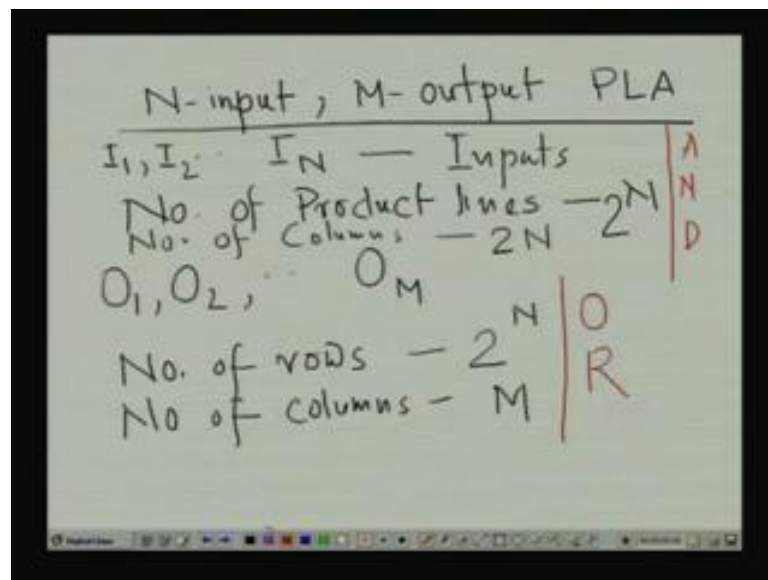


Digital System Design
Prof. D. Roychoudhury
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 28
Programmable Logic Devices (Contd.)

In the last class, we have read the design of Programmable Logic Carry, we have seen that, how one PLA can be designed for a set of output functions. Now, today we will continue the discussion on the programmable logic devices. First we will see that how the, what will be general structure of a PLA design.

(Refer Slide Time: 01:23)

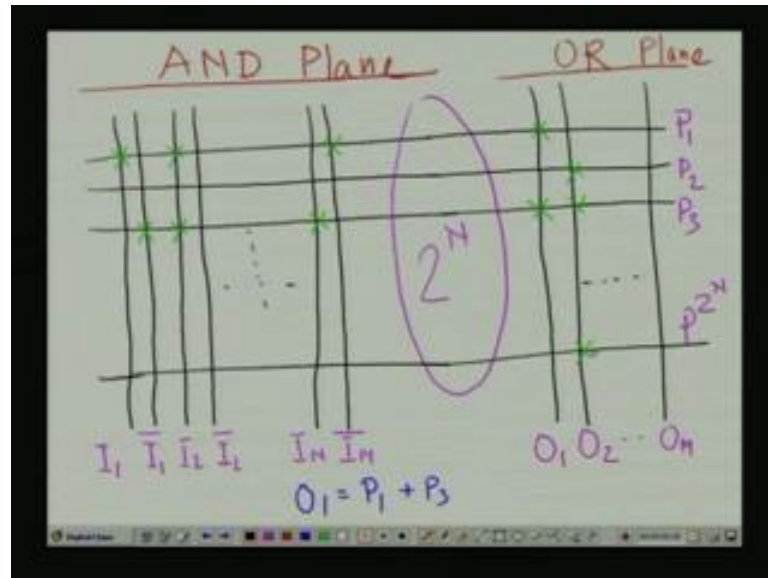


So, if we consider the one N input, M output PLA, then what will be the programmable AND array and the OR array structure, so first we will consider the programmable AND array, as it is a N input, so there will be two N number of lines. Say my inputs are I_1, I_2 to I_N and outputs are, these are my inputs and outputs are O_1, O_2 up to O_M . So, this should be I_1, I_2, I_N , some there will be in the PLA design, there will be N number product lines, so number of product lines or the rows are says N.

So, number of rows in the AND array is, number of product lines is 2 to the power N and number of columns is 2^N as the N number of input variables and the N number of complemented variables are available. So, this is my AND plane specification, this is my AND, now for the OR plane, if there are M number of outputs.

So, in the OR plane the number of rows are 2 to the power N, the product lines and number of columns are the number of output functions that is M. So, these are my OR plane specification, so now, if we draw the PLA, the generic structure of a PLA.

(Refer Slide Time: 04:52)



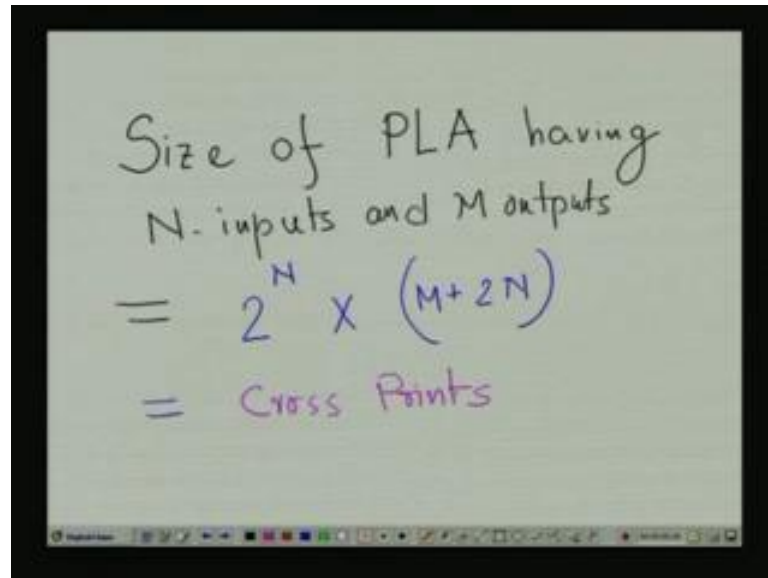
So, it is N number of input lines, so if we mark that as a, say $I_1, \bar{I}_1, I_2, \bar{I}_2, \dots, I_N, \bar{I}_N$, so these are 2^N number of columns. Now, we have 2^N number of product lines, so these are these of the 2^N number of product lines. So, here this is actually 2^N and depending on the program that means, which min terms we have selected say the cross points will be here, and this is row wise; that means, see for this case, it is I_1, I_2 , then say \bar{I}_N .

Similarly, the third one will be, say \bar{I}_1, I_2, I_N , like that, so this is my AND plane, now if draw the OR plane. So, 2^N number of product terms these of the, these will be the rows and there will be M number of columns, so M outputs, these are my M outputs and they are O_1, O_2 up to O_M .

Now, in the OR plane the cross points will be column wise because, from the AND plane the mean numbers, min terms are selected and then, this is the sum of min terms. So, say if this is my min terms and if these are the product lines say P_1, P_2, P_3 and P_{2^N} , then say, one is in this structure, say O_1 is P_1 plus P_3 . So, here similarly the min terms will be selected, so this is my programmable AND plane and this is my programmable OR plane.

And, the size of the PLA is see that here, there are 2^N number of rows and in the AND plane there are 2^N columns in the OR plane, it is N number of columns. So, total columns are $M + 2^N$.

