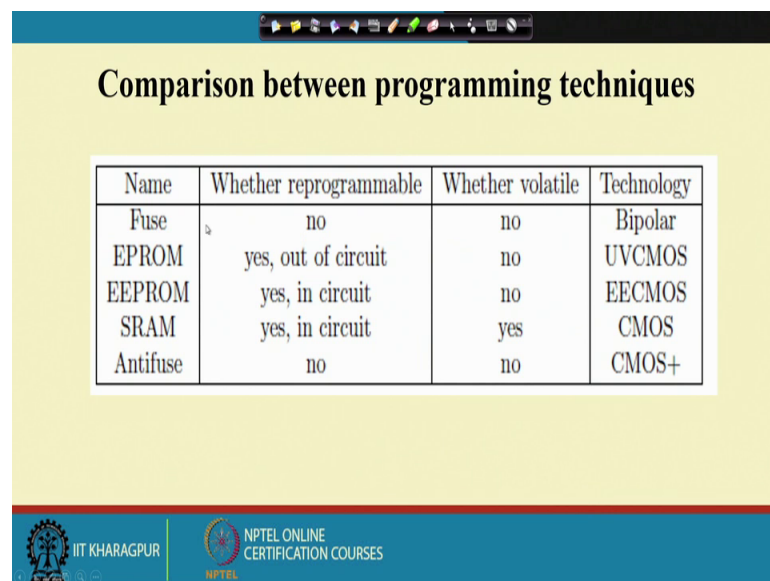


Digital Circuits
Prof. Santanu Chattopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture – 45
FPGA (Contd.)

So if we compare between these programming techniques. So, this the programming technology, if it is a fuse based programming.

(Refer Slide Time: 00:22)



Name	Whether reprogrammable	Whether volatile	Technology
Fuse	no	no	Bipolar
EPROM	yes, out of circuit	no	UVC MOS
EEPROM	yes, in circuit	no	EECMOS
SRAM	yes, in circuit	yes	CMOS
Antifuse	no	no	CMOS+

Then it is not reprogrammable because once the fuse is blown so, we cannot restore it. So, it is not volatile like even if you switch off the power this blown switch will not come back to its original value. So, that is not programmable, not volatile and a technology that uses is the bipolar technology.

Normally, it is a TTL type of technology Transistor Transistor Logic type of technology that is used in the fuse; then EPROM. So, EPROM it is also programmable definitely, but you have to take it out of the circuit, because for reprogramming. So, we have to take it out of the circuit and expose it to ultraviolet light and then only the content will be reset and then you can program it again.

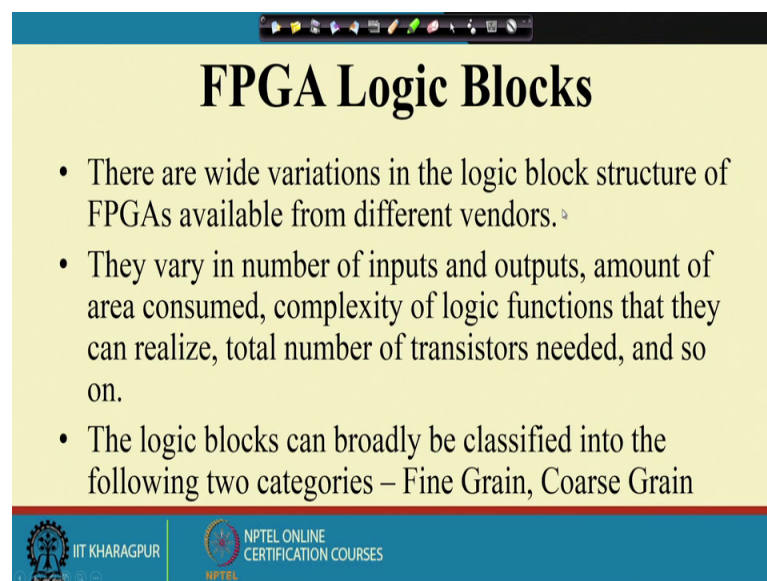
So, that way it is EPROM is reprogrammable, but it is out of circuit, it is not volatile because, the content will never be lost. The technology that uses is UVC MOS ultraviolet

CMOS technology then we have got EEPROM technology for a programming technique. And this is programmable and it is in circuit programmable, because you can just program this we can apply a suitable voltage. And then the then the PROM content will be modified.

