we are not able to catch them during testing then we say that our testing method is not effective . And these kinds of quantitative quantitative assessments we can make after we have modeled the defect as a fault. So the fault model basically transforms the problem of defect detection to the problem of fault detection .

Now there are various kinds of fault models that we use in an in a for in an integrated circuit. Now let us look into one of the most popular fault models and that is known as the stuck at fault model. Now what is stuck at the fault model? Stuck at fault model assumes that defect causes the signal to be permanently stuck at a constant logic . For it transforms the defect to a logical fault model and there are two types of faults. The first one is known as stuck at logic 0 or stuck at 0 and we can use the short form S a 0 to say that a given signal is stuck at logic 0 .

And similarly there is another type of stuck at fault which is known as stuck at logic 1 fault and it is in this case the given signal is a stuck at 1 and we abbreviate it as S a 1 fault . So now let us take an example, suppose we are given this NAND gate it has got four inputs a b c d . Now if there is a stuck at fault or stuck at 0 fault at the pin a if the pin a is say stuck at it is shorted to the ground line . Then we model it as a stuck at 0 fault or we say that a is stuck at 0 when we write stuck at 0. Similarly if the a line was shorted with V d d or power supply line in that case we will say that a is stuck at 1 and and in short form we say that S a 1 is there at the line a .

Now there is another fault model which is known as the single stuck at fault model which is very popular. Now what is single stuck at fault model? In the single stuck at fault model we assume that there is only one fault active at a time in our circuit . And why do we use the single stuck at fault model? So we use it because it reduces the complexity of test pattern generation significantly . Now given our circuit, say our circuit which consists of say millions of gates and millions of nets is it fair to assume that only one fault is active at a time . So on the surface it seems that it is very unfair because if there are say millions of nets in our design and the fabrication process is a kind of or the or getting defects during fabrication is a random event then we can have multiple faults simultaneously on our chip .

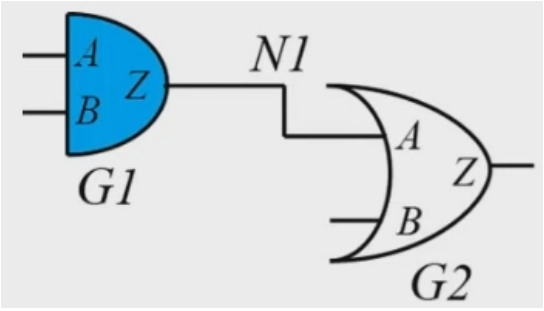
So the single assumption that only a single fault is active at a time in our circuit may not be may not be correct . But still we use a single stuck at fault model why? So the reason is that the test patterns or the set of test patterns that we derive using the assumption of single stuck at fault model those are able to capture 99 or those are able to cover 99 percent of the multiple stuck at fault in our circuit if there are multiple outputs in our circuit and the logic structure is fairly complex which is there in our industrial designs in those cases in the test patterns that are derived using single stuck at fault model assumption those are able to cover 99 percent of the multiple stuck at fault also. Therefore the quality of the testing does not suffer by making this assumption and therefore the single stuck at fault model is widely used in the industry because it reduces the complexity of test pattern generation without sacrificing the quality of the test. Now let us understand where faults occur in our circuit and what are fault sites? So the point

where a fault can exist or we assume it to exist is known as the fault site. Now how do we emulate a stuck fault in a circuit? So when we have a circuit and we want to emulate it, suppose we have an inverter driving another inverter .

Now if we say that the input A sorry input of this inverter I2 let us name it as 1 and we say that this one was stuck at say 0. Now when we want to do a circuit simulation or fault simulation for the case when there is a fault occurring in our circuit how do we emulate those conditions . So we emulate it by disconnecting the corresponding source meaning that we disconnect the line between I1 and I2 we disconnect it and tie it to the constant logic . We say that this input is being held to constant logic 0 . In this case it was stuck at 0 therefore we tied it to 0 if it was stuck at 1 we will tie it to 1 .