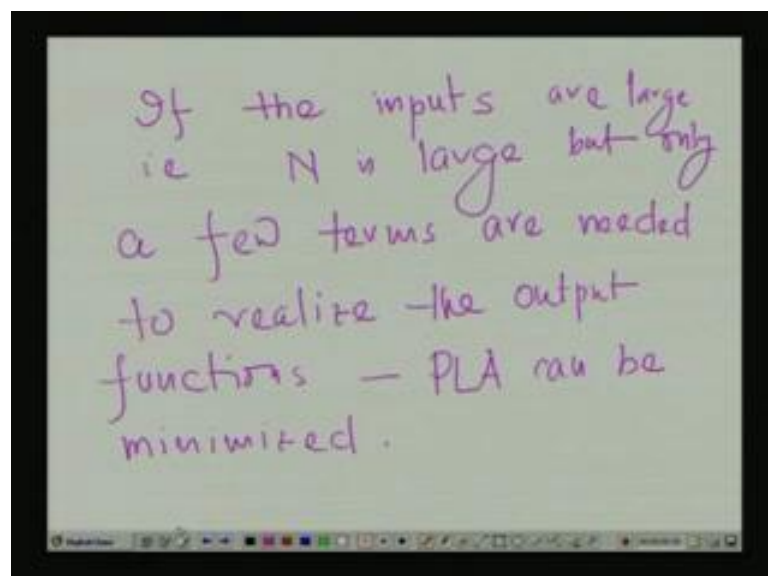
(Refer Slide Time: 09:04)



Size of PLA having
 N -inputs and M outputs
 $= 2^N \times (M + 2^N)$
 $= \text{Cross Points}$

And that so, the size of PLA having N inputs and M outputs is equal to the power M by this is the number of rows, my $M + 2^N$ number of columns. So, this is a size, and so, these many number of cross points are also available, so these are my number of cross points, maximum number of cross points.

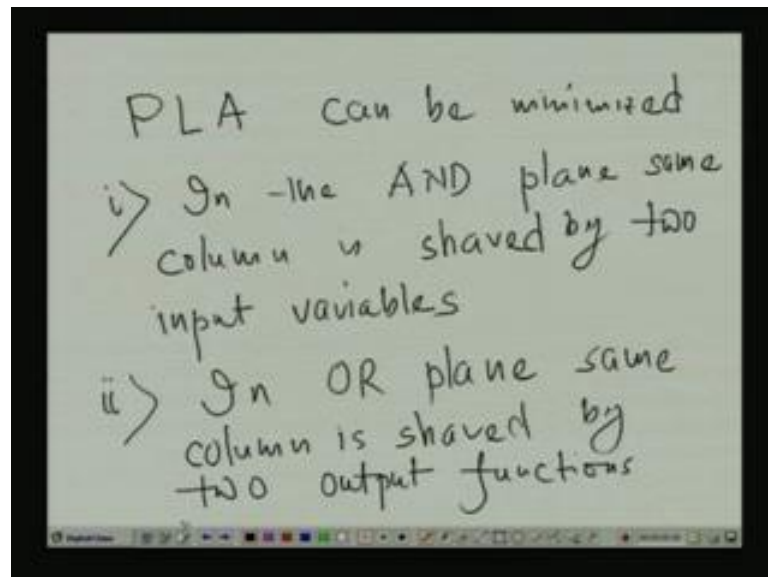
(Refer Slide Time: 10:18)



If the inputs are large
ie N is large but only
a few terms are needed
to realize the output
functions — PLA can be
minimized.

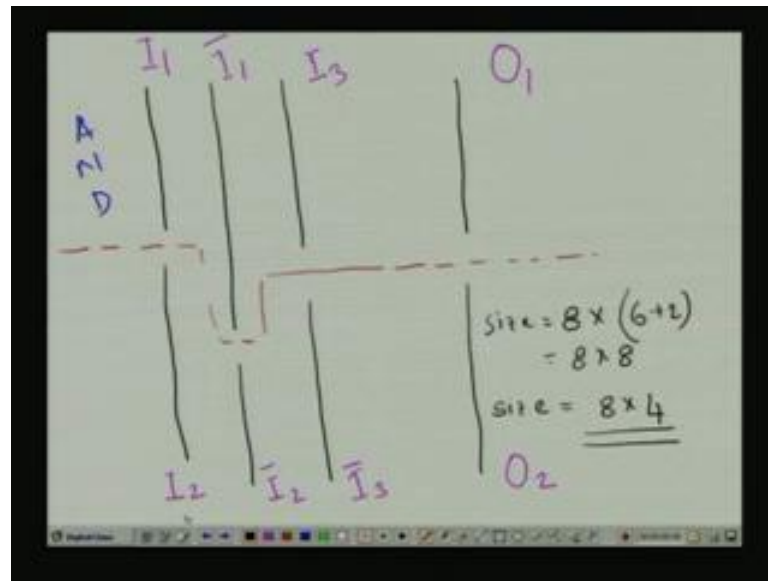
Now, as I already mentioned that, if the number of min terms, say if the input variable of the inputs are large means, that is N is large, but a , only a few min terms are needed to realize the output functions. Then, the PLA can be minimized and in this case the PLA is efficiently utilized, now another case is the, for PLA minimization.

(Refer Slide Time: 11:39)



Say PLA can be minimized, that number one that, in the OR plane or first we see the AND plane, in the AND plane same column is shared by two input variables. Or similarly in the OR plane same column is shared by two output variables, output functions two output functions, in this case, actually the PLA is partitioned row wise.

(Refer Slide Time: 13:23)

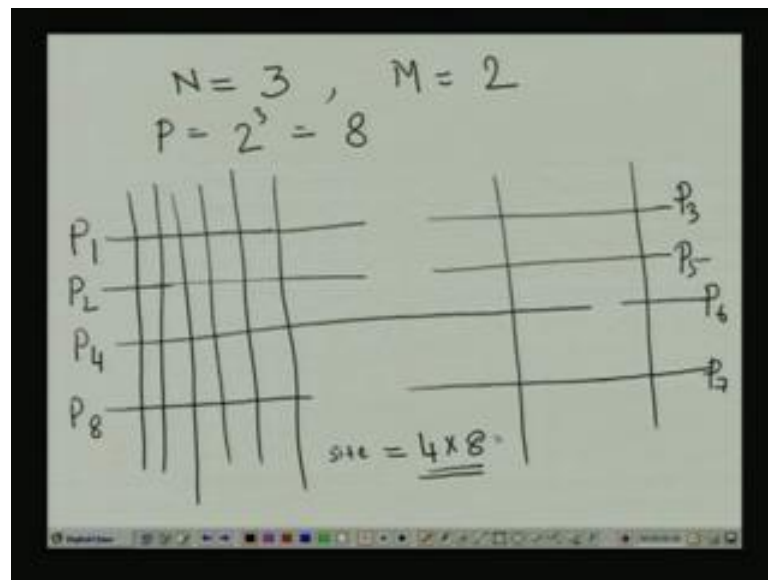


Means, see if I give a, say I have columns, say there are see here I 1, say I 1, say this can be I 2, this can be I 1 complement, this can be I 2 complement. So, this I 3, this is I 3 complement, so here the partitioning has been done like that, this is in the AND plane structure. And, similarly if the OR plane, say if there are only two outputs and it is possible that this is 1, so this is O 2, so this can be partition like that.

So, in this case as it is a three input two output function, so the size, actual size needed was size would be the eight product terms 2 to the power 3, 8 by 6 plus 2. So, these will be because, 2^N means 6, two output, so the actual size or original size should be 8 by 8, 8 by 8 structure. Whereas here we see that it is only the, in this case the product terms we are not sharing.

So, rows are same, so the modified size will be only 8 by 4 because, here columns are only 4, so in this case, the example I have shown here actually the columns are sharing by the input variables and the output functions. Similarly, what we can do, the other partitioning for PLA minimization that can be the, if the product terms are shared. So, what we can do, say I have if for N input variables I have 2 to the power N product terms.