So, it is also not volatile and the technology is electrically erasable CMOS technology then this SRAM. So, this is also reprogrammable and it is in circuit reprogrammable. So, that is fine it is volatile so if you switch off the content will be lost. So, the FPGA chip has to be programmed again and the technology it uses is the CMOS and this anti fuse. So, anti-fuse it is not programmable it is not volatile and it uses technology it is CMOS plus a CMOS plus that, we have said that a silicon oxide nitride type of connections.

So, this is some advanced masking will be required over and above this normal CMOS fabrication. So, this is the summary of comparison between these programming techniques. Next, we will be looking into this FPGA logic blocks.

(Refer Slide Time: 02:29)



FPGA Logic Blocks

- There are wide variations in the logic block structure of FPGAs available from different vendors.
- They vary in number of inputs and outputs, amount of area consumed, complexity of logic functions that they can realize, total number of transistors needed, and so on.
- The logic blocks can broadly be classified into the following two categories – Fine Grain, Coarse Grain

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, logic blocks are the heart of this FPGA chip. So, they are the they are actually the modules which will be implementing the logic function that we are going to have. So, there are wide variations in the logic block structure of FPGAs available from different vendors. And here actually there are you can find the different FPGA vendors, they are come up with different type of logic block structures.

And they vary widely they vary the number of inputs and outputs amount of area consumed complexity of the logic functions that, they can realize, total number of transistors needed etcetera. So, that way there is the wide amount of variation.

The logic blocks can broadly be classified into 2 categories 1 is called fine grain logic block, another is called coarse grain logic block. So, as the name suggests so, you can understand that when we have got very fine very simple logic block consisting of less number of logic gates in them. So, they will give rise to this fine grain logic block and when we have got this complex far logic block. So, having individual logic block having more number of functionality implemented within it so, they will be called coarse grained logic block.

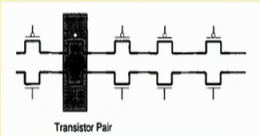
Now, we will see that trade-off between them and which one is superior. So, it is difficult to answer so, each one have got their own advantage and disadvantage so, but we will. So, the choice is depends on many things like first of all the availability, then the familiarity of the designer right, if I am familiar with the Xilinx FPGA is more. So, maybe in my future design I will be using Xilinx FPGAs only.

So, that way it is like that. So, choice is more determined by availability, familiarity and in some cases, the price also.

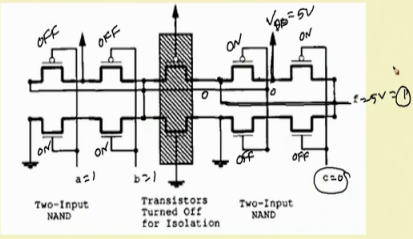
(Refer Slide Time: 04:25)

Fine Grain Logic Block

- The block contains a few transistors that can be interconnected via programming.
- *Crosspoint* FPGA uses a single transistor pair for each Boolean variable in the logic block.





Transistor Pair




Two-Input NAND Transistors Turned Off for Isolation Two-Input NAND

$f = (ab) + c'$

 IIT KHARAGPUR

 NPTEL ONLINE
CERTIFICATION COURSES



So, first we look into this fine grain logic blocks so, this is a logic block from cross point it. So, this logic block contains a few transistors that can be interconnected via programming.

So, for example so this, cross point FPGA so, it uses a single transistor pair for each Boolean variable in the logic block. So, this is the thing like so, you have got a type transistor and an N type transistor. So, they that is used for 1 Boolean variable, you have got a chain of such transistors and you can establish these connections. So, you can you can do metallization, so that this programming so, that these gate lines so, they can be connected they can be connected to some inputs.

Similarly, these drain and source lines they can be connected to some they can be connected in some fashion. So, that can happen so, here is a small example suppose, we want to realize the function f equal to $a b$ plus c bar. So, you see so, from this function you can understand. So, if c is equal to 0 then the functional the function must output a 1. So, if you look into this design if you see if c if I put c equal to 0 then what will happen?

So, this transistor will be off so, this transistor will be off because this is N type transistor and giving 0 to the gate. So, this will be off and this upper transistor will be on because this is a P type transistor and I am giving a 0 to the gate. So, this transistor will be on so, this up arrow means. So, it is connected to the supply voltage V_{CC} equal to 5 volts. So, we normally call it V_{DD} in case of CMOS.

