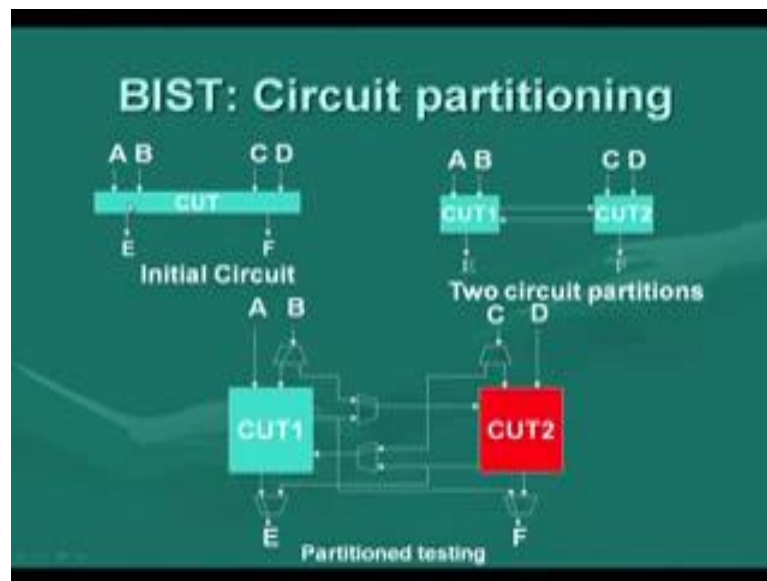


0020Digital VLSI Testing
Prof. Santanu Chattopadhyay
Department of Electronics and EC Engineering
Indian Institute of Technology, Kharagpur

Lecture - 34
Low Power Testing (Contd.)

(Refer Slide Time: 00:24)



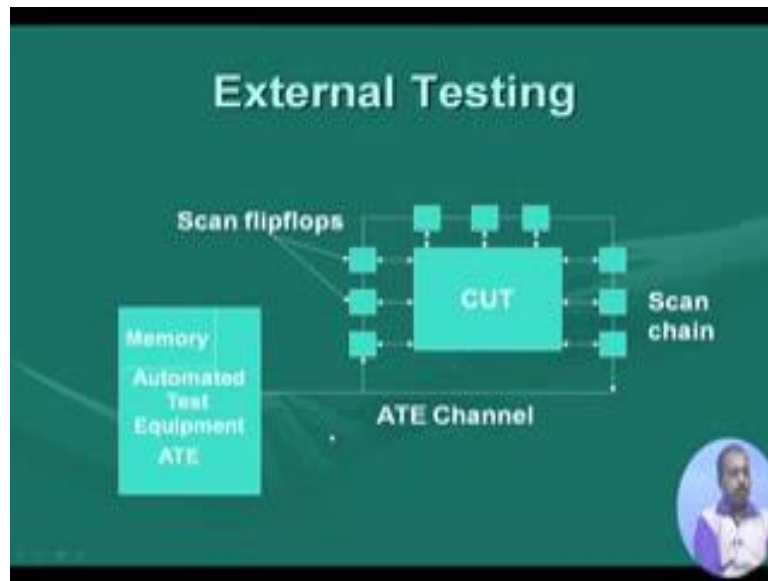
As far as power reduction in BIST is concerned, another possibility is to partition the circuit. For example, if suppose we have a circuit which has got inputs A, B, C, D and there are 2 outputs E and F, now it may be possible to divide the circuit into 2 parts circuit 1 and circuit 2. So, there will be naturally since they are part of 1 circuit. So, there will be lines running between the 2 circuits like some signals, from circuit 1 will go to circuit 2 something from circuit 2 will come to circuit 1. If it is done then suppose we are trying to test this circuit 1 when we are trying to test this circuit 1, we put a number of multiplexers it may be the case that this input B is common to both circuit 1 and circuit 2. So, what we do? We set this multiplexer in such a fashion that when we are testing circuit 1, this B input comes to this one, whereas this input is not reaching there. So, this is basically some 0 or some previous value or whatever it is depends on whether it is the last input or some gated input something like that. So, it will remain at the same value at circuit 2.

So, this comes to, the B input comes to circuit 1. Similarly it maybe common that C input is there for both circuit 1 and circuit 2. So, C input is obtained from this, the slide is not shown explicitly from C to this. So, this may be coming to circuit 1 if I have configuration of this multiplexers. So, that way this A, B, C, these are the three inputs given to circuit 1.

Now we can see that some other inputs which are common between some other signal lines which are common between circuits 1 and circuit 2 maybe programmed to reach circuit 2 when we will be testing circuit 2. So, by setting this multiplexers properly. So, we can partition the circuit into circuit 1 and circuit 2 and we can test them separately. So, when we are testing circuit 1. So, this multiplexer is set in such a fashion that these A B C inputs and the lines from circuit 2, they come to circuit 1. Similarly the other way that is when you are testing circuit 2 the other thing happens. Similarly E and F are 2 outputs, it maybe that these output of circuit 2 that is coming here so that may be redirected to 2. So, this of course, depends on the detailed circuit that we have this is just an example.