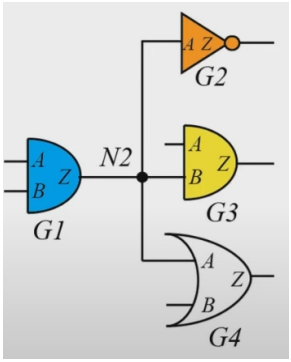
Now for a given net how many fault sites can be there . Now if the net is a fan out free net meaning that for this net the number of pins in the fan out is 1 . So if the fan out is 1 or it is of this kind of nets are known as fan out free nets in that case the number of fault sites is 1 . If this one is stuck at 0 if the input is stuck at 0 we or say this z is or the z pin is stuck at 0 both are considered as the same . And therefore the number of fans out the fault site for a fan out free net will be 1.

But what about the nets which have fan out more than 1 for example this net N2 . In this case how many fault sites will be there . Now in this case suppose there is a fault at this point at the output of G1 suppose this one was stuck to 1 . If this one was stuck to 1 then it means that all the driven pins all the driven pins will get a value of 1 . So in this case this stuck at 1 at the pin z of G1 will lead to fault in all the driven pins all in the fan out.



However suppose there is a fault at this pin only the input pin that is the input pin A of G2 suppose this was much stuck to 1 . Then in this case even though the G2 is stuck to 1 the other gates for example G3 and G4 those may be working fine . So we have to differentiate whether this one the input of G2 is stuck to 1 or input of G3 is stuck to 1 or input of G4 is stuck to 1 . So there can be 3 there can be 4 different ways in which fault can happen . It can happen at this point it can happen at this point at this point 3 in the fan out plus it can have happened in the output of G1 .

So if there are n fan outs so in this case there are 3 fans out there then how many fault



sites will be there it will be n plus 1 . In this case there were 4 fault sites 1 fault site second third and 4 . So depending on how many fan outs are there for a given net the number of fault sites will be either 1 or n plus 1 . Now let us take an example of a circuit and then understand where the faults can occur and what kind of how many possible faults there are under the assumption of single stuck at fault model ok. Now consider this circuit, suppose this circuit was given to us and we are asked the question that how many singles stuck at fault can be possible in the circuit .

Now to find out the number of possible faults in the circuit we need to first find out how many fault sites are there for the circuit . So in this case of course there will be the fault sites that are the input port these are the fault sites . And then if we look into the net A this one there is only one fault site because it has got a fan out of 1 . And for the net which is connected to B how many fault sites are there one at this input and another at this .

And of course the fault can come B also be at B. Now for C how many fault sites are there only one because there is a fan out of 1 . Now for F 4 there is only one fault site, for F 5 there is one fault site and for Z there is one fault site . So how many total fault sites are there one is A B F 1 F 2 F 3 1 2 3 4 5 6 7 8 9 . Now on each fault site we can have two types of fault stuck at 0 and stuck at 1. So total how many faults are possible 9 into 2 that is 18 faults are possible 18 single stuck at faults are possible for this circuit.

Now let us look at how we can detect a fault in our circuit. So we detect faults in our circuit using test vectors. Now what are test vectors? So test vectors are any input pattern or a sequence of input patterns that produces a different response for a faulty circuit and a fault free circuit. For example let us assume that this is a certain argument circuit and these are the inputs . So we can apply a pattern a sequence of 0s and 1s at the input such that the output that this circuit produces for a faulty circuit and a fault free circuit are different meaning that say for faulty circuit it is producing 0 and for a fault free circuit if it is producing 1 then the input pattern that we have applied to our circuit that is known as the test vector or test pattern .

Now for a given circuit of an element for example say we have a NAND gate if we want to exhaustively test the functionality of this NAND gate will require 2 to the power n number of input combinations where n is the number of inputs. So in this case since there are 4 inputs it will require 2 to the power 4 or 16 input combinations . But can we derive a set of test vectors which is much less than 16 and are able to capture all the

defects in our or faults in our circuit. So the answer is yes and for that we take help of the fault models. So what fault models or for example the single circuit fault model does is that it makes the number of faults that need to be detected in our circuit as linear in the number of circuit elements .

