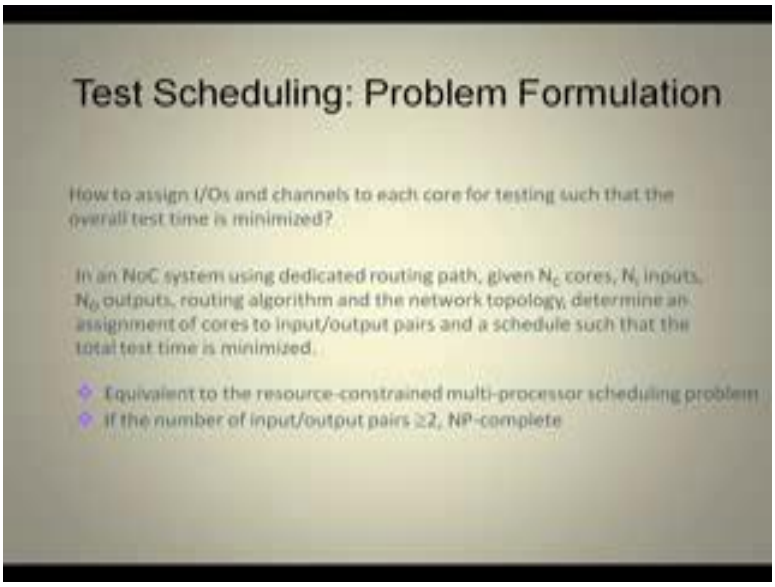
Digital VLSI Testing
Prof. Santanu Chattopadhyay
Department of Electronics and EC Engineering
Indian Institute of Technology, Kharagpur

Lecture - 53
System/Network – On - Chip Test (Contd.)

So, next we address the test scheduling problem, how to schedule tests; testing of all the components.

(Refer Slide Time: 00:21)

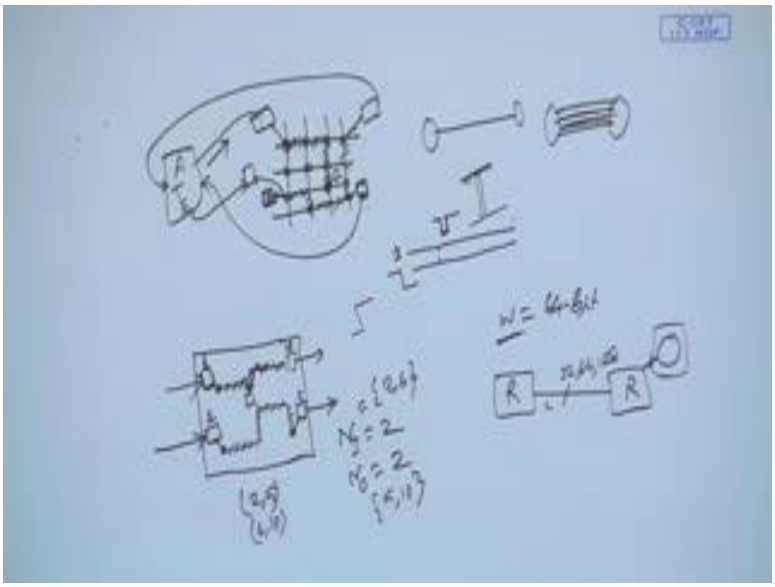


So, when in a (Refer Time: 00:26) testing, when we talk about test scheduling we are talking about testing of cores alone, but now we need to talk about testing of cores, testing of routers, testing of links, even the testing of this network interfaces, all of them are to be addressed. So, of course, the major one is the core testing and then we have to go for the other routers and all those things. So, the basic question that we need to answer is how to assign IOs and channels to each core for testing such that the overall test time is minimized. Since the core has got maximum number of test patterns so we have to reduce this testing time of the cores.

So, compared to the routers which are very simple structure, the amount of testing time needed for the routers may be less so that is patterns are also less. So, that way the major part, major test time is occupied by the cores, by the testing of the cores. So, this is this is. So, try to minimize the core testing time. So, in an NoC system using dedicated

routing path given N C cores, N I inputs and N O outputs routing algorithm and the network topology determine an assignment of cores to input output pairs and a schedule path such that the total test time is minimized. So, this is the overall problem.

