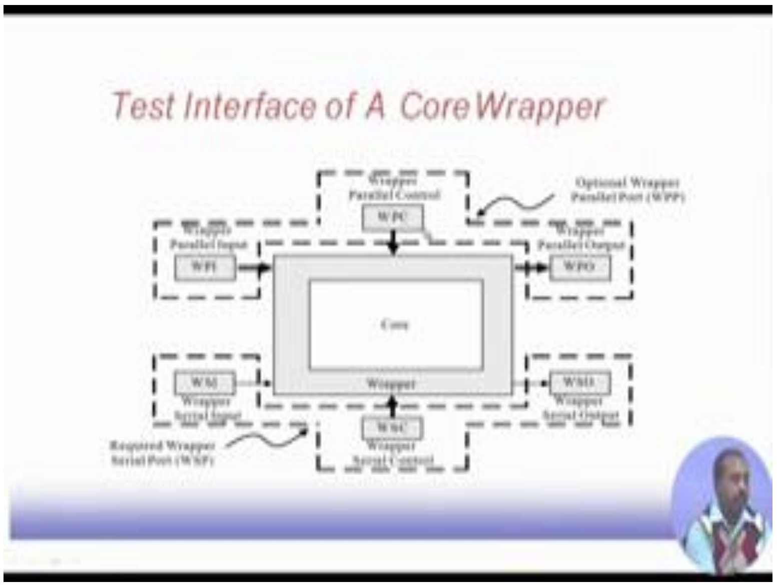
Digital VLSI Testing

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

Lecture - 43
Boundary Scan (Contd.)

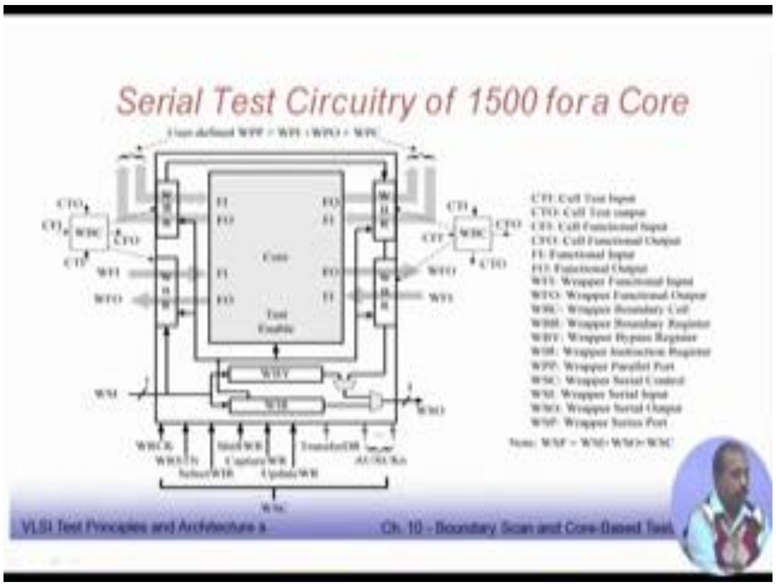
(Refer Slide Time: 00:21)



So, this test interface of a core wrapper, so there are 2 parts you can see that it has got a serial part. So, this is the wrapper serial control, we have got a serial part of the wrapper and there is a parallel part of the wrapper. So, there are unlike 1149 which are only serial interface; 1500 it has got parallel interface as well. Why this augmentation has been done, because for core testing where the SOC testing, we have got we have got large volume of test data. So, if you rely on serial transfer of test data through the chip then it will take enormous amount of time. So, some; so we need to transfer test patterns parallelly. So, test bits have to be transferred parallelly, so that these testing time can be reduced significantly, so that is how this is done, so that is why this is done.

So, we have got this wrapper serial port, so that can be used for transferring some small amount of test data, some configuration information like that. But there is another part which is wrapper parallel interface, so which will have this wrapper parallel input wrapper parallel output this type of registers plus there is a wrapper parallel control; the signal lines will be there in WPI, WPO and WPC. Where this WSI, WSO and these are serial lines, so these you can understand that these are parallel lines, so a large number of a good number of bits can be transported through them. So, if my ATE has got a channel that has got more number of bits, so they can be put into this WPI lines, so that it can transfer a number of bits parallelly for testing.

(Refer Slide Time: 02:09)



Serial Test Circuitry of 1500 for a Core

CTI: Cell Test Input
CTO: Cell Test output
CFI: Cell Functional Input
CFO: Cell Functional Output
FI: Functional Input
FO: Functional Output
WFI: Wrapper Functional Input
WFO: Wrapper Functional Output
WBC: Wrapper Boundary Cell
WBR: Wrapper Boundary Register
WBY: Wrapper Bypass Register
WIR: Wrapper Instruction Register
WPP: Wrapper Parallel Port
WSC: Wrapper Serial Control
WSI: Wrapper Serial Input
WSO: Wrapper Serial Output
WSP: Wrapper Series Port