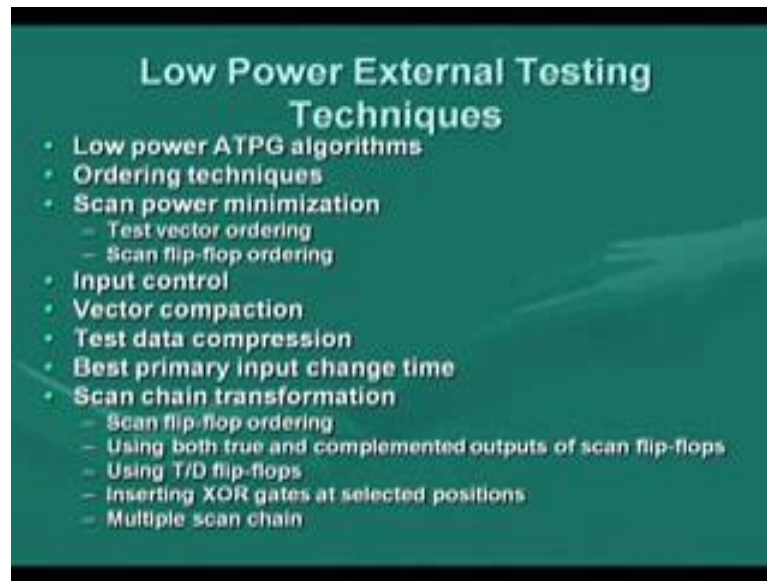
So, if we do that then this circuit will be partitioned now we can have BIST which will be of lesser, the test pattern generator for BIST will be of lesser width and, we have got this you know that if I have got an n input circuits, so in ideal case I will require an n bit LFSR to generate the test patterns. Now if I reduce this n to some other value say n by 2 then $2^{n/2}$ plus $2^{n/2}$ by so many patterns will be needed for testing the 2 circuits separately. Of course, some patterns will be needed to test the multiplexers as well, but the total sum will be much less than 2^n . So, that way this circuit partitioning can reduce test time and since the circuit 2 is not excited when circuit 1 is being tested circuit 2 can be totally in a non excited mode inputs are not modified there, so it can, it will consume very little power maybe only the leakage power part will be consumed, but no dynamic power will be consumed. So, that way power reduction can be achieved.

(Refer Slide Time: 04:05)



Another way of testing the circuits is via external testing. So, when we have got the test equipment coming separately. So, this test patterns are stored in the memory of the ATE and then this circuit that we are going to test so they are, so the scan chain the scan flip flops of the circuit they form a chain - scan chain and it is also assumed that this circuit has got some primary inputs also, but primary inputs are also assumed to be connected over a scan chain. So, it is in this case it is taken as the full circuit all the inputs are coming through the scan flip flops. Now from the ATE channel the bits will be shifted through this scan flip flops, the test pattern bits will be shifted through this scan flip flop and the response bits. So, they will be collected again on to the scan chain flip flops and they will be shifted out through this ATE channel. So, this is the method for external testing that we have seen in previous classes.

(Refer Slide Time: 04:57)




Now, there can be several techniques by which we can do this external testing in a low power manner. So, we can have some ATPG algorithms that target low power, it will try to generate test patterns. So, the power consumption is less then there are some ordering techniques so we can change the order of these test patterns to be applied to reduce the circuit transitions. Then we can go for some scan power minimizations, so scan power minimization it can talk about test vector ordering, scan flip flop ordering, then we can control the inputs we can do some vector compaction, can take help of test data compression. Then another possibility is the best primary input change time, so what it tells is that we have got this any test pattern has got one part which is primary input the other part constitutes the pseudo primary input. So, the pseudo primary input part is actually shifted through the scan chain. Now since that shifting will take time, so when exactly we are going to change the primary input part. So, that may determine the power consumption the transitions in the circuit.

Then there are lot of scan chain transformation techniques like scan flip flop ordering, then modifying the flip flop types like T to D or using both q and q bar outputs of the scan flip flops, then we can put some XOR gates to the scan chain at some selected positions and also we can go for multiple scan chain.

