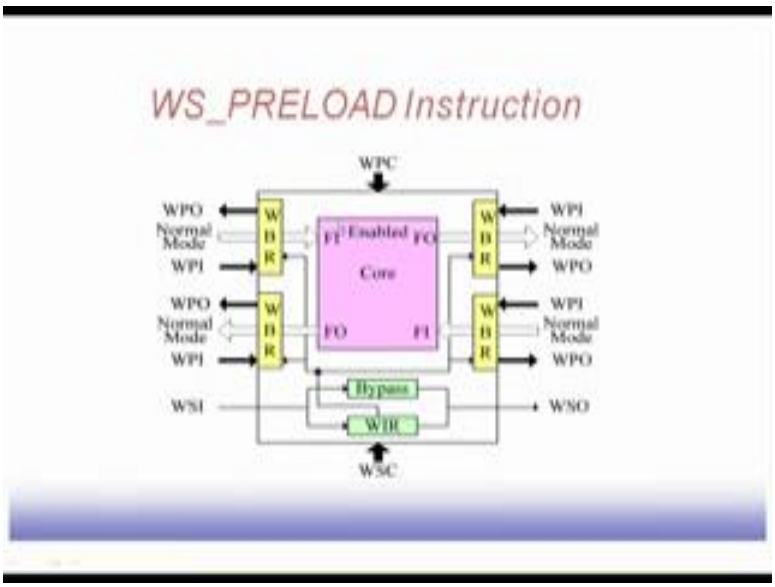
Digital VLSI Testing
Prof. Santanu Chattopadhyay
Department of Electronics and EC Engineering
Indian Institute of Technology, Kharagpur

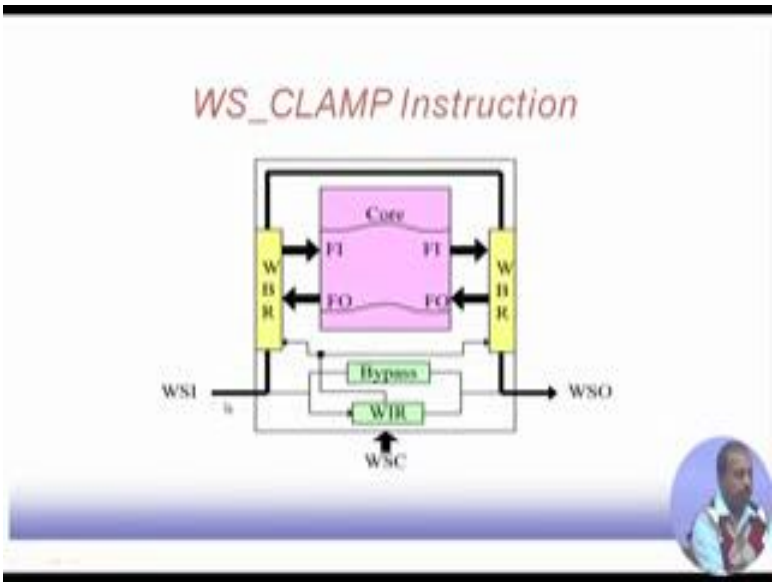
Lecture - 44
Boundary Scan (Contd.)

(Refer Slide Time: 00:28)



So, wrapper preload instruction, the WS preload instruction, so that can also be used. So, in this case, whatever it is a serial preload, so if it is serial preload then whatever is coming in that WSI line, so that is actually loaded into this WIR register, so that way rest of the thing is not active. So, the serial control will be loading this say this WIR register. Similarly, this parallel control can be used to load this WBR registers, but whole thing is the preload operation. So, it is not sent to the core under test. So, this is basically that R 1 flip-flops of all these WBRs, WPCs, so they are actually configured into chain and they will be loaded the values. So, this WPI, WPO lines, so these values will be loaded as a preload instruction.

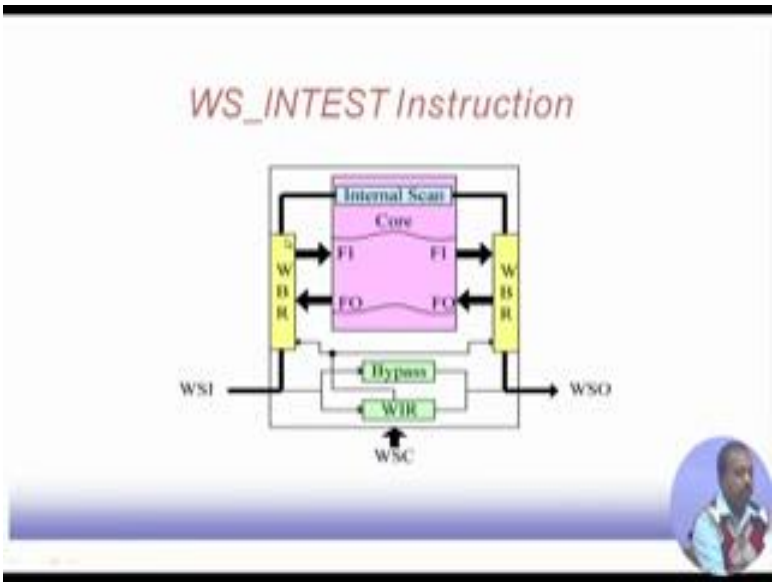
(Refer Slide Time: 01:17)



So, then there is a parallel preload WP preload operation, so this is again the same thing that WSI will be put into the bypass mode. So, this wrapper serial input is not going to affect this WBR, so that is going into bypass mode. And whatever has been loaded into this WBR, so they are available as preloaded state. So, similarly this WBR content is also available as a preloaded state so whatever values were preloaded, so they are available at the output of those parallel outputs.