We in the MOS circuit, so you called it is drain voltage V_{DD} . So, that is maybe equal to 5 volt so, what will happen? So, this is this point is at 5 volt and this transistor is on. So, you will get f equal to 5 volt. So, that is equal to logic one. So, you see if I put c equal to 0 I am getting f equal to 1. Now, this a b part. So, a b part is realized by the first this section.

So, this realizes the product term a b how? Suppose, a equal to 1? Suppose a equal to 1 and b equal to 1. So, both of them are equal to 1 then what will happen? Since a equal to 1 so, this transistor is on this is off because this is N type transistor and if I give this gate voltage equal to 1. So, the transistor will be on. Similarly, this b being equal to on, this is b being equal to 1. So, this transistor is on this transistor is off fine.

So, what will happen? So, this point is grounded. So, you see that this ground potentially ground voltage is available here and this ground through this line. So, it will be coming so this line is made equal to 0 and you get a 0 voltage here. So, in this 0 comes so as a result so, this transistor this is a p transistor with gate value 0.

So, this is turned on and this is an n transistor with gate value 0. So, this is turned off. So, you see that this transistor being on. So, this voltage this VDD that we have is available at this point. Because, this transistor is also this drain and source will be at the same potential.

Since, the drain is at 5 volt, so, this source will also become at 5 volt. So, as a result this line will be become becoming equal to 1 again. So, on the other hand if I have this say a equal to say 0.

(Refer Slide Time: 08:19)

Fine Grain Logic Block

- The block contains a few transistors that can be interconnected via programming.
- *Crosspoint* FPGA uses a single transistor pair for each Boolean variable in the logic block.

Transistor Pair

Two-Input NAND

Transistors Turned Off for Isolation

Two-Input NAND

$f = ab + c'$

$a = 0$
 $b = 1$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, a equal to 0 and b equal to say. So, a is 0 b is 1 so as a result, this transistor is on this is off this is off and then I have got this transistor as off this transistor is getting on so, this is off and this transistor lower transistor N transistor so this is on.

Now since, this transistor is on so you get this one value coming here. So, these so this is also connected to this is actually whenever, we have got this up arrow means it is connected to VDD so it is connected to VDD.

So, this is these 2 points potential are same so I get a 5 volt here, and that 5 volt get transferred to this point. So, I when I get 5 volt here so this transistor will be on and this transistor will be off as a result this ground value 0 will be coming to this point and that will be so the ground value will be 0 and this 0 will be communicated to this point this f.

So, in this fashion so we can have different input combination as a result, this individual stages of this transistor. So, they will be is propagating the signal or they will not be propagating the signal. In between, we have got this isolation stage.

So, this isolation stage so this will be this is for separating the 2 product transfer. So, this a b and c bar they are separated by this isolation states. So, this is this you see that this is an N type transistor and I have connected it is gate to ground, the P type transistor I have connected is gate to high. So, as a result both the transistors are off.

So, this actually establish an isolation between this ab term and the c term and. So, the logic is transferred from one side to another side by means of these interconnects that we have between these transistors arrays. But, otherwise the transistors are not connected to each other by means of the isolation trans isolation stage so, that is ensured.

So, this way we can have this cross point FPGA. So, they realize this logic functions and you see the logic block is very simple. So, you have got only 2 transistors per Boolean variable ok. So, next we will be looking into a more.

(Refer Slide Time: 10:54)

Coarse Grain Block - XC4000 from Xilinx

5-input
Any 4-input
 $2^4 = 16 = 0.5536$

$f(x_1, x_2, x_3, x_4, x_5) = x_1 \cdot x_2 \cdot x_3 + x_1 \cdot x_4$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Complex ahlogic block structure which is from Xilinx, which is Xilinx has got a family of FGPS.

So, this is one of the initial families I should say. So, today if you look into the advanced families of Xilinx. So, they are very complex we will see some of them. So, this is the XC4000 series from Xilinx, Xilinx FPGA. So, what it has this is a coarse grain block because, you will see that the logic function that that you can get is quite complex. So, compared to only 2 transistors, that we had in cross point FPGA so here, we have got this type of lookup table. So, lookup table means so this has got 4 inputs G1 G2 G3 G4.

So, it can realize any of any 4 variable function. So, this lookup table can realize any 4 input functions, any 4 input function. Similarly, this lookup table also can realize any 4 input function. So, this so I can so if I if I look took take them separately then I can realize 2 4 input functions directly from the lookup table.

So, number of 4 input Boolean functions that you can have is 2^4 that is 2^4 power 16. So, it is 16.