(Refer Slide Time: 06:29)

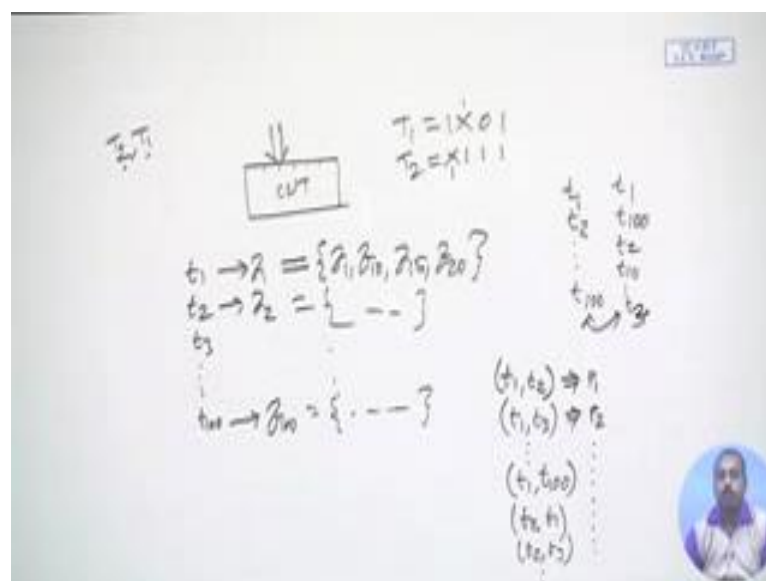
Low Power ATPG

- Modify ATPG algorithm (like PODEM) to fill-up the don't cares judiciously so that the number of transitions between two successive test vectors is minimized,
- Generating redundant set of patterns
 - A number of patterns are generated for each fault
 - For sequential circuit testing, patterns are to be applied in a sequence only.
 - Each such sequence may have different power requirements
 - Select those sequences requiring lesser power, without sacrificing fault coverage.



So, we will look into some of this strategies low power ATPG, low power ATPG. So, any standard ATPG so that will leave a number of do not cares not fields. So, actually those do not cares can be exploited so that we can we can minimize the transition in a circuit.

(Refer Slide Time: 06:54)



Handwritten diagram illustrating Low Power ATPG. It shows a circuit block labeled "CUT" with inputs T_1 and T_2 . The inputs are defined as $T_1 = 1X01$ and $T_2 = X111$. Below the circuit, a sequence of test vectors $t_1, t_2, t_3, \dots, t_{100}$ is shown. The vectors are mapped to a 4-bit output space $\{z_1, z_{10}, z_{10}, z_{20}\}$. The mapping for t_1 is $\{z_1, z_{10}, z_{10}, z_{20}\}$. The mapping for t_2 is $\{--\}$. The mapping for t_3 is $\{--\}$. The mapping for t_{100} is $\{--\}$. A sequence of pairs $(t_1, t_2), (t_1, t_3), (t_1, t_{100}), (t_2, t_1), (t_2, t_3)$ is shown, indicating transitions between vectors. A speaker icon is in the bottom right corner.

So, what I want to mean is this is the circuit. So, this test pattern generated has given me 2 test vectors say t_1 and t_2 . So, first t_1 will be applied to the circuit and then t_2 will be applied to the circuit. Now if we can ensure that between t_1 and t_2 many bits are not changing, if many bits are not changing then it is possible that the many of the gates which are getting input from the primary input of this; from the primary input will not do a transition and subsequently the gates at successive stages they will also not make much transition if the first level gates are not doing transition.

Now, we have told previously that in a test pattern generation process that successive test patterns are highly uncorrelated that is the number of similarity bits between 2 test pattern is generally very low to make sure that new faults are being excited, but it may so happen that though they are different, but many of the bits may be do not care in one of the patterns. So, T_1 may be something like say 1 X 0 1 and T_2 maybe X 1 then this is say 1 and this is say 1. So, there are changes, but you see that, if I fill this X by 1 and fill this X by 1 then the hamming distance between the patterns can reduce compared to the case where this X will be filled up randomly. So, what we can do? This any ATPG algorithm it can have a post processing stage in which the don't care bits will be filled up judiciously so that number of transitions between 2 successive test vectors get minimized. So, this is one possibility.

