

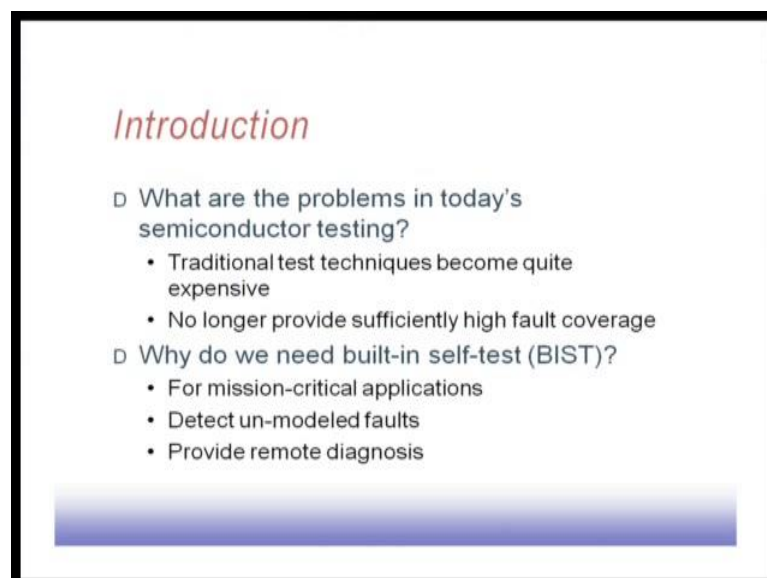
**Digital VLSI Testing**  
**Prof. Santanu Chattopadhyay**  
**Department of Electronics and EC Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 24**  
**Logic BIST**

In this case what happens is that the test pattern generated and the response analyzer they are all integrated along with the chip. In this case no external test pattern is necessary. So, whatever test pattern needs to be applied they are generated on chip, and the responses also obtained and it is analyzed on chip. So, that way it relieves the use of 80 and particularly 80s are going to be very costly, so if we want to reduce the test cost when this may be one approach.

And in many cases it is not possible to access the chip, if it is say deep inside a system or if it is a part of some environment where it is difficult to access the particular chip for test application, but at the same time it is very much necessary to test the chip periodically. So, in those situations this built in self test philosophy that is going to help in the test generation process.

(Refer Slide Time: 01:23)



*Introduction*

- ▷ What are the problems in today's semiconductor testing?
  - Traditional test techniques become quite expensive
  - No longer provide sufficiently high fault coverage
- ▷ Why do we need built-in self-test (BIST)?
  - For mission-critical applications
  - Detect un-modeled faults
  - Provide remote diagnosis

So, to go back, what are the problems in today's semiconductor testing? Traditional test techniques become quite expensive. As have seen, so it is the test pattern generation itself is a cumbersome process and it requires a good algorithm to run and it may require

quite some time to generate the test patterns, and naturally application of those test pattern. And at the same time as the complexity of the circuit is increasing it is no longer provides sufficiently high fault coverage.

So, if you try out if some random pattern, you will not get good fault coverage. If you are trying out dedicated test pattern generation algorithm, then also it is costly because of this test generation process. And of course, the test application is becoming another issue like this automatic test equipment that we have, so the cost of the systems is going to be high with the increasing memory requirement and the number of pins that the 80 channel we have. So, that way it is going to be costly. So, we need some cheaper solution. And if we are trying to reduce the system cost.

Why do we need built in self test? One special reason is that for mission critical application. So, mission critical application are those where we need to be sure about the correct operation of the system during the running of the application itself. So, many times we need to test the system again and again for particularly say- when we have got medical application. So, it is required that we check that medical appliances so they are working properly. Similarly when it is one some say critical operation like say some quiz control type of application there also we need to check the correctness of the system very often. And in that case if it is required to shutdown the system and put it on to a test mode and then test it by applying some external pattern it become a cumbersome.