So, this is the total number of 4 variable functions that we have that is the total number of 4 variable functions that are possible. So, you see this is a huge number of functions that you can realize using this for input look up tables plus there is another lookup table after this.

So, there the output of these 2 look up tables are fed and also this from this C1 the C1 C2 C3 C4 through this selector, you can select one of these inputs to come to this lookup table. So, if I take these 3 look up tables together if I take these 3 lookup tables together then I can realize any 5 input Boolean function.

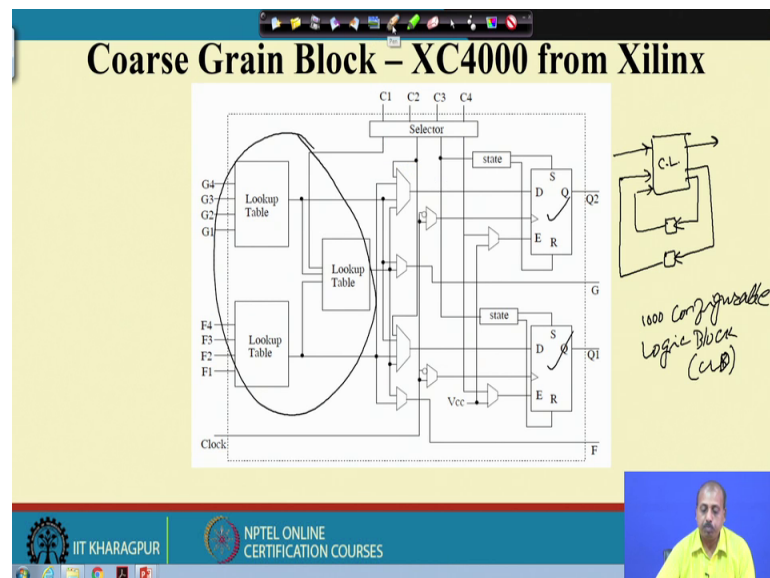
Any 5 input Boolean function because if I have. So, if I have got a 5 input Boolean function $f(x_1, x_2, x_3, x_4, x_5)$. So, we know that by Shannon decomposition. So, I can write it as $\bar{x}_1 f(x_1=0) + x_1 f(x_1=1)$. So, this x_1 line can be connected to this C1. Now, this can realise so, this lookup table can be used to realize $f(x_1=0)$ and this lookup table can be used to realize $f(x_1=1)$ and this selector can be programmed such that it comes here ultimately.

So, this lookup table realizes the 3 variable function consisting of the variables x_1 f x_1 equal to 0 and f x_1 equal to 1. So, these 3 variable these 3 functions can be 3 these 3 variable function can be realized by this lookup table. As a result, combining these 3 lookup tables you can get.

Any 5 input function realized so, that is very that is very powerful. So, after the lookup table after the basic combinational logic function that we can have. So, we see that there are other elements for example, there is a multiplexer here and this from this multiplexer you can either pass one of these lines through this to or you can pass this output of this G or you can pass the output of this f lookup table or you can pass this output of this lookup table. So, this is a 4 input multiplexer so, you can pass any of these 4 inputs to this to this d flip flops.

So, this is similarly [vocalized-noise to this d flip flop similarly, for the lower side also we have got another d flip flop and this you can. So, you can pass this flip flop values you can pass this combinational logic into these flip flops and. So, as a result the value will be stored. So, as you know that whenever we are trying to realize this finite state machine type of structures.

(Refer Slide Time: 15:02)



So, what we have is we have got some combinational logic and that combinational logic feed some flip flops. So, they feed some flip flop for the, this next state values. So, this is the structure of a finite state machine. Now, you see if you look into this structure. So, it

is very much likely that this combinational logic, I will be able to realize using these lookup tables and I have got 2 flip flops. So, this 2 flip flops. So, these 2 flip flops can be used to do that.

So, you see a sizable amount of finite state machine can be realized by a single FPGA logic block the off Xilinx and of course, there are large number of such logic blocks available. So, if you in your design if there are thousand in this family XC 4000s family.

So, if they are there are thousands such configurable logic blocks. So, they are called configurable logic block or CLB then these thousand such CLBs. So, they will ensure that you have got so there are large number of flip flops in the design in the in the chip.

So, normally we can use them for realizing any complex finite state machine with large number of states and all. So, that can be done very easily. So, this makes this Xilinx family of FPGS quite popular because with this coarse grain logic block. So, you can put good amount of logic into individual blocks and of course, we can connect them with one another.