(Refer Slide Time: 01:51)



So, as we have said that if this is the full NoC that we have now there are some input output pairs because say this is a router here with that we have got a core and that core is connecting to one of the input channel of the ATE. Similarly, there may be another router here to which we have got an attached core and that acts as the output channel for the ATE. So, this pair, suppose this is core number say 2, this is core number 5. So, this 2 5 it forms a input output pair. Similarly if there may be another input output pair here, this may be output and this may be input so that may be say 6 and 10. So, that is another input output pair.

So, in this way this we have got a number of input output pairs. So, one particular core, if a code is located somewhere here and that we are going to test. So, the path through which this test patterns will be transported very much depends on the input IO pair via which we are going to test it because if we are going to test it via this pair 2 5 then the test patterns will be communicated like this and the responses will again be communicating from this core like this. So, this is the path that needs to be reserved for in that case, this is the path that needs to be reserved for that case.

On the other hand if the testing is done using say this IO pair, the IO pair 6 10. So, this is 6 this is 10, if it is done using 6 10 then the reservation has to be done like this. So, if this is the corresponding router. So, I have to reserve all these things. So, this part has to be reserved and then the response may be going from here and then somehow it has to go there. So, this entire path has to be reserved. So, the reservation differs. Now depending upon the IO pairs that we are using, the path will be differ and as a result the parallelism will get affected. So, we cannot put, we cannot test 2 cores parallely if there some paths are common, if IO pairs are common then definitely we cannot test them parallely.

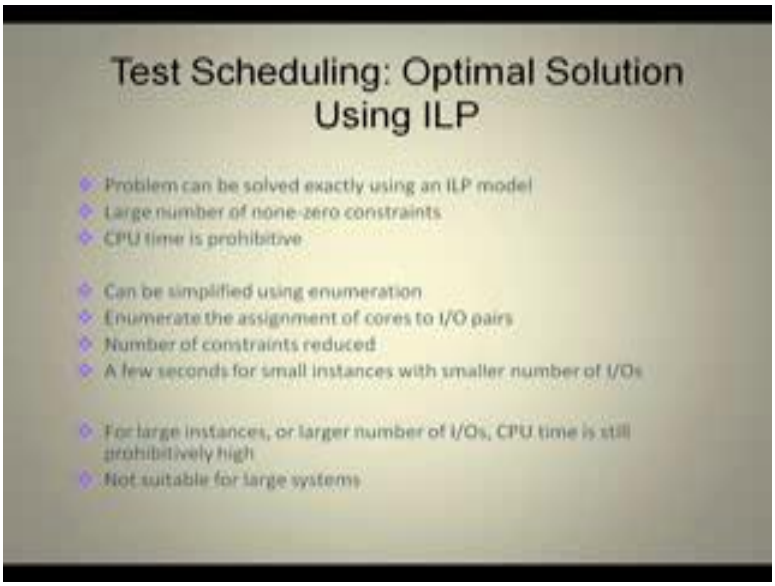
Even if IO pair are different if some part of the path is overlapping then we cannot test the 2 cores simultaneously because we are assuming a preemptive schedule of the testing process. So, we cannot stop one test for some time to the other or multiplex between 2 tests so that is not permitted in a non preemptive testing policy.

So, we have to determine this assignment of cores to IO pairs and also a schedule. So, what are the things given as input the N C the cores the N I inputs and N O outputs. So, how many inputs and outputs are there, and then in the sense that in that input and output core like how many input output cores are there then N I, so in this case we have got N I equal to 2. In this particular example that we are talking about we have N I equal to 2 and N O equal to also 2. So, N I is basically the set 2 and 6 and N O is the set 5 and 10.