(Refer Slide Time: 16:30)

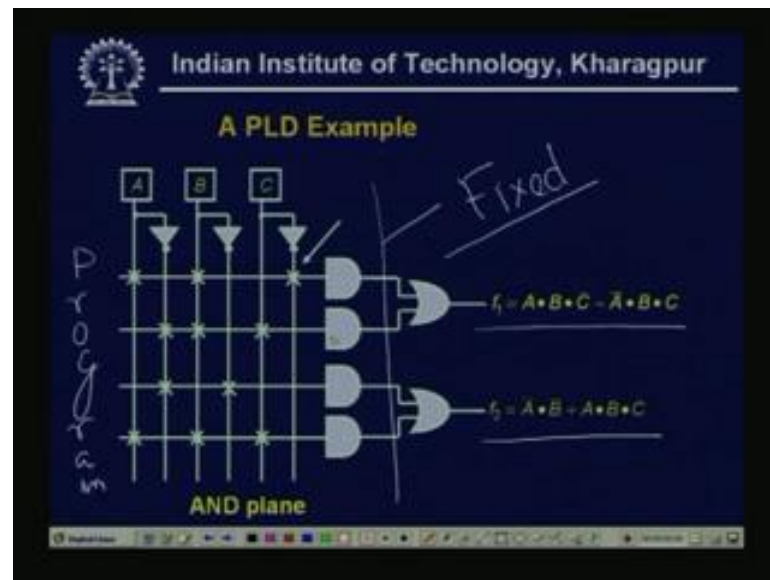


Say in my guesses, here N equal to 3 and M equal to 2, what we have seen, so 2 to the power 3, there are product terms, P is 2 to the power 3 equal to say 8. Now, if the design is such, that it is possible to share the product terms, then instead of 8, what we can do that, say this is as if this is possible. Now, say this is P_1, P_2 say P_4 and P_8 , and it P_1 is shared by P_3, P_2 is shared by P_5 , this is P_6 and say this is P_7 .

Now, the columns are as it is, say it is 6 num 6 2 N ; that means, there are 6 columns in the AND plane and say in the OR plane, there are two functions see like this, these are the two functions. So, in this case the instead of 8 by 8, we will get the size is, see here it is only four columns, 4 divided 4 by 8, it is 4 by 8, earlier cases it was 8 by 4.

If it is row wise shared or row wise partition; that means, columns are shared, here it is column wise partition; that means, product terms are the rows are shared, so in this way the PLA can be minimized.

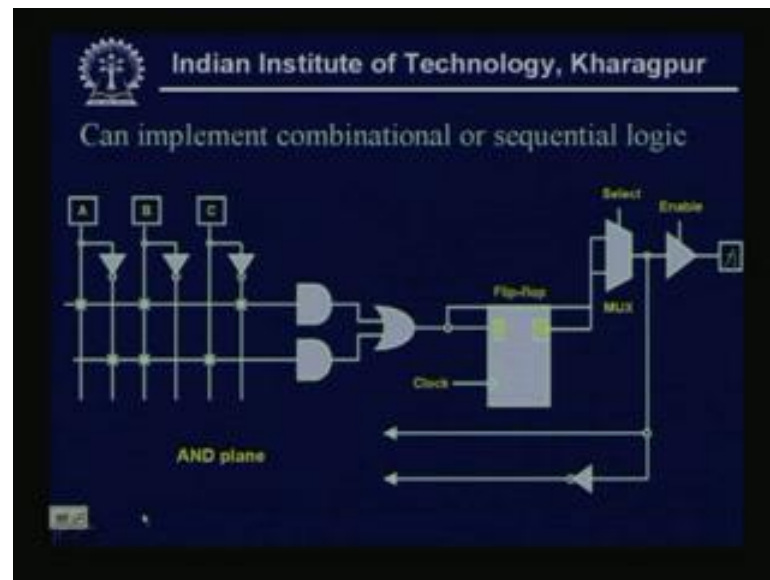
(Refer Slide Time: 18:59)



So now, we have seen the programmable logic devices, we have mainly discussing on different type of programmable logic devices structure. So, the PLA structure we have seen and how it can be minimized, now there are two other different PLDs are available. One is CPLD, the complex programmable logic devices and the other is field programmable gate array.

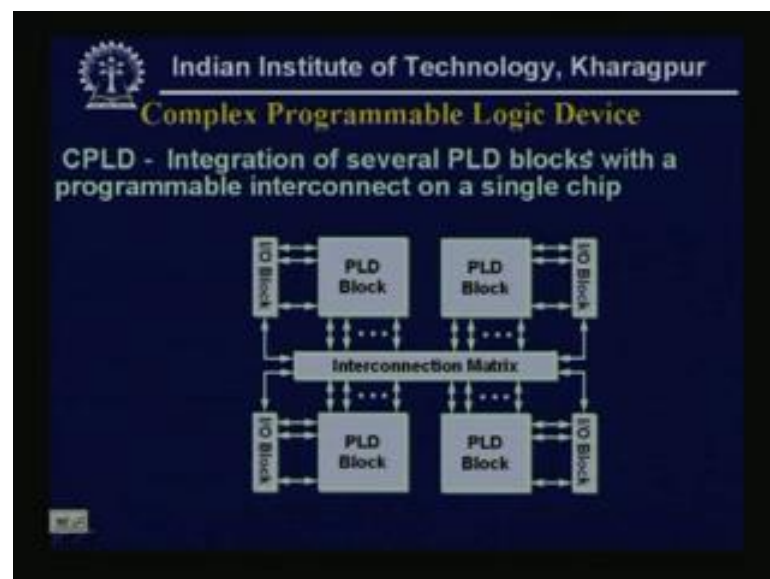
Now, we will see details the design architecture of these two type of PLDs, as already we have seen that PLD means, that say for this particular design, this is the AND plane, this is the programmable AND plane. So, we have see this is a programmable AND plane, and this is a fixed OR plane and in these cases, the two functions f_1 and f_2 are realized.

(Refer Slide Time: 20:26)



Now, not only combinational we have seen that sequential part also, can be realized by using this programmable and on the fixed type OR plane. So, any one of this, so far we have discussed the PROM, PLA, PAL, can be the PLD block.

(Refer Slide Time: 20:51)

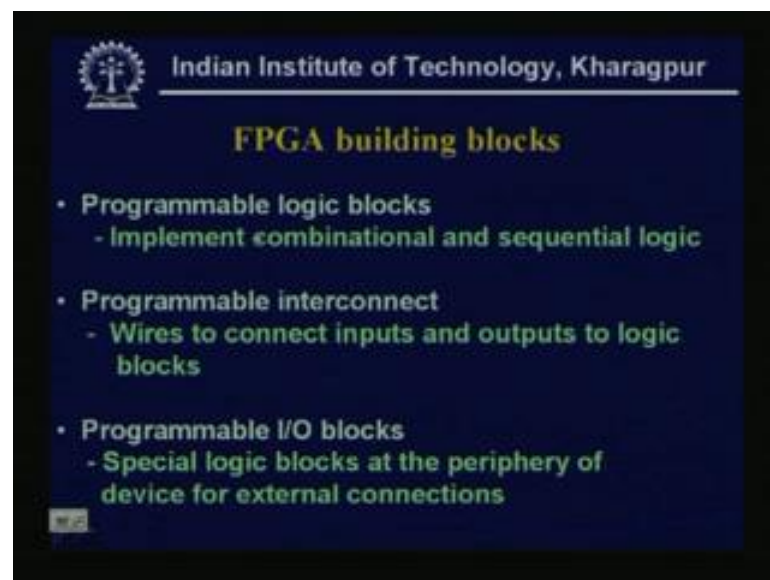


So, CPLD the complex programmable logic device, we can define as if, this is the integration of several PLD blocks with a programmable interconnect on a single chip. So, these are the PLD blocks and these PLD blocks can be any PLA, PAL or PROM type

of design. Moreover that IO blocks; that means, for the IO interconnections that can also be programmable.