Note: WSP = WSI+WSO+WSC

VLSI Test Principles and Architectures

Ch. 10 - Boundary Scan and Core-Based Test

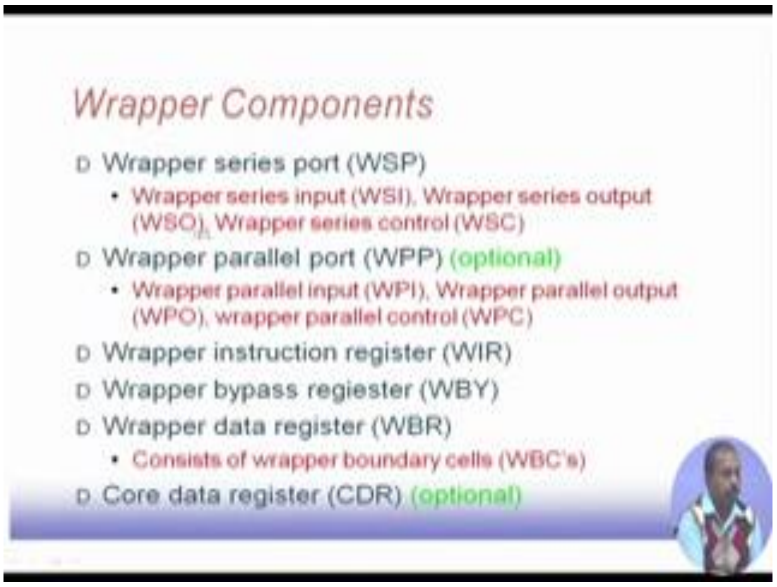
So, serial test circuitry of this 1500, so you see that we have got this wrapper buffer register, so that is some similar to the boundary scan register - BSR cell, so that is same as that. So, a number of buffer registers, so it will have these type of WBC which is wrapper boundary cell, a number of such boundary cells will be there that will be constituting one register, so that can feed this functional input that can negate result from this functional output. Now, this each WBC, so it has got this CTI which is cell test input then CTO - cell test output, CFI - cell functional input and CFO - cell functional output. So, when this circuit is put into normal mode of operation then this CFI line will be going through to CFO, and that will be ultimately applied to the functional input tins of the core.

Now, similarly these CFO lines, so they can be collected into the through this functional input line and that will be available here. So, that can be transferred through other points, so that destination cells; it can be transferred. And if you are trying to do a serial shifting of test data, then it can be done like cell through this CTI. So, it can be transferred serially through this, through this cell test input line. So, it can be transferred serially. So, this WBR part is similar to this boundary scan register excepting the fact that ok, this has got this parallel input also from this functional input it can come there.

Similarly, we have got this bypass register which is same as this boundary scan bypass register that we had. So, here also if this boundary register bypass is if this wrapper bypass register is activated then this serial input, so it will go to this WBY bypass and through this bypass, so it can be selected through this multiplexer, so that it is going to this WSO. The WSI in one cycle it can go to WSO, so that is the bypass. Then similarly we have got this instruction register similar to this WBI, similar to these lines of boundary scan circuit, so boundary scan register. So, this WIR is the wrapper instruction register.

Then we have got this wrapper scan control, wrapper serial control port that has got a number of inputs like this WRCK. So, this is basically the wrapper clock then this is reset. So, you can do a reset or you can have this select WIR, so you can the instruction register you can select or shift WR. So, these are the various operations shift, capture, update, transfer, so these are the various inputs that can be that these are the different commands select WIR and shift WR, capture WR, update WR and transfer DR, so these are the various commands that are given to the serial control. And there are some auxiliary clock inputs, so that can also be used in some cases. So, if some extra clock signals are necessary, so that can be through this auxiliary input.

(Refer Slide Time: 05:54)



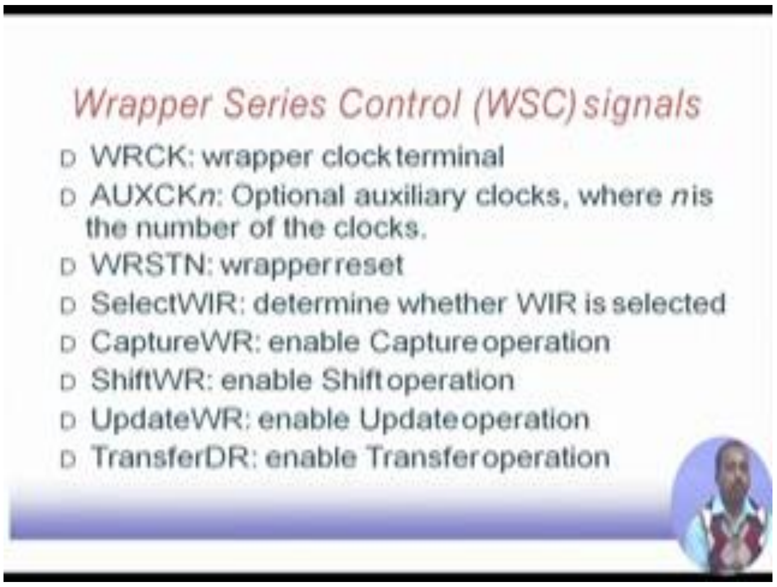
Wrapper Components

- Wrapper series port (WSP)
  - Wrapper series input (WSI), Wrapper series output (WSO), Wrapper series control (WSC)
- Wrapper parallel port (WPP) (optional)
  - Wrapper parallel input (WPI), Wrapper parallel output (WPO), wrapper parallel control (WPC)
- Wrapper instruction register (WIR)
- Wrapper bypass regiester (WBY)
- Wrapper data register (WBR)
  - Consists of wrapper boundary cells (WBC's)
- Core data register (CDR) (optional)

So, wrapper components, first we have got the wrapper series ports - WSP. So, wrapper series wrapper series port is a WSP, it has got WSI, WSO and wrapper series control lines. So, this is the WSI, WSO and wrapper series control lines. So, all this part forms the series port. Now, there are wrapper parallel ports, this is optional. So, as I was telling that the serial port is must, but this parallel port is optional. So, in the parallel port, if you have it then you have got wrapper parallel input, wrapper parallel output and wrapper parallel control, so these are the things that are there. So, this is the thing that is wrapper parallel process WPI, WPO and WPC. So, they will be actually coming to this part. They will also be transferred through this boundary cell, so that this can be applied parallel to the functional input or functional outputs can be collected and all, so that is the parallel interface.