So since the number of faults that we need to detect in our circuit becomes linear, the number of test patterns that will be required to test our circuit also reduces drastically . So let us take an example and understand how the assumption of a single circuit fault model helps in reducing the number of test vectors. So let us take a case in which we have say 4 input NAND gates and we want to assume that a single circuit fault model is valid for this case. So since there are 4 input pins and 1 output pins and each of them can have a stuck at 0 and stuck at 1 fault. So the number of faults that will be there that needs to be detected for this circuit is 5 into 2 that is 10 faults .

So that is there are 5 pins in this component and each can have 2 types of faults stuck at 0 and stuck at 1 will have 2 into 5 that is 10 numbers of stuck at fault that need to be detected. Now let us derive the test patterns for detecting all this 10 10 10 stuck at fault. So first let us see that if if our circuit is fault free and this is a NAND NAND NAND gate then since there are 4 inputs we have 2 to the power 4 or 16 input combinations and for the for for the for the case when all the inputs are 1 we have a value 0 and in other cases we have the output value as 1 . Now let us assume that in our circuit a fault at 0 fault occurs at this point at this point this is stuck at 0 this is grounded.

So A becomes 0 now if A becomes 0 then what will be the value of Z. So in that case all input combinations will have the output as 1 and then from this truth table we can derive what is the test pattern. So remember that or what is the test vector. So remember that a test vector is the input combination that produces different output for a faulty circuit and a fault free circuit. So we see that only for the test pattern 1 1 1 1 1 sorry 1 1 1 1 we have the the value produced at the output Z is 0 for a fault free circuit and 1 for a faulty circuit and therefore 1 1 1 1 is a is a test vector to detect a stuck at 0 fault at the pin A . Similarly the to detect stuck at 0 fault at pin B C and D we can use the same same same test pattern that is 1 1 1 1 .

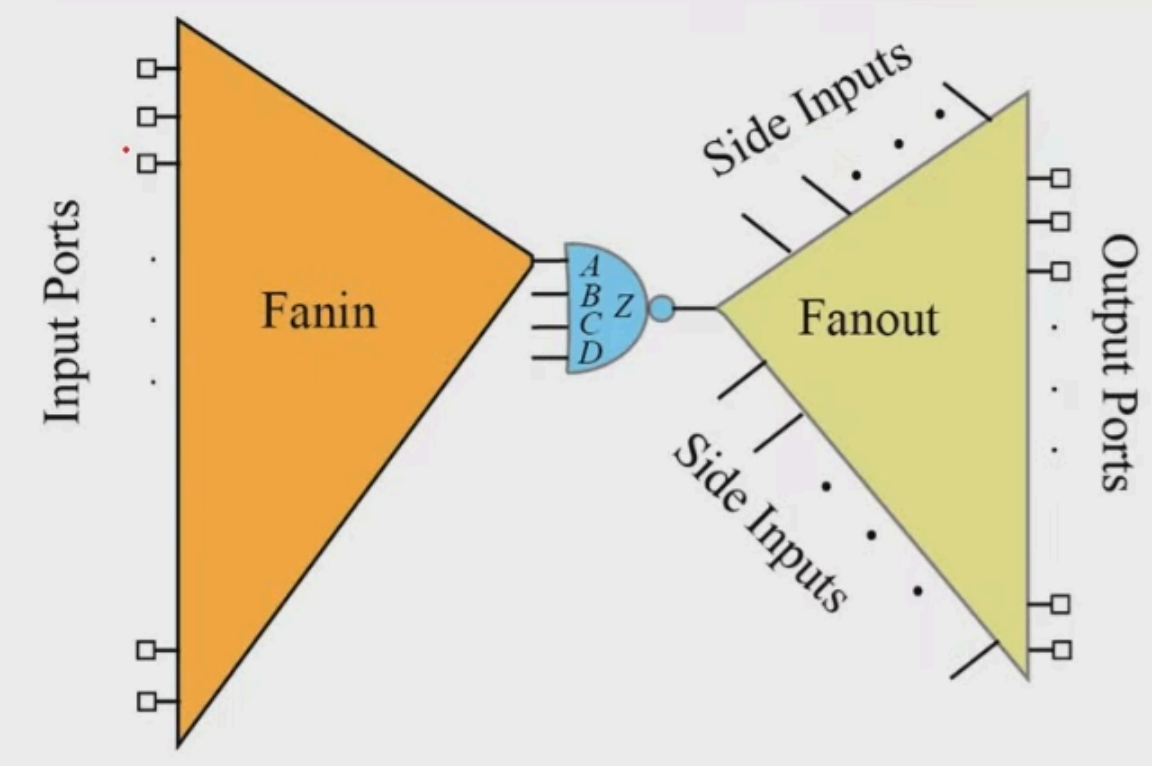
Now for the stuck at fault stuck at 0 for stuck at 0 at the output 0 sorry output Z what will be the test pattern in this case for any input combination other than 1 1 1 1 the output will for a faulty circuit is 0 and the fault output for a fault free circuit is 1 . So all these are valid test patterns except this one . Similarly we can derive the test pattern for the case when A is stuck to 1 B is stuck to 1 C is stuck to 1 D is stuck to 1 and Z is stuck to 1 . So these are the these are the patterns for example if we take A stuck to 1 if A is stuck to 1 then the for the test vector 0 0 1 1 1 1 the output produced by a good circuit