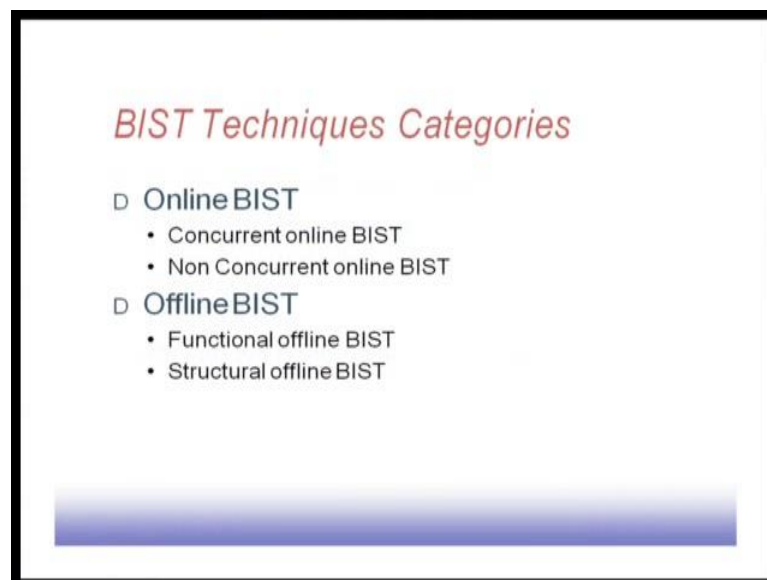
Second important issue that we have is to detect unmodeled faults. Actually, if we recollect, what happens in a system is the occurrence of some defect. And when some defect occurs we actually try to detect those defect, but for our understanding what we do is we just we module those defects as faults. Now they are may be many unmodeled defects which manifest as unmodeled faults. As the technologies are changing, there are newer fault models are necessary as we have discussed previously, but it is not very easy to come up with a comprehensive new fault model. So, that way detection of unmodeled faults becomes an important issue.

At the traditional test pattern generator they are targeted to a particular fault model. So, stuck-at fault, bridging fault or delay fault, like that they are targeted to a particular fault model. So, the test patterns which are generated by them, so they are suitable for those

fault models only. So, if there are some unmodeled faults it is a bit unlikely that those test patterns will be able to detect those faults. So, this is another problem.

Third problem is to provide remote diagnosis. As I was telling that many times a system may not be accessible, it may be the environment may be such that it is not possible to access that system, but still we need to check the correctness of the system. So, for this type of application we need this remote diagnosis and then this built in self test is going to help us, where the test pattern is generated in the circuit itself and the responses are also analyze along with the circuit.

(Refer Slide Time: 05:22)



So, if we try to see the categories of this BIST, which is the full abbreviation of built in self test. They can be classified broadly into two categories: online BIST and offline BIST.

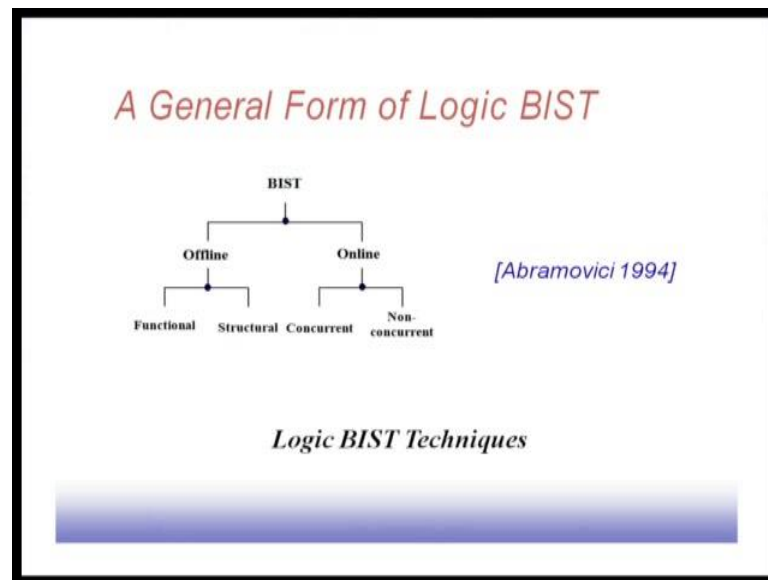
So, online BIST means when the system is on at that time it is going to test the system, so it is testing is done concurrently from the system is on. Again when the system is on there can be two situations: a system maybe on, it may be working doing it normal work or it may be in an idle mode. Particularly say, when we have got a pager for example; when it is doing the paging operating it is in the operational mode, but the pager more than there is no message to be paged it remains idle. So, one possibility is that we extract those idle periods to do the testing of the pager.

So, both are online; in both the cases the pager is on and it is operating, but in one case it is concurrent online testing where the system is operating, side by side we are extracting some information from the system and telling that we analyze the response and try to see whether the systems is working properly or not. So, that is concurrent online BIST. And non-concurrent online BIST, they actually mean that we are putting the system is in idle mode and in the idle mode instead of applying say normal input you are applying the test input and this test inputs are actually being evaluated by the system and the responses are getting compared.