Now, there is wrapper instruction register WIR; we have got wrapper bypass register – WBY, and wrapper data register -WBR. So, this wrapper data register, so it has got a boundary scan cells WBCs. So, these WBCs are there; in each WBR, so there will be a number of WBCs, so that will be defining the wrapper register. And there is a core data register, which is optional well like core may be of producing some data that may need to be transported for testing purpose, so that is the core data register or it may be for configuring the core in certain mode, we have to give some command to the core. So, that way there may be another on core data register which may be programmed, so that is in some cases some application it will have, so and some other cases it will not have.

(Refer Slide Time: 07:50)



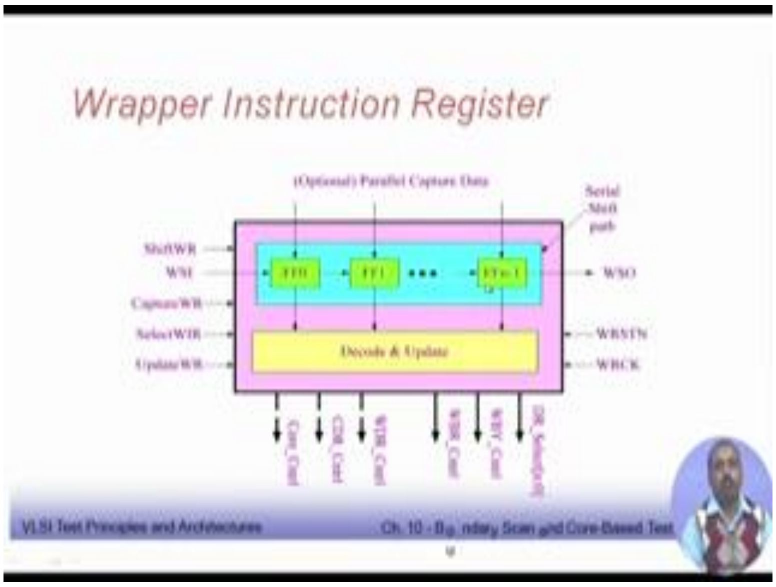
Wrapper Series Control (WSC) signals

- WRCK: wrapper clock terminal
- AUXCK*n*: Optional auxiliary clocks, where *n* is the number of the clocks.
- WRSTN: wrapper reset
- SelectWIR: determine whether WIR is selected
- CaptureWR: enable Capture operation
- ShiftWR: enable Shift operation
- UpdateWR: enable Update operation
- TransferDR: enable Transfer operation

Then this wrapper series control signal that is WSC, it has got a number of signals in it; WRCK is the wrapper clock as the name suggest then there are a number of auxiliary clocks. So, these are auxiliary clocks. So, it may be necessary for operation of certain shifting, so that is there. So, normally they are not used, but this provision has been kept, so that if some auxiliary clock is needed for some operation. So, it can be taken care off.

Then WRSTN is the wrapper reset. Then select WIR, so it determines whether WIR is selected or not. So, if an instruction is for data part or for the control part, so that is actually the select WIR. So, if this is selected then the data that is coming through the TDI line will go to the WIR instead of instead of going to the WBR. Then this capture WR is enable capture operation; shift WR is enable shift operation; update WR is enable update operation, so the boundary scan cell structure and the wrapper scan cell structure, so they are similar. So, all the modes that we have in boundary scan cells. So, we will be having here as well. So, all this 5 operations are possible.

(Refer Slide Time: 09:14)



Wrapper Instruction Register

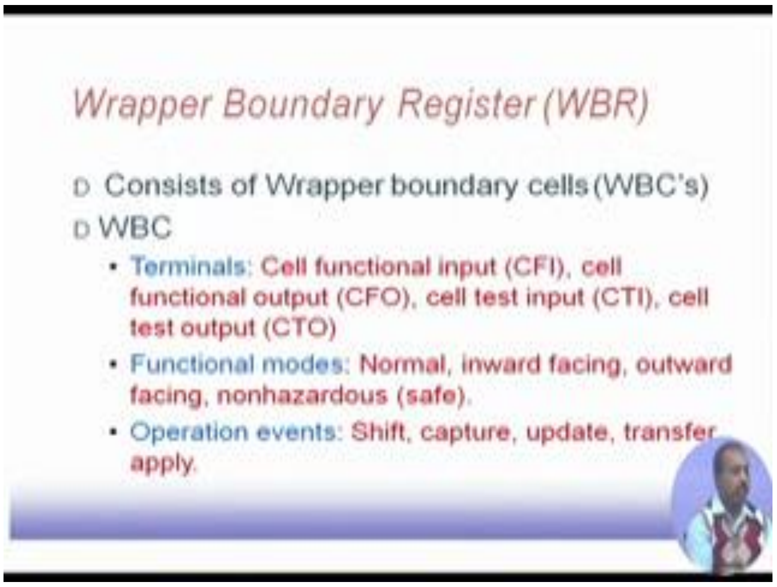
VLSI Test Principles and Architectures

Ch. 10 - Boundary Scan and Core-Based Test

Then these wrapper instructions register, so it typically has got a structure like this. So, we have got this flip flops, so depending upon your instruction with, so we can have a number of flip flops there, and they are consisting of the say flip flop zero to flip flop n minus 1. Then this wrapper serial input that comes as a input here, and it goes as wrapper serial output from the other side. Now, this parallel, so these wrapper instructions register, so it can be loaded parallelly as well, this parallel captured data, so it may be coming from parallel port so, but or some other source, but this provision is there, but it is optional. So, may be from the parallel port, we want to load this instruction register directly with some instruction, so that can be done.