will be 1 the output produced by the good circuit will be 1, but output produced by the bad circuit or faulty circuit will be 0 because A is held to 1 . So a valid test pattern to detect a stuck at fault at stuck at 1 fault at A is 0 1 1 1 .

So we see that the number of tests or the test vectors that can test all the ones stuck at fault in our circuit is 1 1 1 1 0 1 1 1 1 0 1 1 and 1 0 1 1 0 1 and 1 1 1 0 . So these 5 test vectors can cover all the stuck at fault in our circuit . So instead of 2 to the power 4 that is 16 combinations input combinations we need to apply only 5 combinations . So since there were 5 sorry 4 input pins we required 5 test vectors to detect all the ones stuck at fault. So in general if there are say n inputs for a NAND gate then will require n plus 1 number of test vectors to cover all the singles stuck at faults in the NAND gate .

Now this is the story about 1 component 1 . Now we want to test this NAND gate when it is lying inside our circuit . Now when this NAND gate is sitting inside our circuit we need to apply the test pattern to it . Now if this NAND gate was directly connected to the input port the AT equipment AT could have directly applied the test patterns at the input pins . For example, suppose the test vector we needed to apply was 1 1 1 1 and this NAND gate was directly connected to the input port then the AT could have directly applied 1 1 1 1 to this NAND gate . And testing would have been easy or the application of test vectors would have been easy .