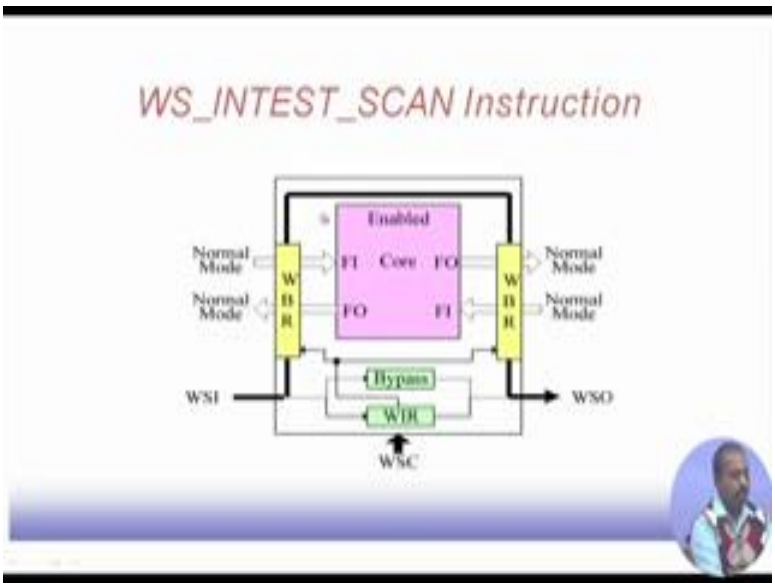
Then there is a clamp. So, clamp means that this WSI and WSO lines, so those values have been applied, but this is they are applied to functional input and functional output that passes through this core actually the functional input that are available at WBR, so that is passing through the core. Similarly, so they are affecting the functional output and then these functional outputs are loaded onto the WBR, this direction should be reverse. So, FI should be this side FO should be that side. So, anyway this is basically the pattern passes through the core, the clamp mode.

(Refer Slide Time: 02:39)



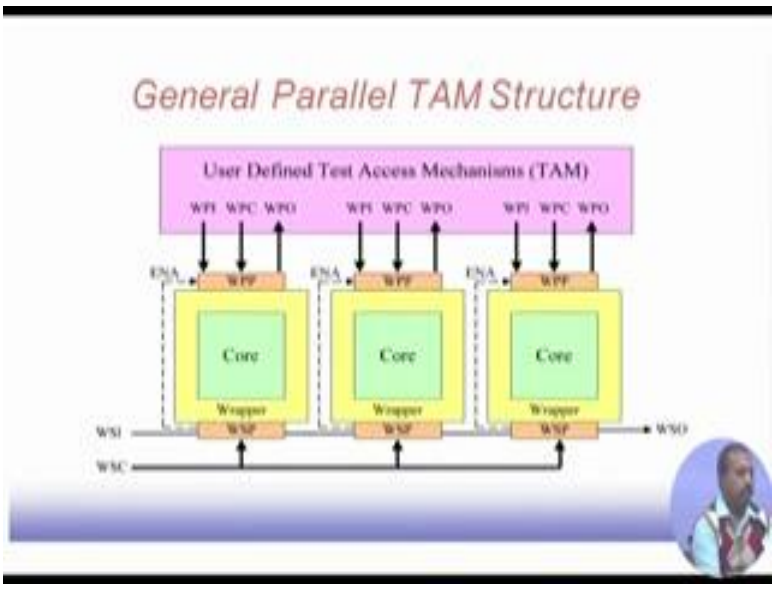
Then the INTEST mode, so in the INTEST mode also it is similar. So, this wrapper this boundary registers wrapper boundary registers, so they are going to be connected through this internal scan chains. So, they are in the previous case only functional input and functional output were involved. Now, this internal scan chains are also involved. So, internal scan chain will get the values and accordingly this core will get the full setup inputs and that will be passing through these WBRs and then that values will be available at this WSO line. So, serially this WBR values will be available in the scan outlines.

(Refer Slide Time: 03:26)



INTEST scan. So, INTEST scan means here the functional inputs will not be fed, but this WSI lines, so they will be passed the scan chain will be formed the scan chain. So, it will pass these lines through this scan chains. So, this is basically for checking the boundary scan register. So, some pattern that we need to pass through this scan chain to see whether this scan cells are ok or not. So, these boundary scan cells are ok or not. So, some pattern may be fed through this and that will check the scan part. So, this whether scan chain part will be tested, so that is the scan instruction.