So, the comments that are the other signal inputs that are coming to shift WR. So, these are actually the instructions the shift WR, then capture WR, select WIR, update WR, so these are the instructions that are coming. And then we have got this signal WRSTN, WRCK, so these are the clock, this is the reset signal. And we have got this decode and update, so all these inputs, so they are decoded and accordingly the values are updated. So, this DR select, so this is basically which register we are going to update. So, data register select, so 0 to n if there are n registers, so which register you are going to update also. So, W bypass control, so bypass control is if this is 1 that means, it will be wrapper will be in bypass mode then this BR control, so this whether we are going to do this bypass register control. Then there is a data register control that the cell this core data register control and core control, so these are various controls are there this may be used for some cases. So, whole documentation you can get in the 1500 manual, so that will give a more detailed idea.

(Refer Slide Time: 11:23)



Wrapper Boundary Register (WBR)

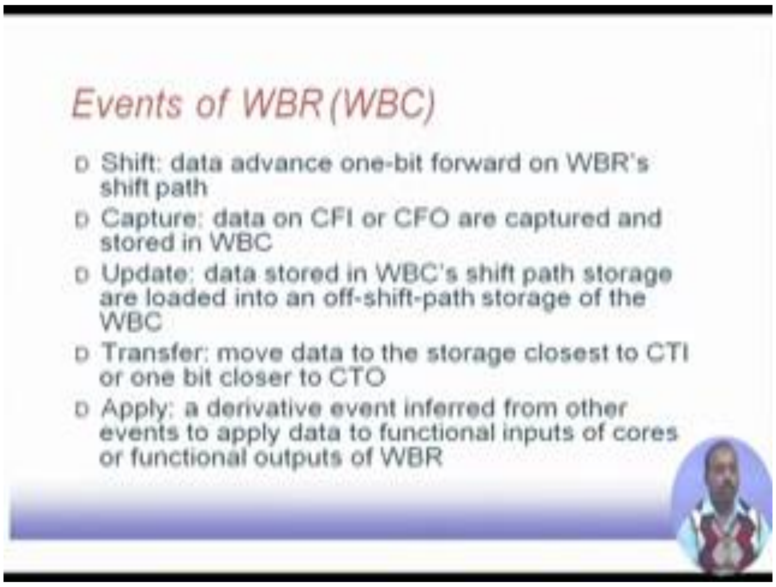
- Consists of Wrapper boundary cells (WBC's)
- WBC
  - Terminals: Cell functional input (CFI), cell functional output (CFO), cell test input (CTI), cell test output (CTO)
  - Functional modes: Normal, inward facing, outward facing, nonhazardous (safe).
  - Operation events: Shift, capture, update, transfer apply.

Then this wrapper boundary register, so if we look in more detail, so it consist of wrapper boundary cells, actually some structure earlier we have seen this thing. So, it has got, yes. So, this wrapper boundary cell it has got these CTO, CFI, CTI and CFO. So, let us see, what are those things? So, CFI is the cell functional input; CFO is the cell functional output, cell test input and cell test output. So, functional inputs, so they will be coming from the when the circuit is in normal mode those mode will be coming. So, this as you can see that there are several functional modes normal, inward facing, outward facing and nonhazardous.

So, normal means whatever is coming as whatever is coming on the functional input lines, so CFI lines, so we will pass it on to CFO line, so that is the functional part. Then we have got inward facing. So, there is a difference inward facing and outward facing what can happen is that when we are testing a particular core the pattern that the test bit that is there in the this wrapper scan cell, this wrapper boundary cell, so that is actually going to be applied to the core. So, that way the core should be directed inward that particular cell is directed inward as if its content will be put inside the core for testing. On the other hand, if you have to say testing say an interconnect between 2 ports then the content of a particular cell, so this will be actually is a outward because that will be taken out of the core and applied to the input of another core, so that is the outward facing. So, we have got this functional mode, so it has inward facing and outward facing. And nonhazardous means it is a safe mode. So, this may be put into this mode, so that by mistake the WBC does not get configured to some other mode.

So, the operation events that are possible, shift, capture, update, transfer and apply. So, shift is naturally the shift operation, shift the content by one cell position. Capture is capturing the value, after the circuit has operated functional mode for one cycle. So, the value is captured onto the scan set, so that is the capture operation. Then update is transferring this content of R 1 register to R 2 register. And then we have got transfer, so transfer will be may be transferring on the parallel line or the serial line. And apply, so may be some instruction has been loaded onto WIR and then it is given the apply operation, so that operation will be done on the scan cell content.