So, this is the online BIST, now there is offline BIST. In offline BIST we put the system in offline mode, so it is not doing the work currently so it is put into a test mode you can say. And then the test patterns are applied to the circuit and the responses are analyzed. So, at this offline mode system is not doing its normal work; it is normal work is suspended. Now there can be two types of offline mode: one is functional offline BIST and one is the structural offline BIST. So, functional offline BIST and the structural offline BIST the major difference is that how do you configure these BIST or what type of patterns are used by the BIST. In functional offline BIST it is actually trying to test the system functionality. So, test patterns are generated keeping the system functionality in mind, so it is actually doing a functional testing of the system.

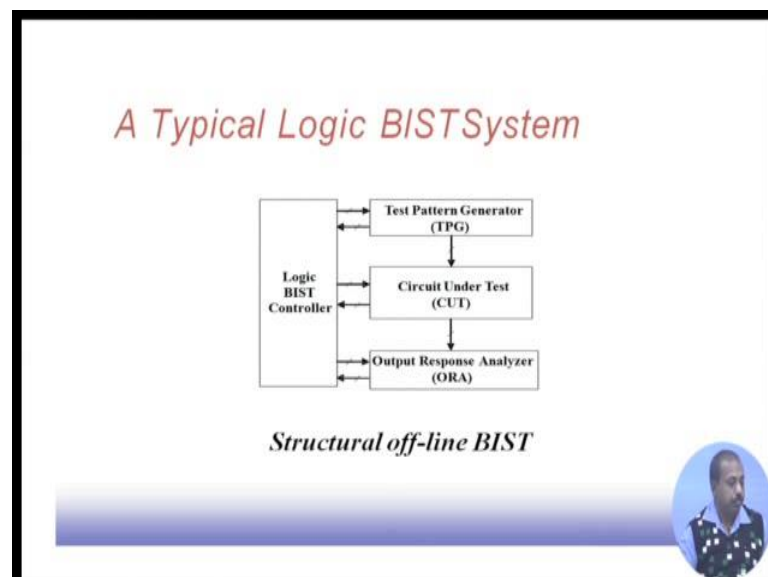
Now, a system may be viewed from two angles: one is the functional angle where we are talking about the system functionality; another angle is the structural angle which means that we are looking into how this test system has been built, what are the components in the system. So, if you are looking from the structural angle you can have a BIST from the structural parts. So, maybe my system has got two three sub modules and for each of the sub module they have got we know they are structures. And for testing those structures we apply some patterns. So, that is the structural BIST. So, based on the structure of the system and functional offline BIST this is based on the function of the system.

(Refer Slide Time: 08:55)



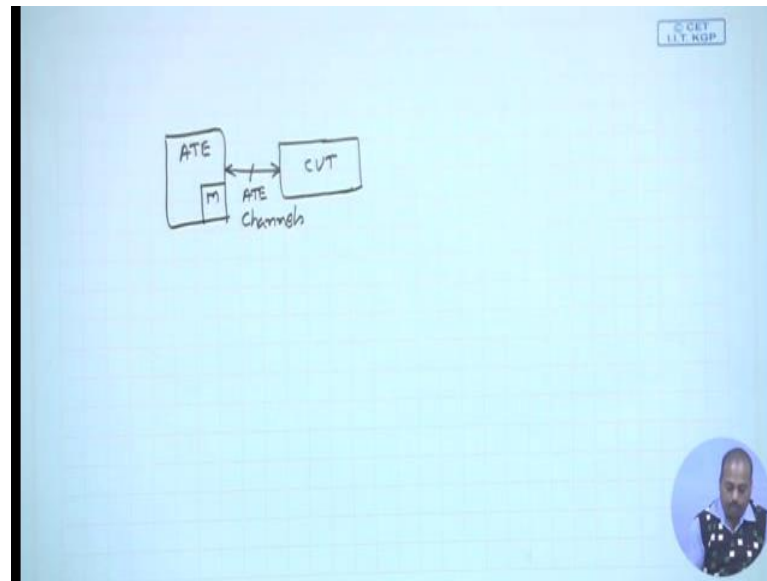
So, this is the classification. So, BIST can be classified into offline BIST and online BIST. Again the offline BIST can be classified into functional BIST and structural BIST; online one can be classified in concurrent and non concurrent. So, this classification was done by a Abramovici in 1994.