(Refer Slide Time: 04:08)

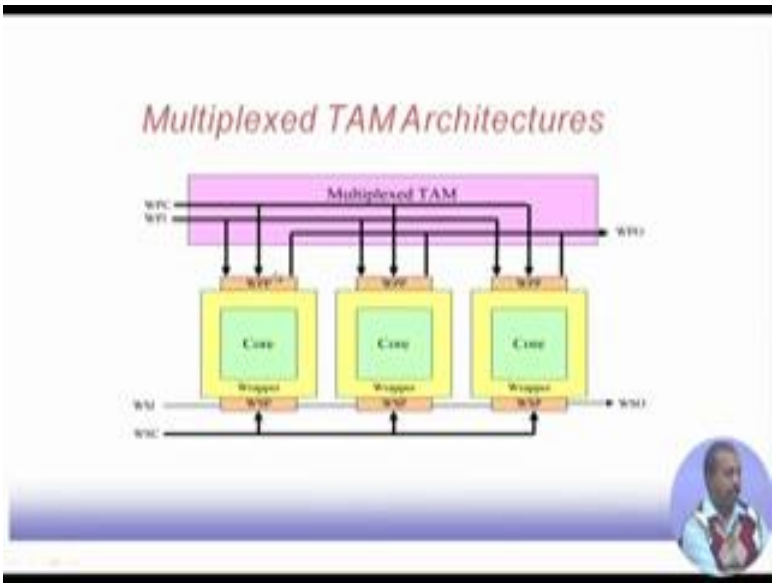


Next, we will look into what are the possible test access mechanism architectures that are there. So, we have got this serial input. So, core has got a wrapper it has got one serial port and a parallel port. So, there are 2 such ports. Now, the serial input that is coming, so it can go through this, it will go through this serial interface and these parallel inputs they will go through this parallel interface. So, this WSP register, the serial port register the serial port interface, it will have an enable line to the parallel port at least through this WSI line, so we have to give some instructions, so that this parallel loading or parallel operations are enabled. So, this that is why the serial part is mandatory of this 1500 standard, but this parallel part is not mandatory, but parallel part may be used for doing the loading.

So, this WPI, WPC, WPO, so these are the parallel parts, so how these will be connected, so that is determined by the user. How this WPI, WPO lines of individual cores they are

going to be connected, whether it is on a bus, whether it is some distributed lines etcetera, so all those will be decided by the user. But whatever it is then that loading and unloading has to be enabled by the serial interface, so that is why the serial part is must.

(Refer Slide Time: 05:43)



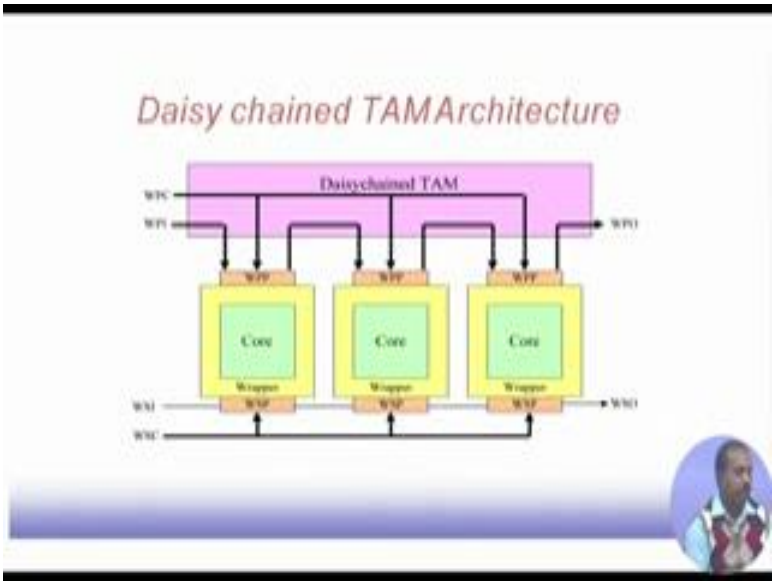
So, one possibility is that we have got multiplexed TAM architectures. So, multiplexed TAM architectures means out of this WPI, WPO and WPC lines, so they go to all the places like say WPI line is going to all the 3 core interfaces. This WPO line they are coming out of all these parallel interfaces and going to the WPO line; and this WPC line So, it is going to the again going to control individual wrapper parallel wrappers the parallel part of it. Whereas, this WSI line, so it is connected to serially one; and this WSC the control line, so it is going to control this WSP the serial port serial part interface and naturally there has to be some control for that control for this WPP from the WSP.

Now, at one point of time, so you can test only one core because they are multiplexed in nature. So, we do not need large number of inputs only, so whatever be the WPI value only that much bit is needed for sending in the test pattern and similarly for getting up. So, you can say that altogether we need WPI plus WPO number of bits for the test bus that we have and the multiplexed TAM is sorry this WPC lines will also be necessary, but that is the maximum line that you are going to allocate to any of these cores.