(Refer Slide Time: 16:42)

The slide displays a logic diagram for the ACT1 block. It features four inputs: w, x, y, and z. Inputs w and x are connected to a 2-input multiplexer with select line s1. Inputs y and z are connected to another 2-input multiplexer with select line s2. The outputs of these two multiplexers are connected to a 4-input multiplexer with select lines s3 and s4. The output of this 4-input multiplexer is connected to an OR gate with inputs s3 and s4, which produces the final output f. Handwritten notes in black ink on the right side of the diagram state: "3-input mux", "4-input mux", and "~ 700-800". The slide footer includes the logos for IIT Kharagpur and NPTEL Online Certification Courses, along with a small video inset of the presenter.

Next we will be looking into another coarse gain logic block from actel which is known as ACT 1 logic block.

So, compared to Xilinx FAGP so, they are not that much that much logic reach I can say, but it is ah, but it is it is quite simple. So, we have got 3 multiplexers connected in this

fashion. So, these 3 multiplexers are connected and there is one or gate. So, these 3 multiplexers and this or gate so they can realize some functions.

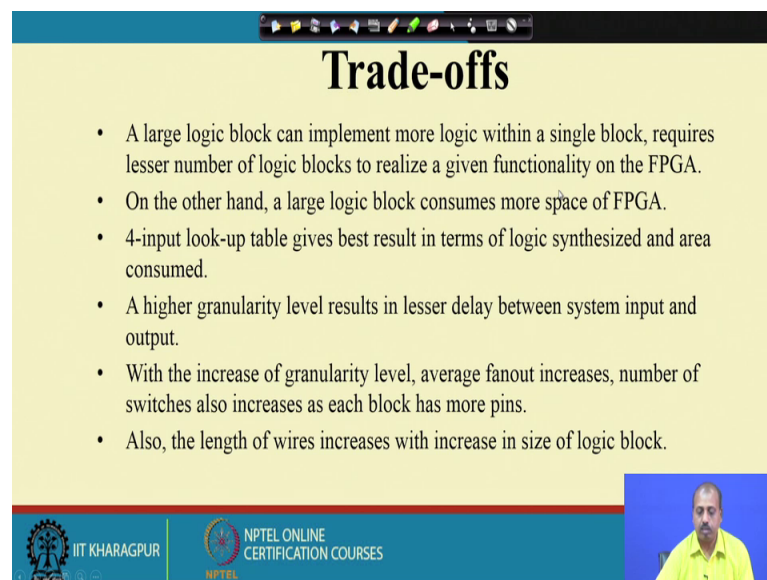
So, these multiplexers can be programmed like say. So, overall logic block has got this 3 plus 3 plus 2 total 8 inputs. So, this overall logic block is of 8 input and 1 output.

So, of course, it cannot realize all 8 input functions, but if you analyze this particular block. So, it can be shown that it can realize up to all 3 input functions. So, up to all 3 input functions can be realized by this logic block. So, it can be proved similar for 4 input so it cannot realize all. So, it is not all, but many of the 4 input functions can be realized so this way if you enumerate.

Then, there are around 700 unique 700 to 800 unique functions that can be realized by this act 1 logic block.



So, that is also quite large number. So, because of the advantage that these individual logic blocks are pretty small, you can have more number of logic blocks within the FPGS chip, but compared to these Xilinx FPGAs. So, they are much simpler.

(Refer Slide Time: 18:33)



Trade-offs

- A large logic block can implement more logic within a single block, requires lesser number of logic blocks to realize a given functionality on the FPGA.
- On the other hand, a large logic block consumes more space of FPGA.
- 4-input look-up table gives best result in terms of logic synthesized and area consumed.
- A higher granularity level results in lesser delay between system input and output.
- With the increase of granularity level, average fanout increases, number of switches also increases as each block has more pins.
- Also, the length of wires increases with increase in size of logic block.

 IIT KHARAGPUR  NPTEL ONLINE CERTIFICATION COURSES

. So, next we will see how this what are the trade-offs between this coarse grain logic block and the fine grain logic block whenever we are looking for this coarse grain logic block a larger logic block and implement more logic within a single block and requires lesser number of logic blocks to realize a given functionality on FPGA.

So, individual logic blocks are having more number of functionality that can be implemented there. So, I will require less number of such logic blocks. So, the number of interconnects that will be needed will be less. So, it will require that delay of the design will be low because the interconnects. They contribute significantly towards that delay and with the increase in the length of the interconnect the delay increases by the square of that. So, as a result so if the length of the interconnects are small then I will the delay of the design will be less.