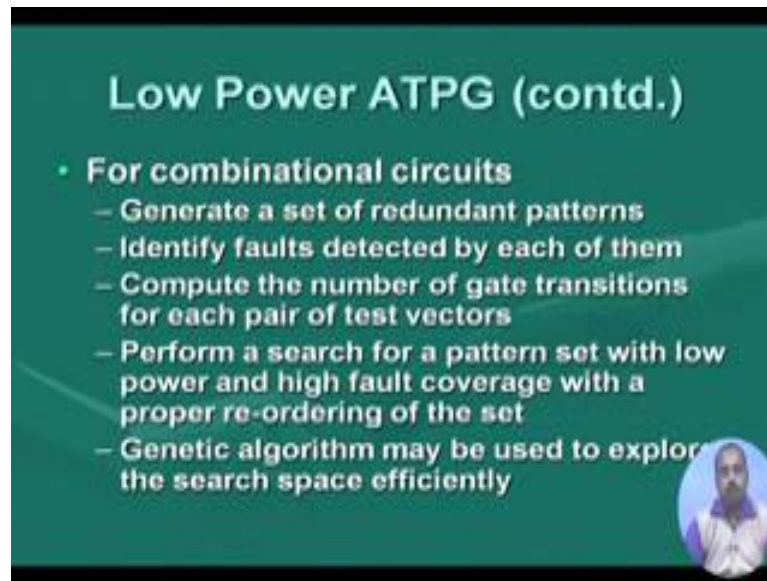
Then generating redundant set of patterns, so what it says is that you know that sometimes in our class we have discussed that n detect test patterns set. So, n detect means a fault is not dropped from the set of faults to be considered by the ATPG algorithm as soon as it gets detected by a single pattern rather it is kept in the fault list till it has been detected n times. So, n may be equal typically equal to 2 or 3, but what we get is for every fault we are getting a number of patterns. So, in this way we can generate a redundant set of patterns. Now most of the ATPG algorithms when you generate these n detect pattern set it does not do any compaction of the test set. So, for every fault I get say n equal to 2, so 2 pattern. So, if there are say k number of faults in the circuit then I get a total of $2k$ number of patterns.

So, there is no compaction that is done. So, what will happen is that, so many of this patterns they have the capability to detect number of faults. So, though it has been taken

into the set because it could detect one particular fault, but it is a they have the potential to detect more number of faults. So, once we have got a redundant set of patterns. So, we can take, we can do something so that we can chose pattern, so that they are they consume less power. So, a number of patterns are generated for each fault depending upon that n detect principle. For sequential circuit testing patterns are to be applied in a sequence only for sequential circuit though we did not discuss about sequential ATPG in general, but what happens in that case is the first pattern or a set of patterns will take the circuit to a particular state and then the final pattern it will actually apply excite the fault. So, that way there is a sequence of patterns that are present which will be doing the test excitation and observation.


So, sequential circuits patterns are to be applied in a sequence only and each such sequence may have different power requirements. So, for the same fault I may get different sequences of test patterns, so which sequence will pick up so that there may be option. So, maybe some sequence apparently seems to be longer than some other sequence, but it may so happen that the second sequence between the successive patterns hamming distance is low as a result they do not create lot of transitions in the circuit inputs. So, lot of gates do not switch that a power consumption maybe low. So, we can we can go for the sequence which may be longer, but power values are low. So, if we select these sequence is requiring less power. So, if the fault coverage is not sacrifice because for every fault I have got options of sequences and for I am including at least at least one sequence for each for fault, so fault coverage is not dropped, but our power consumption may be reduced.

(Refer Slide Time: 12:07)



Low Power ATPG (contd.)

- For combinational circuits
 - Generate a set of redundant patterns
 - Identify faults detected by each of them
 - Compute the number of gate transitions for each pair of test vectors
 - Perform a search for a pattern set with low power and high fault coverage with a proper re-ordering of the set
 - Genetic algorithm may be used to explore the search space efficiently



For combinational circuits the situation is slightly more difficult as far as power consumption is concerned, first of all we were the first few steps are same like we generate a set of redundant patterns. So, for every fault we generate n number of pattern then we do a fault simulation. So, after fault simulation we can see that every pattern suppose we have generated redundant set of patterns consisting of t_1, t_2, t_3 up to t_{100} where t_1 is targeting fault f_1 , t_2 is targeting fault f_2 , like that t_{100} is targeting fault f_{100} .

Now after this it passes through a fault simulation stage, after doing fault simulation we may find that t_1 detects a number of faults f_1, f_{10}, f_{15} and f_{20} ; though originally the ATPG 2 let had generated a targeted to f_1 only. So, this way for every fault we see a set of faults every pattern we see a set of faults that the pattern can detect. So, identify the false detected by each of them so that is basically via fault simulation and we compute number of gate transitions for each pair of test vectors.

Now what will happen is that since this is a combinational circuit so to detect a fault it is sufficient just to apply the corresponding pattern, it is not dependent like what was the previous pattern that was applied particularly if you are targeting starkard faults so this is particularly true. For delay fault it is slightly different because delay fault is a 2 pattern

test so there I have to consider groups of 2 patterns, but if you are restricting to say starkard faults which covers most of the other faults as well the test patterns generated by starkard faults covers most of the other faults as well.

So, we can see that supposed we is first applied t 1, now I have a choice I can apply t 2, I can apply t 3, I can apply t 4, I can apply t 100 it does not matter. For somebody applying patterns in this sequence t 1, t 2, t 100 and another person applying it in this sequence t 1, t 2, t 100 then t 2 then t 10 then finally, say t 2 sorry t 3 at the end. So, all the patterns are applied to a combinational circuit, the fault coverage will remain unaltered.