We can have some set of daisy-chained structure. So, WPI line face it interfaces the first core then this WPO line of first core connects to the WPI of the second one. Now, if you put them in a bypass mode then this WPI line will be going through this one, then it will go to this WPI of second one, so it will it will go like this. So, this particular architecture is known as daisy-chained TAM architecture. So, we have got this daisy-chained policy and then of course, it takes time. So, whenever so for testing this core it has got the highest priority whenever this test core test is scheduled. So, it will happen immediately it will get the WPI line. Similarly, if it is, so I can test this core only when this one is not being tested; similarly I can test this core only when the first 2 are not being tested, so that the schedule test scheduling become constant by this TAM architecture.

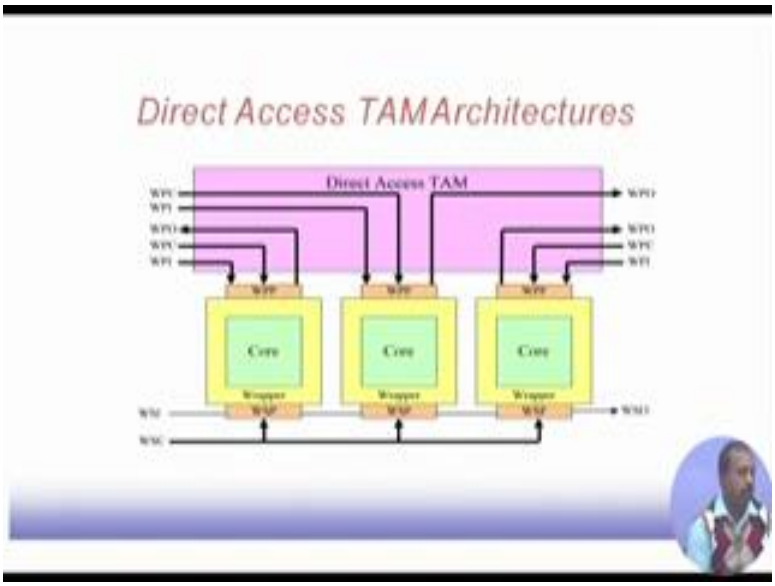
Whereas, in this previous case, so since it was multiplexed, so I can do multiplexing in any order. So, I can if the cores they support preemptive testing in the sense that I can apply say one core into a 100 test pattern, so I apply only 50 of them now and 50 of them later. So, that way that type of interleaving if it is possible then also this multiplex time is going to help.

(Refer Slide Time: 08:43)



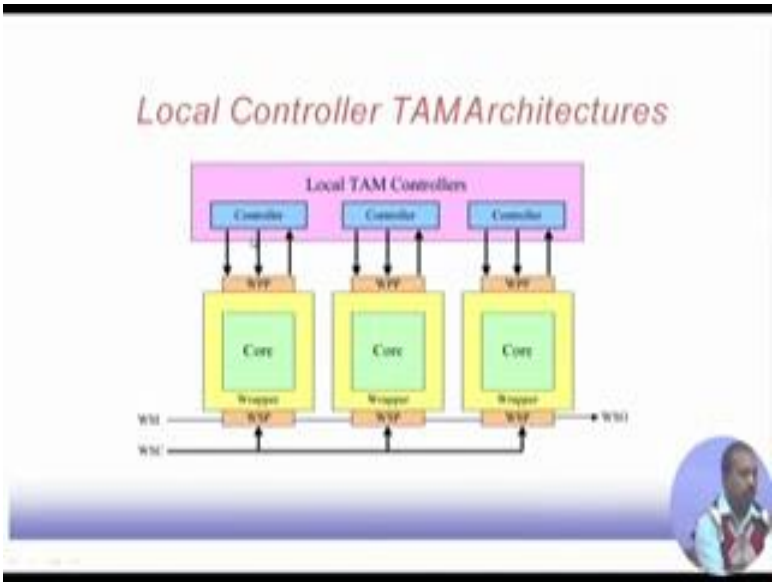
But here in this case once a test is going on, so nobody else can be tested, so that is the problem with these daisy-chained structures.