(Refer Slide Time: 14:19)

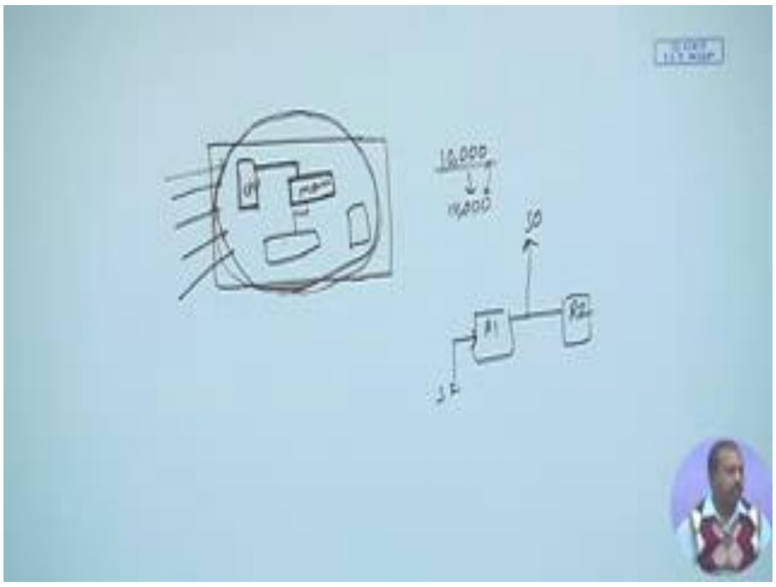


Events of WBR (WBC)

- Shift: data advance one-bit forward on WBR's shift path
- Capture: data on CFI or CFO are captured and stored in WBC
- Update: data stored in WBC's shift path storage are loaded into an off-shift-path storage of the WBC
- Transfer: move data to the storage closest to CTI or one bit closer to CTO
- Apply: a derivative event inferred from other events to apply data to functional inputs of cores or functional outputs of WBR

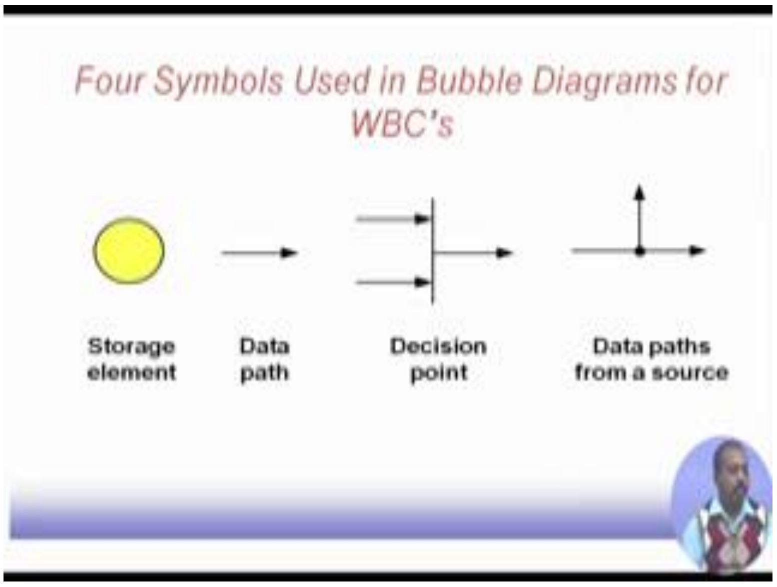
So, shift is data advance one-bit forward on WBRs shift path. So, as I was telling that WBR is consisting of WBC's, so this through this SI SO line. So, it just shifts to the next position. Capture, data on CFI or CFO are captured and stored in the WBC. So, this cell functional input or cell functional output depending upon whether it is capturing an input or capturing an output from the for the core. So, it will be CFI or CFO, so that will be captured into the WBC. Update, so it says that the data stored in WBCs shift path storage are loaded onto an off shift path storage in the WBC. So, we have seen previously that in these structures, so there is one R 1 register and this R 1 register, so this S I input can come here and this S O input goes from there.

(Refer Slide Time: 15:22)



And it also there is one R 2 register to which the content may be stored. So, this is basically that update operation. So, data stored in WBC's shift path, so this shift path consists of R 1 registers only. So, from this R 1 registers, so it is loaded onto R 2 which is not on the shift path, off-shift-path storage of the WBC. Then transfer, so it says that the move data to the storage closest to CTI or one bit closer to CTO. So, from CTI, so it will go to the next input or from the CTO, so it will go to the one bit toward the CTO, so it will go by one cycle. So, the cell test input and cell test output, so they advance by one position. And apply is a derivative event inferred from other events to apply data or functional inputs of cores or functional outputs of WBR. So, as I was telling some instruction, we have to load on to IR and then we have to tell apply, so apply will do that operation. So, this is basically a combination of number of operations.

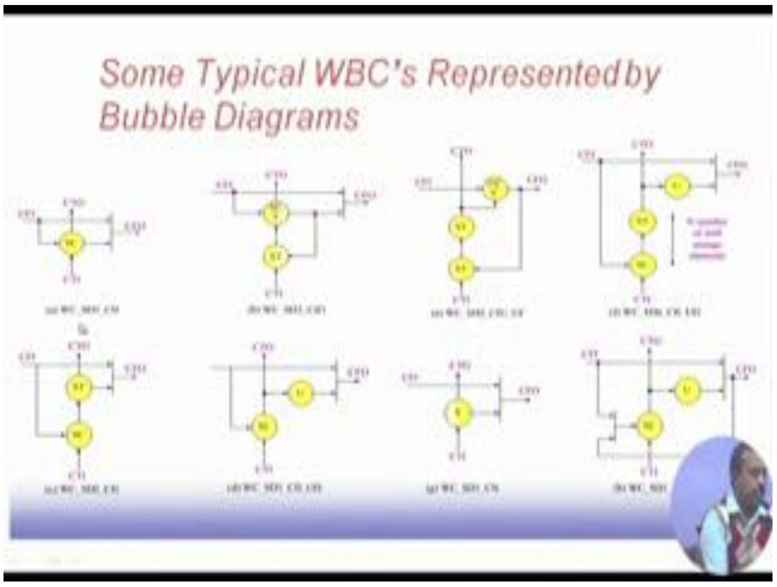
(Refer Slide Time: 16:38)