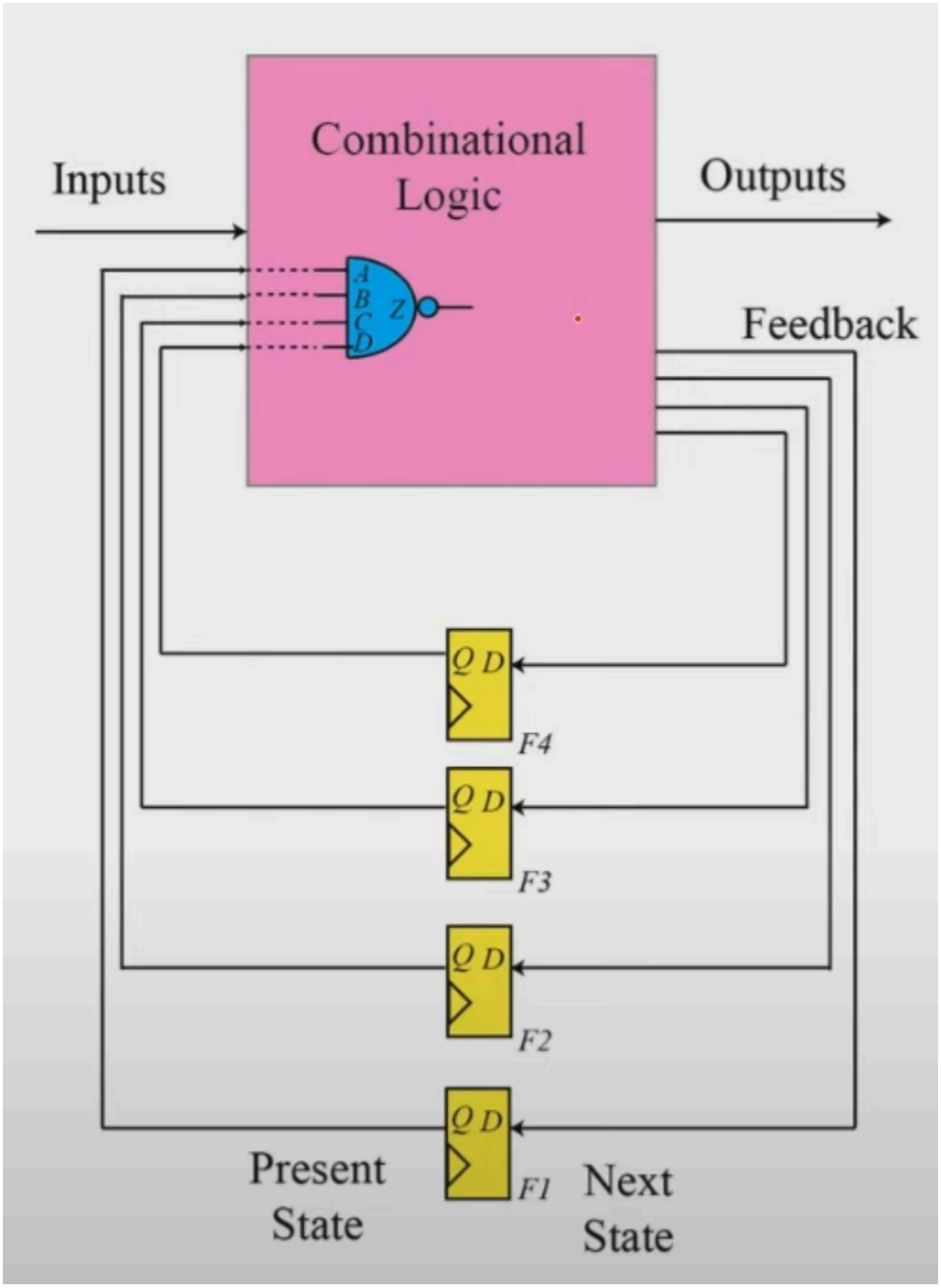
But as we move this NAND gate deeper into our logic for example, here we have shown that there is a NAND gate and in its fan there is a complicated logic structure, a Boolean function which is implemented using a logic gate. There can be several levels of logic gates in its fan . Now in this case if we want to apply say 1 1 1 1 test vectors to at the input pins we need to find out what would be the combination of values such that we get the at the combination the correct value at the input port input pins of this NAND gate . So, as the gate moves deeper inside our circuit the application of that of the required test vector on that component becomes more and more difficult. Because intervening logic structure will not be a may not allow us to pass some test pattern directly to the input of the logic input to the component we want to test .