So, given this example N I and N O. So, we have to identify the assigned the cores to the IO pairs and find out the time at which individual testing will be done. So, that test time is minimized. So, it is equivalent to the resource constrained multi processor scheduling problem and if the number of input output pairs is greater or equal too then the problem becomes an NP complete problem. So, that way this is also a very hard problem to solve and that no efficient algorithm is known that runs in polynomial time and give rise to, gives us the optimum solution.

So, if it is more than, IO pairs are more than or equal to 2 then it is no more a simple problem. So, naturally in most of the cases to do test should; to do the this testing fast we want to do parallel testing as a result number of input output pairs will anyway be greater or equal to so naturally the problem that we have to solve. So, they are hard problem. So, they are input output problems.

(Refer Slide Time: 06:23)



So, optimal solution can be obtained using integer linear programming, because we have seen for this SOC testing also we can formulate this SOC testing problem as using ILP integer linear programming. So, here also we can solve this problem using ILP formulation and there are many such formulations are available in the literature

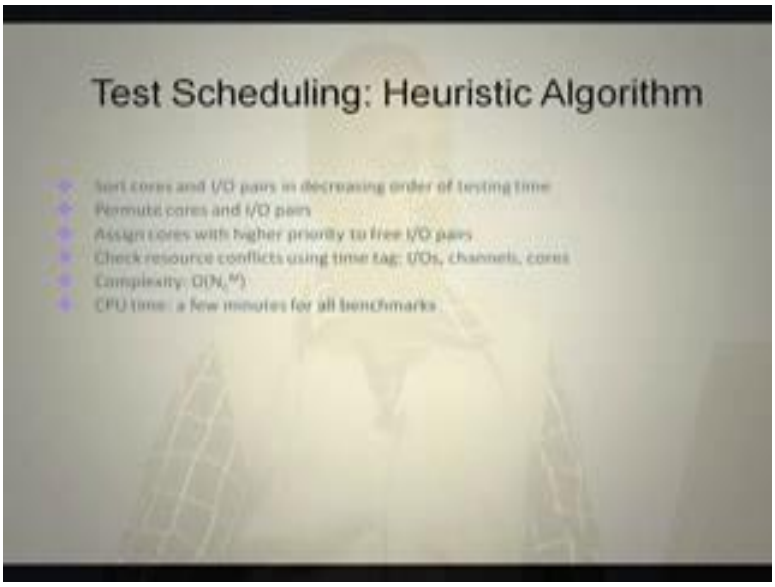
So, the problem that we have first in solving this case with ILP is that there are large number of non zero constraints. So, there are many numbers of constraints so which are non zero, they will be giving rise to some nontrivial constraint that we cannot ignore and as a result CPU time is becoming prohibitively high. So, for a reasonably size problem the ILP formulation will have large number of variables and large number of constraints sp the solution become difficult. So, the CPU time becomes prohibitively high. So, we can solve, we can simplify this formulation by means of enumeration, enumeration means, this ILP formulation is trying to get some solution it is trying to assign values to some variables and get some solution. So, we have some tables pre-computed which tells that if you put it on to this say for this NoC testing.

So, if you are putting on to this IO pair then this is the path that needs to be blocked. So, instead of making the ILP identify the path. So, we can have a table it because it is following a x y routing. So, we can very easily form a table that will tell us like what are the components to be; what are the components to be reserved. So, that may be that is actually the type of enumeration. So, given this it can help us in doing the operation. So,

you can enumerate the assignment of cores to IO pairs. So, we can tell that, if this is assigned to this IO pair then what is the path if it is assigned to another IO pair then what is the path. So, instead of making the ILP identify the paths so we can get this thing.