(Refer Slide Time: 08:51)



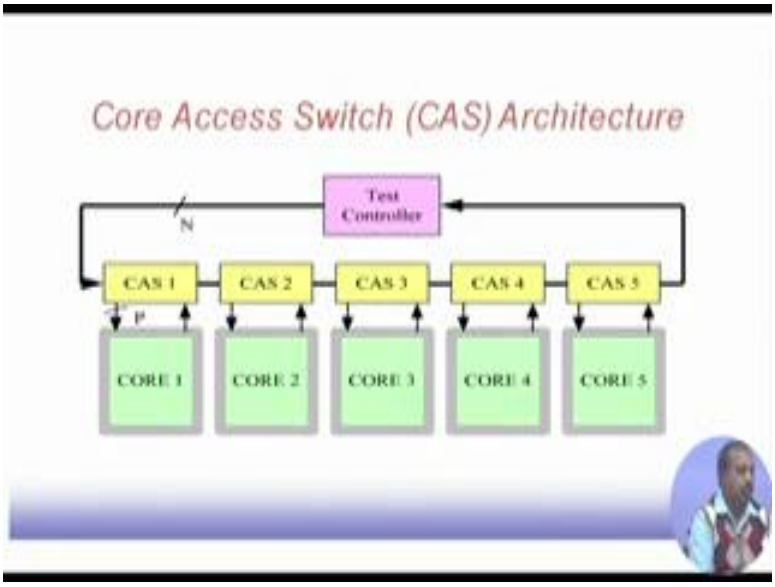
Then we have got this is the most flexible structure the direct access TAM. So, this WPI, WPO, WPC lines, so they are dedicated for every core, so they are given separately so this triple WPI, WPC, WPO is for core one, this is for the second core, this is for the third core like that. So, naturally here the advantage is that the number of ATE channels needed will be large, because a large number of bits will be are to be fed for testing though we can do parallel testing, but a large number of bits need to be fed to the circuit. So, as a result we have got this ATE cost will go up, but test time we can say that it will get reduced.

(Refer Slide Time: 09:46)



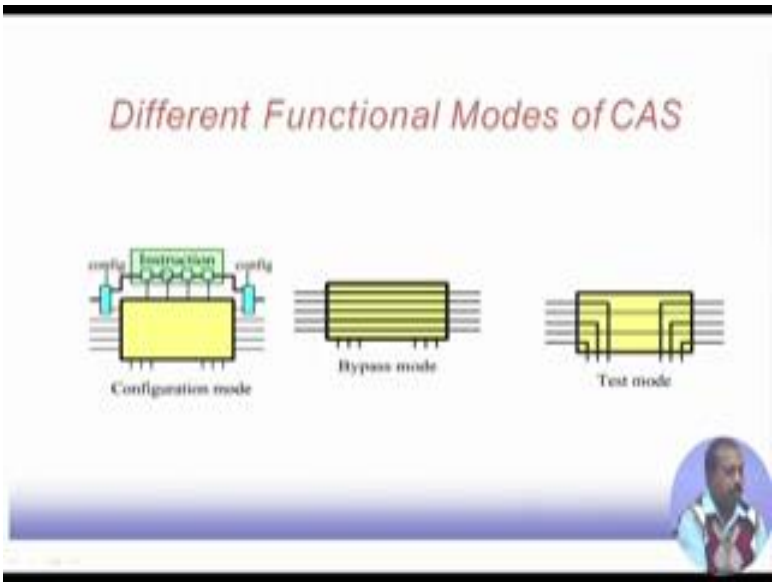
We can also have some sort of local controller for TAM architecture. So, it says that we have go so this controller. So, this controls this operation. So, this WPI, WPO and this WPC lines, so they may be controlled by this local controller. So, we may be from a global point of view, so we can give instruction to these controllers to operate. So, we can give some top level command to this controllers and then these controllers, so they will give commands to the they will control the actual test session, so that can be done. So, you have got this type of local controller TAM architecture, so that can be utilized.

(Refer Slide Time: 10:36)



We can also have some sort of core access switch. So, core access switch means, so there is a test controller, so it will pass this access of this WPI, WPO lines into this switches through the switches. So, if this switch is done, so there are N lines coming from the test controller some part of it is for selecting the switch and some part is actually there those TDI, TDO lines. So, once this switch is on, so this connection will be to this core. Similarly, when this switch is on, then this connection will be to this core. So, we have got a controller and there are a number of switches core access switches through which this control will be this access will be controlled. So, that is the core access switch.

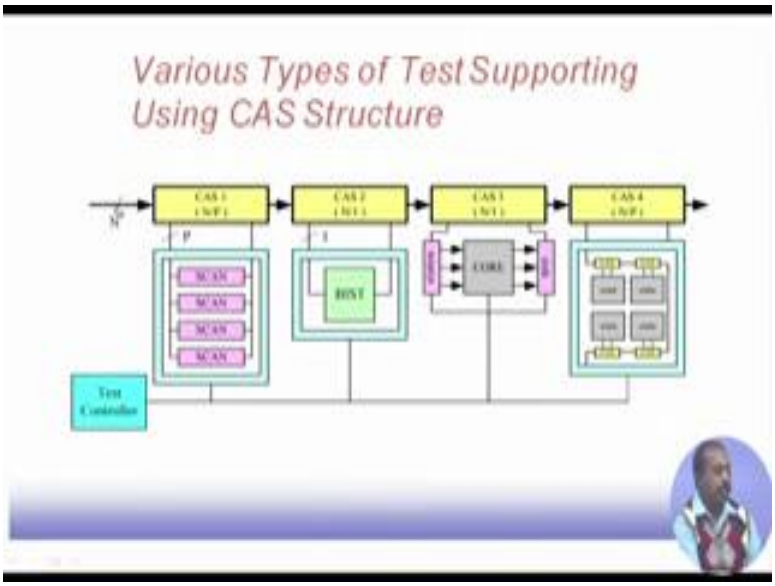
(Refer Slide Time: 11:30)