So, a large logic block can implement more logic function within a single block. So, it requires lesser number of logic blocks to realize a given functionality on FPGA the other hand, a large logic block consumes more space on FPGA as I said that say for example, these Xilinx XC4000 logic block that we have seen. So, this is a quite powerful no doubt it has got large number of large number of flip flops within it. So, we have got say we have got say 10000 logic blocks and each are having 2 flip flops in there is a 2000 flip flops.

So, are you going to use really these 2000 flip flops in our design. So, if you are not using that then of course, there is a wastage and there is no way to tell that I will not use these flip flops. So, they will remain whether you use it or not. So, they remain in the FPGA chip so, so that is there so, there that that is a wastage of space. So, the large logic block consumes more space on FPGA. Now, there are there is a study that shows that among these lookup tables that we have.

So, 4 input lookup table they gives best result in terms of logic synthesized and area consumed. So, if you make a trade off plot so, are some early researchers they have shown that that this 4 input is the ideal value 3 input if you make. So, that is also not good 5 input also we cannot they are not very much useful, but these 4 inputs are better a higher granularity level.

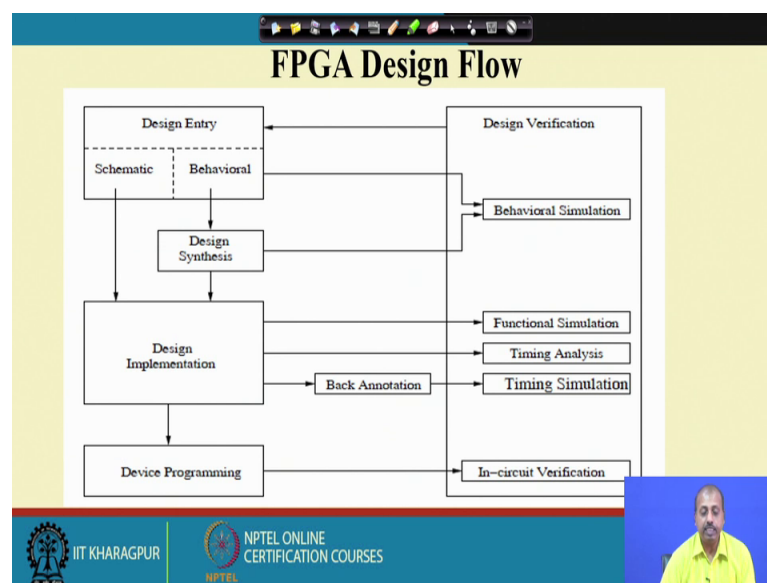
Results in, lesser delay between system input and output this point I have already said that we have got this interconnect length will be small ah. So, as a result that delay will be less with the increase of the granularity level average fan out increases and number of switches also increases as each block has more pins. So, one per one output may be one output of a logic block may be going to 10 different places.

So, as a result the fan out will be high. So, that way again that is another point so, if fan out is high so, we have to put some buffer in between. So, as a result the overall logic requirement will also go up and length of the wires increases with increase in the size of the logic block so, the logic block is.

So, if you look into this FPGA chip as we have shown you that if these are the logic blocks and say this is this is another row of logic block. So, we have got interconnects running between them. So, if these logic blocks sizes are small or large then naturally this interconnects length is also going to be large. So, that is why we have got that is another problem.

So, if you have got larger sized logic block then the length of the wires will also increase. So, that way it may be better that we go for smaller size logic block or final logic blocks.

(Refer Slide Time: 22:18)



So, the overall FPGA design flow is like this. So, no FPGA based designs are done manually. So, the first of all, it goes through a design entry phase. The Design Entry phase it can be either by a Schematic Entry or it may be a Behavioural Entry. So, Schematic Entry means. So, you draw your circuit in terms of AND OR NAND NOR gates and say flip flops and all.

So, you do draw your design in terms of that normally FPGA vendors they provide you some logic libraries. So, we take the elements from that logic library and draw our

schematic in terms of that then. So, other possibility is that you describe your design in some language like VHDL or very log type of language and then from there you go through a synthesis states.

So, that will convert your description into circuit. So, whatever you do whether you directly enter the schematic of your design. So, normally for smaller designs and for more if we are looking for optimized design so, we go for this schematic based our strategy.

So, that the overall control is in the hand of the designer or we can have this behavioural entry. So, behavioural entry is normally there when we have got the entries from more complex systems. So, you want to it is difficult to draw the circuit directly. So, go for this behavioural design so whatever and design synthesis.