So, we can compute what are the, so bit; so we compute this type of transition like first t 1 followed by t 2. So, when t 1 is applied circuit gates are at some values after that if I apply t 2 some of the gates will transactions; you count how many transitions are occurring if I applied t 2 after t 1. Similarly t 1 followed by t 3. So, this way up to t 1 followed by t 100 then t 2 followed by t 1, t 2 followed by t 3. So, this way if we compute all these pairs of vectors, all these pairs of test patterns and see what are the transitions that are occurring if I first apply the first pattern and followed by the second pattern? So, this way it computes the number of gate transitions for each pair of test vectors. Then it perform a search for a pattern set with low power and high fault coverage with a proper reordering of the set. Now what? Now I have a choice suppose it says that t 1 followed by t 2 it consumes power p 1. So, in terms of gate transitions maybe it is now p 1 number of gates to transitions say then this may be p 2.

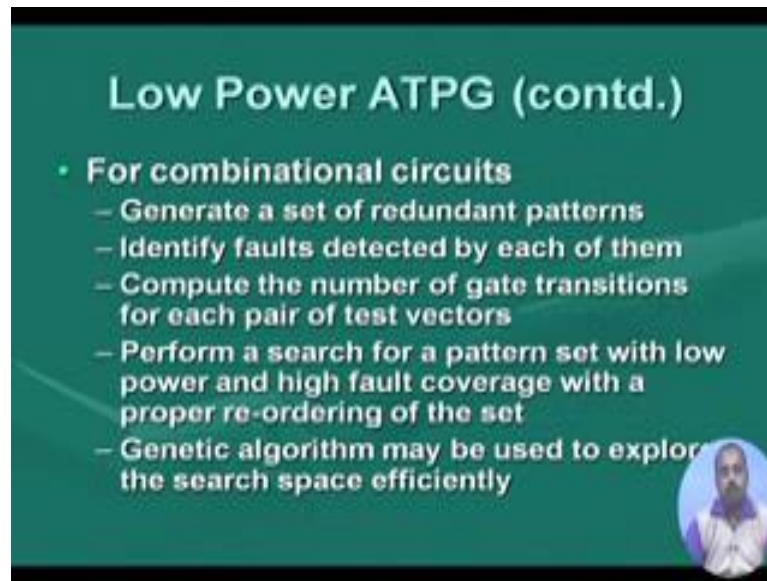
So, like that I have got n number of our for each test pattern pair I have got the number of transitions that are occurring in the circuit and number of transitions you know that is an indicator of the dynamic power consumption of the circuit if the patterns are applied in that sequence. Now from this we can take which ever pair is consuming the minimum power suppose the pair t 50 t 60 that takes the minimum power. So, we start the say test pattern set first t 50 then t 60. Now once we have chosen t 50 t 60 pair, we have got certain faults already detected by from this from this table will know what are the fault that are covered by this t 50, t 60 pair. So, those faults I can drop from my consideration.

Then from the remaining patterns set I can find out after $t = 60$ which pattern combination can give me the next best power consumption, maybe if the $t = 60$, $t = 40$ pair gives the next minimum power consumption. So, we will take up to $t = 60$, $t = 40$ or other will apply $t = 40$ after this provided $t = 40$ identifies a good number of new faults which are not covered by $t = 50$ and $t = 60$. So, this way we can have some sort of, we can search for a pattern set with low power and high fault coverage. So, there will be 2 cost function component one is the power consumption and the other is the fault coverage. So, if we do not do a fault coverage analysis then of course, it will select patterns which consume pairs which consume low power, but the fault coverage value will be low. So, as a result we need to apply a large number of patterns maybe finally, we will end up applying more patterns and as a result the power consumption is getting increased.

So, we perform a search for a pattern set with low power and high fault coverage with a proper reordering of the set, so the procedure that I have told, this definitely gives us the ordering also. So, the order in which the patterns set has to be applied. So, that the power consumption is less. Now this algorithm is a bit directed like it is starting with the least one then it is doing something so that the next pattern selected is low power one and that way it is working now we can this one, but this problem is NP-hard problem. So, you cannot have any determinist, there is no non deterministic algorithm that can solve this problem optimally in polynomial time.


So, there are several works the talk about a genetic algorithm solution for this problem as well. So, genetic algorithm based solution, it will select a set of patterns from the set and after that the ordering will be done using the procedure that I have just discussed. So, this way it can explore the search space efficiently.

(Refer Slide Time: 18:37)



Low Power ATPG (contd.)

- For combinational circuits
 - Generate a set of redundant patterns
 - Identify faults detected by each of them
 - Compute the number of gate transitions for each pair of test vectors
 - Perform a search for a pattern set with low power and high fault coverage with a proper re-ordering of the set
 - Genetic algorithm may be used to explore the search space efficiently



To take a typical examples how this vector reordering can help in the transition deduction process. Suppose I have got a three input circuit with inputs A, B and C, A, B feeding an AND gate and this is I finally feeding on OR gate. Now if we are going for an exhaustive test for example, now typical example can be that I use some sort of a counter that mod 8 counter so that generates all the 8 test patterns exhaustively. So, it starts with 0 0 0 0 1 etcetera if you just count the values at X and Y. So, this X values are 0 all together all up to this 1 0 1 and then the next 2 patterns it will be 1 and Y is like this.