So, this core access switch, so it can be operating in the configuration mode. So, here in the configuration mode what happens is that the instruction may be loaded. So, now, the core is not being tested, but only the instruction is being loaded into the instruction register. So, this configuration selection point, so it can say it can select that it is now it is now putting the data onto the configuration onto the instruction register. So, this configuration bit can be controlled accordingly. So, that instead of that pattern reaching the core, so they will be reaching the instruction registers only of the wrapper.

Now, in the bypass mode, so they are just bypassed to the output. So, this parallel inputs, so they are outputted here. So, bypass mode or it may be in the test mode. In the test mode may be I will apply the first test pattern get the response similarly, sorry this first bit is going to connect to the first input then may be for this particular core testing, only these 3 lines are important. Similarly, say only these 3 lines are important for output, so rests are not. So, rest of them may be passing directly through this switch, but we can have only these 3 lines feeding the core that we have downwards. So, some of them are input lines, some of them are output lines. So, input lines will be connected to this output lines connected to this. So, as a result, whatever test pattern has to be applied, so it will be applied and the response will be collected on these 3 lines. So, remaining 3 lines may remaining 2 lines it may be used for testing another core simultaneously or it so depending upon our scheduling policy, so we can select those lines.

(Refer Slide Time: 13:23)

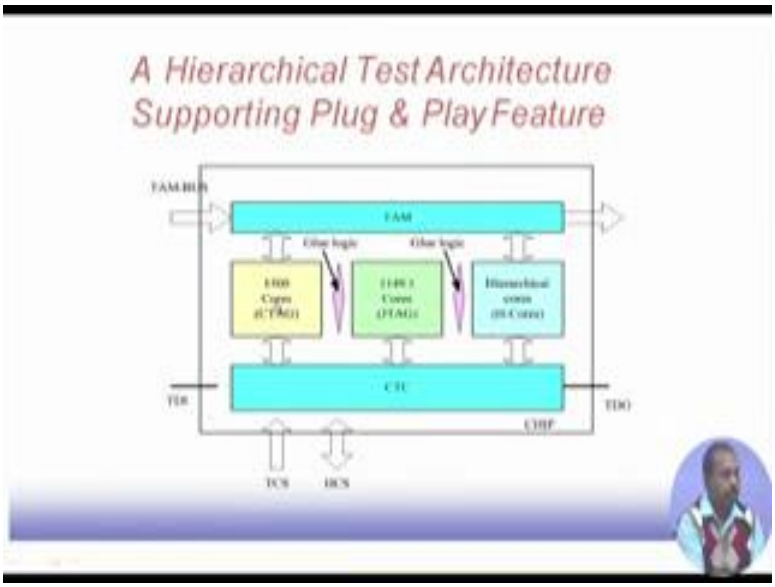


So, there can be various types of tests supported using this CAS structure. Now, one possibility is that I have got a core that has got scan chains. Suppose, there are P numbers of scan chains, so from these N lines, so P bits are to be used for feeding this scan chain. Now, this part is second core, it may be a BIST structured core. So, from this input, I need to just I just need a command that it has to go in a BIST fashion. So, it will generate its test patterns in a from the serial random pattern generator and response will get analyzed by MISR. So, all those things will be done by the BIST that we have integrated with the core.

Thirdly, so this core may be it has got its own test pattern generator and sink loaded with; it may not be a BIST. So, BIST may be using a pseudo random pattern generator. So, it may not be using a pseudo random pattern generator. So, it may be some specific pattern sequence that is generated by some hardware, maybe it is stored in some memory there or may be generated by some hardware. So, that is generated applied to the core and the response is collected at the sink, so that way we can do these things. So, this is the core access, this is the core access which this can be configured, so that it will feeding this source. So, again only one bit is necessary to tell the source that this is the thing; similarly this sink pattern will be response will be collected, it may be shifted serially or may be the final compacted pattern will be shifted, so that way it can be done.

And another possibility is that it may be a hieratical structure. So, instead of having a single core, there may be 4 cores for example. So, here also I have got this type of core access switches available with these individual cores. So, there may be the line will connect it in that first switch that switch may be connecting to the other switch, so that way it may be connected. And finally, it is going out of this core and going to this line. So, what happens is that this core this ATE has given us N bit for coding this pattern for coding the operations to be done at various places. So, if we can judiciously do a planning, you can judiciously do some test scheduling, so that I can use P bits for feeding this scan chains, I can use one bit for feeding this one, again one bit for feeding this and again p bits for feeding this CAS lines. So, that way I can be able to test a number of modules parallelly, because they are using different testing mechanisms. So, I can use them parallelly, and I can get a faster testing of the whole chip.