(Refer Slide Time: 09:19)



So, how does a typical logic BIST system look like? So, this is the circuit that we have, the circuit under test. Now in deterministic testing or a TPG based testing what we had is.

(Refer Slide Time: 09:34)



If this is the circuit under test then there was an equipment called ATE, automatic test equipment. And this automatic test equipment there was a memory which was holding the test patterns. Now, from the ATE the test patterns are applied to this circuit under test and the responses were obtained from this. So, these lines are known as ATE channels; they are called ATE channels.

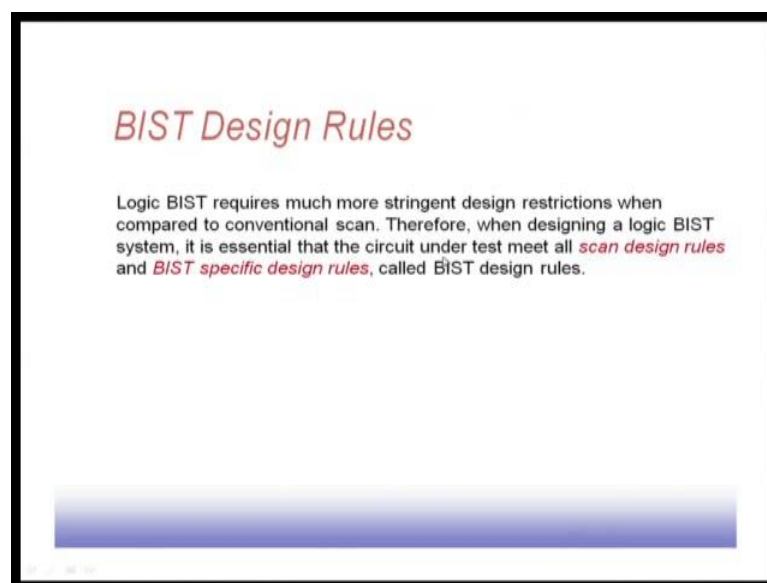
So, this was the situation. We apply the pattern that is stored in the memory, so there is a controller which applies the patterns one after the other from the memory to this circuit under test and the circuit under test returns the response and this response is again compared with the golden response that is stored in the ATE. That was the situation for this external testing or this ATE based testing. In case of BIST; we do not have any other ATE, but there is a test pattern generated which is integrated with the circuit and there is an output response analyzer which is also integrated with the circuits. So, in a deterministic case this TPG and ORA, so they were part of the ATE; now I do not have this TPG, ORA separate but they are part of the circuit itself.

The advantages that from outside world you just need to tell the system to start testing. So, there is no further requirement of transferring test patterns to the circuit or getting the responses and analyzing them, so those requirements are not there. From outside world we can just give the command to start testing. And when it starts testing then after this response analyzer ultimately comes up with the answer yes or no; the system may be

correct, the system may be incorrect so there may be some faults. So, those are actually detected by this response analyzer.

So, there is a logic BIST controller which will tell the system like when to start testing and maybe it needs to stop somewhere or there may be some other controls that are needed. So, we will see when you go to these different types of BIST and all that, some other controls may be necessary which are provided by the logic BIST controller. So, this is a typical structure of a BIST system.

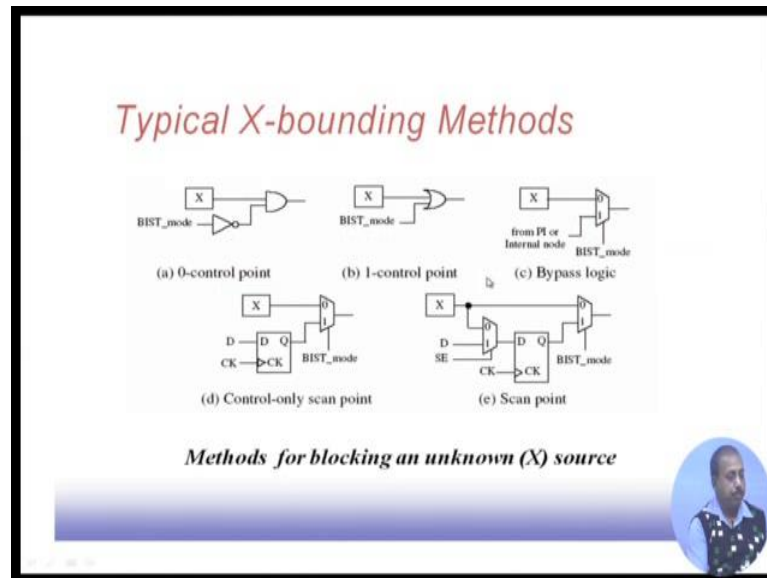
(Refer Slide Time: 11:54)



BIST design rules: there are certain design rules that must be followed while having this logic BIST. It requires more stringent design restrictions when compared to conventional scan. Therefore, designing a logic BIST system it is essential that the circuit under test meet all scan design rules and BIST specific design rules called BIST design rules. So, this scan design rules must be satisfied, because the most of the cases in this test from the test pattern generator this generates a bit streams to feed the scan chains of the circuit under test. So, whatever scans based rules we have they must be satisfied.