And this some interconnection matrix is there, so this is the overall architecture of the CPLD. Now, we will read the field programmable gate array and we see that actually the basic architecture is same, only the technology is different.

(Refer Slide Time: 21:45)



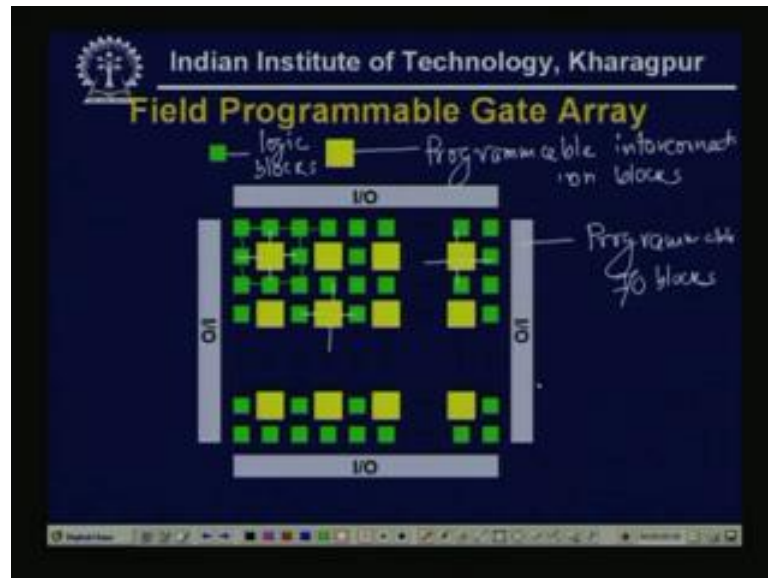
So, the FPGA building block, that it has some actually it has three types of blocks are available in the FPGA. The programmable logic blocks, which implement combinational and sequential logic, as already we have seen that PLD blocks can realize combinational as well as sequential blocks, now programmable interconnect. So, wires to connect inputs and outputs to logic blocks, just now we have seen.

See this logic blocks for this wire, this can be, this can also be programmable and for this purposes just to program the interconnection, actually this interconnection matrix is controlled from outside. So, there are some control lines or this can also be programmable or the interconnections are programmable, now this, programmable IO blocks, so special logic blocks at the periphery of device for external connections.

So, not only the logic blocks and the interconnection blocks are programmable, but the IO blocks; that means the, which input will we want to fill and which output we want to

take from the design, that IO blocks is also programmable. So, these are the three programmable logic blocks are the basic building blocks of FPGA.

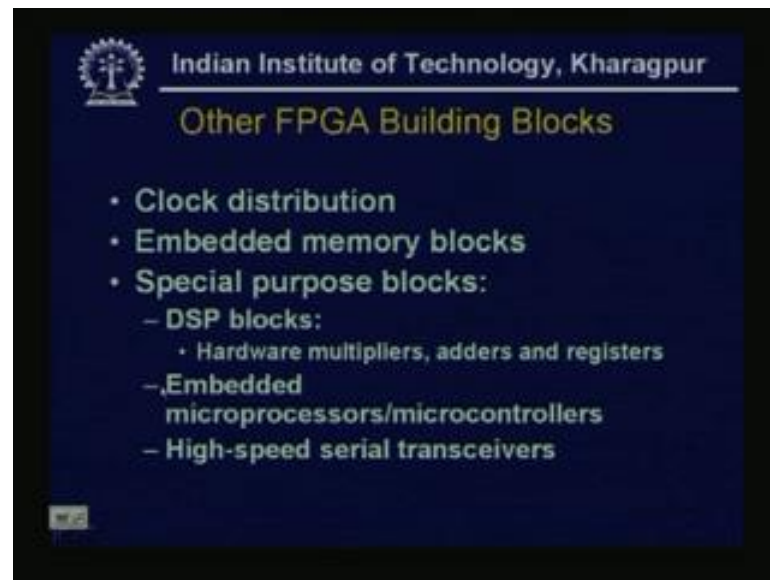
(Refer Slide Time: 23:34)



So, already this overall architecture we have once we have seen, now I describe one by one, so this is again a 2 D type of gate structures and if I consider these as a matrix. So, these as if, this small green one and the larger yellow blocks, these are the two types of matrix elements. And, the yellow one, we are telling this is my inter switch blocks or the programmable interconnection blocks, so this is my programmable interconnection blocks and this is my logic blocks.

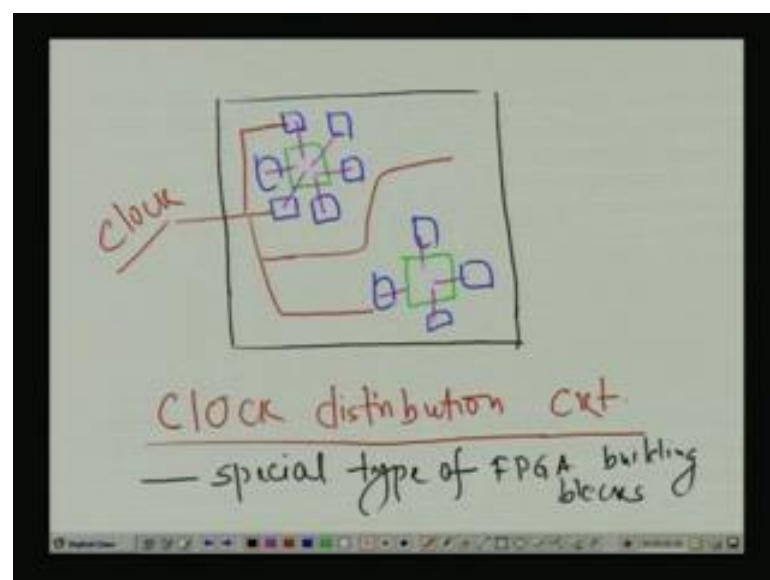
And these are the IO blocks, four IO blocks as if, they are arranged in a, in the boundary, so these are my programmable IO blocks. So, these are actually connected, this one is connected, these are all by directional and they are connected with the switch block also, so this connections are also. Similarly, here also I can connect, so these types of connections are available and all the three building blocks can be programmable from outside, so this is the basic architecture.

(Refer Slide Time: 26:04)



Now, there can be a other type of FPGA building blocks and they are for the clock distribution, the embedded memory blocks, special purpose blocks of DSP blocks, Hardware multipliers adders and registers, embedded microprocessors, micro controllers and high speed serial transceivers. So, clock distributions that, this is one circuitry and always these are needed, see if I have the, if we draw these things.

(Refer Slide Time: 26:47)



Say, as if this is my full FPGA and the one switch box, what I can draw, this is one switch box and these are my programmable, say these are my programmable logic

blocks, see this type of things. Now, all are connected, see these types of connections are there see I have other switch box here, which are connected to logic cells. Now, see this one need a, if it is a sequential circuits, so these needs a logic cells clock.