(Refer Slide Time: 16:38)

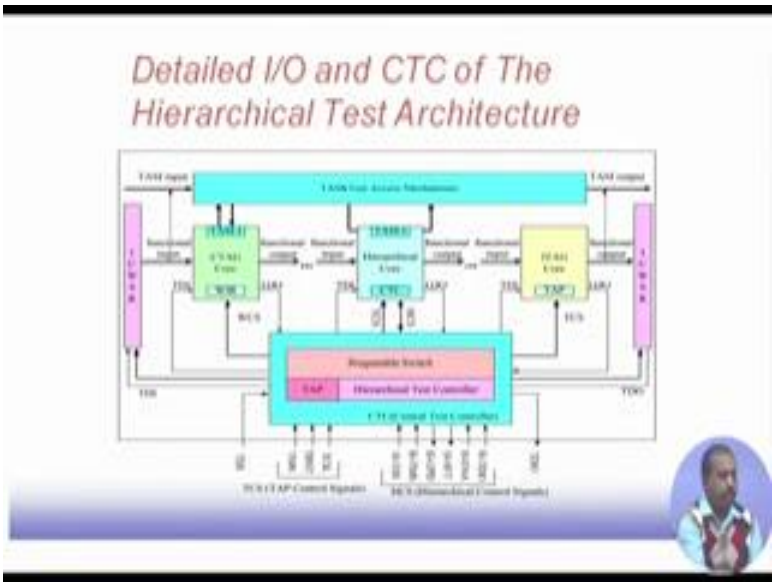


Next one is hierarchical test architecture. So, this is basically, so this is a typical situation. We have got these cores supporting this 1500 series. So, they are called c tag cores, because they are supporting this core interfacing. So, they can be directly connected to this they can be they can have both serial connection as well as parallel connection. So, they can be connected through this TDI line serially or they can be connected with a TAM-BUS. So, I have both of them can be used.

Then this one there may be some courses, which are not which are not compliant to 1500. So, they may be supporting only 1149.1, so that is only the boundary scan standard is supported. So, if only boundary scan is supported, they have to be connected to this serial input part. So, serial testing and all that are to be done. And there may be some cores which are hierarchical cores. So, hierarchical cores means they have got some cores which themselves are assesses. So, they have got their own test architecture embedded and then for those cases it becomes more complex. So, they are so there may be the test patterns available through the serial input as well as this test access mechanisms also may be having these things. So, this way we can have this whole chip consisting of different types of cores. So, different types of cores from the view point of their compatibilities and that way we have got different types of cores and it is like this.

Now, so this is founding my TAM bus. So, this is founding my serial TDI TDO serial interface and that there are some test control lines, so that will be coming to this CTC and there is a hierarchical control line. So, that will also be coming to CTC for the control operation. So, this way if we can support all these modes, all these interfacing standards then we can go for a plug and play type of architecture. So, any new core that we want to put into the system, so if it is compatible with say 1500 may be we will like to test it using TAM; if it is not that, so we may like to test it using a boundary scan. So, like that depending upon the interface supported, so we can go for that. And plus this glue logic. So, they are also to be tested, but the glue logic, they are discrete gates discrete flip flop gate like that, so they do not have any compatibility with this standards. So, they are to be tested in a different fashion. And this system integrator actually has to bother about how to test this glue logic part. So, that may be by means of some extra EXTEST mode and all that.

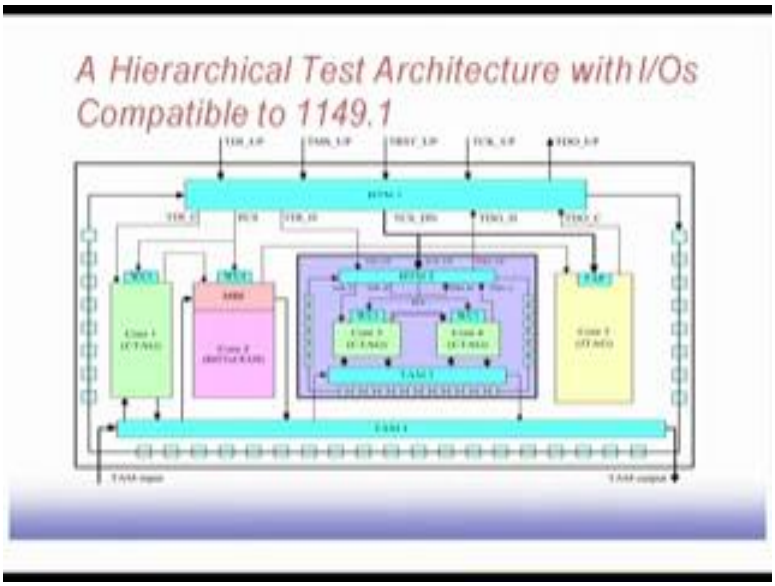
(Refer Slide Time: 19:40)



So, this is the hierarchical test structure that this may be one possible hierarchical test structure. So, we have got this hierarchical core, so hierarchical core it has got its own CTC the control test control point. And it has got these TDI and TDO lines connected. So, this will be this may also connected to this test access mechanism. So, inside it will generate its own hierarchical test information. So, it has to control the individual 1500 or this 1149 interfaces of its own individual cores, so that is not shown here, so that has to be done.