Four Symbols Used in Bubble Diagrams for WBC's

Now, in case of this WBC, the operations are often specified by means of some symbols. So, they some diagrams, so they are known as bubble diagrams. So, in a bubble diagrams, so we have got this sort of structure. So, storage element then this is a data path then this is a decision point, so 2 data paths, so they will come here, and based on some select line, so either of the 2 data paths they will be passing to the output. So, that is the decision point. Then we have got data path data paths from a source. So, at this point, so it is 2 data paths are emerging from this particular point, so that is a source point. So, whatever different type of instructions that we have looked into, so they can be modeled by means of this bubble diagrams, and they can be used to specify, they can be used for specifying the operation mode clearly.

(Refer Slide Time: 17:36)

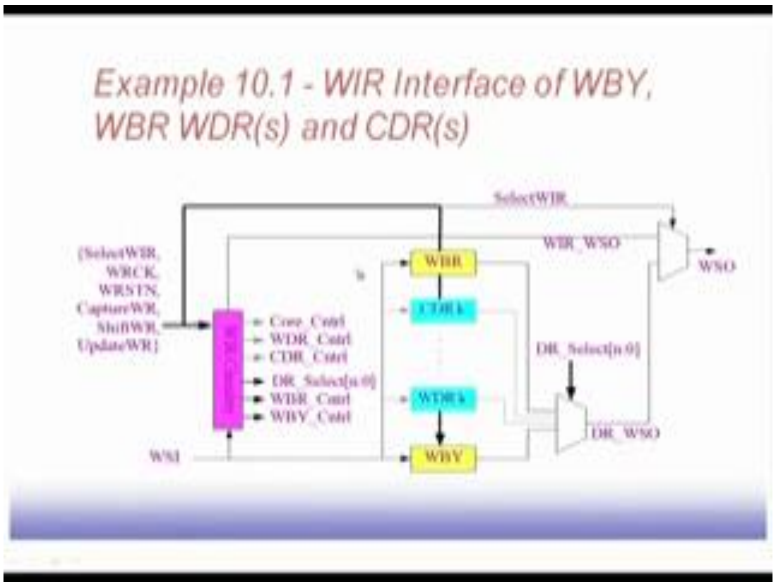


Some Typical WBC's Represented by Bubble Diagrams

Like say this one. So, it says that shift data one to CII. So, this is s and d, I think there is some description. So, what it says is that, so here this can do this bubble, so it is basically a storage element and the storage element has got points S and C. So, S stands for shift, C stands for capture. So, this cell can do 2 operation can be a shift operation or may be a capture operation. So, say since it is coming from CTO, from say from CTI it can be, so it can do a scan operation. So, CTI to it can send to CTO or it can do a capture operation. From CFI, so it can come to this cell it can be combined here, and then the finally, the normal mode, so this CFI will go to CFO and this in this capture mode, so this content may be available at CFO. So, as a result, there is a decision point at this point, so this finally, realizes this particular operation where this cell is a this can do scan or this can do capture.

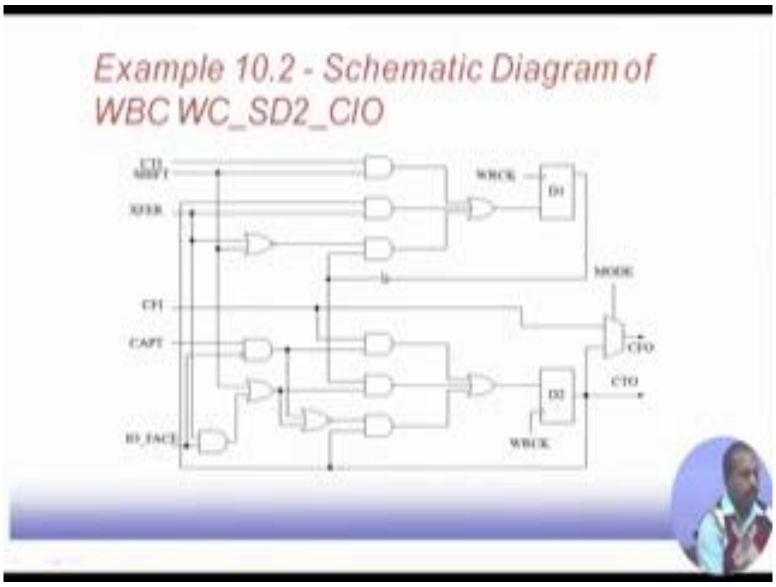
Then this one, so it can do scan, capture and transfer. So, this cell can do scan, capture and transfer. So, this cell can do this flip flop storage can do scan and transfer. So, from CTI, so this scan, transfer, so it can scan input, it can do the scan operation, this can also do the scan operation, and finally it may be connected to the CTO. So, this may be one type of operation for this particular instruction. Then similarly functional input can go there. Then this functional input it can be captured here and then it can be transferred. So, it can be transferred to this CFO or it can be transferred to this next level of elements. So, we have you can see that this is some sort of that R 1 register and this is some sort of that R 2 register by which this operation is taking place. So, in this way, all these cases, so they correspond to some instruction being executed by the WBC. So, the features of this individual flip flops and how they can be utilized for realizing the functions. So, these are actually the comments of this 1500, and how these comments are actually implemented, so that is there.