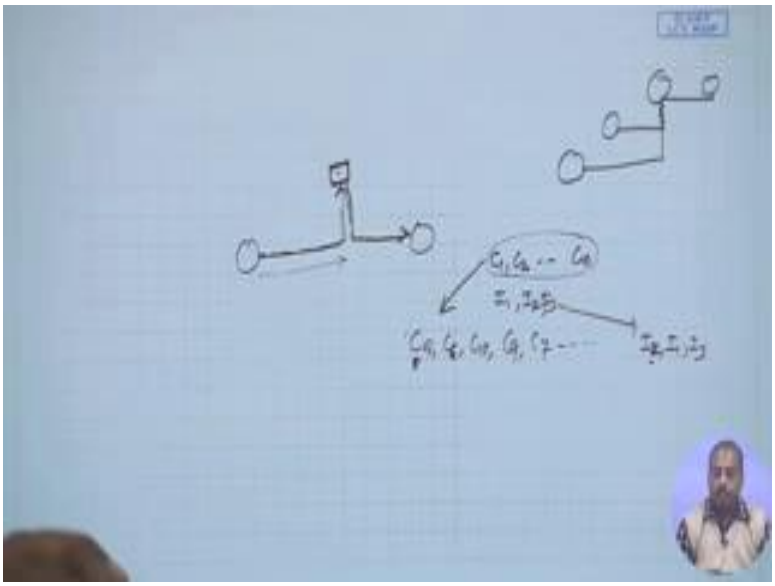
So, if we do that a number of constraints gets reduced. So, that way it simplifies the problem, and the as a result the ILP formulations that have been reported in the literature. So, that may take much less time for small instances with smaller number of IOs for larger instances with larger number of IOs CPU time is still prohibitively high, so it is not suitable for large systems. So, that is the basic problem that we have, so there we have got this system requiring lot of time to get the solution.

(Refer Slide Time: 09:18)



So, there are some heuristic algorithms that have been developed. So, one possible heuristic is it works like this - first of all we sort the cores and IO pairs in descending order on of testing time. So, it is like this if I am testing on particular core using some IO pair say I am testing say this core using some IO pair here, this 2 are the IO pair. So, basically the path will be going like this from here to here and from here to here through intermediate router.

(Refer Slide Time: 09:40)



So, we know how many routers are there, what are the delays of individual routers, how many links are there, what are the delays of individual links, moreover how many test patterns are there so that this is going to reach this thing. Now since we have done reservation we have, there is no conflict. So, at the time of testing the time at which we schedule the test of this, these resources are definitely available. So, test patterns they will go through this channel in an uninterrupted fashion. So, unlike the normal operation in an NoC what happens is that if you are sending one packet from here to here. So, it is following this path now there may be another communication which is going on. So, may be another communication from say here to there, then it also follows this path.

Now this part of the path is common between the 2 communications. So, there will be a conjunction. So, there will be competition between the 2 paths, 2 communications to proceed. So, as a result the conjunction will occur and that the time required for communication, it depends on whether this 2 communications are occurring together, they are occurring one after the another and all that, but in case of testing that problem is not there because we have done reservation. So, we have reserved this channel for this much of time, for the entire test time of the core. So, there is reservation so we do not have this thing. So, we can compute what will be the test time required. So, for every core and every IO pair assignment we can find out what is the test time that is needed.

So, we sort cores and IO pairs in decreasing order of testing time. So, whichever core takes highest time so that is put at the beginning. Then we permute cores and IO pairs, permute cores and IO pairs means we try different sequence of cores core testing and on to different IO pairs.

So, it may be we assign some of the cores on IO pair 1 some other cores and IO pairs 2 and in some other combination. So, we assign some other set of cores to core 1 while another fourth set of cores to IO pair 2. So, that way we can try out different permutations of IO pairs. Then assign cores with higher priority to free IO pairs. So, after we have taken that will consider the cores in some order here and we will consider the IO pairs in some ordered here like I have got say 10 cores. So, C 1, C 2 up to say C 10 and I have got the IO pairs I 1, I 2, I 3 suppose there are three pairs.