So, these are the cat tools so that will convert this description on to this implementation. So, after we have got this implementation so, implementation is by means of this logic elements that the library supports and the interconnections that we do. So, we do some timing verification and all so we will come to that and then it goes to device programming. So, once this design has been implemented so this device programming so, it generates the control program.

For the FPGA, and that control program is used for say if it is an a SRAM based FPGA. So, it will be used it will be downloaded onto the FPGA chip for functioning if it is anti-fuse or this one time programmable FPGAs then these device programs. So, they will be used to burn the appropriate fuses or anti fuses and that way this connection the FPGA will become functional. So, this is the design process.

So, design whenever we are doing the design digital design the verification also comes into question. So, if you are having this behavioural specification or the. So now, we can do a simulation of it. So, we can we can put some sort of we can put some sort of tested vectors and see like whether it is giving me proper result or not, that is via behavioural simulation.

Similarly, for the Design Synthesis also so, we can go via some simulation and do that and for the once the design has been implemented. So, you can go for functional

simulation. So, to see whether functionally it is giving correct result or not you can do a timing analysis so now may be in your design entry.

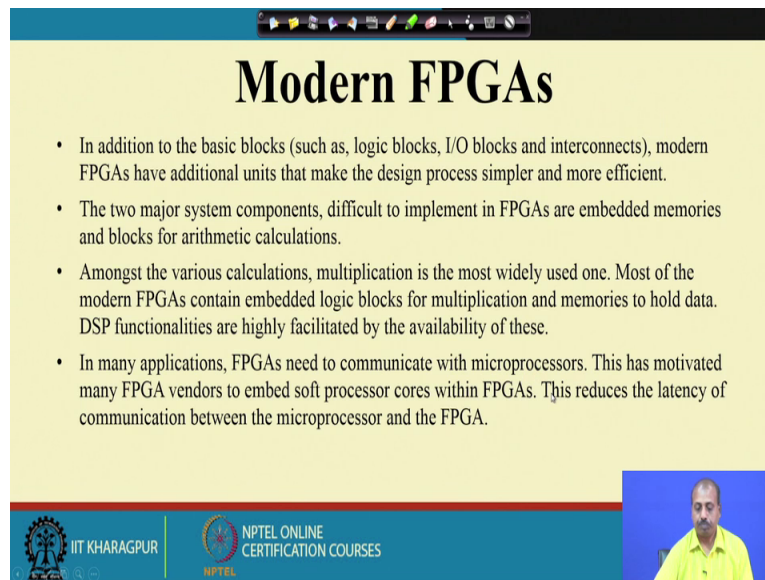
So, we have put output of 1 block to connected to the input of another block but, while you have done the implementation so, these 2 blocks are placed far away on the FPGA chip and they are going through a number of interconnect stages for doing the for getting the connection between the 2 points.

So, as a result the delay of that where will be more so, that will be captured in the timing analysis phase. So, the timing analysis it will find out whether there is any timing violations etcetera and then it will be giving all the wired delays and those wire delays will be back penetrated into the original design. So, that is the stage of back annotation so, the delay values are put onto the design and then we do another timing simulation so, after timing simulation.

So, if we are satisfied with the timing simulation then only this device programming is done and once the program control program has been loaded on to the FPGA device. So, we can go for in circuit verification. So this in circuit verification, so, they will be it will be it can verify if there is any problem in the actual running of the system or not.

Normally, if your design is through this timing simulation so, we can be sure that the design will work in the final implementation as well if you look into the modern FPGA. So, in addition to the basic blocks such as logic blocks IO blocks and interconnects.

(Refer Slide Time: 26:50)



Modern FPGAs

- In addition to the basic blocks (such as, logic blocks, I/O blocks and interconnects), modern FPGAs have additional units that make the design process simpler and more efficient.
- The two major system components, difficult to implement in FPGAs are embedded memories and blocks for arithmetic calculations.
- Amongst the various calculations, multiplication is the most widely used one. Most of the modern FPGAs contain embedded logic blocks for multiplication and memories to hold data. DSP functionalities are highly facilitated by the availability of these.
- In many applications, FPGAs need to communicate with microprocessors. This has motivated many FPGA vendors to embed soft processor cores within FPGAs. This reduces the latency of communication between the microprocessor and the FPGA.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