And apart from that some more rules are to be satisfied. So, this scan design rules we have seen in previously, we have to look into this BIST specific design rules called a BIST design rules.

(Refer Slide Time: 12:48)



So, a basic problem with this BIST based analysis is that there may be some x values, some unknown value or do not care values that may that may come, basically the unknown values. So, what happens is that in this case, this response analyzer: so the test pattern generator is generating the patterns one after the other so they are applied to the circuit under test. And this response analyzer will try to compact all those responses into some sort of a signature. And then finally, it compares the signature with the golden signature: if the circuit is fault free then due to application of the test pattern generated by the generator, so what is the response, and in the presence of some fault what is the response. So, these two are compact. If it is faulty then it will find out that this signature is going to be different.

Now, the problem is that if there are some sources of unknowns that are there in the circuit under test then this response analysis cannot be done properly, because the values during operation it may pick up any values 0 or 1. Naturally, we cannot consider it in the BIST section. So, what is required is that; for example if we have got one unknown source x then one possibility is the BIST mode we make it 0. So, in BIST mode this is 1, this is a 0, so the AND gate output will be 0. Whereas, in the normal mode of operation this BIST mode input is 0 as a result whatever be the value at this source it will come to this point.

So, in this way whenever we have got one or unknowns source  $x$  we put one invert followed by AND gate so that in BIST mode it will always result in a 0. So, this is called a 0 control point. So, 0 control point has to be may be inserted into the circuit so that this  $x$  is do not come in the circuit during BIST operation. Similarly, we can also have one control points, where we use instead of AND gate we use OR gate, so this whatever with this  $x$  sourcing a BIST mode this output is 1. So, disappoint is power always 1, so it is not on  $x$ . For normal operation this BIST mode is 0, as a result is whatever be the value at this source it will come to signal line. So, there is no problem with the normal operation of the system.

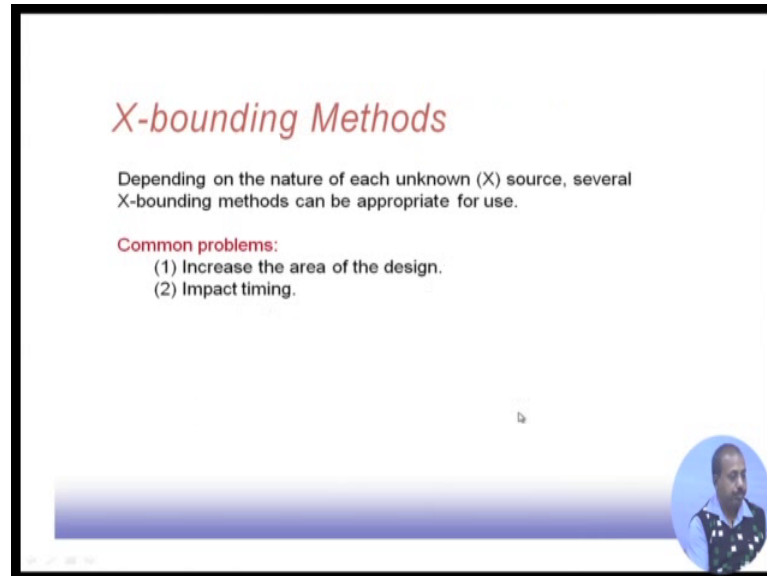
Sometime, another possibility is to use bypass logic. In bypass logic what is done? Maybe from a primary input or some internal node the value is coming and this  $x$  value is coming, so in the BIST mode it will bypass this particular  $x$  value. BIST mode it will take the value from either primary input or some internal node which is not  $x$ . So, the circuit is slightly modified definitely, because now it is taking value from some internal node if not from primary input.