(Refer Slide Time: 20:20)


Example 10.1 - WIR Interface of WBY, WBR WDR(s) and CDR(s)

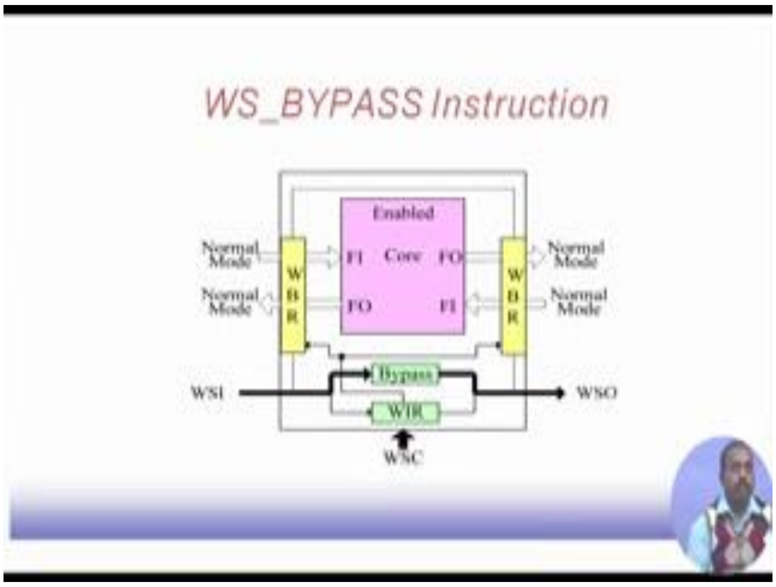
Now, a typical WIR interface may be like this that we have got this WIR circuitry from this wrapper serial input. So, this WIR will be loaded and this WIR will produce all these controls core control, data register control, core data register control then this data register selection which data register we are talking about, so then this WBR control and WBY bypass control. So, all these inputs are coming. So, based on that this WIR circuitry, so it is going to activate some of them; so if these parts are activated, so that is suppose this DR select, WBR this WBR control and W bypass control, so these are activated then what happens is that, so this part will be getting the data. So, whatever is coming, so these controls are activated, but this CDR cell not activated. So, the value that is coming from this WBR, so that is coming to this multiplexer. And now this DR select which WBR we want to send, so that WBR s content will be available in this WSO line and that finally, that will be going to the WSO line. So, this way we can control this instruction register, accordingly this control signals will be controlling this remaining registers, and the content of the particular register will be going to the WSO line.

(Refer Slide Time: 21:56)


Example 10.2 - Schematic Diagram of WBC WC_SD2_CIO

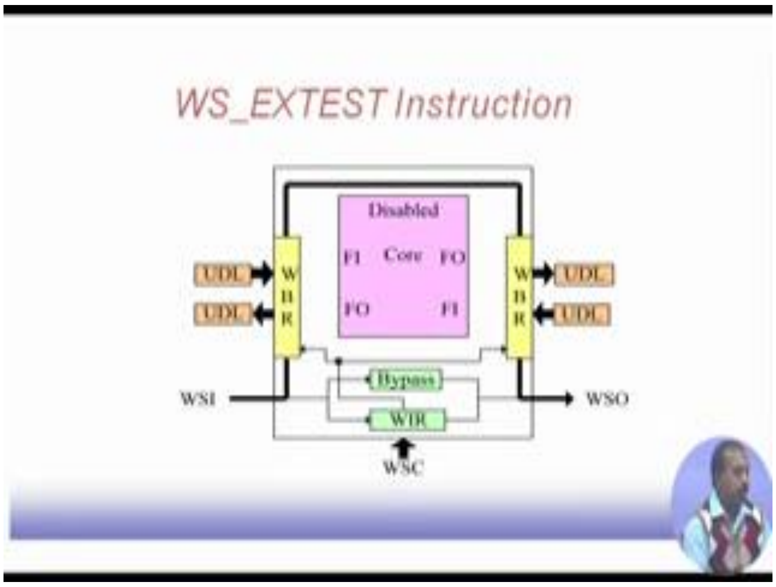
So, this actually realizes the same operation in a schematic diagram, the circuit diagram of a.

(Refer Slide Time: 22:04)


WS_BYPASS Instruction

Now, let us look into some of the instructions like say bypass instruction. So, bypass instruction, how is it to be implemented. So, WBR they will be operating in normal mode. So, the circuit operates in normal mode, but this scan control. So, wrapper serial control signal, so it can tell the WIR that this is a bypass, so this control the bypass control will be high, as a result this will be decoded as a bypass instruction. And whatever is coming on this WSI line will be bypassed and it will be coming to this WSO line. So, this way this bypass instruction can be implemented.