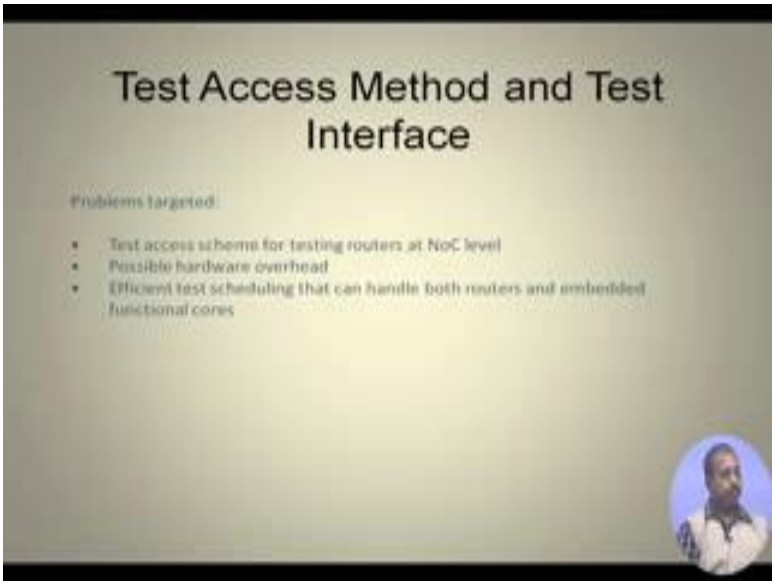
So, which core will be taken up first for scheduling and to which IO pair it will be assigned, it is done like this. So, the first we try out a permutation of this C 1 to C 10 may be the permutation says that C 5, C 6, C 10, C 9, C 7. So, it goes like this and this IO pair permutation is taken suppose the IO pair permutation is like I 2, I 1 and I 3. So, what is done first? This C 5 will be taken up for scheduling. So, we will see, we will try to put this C 5 1 to I 2 ok.

So, C 5 1 to I 2, I know what is the test time and all that then it will be if I 2 is full then it will be then it will be pick up C 6 and it will try to put C 6 on to some IO pair. So, that whichever IO pair is free. So, this says that one that assigned cores, higher priority to free IO pairs and check for resource conflict using time tag. So, if we put a time tag like this, particular channel is busy from time 0 to time 10 when the testing of core 5 was going on, at those time, no other none of those channels can be used for the testing purpose. So, that way by putting time tag. So, we can avoid this type of overlapping.

So, we check for resource conflict using this time tags and this for time tags for IOs channels and cores. So, that is input output IO pairs the channels, the core that we are going to test. So, the complexity of this process is N c M of the order of N c M and this and choosing, and CPU time is few minutes for all benchmarks, for the ITC 0 to benchmarks it has been seen that this type of heuristics they work very fast. So, they may not give a very good solution, but in many cases they are very fast, so we can try to, we

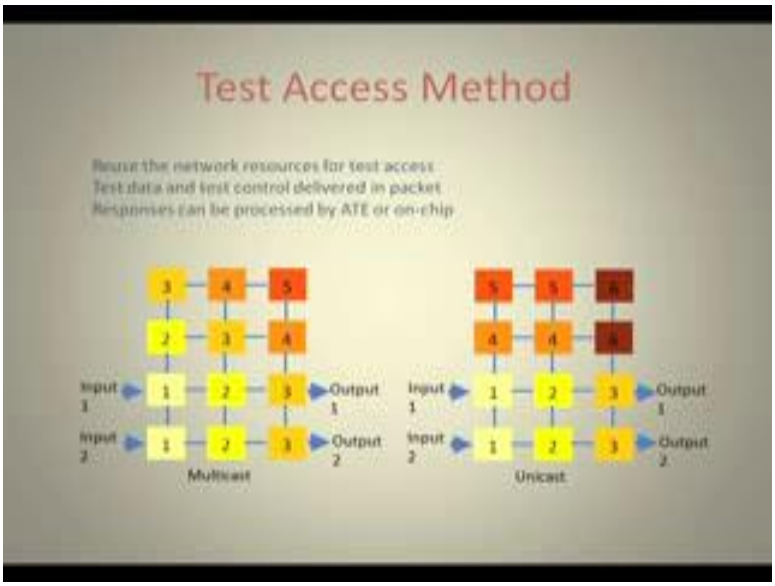
can try out more permutations of cores and more permutation of IO pairs so that we can get some better result.