In that case circuit is a bit modified, but the testing of the circuit can go on properly. And in the non BIST mode this BIST mode control will be 0, as a result this unknown source will come to the output, this line. So, normal operation is not hampered, but BIST mode it is taking some definitely value. Then one possibility is that we can also put a d flip flop and then. So, this scan chain, this a flip flop is a part of the scan chain. Now through this scan chain we can put the desired value on to this. So, there may be a number of points of at which we need to insert this different values at in the BIST session, so they can be put on to a scan chain, and through this scan chain we can put this value to a desired one. Again this unknown source  $x$  is blocked by through this multiplexer and this d flip flop.

Also we can do it like this. So, under this is scan enable signal this d input will come here so that is put into this flip flop. And now in the BIST mode this  $q$  value will be available through this multiplexer to the output. So, in normal operation this unknown source value comes here and also this is scan enable line will be 0, as a result this  $x$  bit whatever be the unknown value so that will come to the flip flop. So, if it is required that this line  $x$  should be buffered; so in this case it is getting buffered in a normal operation. So, scan enable is 0 so it is coming to this flip flop, and then this BIST mode is also 0 so it is through this flip flop it is going this scan input is coming to the output, whereas for

scan operation it is getting buffered through this. So, that way we can insert one scan point.

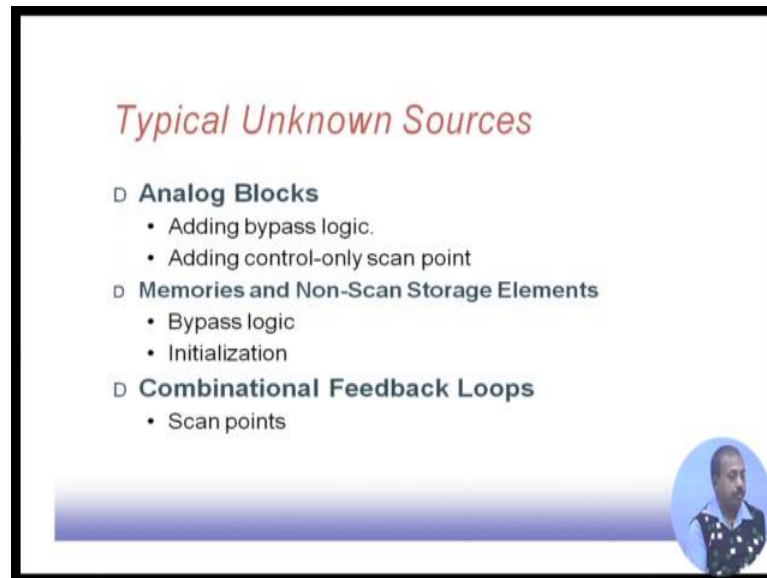
(Refer Slide Time: 17:57)



So, there are several x-bounding methods that are going to be used. Depending upon the nature of each unknown x source several x-bounding methods can be appropriate for use. Common problem: increase the area of the design, because we are introducing some extra logic into the circuit and also timing is getting affected. Because if you look into this everywhere what we are doing is we are putting either extra AND gate OR gate multiplexer or this flip flop and flip flop and multiplexer.

So, as a result this delay of the circuit will change. If they form a part of the critical part and of course the delay of the system will go up. So, timing is going to be affected. What are the unknown sources? We have said that if there is an unknown source we bypass that unknown source by some mechanism.

(Refer Slide Time: 18:43)



So, what are the unknown sources? First unknown source set of unknowns source is the analog block. Now it is a very much unlikely that is system will be completely digital because it has to interact with the environment so there must be some analog component on to eat also. So, this analogue component values at the time of testing those values are not relevant, because we just want to have some typical value and try to see what is going to happen at the in the digital testing process. So, analog block values are not meaningful.