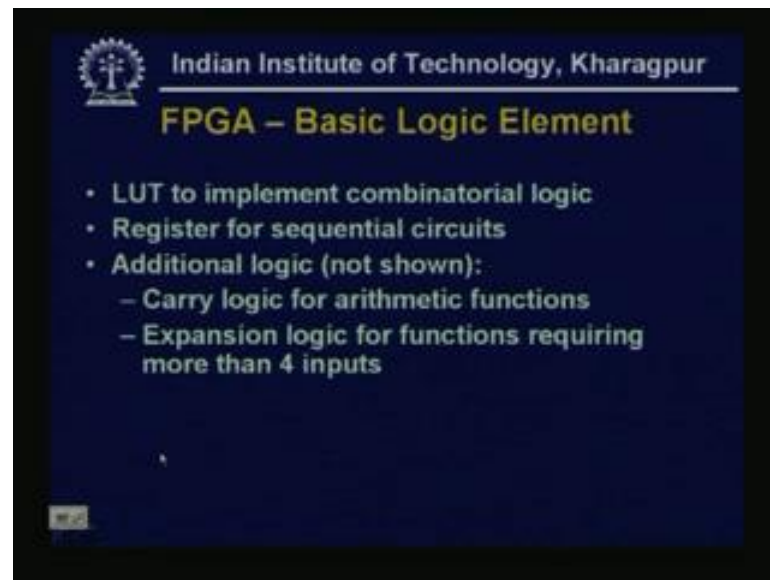
Similarly, here this is also needed clocks, see if this one is needed a clock or this type of things, so actually just to avoid the clock skew because, the same clock always we prefer the same clock. So, then the clock should be distributed, so that we, that clock skew can be avoided. So, we need a special circuit for this clock distribution and now is, the this is, so common for the complex circuit.

So, this clock distribution circuit itself is implemented and that can be treated as a special type of, this is a special type of building blocks, special type of FPGA building blocks. So, this is my clock distribution circuit, now embedded memory blocks, so far, we have discussed the logic blocks are combinational and sequences. But, all of we know, for that complex circuits or any real life circuits, we need the memories, lot of memories.

And these memories are now, they are called the embedded memories within the FPGA, so for that, memory is a special type of building blocks already implemented in the FPGA. Now, so digit, for digital signal processing we need a special type of processing, say some matrix multiplication we needed. So, some matrix multipliers or some fast multipliers, some fast adder or parallel adders and say set of registers, that all these things are needed, these are the normally these are the common processing elements of the DSP blocks.

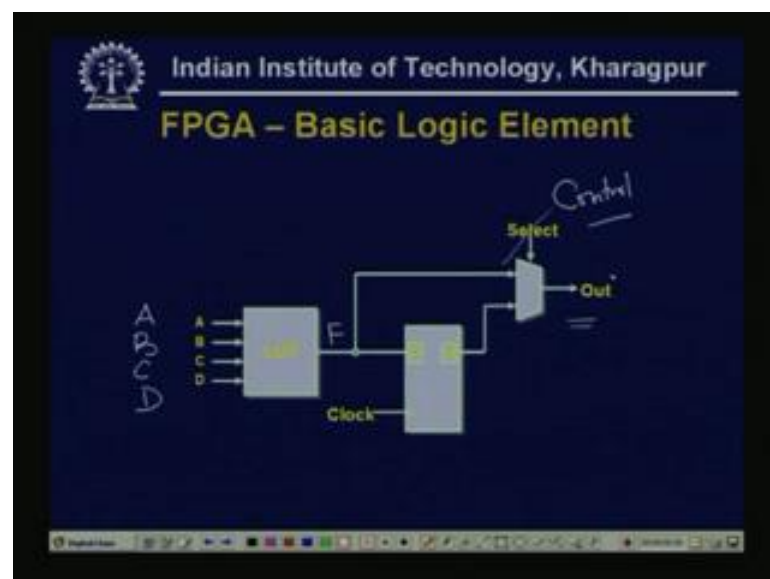
So, they are treated as a FPGA building blocks similarly, embedded microprocessors, microcontrollers that are also treated as a FPGA building blocks. So, nowadays though we call that normally there are only three types of building blocks one is programmable logic blocks, one is programmable interconnection or sometimes we called as switch box and the programmable IO blocks. But, in addition that now a days the modern FPGA chips they contain the, this type of circuitry and again they are treated as a special type of building blocks.

(Refer Slide Time: 31:03)



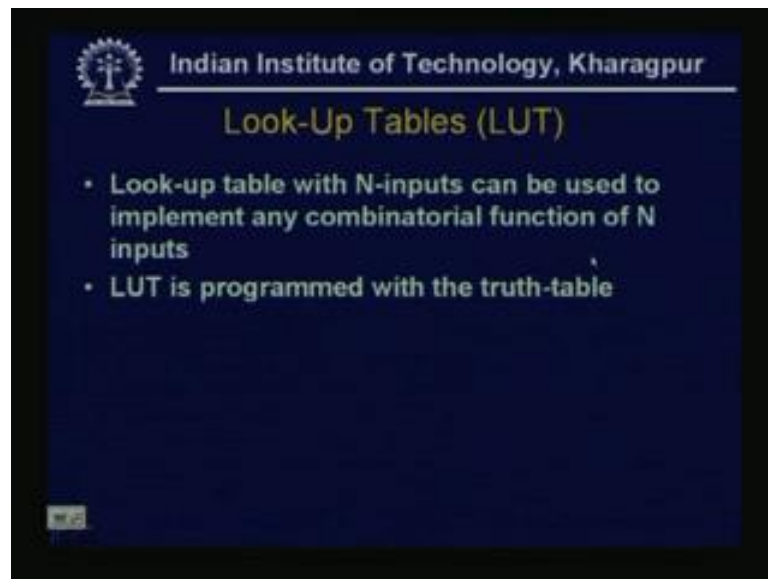
Now, how this FPGA basic logic element are implemented, as in the last class we have seen I mention that normally, look-up table is utilized or is the basic concept of a FPGA logic element. So, LUT to implement combinational logic, register for sequential circuits, additional logical and that this carry logic for a arithmetic function or expansion logic for functions requiring more than four inputs. These are the some common features of FPGA.

(Refer Slide Time: 31:43)



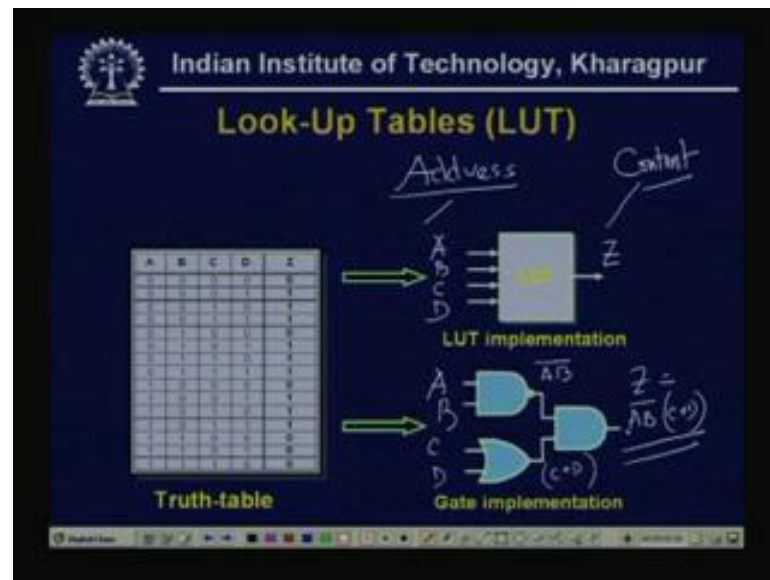
Now, this is one very simple or basic logic element, say I have one look-up table, see it has four inputs, the A B C D. So, the look there are four input variables A B C D are the inputs of the look-up table, see there is only one output, say one function F and that can be last as already we have seen and it or it can be selected. So, using this control line as the select this is my control from outside, when I want to take the output that can be chosen.

(Refer Slide Time: 32:45)



Now, look-up table is one of the important or this is the basic concept, so look-up table with N inputs can be used to implement any combinatorial function of N inputs. And look-up table is programmed with the truth table. So, we see that how the look-up table is implemented.