(Refer Slide Time: 14:42)



So, how to design the test access method and the test interface, so test access scheme has to be design for routers at NoC level. So, apart from testing of cores, we need to test the routers. So, this routers how can we test so that has to be thought about. So, it can add hardware overhead like for this router testing we have to apply some test patterns. So, whether those test patterns are coming from ATE or whether those test patterns are generated at the site itself that becomes a big question. So, whatever we do it will add to some overhead and efficient test scheduling that can handle both routers and embedded functional cores. So, both of them are to be tested. So, these are the problems that we need to address when we are talking about this NoC testing with routers and cores they are going to be tested.

(Refer Slide Time: 15:42)



So, the test access mechanism it says we reuse the network resources for test access and test data and test control will be delivered in packet and responses will be processed by ATE or it can be on chip also.

So, you see what is happening is that. So, this is the input output pair 1 - input 1 output 1 and this is the input output pair 2 input 2 out output 2. So, in a multi cast testing, what we do is that so all the routers being similar, they can be tested by the same set of patterns. So, what is done is that this initially this both of this 2 routers. So, they will get the test patterns. So, at time instant one, both of them will get this thing.

Now at time 2, this can be communicated to this router, the pattern so that way it becomes 2 and then this can be communicated to this router as well and this can be communicated to this router. So, at time 2 these three routers are going to be tested. Then at time three, so this values can be send further the test patterns can send further from 2 to 3, so that these are the cores, these are the routers that are going to be tested at time 3. Then at time 4, it is forwarded from this time 3 routers to the time 4 routers that at the next level at time 4. So, it is (Refer Time: 17:15) transferred to this one and finally, to 5.

And of course, the responses are obtained in this way, so to output 1 and output 2 to see all the responses. Now response compaction may be along with the router itself. So, we can have itself some small module here which will check the router responses by means of some signature analysis and things like that or it may be that this responses are

transferred to the ATE and then it is analyzed at that point. So, both are possible. So, this is the multicast mode of transfer. Now it is not always possible, if the routers are supporting multi cast then only this mode of transfer is possible. So, normally it is a design parameter, so designer in the design if it is required to do some multi cast operation then only the designers will keep the multi cast mode for the routers and one thing we must keep in mind that these routers are much much simpler compared to the routers that we have in computer networks.

In computer networks in many cases an entire computer is dedicated to work as a router, but here you see that we are talking about on chip routers, they are very simple. So, they do not have much intelligence in them. So, whatever functionality is the bare minimum. So, that is put by the designer in to this system. So, if it does not require multi cast in the design phase, so this multi cast facility will not be there, but unicast will definitely there. So, in unicast more what will happen? So, in time step 1, this cores that a test pattern will be sent to this core this router from the first channel and from the second channel it will be sent to this router.

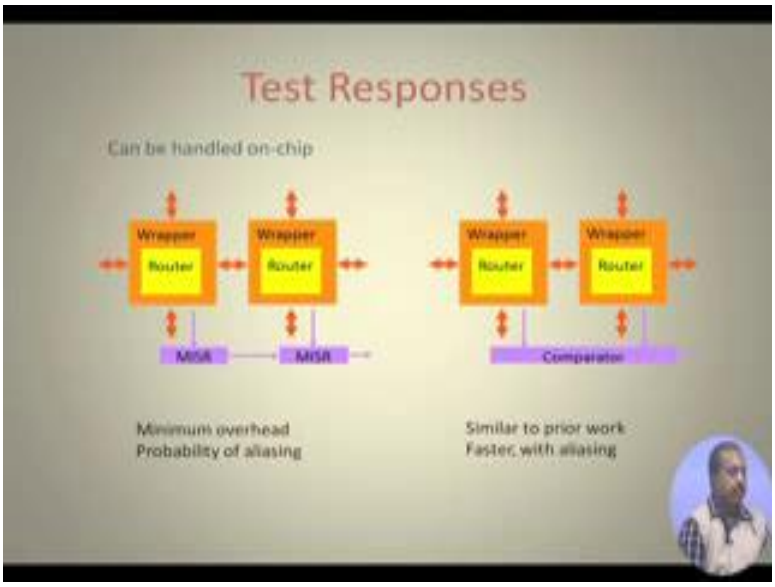
So, in time, so this is unicast, it can be forwarded to only one destination. So, it is forwarded to destination 2 where this router will be tested at time 2. So, this is tested at time 2, then it is going to 3 here and 3 here and once this is over then it can send this pattern to at time 4. So, these 2 communication can go on parallely because this is from this one and this is from this one. So, these 2 communication can be go on parallelly and then they get 4 4. Now this is getting after that, so this will get transferred.