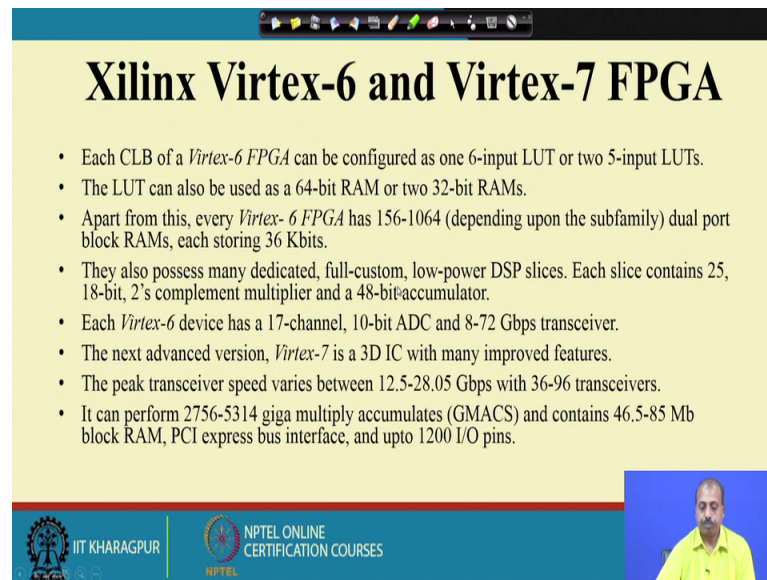
NPTEL

Modern FPGAs have additional units that help in the design process to make it simpler. The 2 major components that we have that we do not have in FPGA are the logic blocks and all one is embedded memory. Another is block for arithmetic calculation. So, the various calculations multiplication is the most widely used one because FPGAs may be used for some signal processing type of applications or multiplication will be very important.

And most of the modern FPGAs they contain embedded logic blocks for multiplication and memories to hold the data. So, DSP functional digital signal processing functionalities are highly facilitated by the availability of this multiplication and multiplier and memory in many applications FPGAs need to communicate with microprocessors. So, the FPGAs a chip that needs to communicate with the some outside chip and many FPGA vendors they have embedded some soft processor codes within FPGAs.



So, this processor codes are there. So, we can you can very easily use the communication mechanism supported by those soft core of the processors that, we have that is embedded into the FPGAs. For example, these Xilinx they have got power pc embedded on to this FPGA chip. So, that way we can have this microprocessor it can establish a communication between the microprocessor and the FPGA.


(Refer Slide Time: 28:20)



Xilinx Virtex-6 and Virtex-7 FPGA

- Each CLB of a *Virtex-6 FPGA* can be configured as one 6-input LUT or two 5-input LUTs.
- The LUT can also be used as a 64-bit RAM or two 32-bit RAMs.
- Apart from this, every *Virtex-6 FPGA* has 156-1064 (depending upon the subfamily) dual port block RAMs, each storing 36 Kbits.
- They also possess many dedicated, full-custom, low-power DSP slices. Each slice contains 25, 18-bit, 2's complement multiplier and a 48-bit accumulator.
- Each *Virtex-6* device has a 17-channel, 10-bit ADC and 8-72 Gbps transceiver.
- The next advanced version, *Virtex-7* is a 3D IC with many improved features.
- The peak transceiver speed varies between 12.5-28.05 Gbps with 36-96 transceivers.
- It can perform 2756-5314 giga multiply accumulates (GMACS) and contains 46.5-85 Mb block RAM, PCI express bus interface, and upto 1200 I/O pins.

 IIT KHARAGPUR  NPTEL ONLINE CERTIFICATION COURSES



So, Xilinx, Virtex 6 and Virtex 7 series of FPGAs so, just to have a brief idea the CLP of Virtex 6 can be configured as 6 input LUTs or 2 5 input LUTs. So, these are the advancement from 4 input it comes to 5 and 6 input. So, LUT can be used for 64-bit ram or to 32 bit rams so, that is. So, that way it gives to gives you to store. So, some RAM space there.

So, Virtexs FPGA has 156 to 1024 dual port block RAMs each storing 336 kilobits. So, the huge amount of space is provided dedicated full custom low power DSP slices are there. So, that can do digital signal processing calculations it has got 17 channel 10-bit ADC 8 to 72 Gbps transceiver. So, 10-bit ADC 17 difference analog signals can be fed into this 10-bit ADC for doing the analog to digital conversion and this transceiver is there for communication Virtex 7 is 3 D IC. So, 3 dimensional integrated circuit so, again it has got much higher capabilities.