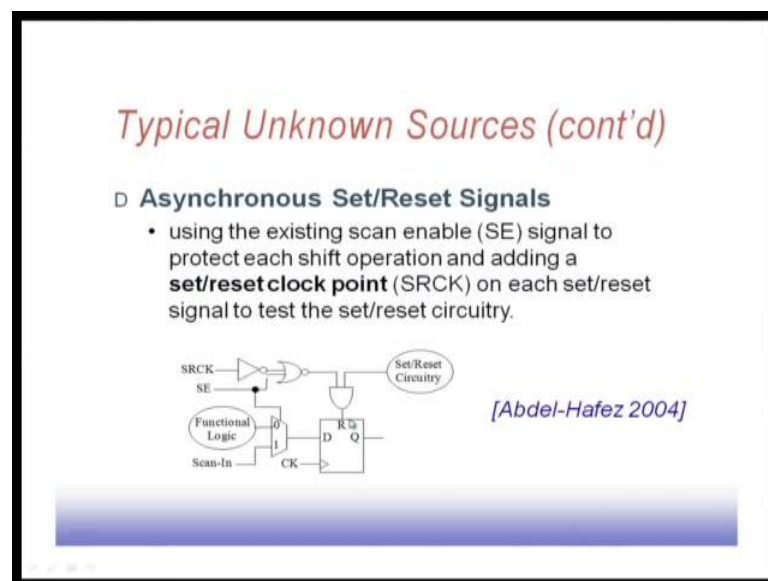
So, what we can do? This analog block we can add some bypass logic. So, this bypass logic edition is by this, so this is the analog block so we bypass it by some other primary input or internal node value. So, that may be one possibility. We can add control only scan point. So, control only scan point is this one, so this unknown source, it is getting blocked by putting this value can be put to this d flip flop and this and this so this can be done.

Memories and non scan storage elements: this is another source of unknown value. So, memory whatever value it will be initialized it depends on the system operation. So, what is the content of memory, how is it going to change, when the system is operating? So that is totally dependent on the actual functionality of the system. So, memories are to be bypassed, because memory content is not known, it is not fixed at all the points and the contents are also not known definitely. So, we have to bypass the memories that are

there. And non scan storage elements; so scan storage we can put it put them on a scan chain and put some values there, but non scan storage elements they will create problem. So, what is the way out? Way out is one is bypass logic, so we can put some bypass logic for them and it can just bypass the content of memory or non scan storage element, or we can initialize them to some proper values. So, that way what are the patterns that we come from them will be known.

Sometimes we have combinational feedback loop. So, combinational feedback loops is another dangerous point, because whenever we are trying to do a fault simulation with is combinational feedback loops. So, it is difficult to come up with stable value. So, it is better that we break that combinational feedback loop by putting some scan point. So, this scan point is this particular structure, so we put some structure so that this loop will be broken and we can put the desired value on to this point via this d put on this flip flop in the scan mode. So, this can be done.

(Refer Slide Time: 21:49)



Another difficulty that we have is with asynchronous set reset signal. So, asynchronous set reset signals what happens is that, in the functional mode these asynchronous set reset signals if it is activated the system will get reset. Now during the BIST operation if this reset signal comes then that creates difficulties, somehow you have to stop that. So, what we can do, we can use this is scan enable signal to protect each shift operation. So, basically this scan enable signal is 1, then this nor gate output will be 0. As a result this

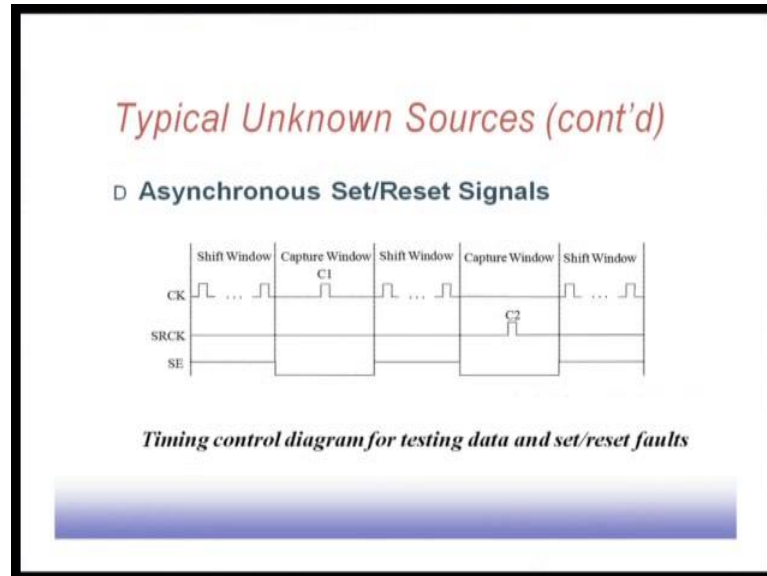
reset line will always get a 0. So, whatever we the set reset circuitry output it is not going to affect this flip flop. So, during the scan operation of this system this flip flop cannot get reset.