Actually you can say that why not 3 to 6. So, that is also possible, but what is required is that the response also has to be collected. So, if you want to do that. So, only 2 of the cores can be tested, 2 of the routers can be tested at any point of time because the response also needs to be transferred to the output channel. So, even if we say that this three will transfer the test pattern to 6, but the response cannot be collected because this 3 cases are there that this 3 routers are going to be tested parallel so that is not possible we cannot collect the response here. And as it is going that reservation based policy, there is no there is no way to reserve the channels that way.

So, only we can reserve channels so that this input to the core to the output. So, it is 1 2 3 it is going like this. So, first it is testing this then from input it will be sending pattern for

this router or it may be forwarded from the previous stage whatever it is. So, it get it is getting the test pattern. So, this way this you see that in this case we will require 6 times steps to do the test access. So, that way it is time requirement may be high compared to a multi cast situation.

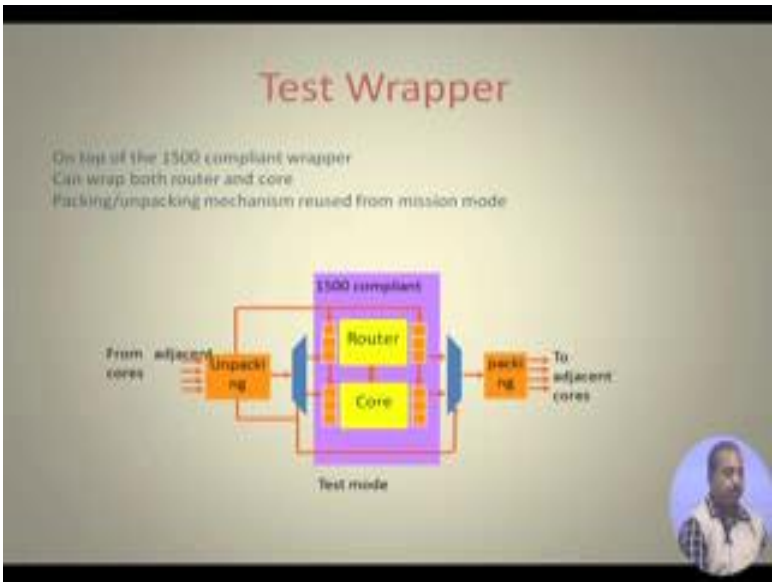
(Refer Slide Time: 20:56)



Now, the response part as I was telling that it can be handled on chip or it can be handled off chip. So, it can be handled on chip by means of like this one, this router whatever is there. So, this is this can give the response to the MISR and similarly this router gives this response to this MISR. So, this MISRs they will be compacting this responses over space and time and then it can be transferred, it will be transferred to the ATE for comparison. So, there this compacting can be done.

And in this case, what happens is that they this responses they are sent to some comparator which does a comparison of the responses and then after doing the comparison this comparator sends the result to the ATE. So, that way this can be faster of course, there is a chance of aliasing here, but here the chance of aliasing may be more because this comparator it just says yes or no compares between the 2 responses. So, it may not be able to identify or pin point the faults that have occurred in some router.

(Refer Slide Time: 22:19)



So, test wrapper design is similar to this 1500 wrapper. So, this test wrapper part, it will be putting the cores into a situation so that it can if the test patterns can be applied carefully to the NoC to the individual cores, but otherwise it is similar to the 1500 wrapper. So, we will continue in the next class.