Now let us see how many transitions have occurred. So, in the X line there is a transition at this point, after applying pattern 1 0 1 when we apply the pattern 1 1 0 so there is a transition, so that is transition count 1. Whereas on the Y input, there are transitions like 0 0 0 to 0 0 1, there is transition this is a transition similarly this is also a transition from 1 to 0, this is also a transactions from 0 to 1, this is again a transition from 1 to 0, this is again another transition from 0 to 1. So, on the Y line there are 5 transitions on the X line there is a single transition on the total number of transition if we do an exhaustive testing in this particular sequence, it is going to be equal to 6.

Now if we can apply that exhaustive test pattern set in a different order. So, naught in the monotonically increasing order, if we can do it like this 0 followed by 2 followed by 4


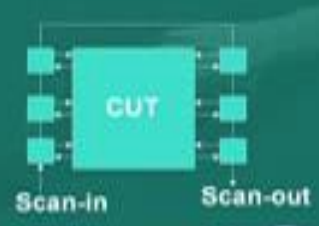
then 1 then 3 5 6 and 7. So, how did I come to this ordering? So, this is just by observation. So, what we wanted to do is to minimize the number of transitions in X and Y.

Now, it may be difficult or it may be easy to generate patterns in this sequence. So, as we know that if we are trying to generate any arbitrary sequence counter the circuitry the pattern generator circuitry that will become complex the counter design will become complex and the counter will also consume some power, so that is another issue. However, if we can do this thing then you see the number of transitions could be dropped to 2. So, there is a transition from 0 to 1 here and there is a transition from 0 to 1 here, there are only 2 transitions. So, compared to 6 transitions there are 2 transitions here. So, if you can measure the power consumed by this circuit when this patterns are applied in this order it will be more than the order in which it is applied here, say in this the power consumed when it is applied in this order. So, this is the small circuit, so then you see that this reordering of patterns set it could reduce the transitions by some value. So, if it is a large circuit though it large number of gates then number of transitions getting reduced with the significantly more.

(Refer Slide Time: 21:37)

Scan Power Minimization

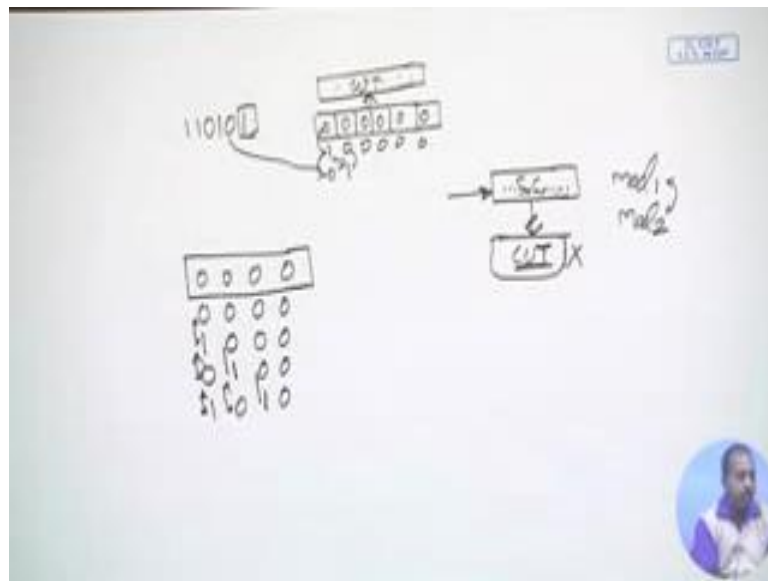
- A good amount of power is consumed when test patterns and responses are shifted through the scan chain
- Ripples cause lot of transitions in the circuit inputs which in turn creates transitions in the circuit gates
- Two solutions:
 - Test vector ordering
 - Scan flip-flop ordering



Now, this analysis is good because I am taking what are the inputs coming to the circuit and primary inputs coming to the circuit and then we are computing the transitions that each gate. Now for scan circuit this cannot be a good approach why? For the scan circuit when I am apart from these primary inputs to the circuit which I have not shown here explicitly assuming that primary inputs and primary outputs are also on the scan chain. So, through this scan chain the test pattern is getting shifted.

Now in each shift so that will generate some transitions in the circuit. So, if we are looking for an exhaustive calculation then what we have to do is that. If I have to shift in the pattern say 1 1 0 1 0 1 like this then initially assuming that the scan chain; the scan chain initially contains all 0, can initially contains all 0, if we assume that this flip flop they have been reset it is containing all 0.