Basically, through the scanning line we are sending the data and this scan enable line is 1, whenever this scan enable is 1 this reset line will be 0. As a result whatever be the operation of the set reset circuitry this scan operation will go on smoothly, so it will not reset the circuit. On the other hand if we are going to check the operation of this set reset circuitry then this SRCK line will make 1, as a result we will get a 0 here, then this scan enable line is also 0. As a result we will get a 1 at this point. So, whatever be the set reset circuitry that will have effect on this flip flop. Now, we can check whether the circuit is getting reset by giving this SRCK signal.

So that is what is said, that using existing scan enable signal to protect each shift operation, shift operations are protected so that they do not get reset by chance. And we add one set reset clock point and set reset signal to test the set reset circuitry, if there may be a number of such set reset points we have to protect them individually, we have to check them to test them individually. So, whichever set reset circuitry we want to test for that input will make the SRCK input 1 so that particular reset will only come.

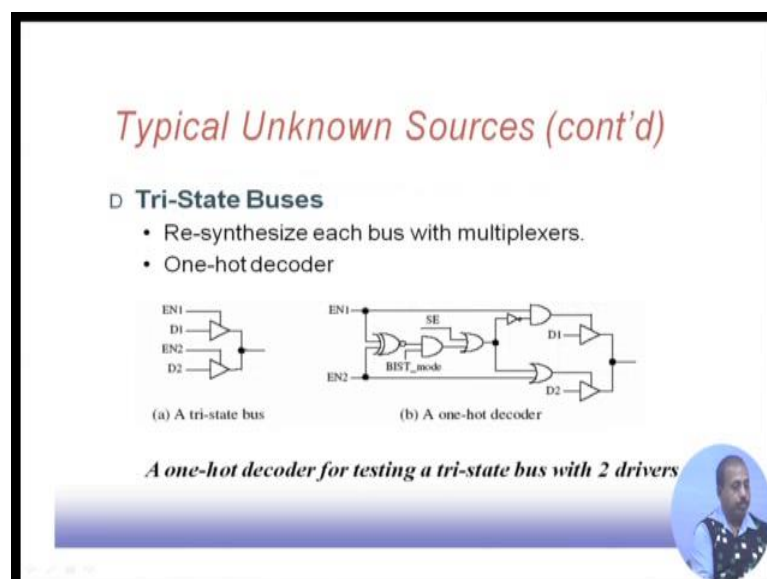
However, if this SRCK is 0 we will get a 1 at this point, we will get a 0. If we do not want that this set reset circuitry should affect the operation now, we can make the SRCK line 0. There are multiple SRCK lines each one for each set resets circuitry. So, we can test them separately.

(Refer Slide Time: 24:34)



So, this is the timing operation that is shown. So, during this scan operation when this scan enable signal is high, this clock will be given for in the shifting operation. And then when this scan enable line is low, one capture signal may be given. So that way this is the doing the normal shift operation. And then if you are interested to do a set reset check, then we can put this SRCK line high at some point of time C 2. If the C 2 is high then this scan enable is low, so the circuit will get reset. Then this of course, in the shift window this line will remain low. So, you will be getting whether the system has been reset or not that can be tested.

(Refer Slide Time: 25:29)



Another source of unknowns is the tri-state bus. Like say, this is the situation one tri-state bus where there are two drivers D 1 and D 2 and there are two control signals EN 1 and EN 2 and it gets the value. Now during normal operation of the system it is ensured that either of EN 1 or EN 2 they are activated. So, whenever we need the content of the bus either EN 1 or EN 2 is activated. However, during test operation it has to be guaranteed that exactly one of them is going to be 1, not both. So, this is modified by means of some one hot decoder.

So, what is done? You see that if EN 1 is 1 then this point will be 0 and as a result this will become as 0. And then if this scan enable is also 0 then this 0 will come here, as a result this point will get 1 and this point will get a 0. So, you see that EN 2; EN 1 being 1, so this signal is coming here so at that time this EN 2 is 0; so it will be coming to this one. But supposed EN 1 and EN 2 both are 1; in that case EN 2 will come here. So, this is an OR gate, so EN 2 will be 1. But this point EN 1 and EN 2 both are being 1 so this is 0, this is 0. And this scan enable signal if it is there then it is 1, this is 0. So, EN 1 will not go in that case. So, these ensure that if both the enables are activated only one of them will be through. And if one of them is one then the corresponding in driver will be turned on. So, that is why it is called a one hot decoder. If both of them are 1 only one of them will finally survive other one will die.

So, this is required because there may be; so when we are applying the test pattern may be EN 1 and EN 2 they become active simultaneously, but that may affect the system operation. So, this is the one hot decoding (Refer Time: 27:47) schedule.