(Refer Slide Time: 33:05)

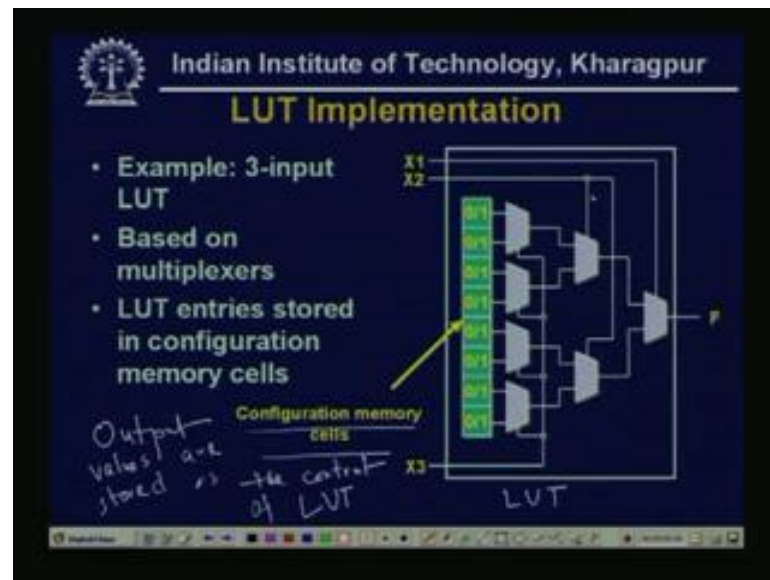


So, this is one example, as in the example I have shown there are one look-up table, it has four inputs A B C D and one output Z. See, this is the truth table, on the truth table this truth table is given the four inputs A B C D and this is one output. Now, see already we have using the Karnaugh map or many other strategies we have read, that we can derive the functions that what how that, what will be the Z, the output functions.

Now, what we can tell or that as, if these A B C D are the addresses of the table and Z is the content of the table, so here this LUT, for this LUT as if A B C D are the locations or the address of the table and Z is the content of the table. And, when if it is Z, Z is some function of A B C D, so using Karnaugh map, we know that always Z can be realize by using gate, this is simple type of gate implementation.

Where see, this is A B bar this NAND, so A B bar and this is a C plus D, CD or, so C plus D, so this is A B bar to C plus D, this type of Z equal to A B bar dot C plus D. So, this is the function realized, now the same thing we can treat as in, this is a that if I give the A B C D values as if that particular content already it is computed and that is kept as a content of the table. So, these mainly, this is the basic concept of the FPGA LUT. So, one thing is this, so mainly LUT has the inputs are treated as address and these of the address and output is content, so this is nothing but a memory.

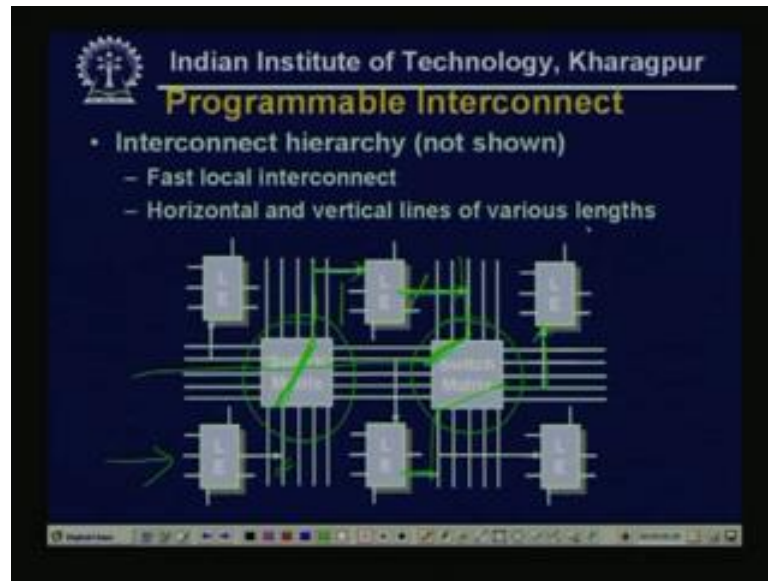
(Refer Slide Time: 36:16)



So, how it is implemented, see this is one multiplexer based, as if LUT entries stored in configuration memory cells and so, these are the contents of the memory. So, these are my, the output function the configuration memory cells means, as if output values are stored as the content of the table LUT and the three inputs X 1, X 2, X 3, they will be treated as the address of the memory.

And in this way, that LUT can be implemented, so this is one possible LUT implementation and this is based on multiplexers, multiplexers are mainly for the programming part as this is a programmable. So, which one I want to select, so it is do in that thing.

(Refer Slide Time: 37:39)



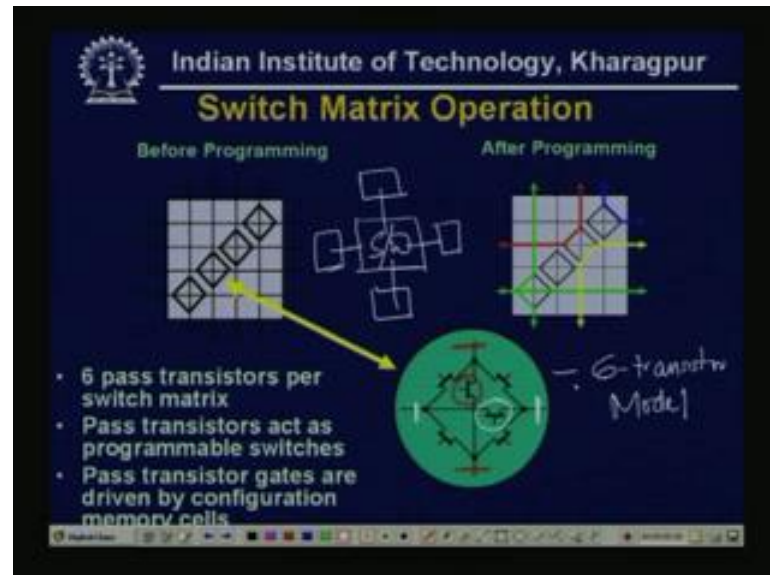
So, now for programmable interconnect, so LUT is the implementation of programmable logic block, now programmable interconnections, so interconnect means, we know that this is nothing, but the switching. So, as if the, I have a fixed wiring I have a 2 D structure of wires there are some rows and there are columns of wires. Now, which wire to be connected to whom, means which row is connected to which column that is nothing, but the switch.

So, that is why the programmable interconnections can be treated as a switch box, so this is one general picture, say as if, see these are some, this is one switch matrix, this is one switch matrix, this is one switch matrix. Now, say I have given, see this is my programming part it is coming from outside, so this is LE means latch enable, so this line is selected and the it is, see this is the switch, see it is going to this, see this is another latch enable.

So, it will be again depending on this, say as if this output is selected, so this can be multiplexer type of thing, again this line is selected, so this is a switch box and this can be a output. So, this can be a direct output, similarly different type of other connections are available here, this. So, mainly that some horizontal and vertical line what I mention that rows and columns of wires are there and in between some switching elements are put and the function of the switching matrix is that which wire to be connected to whom.

So, this is nothing, but that interconnection programmable interconnection part, so in this way the programmable interconnection blocks are implemented in FPGA.