And this programmable switch, so it can be there is a test access port and there is a hierarchical test control, so both will be there. So, this hierarchical inputs, so they will be coming to this hierarchical test controller and this tap control signals, so they will be coming to this tap. So, accordingly this part will be controlling this individual TAMs and individual cores which are supporting this 1500 and 1149; and this has to be done at a more detailed level. So, by breaking the signal, so it has to do its own scheduling, it has to do its own TAM distribution and all those things are to be done, so that depends on what type of cores we have inside this high level core.

(Refer Slide Time: 21:12)

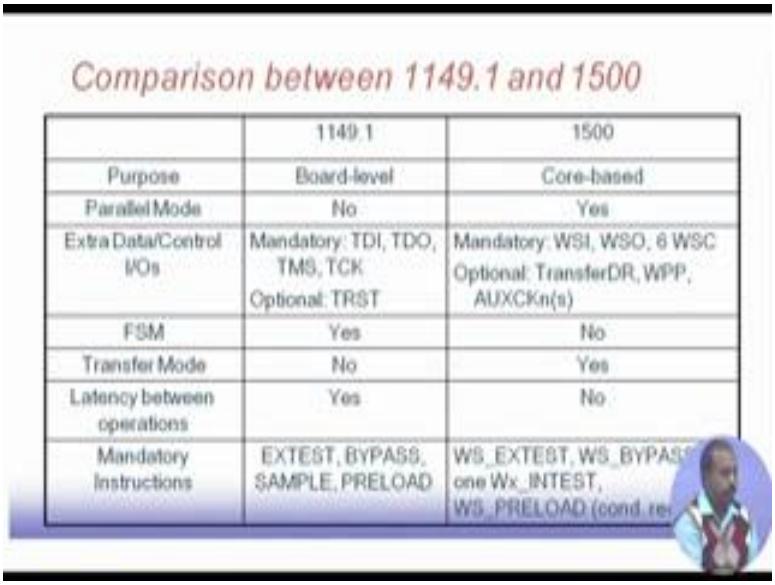


So, if we have got a hierarchical test architecture with IS compatible to 1149 so that means, it does not have this 1500 interface. So, here this is the hierarchical core that we have. Now, this core 3 and core 4, so these 2 cores they do not this core 3, core 4, so they are c tag cores, so they are supporting this hierarchy, so they are 1500 compatible, but this core 5 is j tag compatible, so it is 1149 compatible. So, in that case, we have to we have to this hierarchical test this HTM 1 - top level one, so it has to put some of it in these cores are supporting 1500. So, it has to it can design its own TAM at this point, and it can distribute the signals as it was done here, but for this one it has to give the data through the serial part only. So, this test access code, so this is a serial code. So, it has to get its own TDI line then TDO line like that. So, it has to get it from there.

So, you see that these TDI line, this TAM input line yeah this TAM input line, so it goes to this TAM and then it is put on to this boundary scan also. So, it is coming here then it is going to this TDI for the core test data input for the core. So, it comes here then this is daisy-chained like this and then this line this TDI input goes to this one, so that way this TDS, so this is daisy chained structure and this TDO line of this core. So, it may be it may be it may be put directly on to the TAM or it may be put onto this flip flop chain again to be taken out and going to the TDO, so that way this to the TAM out, so that can be there. So, it is may be coming from here or it may be going from there.

So, the thing is that the cores within a hierarchical core we may have some cores which are supporting 1500. Whereas, there may be some cores which do not support this 1500, they support only j tag. Some of them may be BISTED, some of them may be 1500 compatible, so this way this whole distribution this TAM design has to be done, so that it is taking care of all different types of modes that are there.

(Refer Slide Time: 24:02)



To compare between 1149.1 and 1500, so 1149.1 is a board-level it is targeted to board-level design 1500 is for core-based design, so that is the first difference we have. Secondly, this 1149.1, it supports the parallel mode of operation; whereas, 1500 it does not provide any sorry 1149.1 it does not have any parallel TAM structure, 1500 has got parallel TAM structure. Then this extra data control and I O lines, so they are the mandatory lines at TDI TDO, TMS and TCK, optional line is TRST. Whereas, for 1500 we have got WSI, WSO and 6 W scan control lines. So, they are required and this transfer DR, WPP and auxiliary clocks. So, these are the optional lines.

So, it has got an FSM that can be used for controlling the operation, but 1500 does not have any such control or FSM. Transfer mode is not available in 1149, but it is available here. Latency between operations are permitted in 1149, but it is not so and these are the mandatory instructions EXTEST, bypass, sample preload and these are the instructions that we have for 1500. We will continue in the next class.