(Refer Slide Time: 22:25)



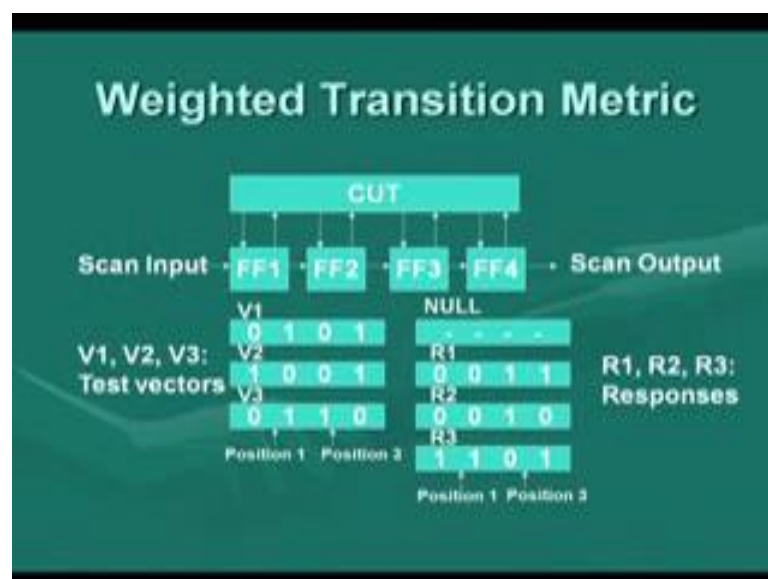
Now first this 1 bit will be shifted into the scan chain, so it will become 1 0 0 0 0 0 then this 0 will go so 1 will, this 1 will come here and this new 0 will be going to this point. So, you see that this process will continue. So, there are transitions like this cell I have already seen 2 transitions, this cell I have seen 1 transition again this way there will be transitions. Now ideally what is required is that this scan cell output this scan chain output will be fed to the circuit. So, they are connected to the inputs to the circuit. So, for

every transition that occurs I need to compute the total number of transitions in all these gates. So, I need to do a logic simulation of this circuit for every shift that I do which is this is a very cumbersome thing, so very complex computation that will be required, so over rate of computation will be very large.

So, this is the first observation that a good amount of power is consumed when test patterns and responses are shifted to the scan chain. So, when I am shifting in the pattern. So, it will create transitions in this flip flops as a result it will see this primary input they will see some transition here and as a result the circuit will start consuming power. So, this is a major source of power consumption rather than your capture cycle, launch and capture cycle is shifting , so that itself consumes lot of power because of this shifting that is occurring through the scan chain. So, ripples that are coming on the scan chain they cause transitions in the circuit inputs which in turn creates transitions in the circuit gates this pointer transitions, those internal circuit is will also transitions.

So, what is the way out? Way out is we can do some test victory ordering or we can do some scan flip flop reordering, but whatever we do the major problem that we have is that computational overhead like we have to do a logic simulation for every scan shift. So, somehow we need to avoid that.

(Refer Slide Time: 24:54)



So, what is done is we come out with something known as weighted transition matrix. So, first of all it has been, there a lot of studies that tell that if you have a scan circuit where you have got the scan chain here and then it is fed to the circuit then the power consumed by this circuit is proportional to the transitions that you see at this point. So, that, so if you measure the transition that occurring at the input to the circuit that in many cases is a good estimation of the power consumption of the circuit and in fact, what we are looking for is we will do some modification to this scan chain.

Now if I have modification 1, if I have got 2 modifications schemes - modification 1 and modification 2, so I do not need the exact power values for modification 1 and modification 2 rather I need a measure by which we can compare between these 2 modification which one will be better. Now it has been seeing that if you compute the transitions that are occurring in the scan chain itself that is a good estimate. So, in case of modification 1, if you see that in that scan chain there are more number of transitions than in modification 2 then modification 1 is expected to consume more power. So, it is the circuit under modification 1 of the scan chain is expected to consume more power than the circuit under modification 2.

So, we do not need to do explicit logic simulation of the circuit that is not required, logic simulation of the circuit is not required you can just compute what are the transitions, how many transitions are occurring in this scan chain when I am shifting in the pattern or shifting out the previous response. So, that is what is going to be measured and that is known as this weighted transition matrix. So, in this case, so this is a situation we have got the circuit under test and suppose you have got a scan chain consisting of four flip flops – FF1, 2, 3 and 4. So, scan input is connected to flip flop 1 and scan output is connected from flip flop 4. Suppose you have got this test vector V1, V2 and V3, V1 is 0 1 0 1, V2 is 1 0 0 1, V3 is 0 1 1 0 and on the other hand this response is like this when we apply V1, response is 0 0 1 1 that is R1, R2 is the response of V2 and R3 is the response of V3.