(Refer Slide Time: 40:21)



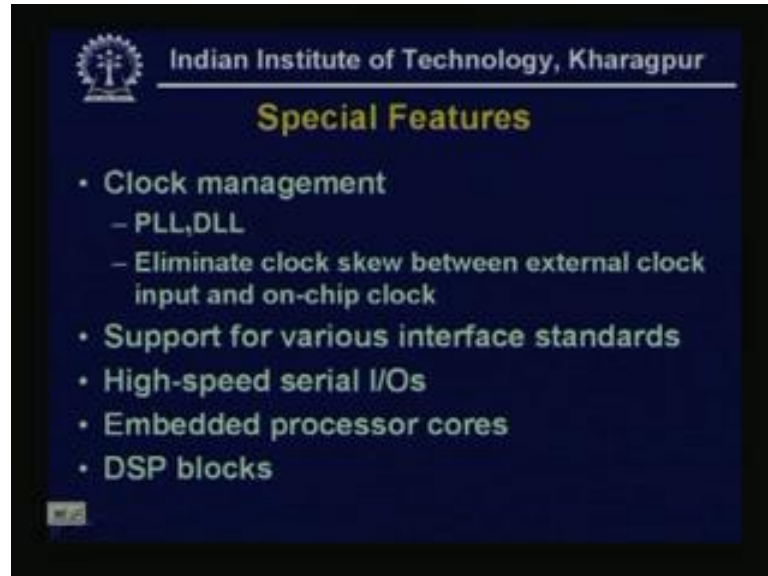
Now, if we quickly see that switch matrix operations because, these are nothing, but a switch. See that, this is before programming as if the switches are kept and this is after programming means which one is connected to which line, as if this is, this line is see this red line means, this is one connection. So, this blue line, this is one connection like that, now this is actually, actual switching is we know that this is by pass transistors and these are act as a programmable switches.

So, pass transistor gates are driven by configuration memory cells, see here that, this is the 6-transistor model, we called this is a 6-transistor model, there can be any many other models available. And, see that from anywhere to anywhere the connections can be, if these pass transistors are selected, then these two points are connected. Similarly, if this pass transistor is connected, then this and this points are connected.

So, in this way for any point can be connected, here the model we have taken is that one, one switch box is here, this is one switch box programmable switch box and as if the logic cells the logic cells are kept or arranged like that and these are the connections. So, if I see, that say as if this four connections, they are the four logic blocks and to whom which logic blocks to which blocks is connected that is the switch box is to in that

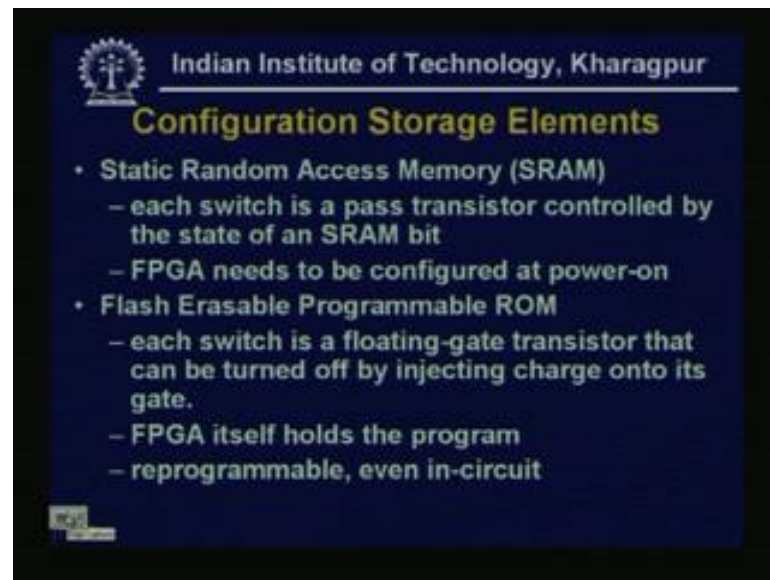
interconnection part. So, this is a 6 transistor model, and mainly pass transistors are being used to do that thing.

(Refer Slide Time: 42:46)



Now, there are some special features, say clock management normally the phase locked loop circuit or the direct link lock and these are being used for clock management which and this clock managements are needed to eliminate clock skew between external clock input and on-chip clocks. And the support various interface standards, high speed serial IO's, embedded processor cores and the DSP blocks, so mainly these are the special feature of FPGA.

(Refer Slide Time: 43:27)



Now, the configuration of storage elements, so as already we have seen that mainly the memory is the basic building block, because the LUT is implemented by using memory. So, the storage elements of the content that are kept in the LUT is the static random access memory SRAM. So, here each switch is a pass transistor controlled by the state of an SRAM bit and FPGA needs to be configured at power on, then flash erasable programmable ROM.

So, here each switch is a floating-gate transistor that can be turned off by injecting charge onto its gate. An FPGA itself holds the program and this; obviously, that can be reprogrammable event in circuit. That means, we can get that for one type of control one type interconnection, I get one design and then, that can be erased and next time for the different interconnection I will get a different design. So, it is the reused, it can be highly utilized specially for the purpose of the academic purposes or to use FPGA in a laboratory, this type of memories are very much needed.

(Refer Slide Time: 45:06)

Indian Institute of Technology, Kharagpur

Configuration Storage Elements

- Fusible Links ("Antifuse")
 - Forms a low resistance path when electrically programmed
 - one-time programmable in special programming machine
 - radiation tolerant

Now, configuration storage elements can be of fusible links, already we discussed that fuse or antifuse is used to get the cross points on or off. So, it forms a low resistance path, when electrically programmed, now one time programmable in special programming machine and this is, this should be radiation tolerant.

(Refer Slide Time: 45:33)

Indian Institute of Technology, Kharagpur

Embedded Memory

Dual-Port RAM

- M512 – 512 x 1
- M4K – 4096 x 1
- M-RAM – 64K x 8

Diagram illustrating the structure of Embedded Memory (Dual-Port RAM) with handwritten labels: Data, Data Out, and address.

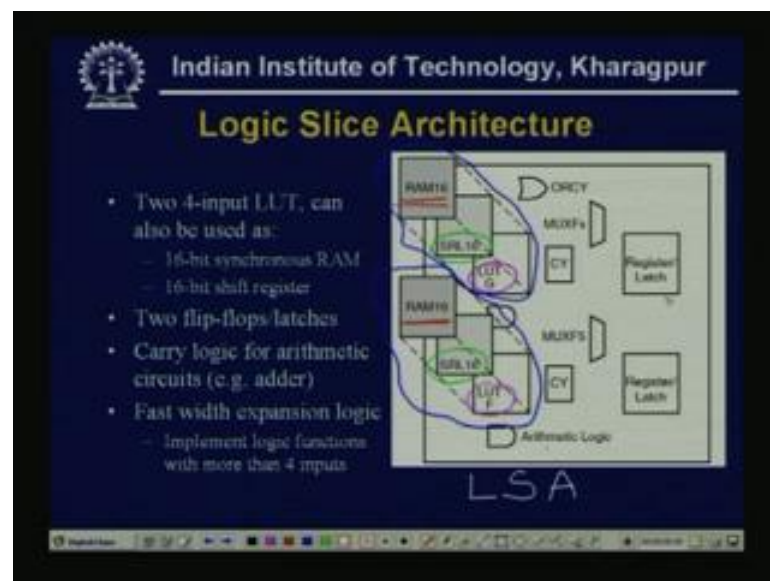
Now, we see some example, all as I mention already, that not only the combinational sequential logics that are implemented by using programmable logic blocks, but there are some special type of FPGA building blocks. One very important

think is the embedded memory and here, we see on this type of structure, see this in this example this is a 512 bit RAM block, this is a RAM.

And, this is a dual port RAM, so 512 by 1 bit and these are the different input output lines as in the memory, we know that normally there should be data in, means this is for my data in for my addressing part or just. We see that which value I want to store, that is my data and the next line is a clock, now this is the data out bit, so when I 1 2 to access the table of the LUT. So, this is my data out or the value of out, the output functions.