(Refer Slide Time: 23:46)

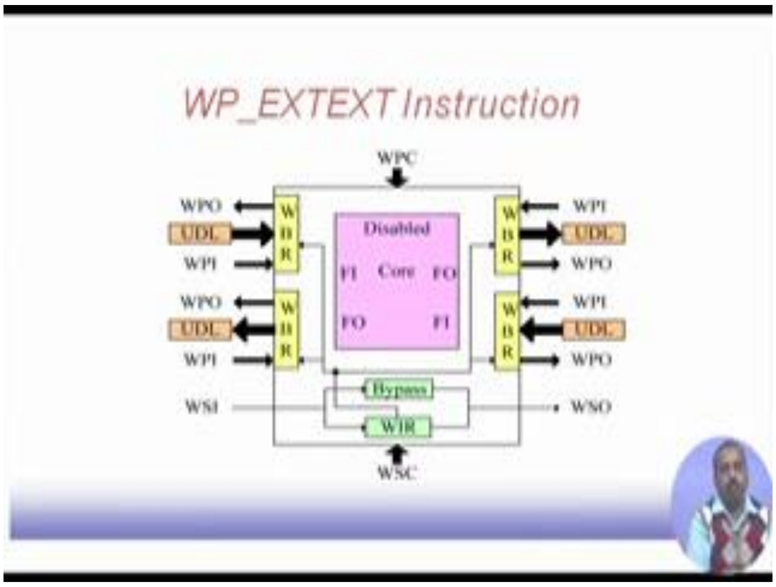


WS_EXTEST Instruction

Then this EXEXT operation, so EXTEXT operation means we are trying to test the external logic. So, here in this particular example, what has been done is that we are testing the additional user defined logic - UDL for by applying some pattern here. So, this you see that UDL that we have shown here this may be this is the part of the full chip, but it is not a core. So, it has not got any functional any other input pin output pin that can be connected to this parallel interface or serial interface like that, so that that is not there. But it is a it is a small piece of circuit, which is close to some chip some core, it is a small piece of circuit which is close to some core. So, what we do through this WSI line, so we send the test patterns that can be used for testing this UDL.

Now, normally this WSI whatever we shift in, so that is going to be a test pattern for the core. But in this case, so this is put in the EXTEXT mode, so this core operation is disabled, so it is not getting to be effected by the changes in the WBR register, but this can be used for testing this one. Similarly, so if I have got say so many UDL's like this may be these 2 constitute one UDL, these 2 constitute another UDL, so from one bypass register, we can feed the input to one UDL and we can get the output of that UDL onto this buffer onto this boundary scan register. Similarly, for this UDL, so we can put this input test input from the boundary scan register and get the response from the boundary scan register, so that will be going out through this WSO line. So, this WSC, where it is putting in the EXTEXT mode, the core is porting to a disabled mode and the operations are being done.

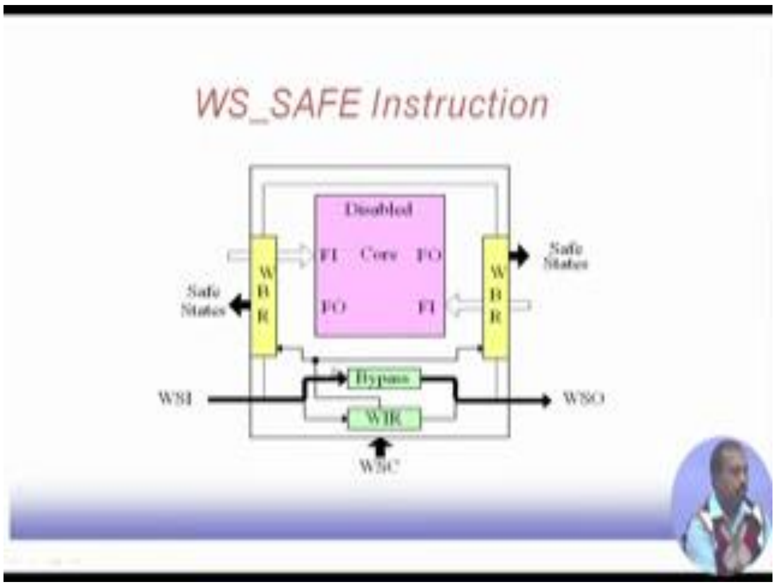
(Refer Slide Time: 24:51)



WP_EXTEXT Instruction

Then we can have this WPW parallel EXTEXT instruction. So, so previously it was WS. So, when it was WS means it was done serially; now when you say WP, so it is a parallel operation. Now, whatever test pattern will be loaded, so that will be loaded parallelly so through this WPI line, so this UDL got loaded the test pattern for this UDL got loaded, similarly for from some other bits of the WPI line, so this boundary scan register got loaded, and then the patterns may be applied to this one. So, then this WPO lines will be collecting the responses and through this WPO line, so we can take out the rest ones. So, here this scan control, it has been told that it will be the parallel operation. So, this instruction register is loaded accordingly that it is a parallel operation. So, when it is a parallel operation, so this serial part will be disabled and this whole thing will go parallelly. So, all these UDLs will be tested parallel by loading the test pattern through this WPI line and getting the responses through the WPO lines.

(Refer Slide Time: 26:11)


WS_SAFE Instruction

Then this is that safe mode that we are talking about. So, in this case we are not very sure about what is going to happen in the core that is why, so we do not want to put any input to the core. So, we just put some value here, so that it is a safe state. So, this bypass is activated, because whatever is coming that has to be passed, so that is the bypass is activated, and we have got this WBRs configured into safe state. So, there may be some safe values for the design may be may be for if I do not want that this core should be active. So, may be that it is powered down or it is clock dated or it may be in particular the clock dated sort of thing. Then these WBRs they may be loaded with some patterns, so that your leakage reduction, leakage power consumption is also low. So, it may be like that. So, that way it is that may be a safe state, so that that depends on many other application process that whatever it is ultimately it means that this is not going to be affected, the core is not going to be affected if it is put into this particular values at the input. So, those are the safe states.