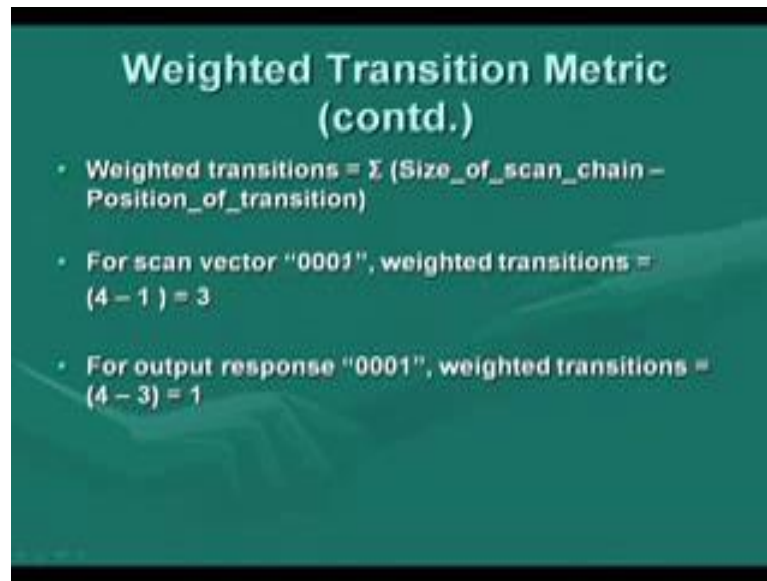
Now, what happens? Like I have got four cells I have got four cells and, this one. So, first if you look into that scan chain values, when I am shifting in this 0 1 0 1 first that 0 goes initially all the flip flops were 0, now 0 comes, 0 it remains 0 0 0 0 first 0 has got

shifted then the next 1 get shifted, so 1 comes here 0 goes there and this remains like this. Then another 0 comes, so 0 comes here and this 1 goes here 0 0. So, this way actually this transition will be taking place; then this 1 0 1 0, it will go like this. Now you see that there are in the original test vector was; original test vector was 0 1 0 1, original test vector was 0 1 0 1 now you see that this particular transition, this 0 to 1 transition; this has gone to how many place? So, this has affected this cell, this has effected this cell, so 3 cell have got affected.

Whereas, the next one that we had. So, 1 to 0 transition this one, so this one has come to only this transition has got affected this 1 0 transition it could not go to the next position, because of the length of the scan chain. Similarly this 0 1 transition, it could not go anymore. So, if I have got a pattern like this 0 1 0 1 you see that in the pattern there are some transitions, but from 0 it is going to 1 then it is going 0 then again to 1, but they do not have the same effect on the scan transition like this 0 to 1 transition. So, it will effect 3 flip flops, this 1 to 0 transition will effect 2 flip flops and these 0 to 1 transition will affect only 1 flip flop.

So, in this way we can see that these scan transitions. So, they do not go to all the flip flops. So, accordingly we can define something called position of transition. So, this is this position if there is a transition. So, we call it position 1, so between any 2 beat we call it position. So, it is position 1, this is position 2, this is position 3. So, for the first V1 you have got transition at position 1, position 2 and position 3 all the three position.

(Refer Slide Time: 30:17)



Weighted Transition Metric (contd.)

- Weighted transitions = $\sum (\text{Size_of_scan_chain} - \text{Position_of_transition})$
- For scan vector "0001", weighted transitions = $(4 - 1) = 3$
- For output response "0001", weighted transitions = $(4 - 3) = 1$

Now, number of transition, weighted transition is computed as size of scan chain minus position of transition. So, you see that if this is the test pattern that we are going to shift this transition is occurring at position 1, but when it is occurring at position 1. So, this will be affecting all the three flip flops all the three flip flop. So, that is that, so this weighted transition is computed at size of scan chain minus position of transition. So, 4 minus 1 equal to 3; if there are more number of 1's here then of course, there those will be added those transition should also be added in fact, in the previous example that you are considering; the previous example that you are considering, this is our position 1 there is a transition, so it is 4 minus 1 three plus position 3, there is a transition 4 minus three that is 1. So, total there will be four transitions that will occur in the entire scan chain; so the shifting of this pattern. So, that way we can have this weighted transition metric computed.

On the other hand for the output response that is not the case. So, output response says the positions are marked from the other side. So, this is if I mark it position 1 to position 3 now this bit will be going out immediately. So, this transition 1 to 0, this can affect only the last flip flop whereas, if there is a transition at this point it will affect all the three flip flops. So, that way position will has to be counted from other side and then this is captured by this formula, again by the same formula the size of scan chain minus

position of transition, but the position is taken from the other side. So, it is from the output side, from input side sorry from the flip flop 1 side is position 1 that way it goes.

So, this way for this entire set of test vectors and responses we can compute this weighted transition for each of the pattern test pattern and the response and those values can be added to get the weighted transition metric.