So if the NAND gate was lying too deep in a circuit it is difficult to apply the required test pattern at the input and this the this ability to set any desired value 0 or 1 on the internal signals of a circuit by applying an appropriate test vector to the primary input this this is known as the controllability . So the the the ability that we apply that we can apply a test vector easily at the input pins of our component or at any given internal signal in our circuit the ability to apply the arbitrarily 0 or 1 at a given internal signal in our circuit from the input ports that ability of a circuit is known as controllability. For example, if this NAND gate was sitting very close to the input port the controllability would have been higher as it goes deeper into the logic logic structure the controllability may become inferior . Similarly the output of the NAND gate will be difficult to observe at any primary output if the NAND gate is lying too deep in the logic structure. For example, we want to see once we apply the test pattern say 1 1 1 1 the the correct the the correct or fault free NAND gate will produce a 0 , but if this was a faulty NAND gate it might be producing a 1 .

Now we want to observe this one on some of the output ports because that is only accessible to the automatic test equipment . So if this component was lying much deeper in our logic then observing this one or the faulty behavior at the output would have been more difficult. So this ability to examine any internal signal by propagating its value to a primary output by applying a test pattern at the input is known as observability . For example, if this NAND gate was sitting very close to the output port its observability would have been higher, but if it is away from the output port its observability is inferior.

Now this problem of observability and controllability becomes more complicated for sequential search .



Now if there is a combined sequential circuit we can model it as an FSM . So we can say that these are the state elements, these flip flops are the state elements and these are driving the combination and the combinational logic cone is being driven by these state

elements . And now when we want to apply test patterns to the test pattern to our combinational circuit elements or components we must produce the correct set of values at the output of the state elements . So the and to and the difficulty in applying correct values at the at the at the output of the of the state elements that is the problem of controllability. Now setting a particular value at any pin in a sequential circuit is more difficult than a combinational circuit because several cycles may be required to write a particular value . And a state traversal can be required and the number of and the and finding such a test sequence is time consuming by sequential ATPG2.

So let me explain what this means . So it supposed we wanted to apply a test pattern 1 1 1 1 at this component and gate . Now these inputs are being driven by the flip flops; these are driven by the flip flops or state elements . Now to get these values at the input of the NAND gate we must have 1 1 1 1 at the output of these flip flops . Now these flip flops can be part of an FSM and getting this 1 1 1 1 at the output or the state 1 1 1 1 at the FSM may not be easy; it may require many cycles to reach that state. For example, suppose these flip flops were part of a counter and the counter starts from 0 0 0 0 .

So to reach the state 1 1 1 1 it will require 2 to the power 4 or in the range of 16 cycles to reach that state . So that is why we say that if we want to get a value at the output of the state elements we will require state traversal in this case state traversal from 0 0 0 then to 0 0 0 1 and so on up to the state 1 1 1 1 . And doing state traversal can be exponential in the number of state elements and therefore reaching the correct state for an FSM that will deliver the required test pattern at the component it may require an exponential number of clock cycles it will increase the test time and the cost of testing. Additionally the finding that tests the correct sequence of this state traversal will give us this test pattern at the combinational circuit element that will also require high computational or require more difficult or challenging computation.

So it will make finding the testing test pattern also more difficult . And a similar kind of problem occurs when we try to observe the problem of fault or the effect of the fault in a sequential circuit to observe the value say 0 or 1 for from for a combinational circuit at the output at an output port we might require again a state traversal and the number of state traversal will be maybe exponential in the number of state elements . So the the controllability and observability of a circuit is very very difficult for or more difficult for a sequential circuit. And therefore we need to make some circuit modifications or we make some changes in our design such that the controllability and observability in a sequential circuit that improves and how we can and what changes we can make and how and what changes needs to be made in our design will be looking that in the subsequent lectures. Now if you want to go deeper into the topics that we have discussed in this lecture you can refer to these books. Now to summarize in this lecture what we

have done is that we have covered or we have looked into some basic concepts related to DFT.

So we looked into what is structural testing or what is a structural testing paradigm and how it differs from functional testing. We looked into fault models. We looked into what are test patterns and we also looked into what is the problem of controllability and observability in a sequential circuit. In the next lecture we will be looking into a design methodology which is known as scan design methodology which basically improves the controllability and observability of the signals in a sequential circuit. Thank you very much.