And this is my address signals of the addresses, means from which locations I want to get the value and another is the control signals. So, normally this is the structure of the embedded memory and these are the lines input output on the address lines are available on the chip of an embedded memory.

(Refer Slide Time: 48:07)



Now, this is another type of architecture we call the logic slice architecture, again this is an example, and actually real life example, See here 2 4 input LUT's can also be used, this is a complex thing say normally. So far, we have seen only one look-up table, now here 2 4 input LUT can also be used as 16 bit synchronous RAM, 16 bit shift registers. So, this is my 16 bit RAM, say 2 4 input LUT, so these are 16 bit RAM.

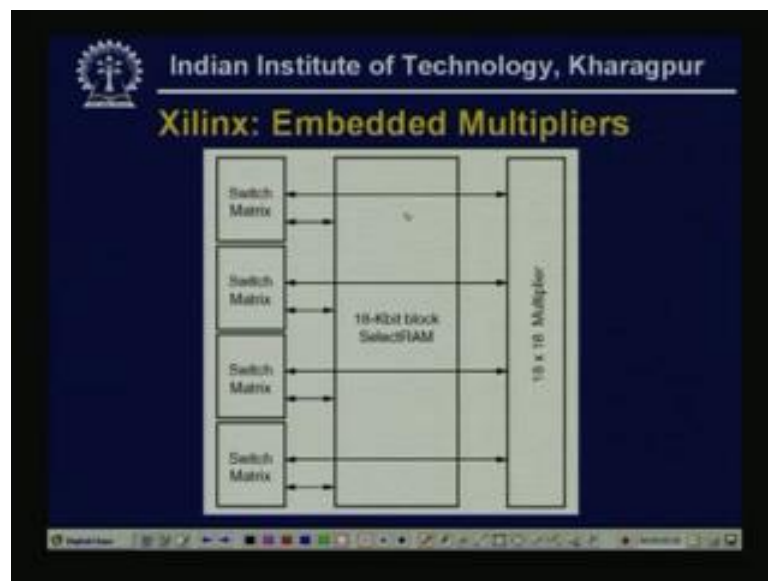
And these are my 16 bit shift registers, this is one shift register, this is one shift register, and two flip flops or latches, so this is one flip flop, one flip flop. So, actually if we see

that, this is one LUT as if, this is one LUT which contains one 16 bit RAM one 16 bit shift register and one flip flop. And this is another LUT, which the, which has the same contents, and there are 4 inputs. So, actually this when that, this is a look-up table how it is implemented; that means, this look-up table is of four inputs bits.

And then fast width expansion logic, so implement logic functions with more than four inputs, so now, these are my arithmetic logics which is also available and this is some register latches. See here actually the connections are not shown, this can be programmed according to the design, so this is a normally call LSA logic slice architecture. So, in this particular example actually 2 4 input LUT are used, but for other complex circuits more than two or actually N number of LUT is can be there.

In real life circuit, actually the switch box, the picture we have seen that two dimensional matrix or 2 D grid structures where one interconnection matrix or one interconnection programmable interconnection block is connected to. So, if we see that thing, see here that one interconnection block is actually connected with the logic blocks, this logic blocks. And this, so number of LUT's can be there in a FPGA. And this is actually how just now what we have seen, that how actually they are kept or they are implemented, so this is logic slice architecture.

(Refer Slide Time: 52:15)

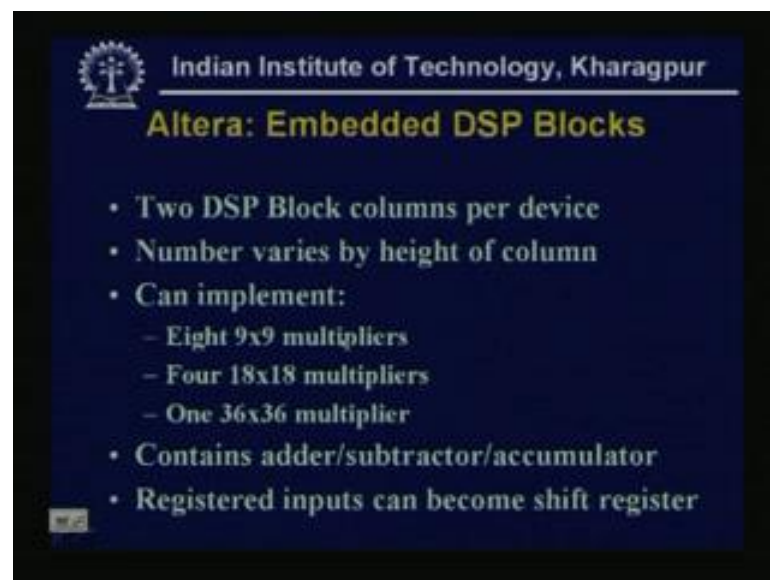


Now, we have seen some embedded memories, nowadays for say DSP type of circuits multiplier is a common processing elements. So, the multiplier itself can be pre-designed

and can be kept as a basic building block in the EPGA, so this is one xilinx example of embedded multipliers. See here, so that switch matrix is connected to 18 K bit blocks, select RAM.

So, this is my RAM and this is 18 by 18 multiplier, as if this is the xilinx the pre-designed multipliers that it kept as a embedded multiplier in the EPGA. One thing to be noted that, this can be a different type of architecture because, when we read later the different type of design of multipliers will be seeing that, this architecture can be changed, this is one example.

(Refer Slide Time: 53:30)

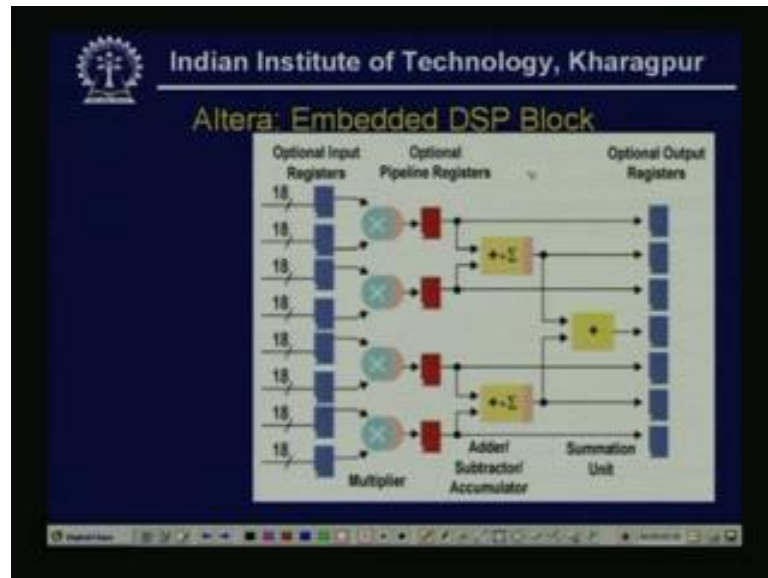


Now, this is different embedded DSP blocks, that Altera family has that they can be kept as a EPGA building blocks, so here, tow DSP block columns per device number varies by height of column, and they can implements. So, actually what are the processing they are doing or the, what are the function that eight 9 by 9 multipliers. So, they can implement, multipliers four 18 by 18 multipliers, one 36 by 36 multipliers. So, these are three different types of multipliers, see the numbers are also different and that can be implemented.

These contains adder, subtractor or accumulator and then register inputs can become shift register. So, this can be normally for made digital signal processing we need the multiplication, addition, subtraction and for that purposes as if the building blocks are

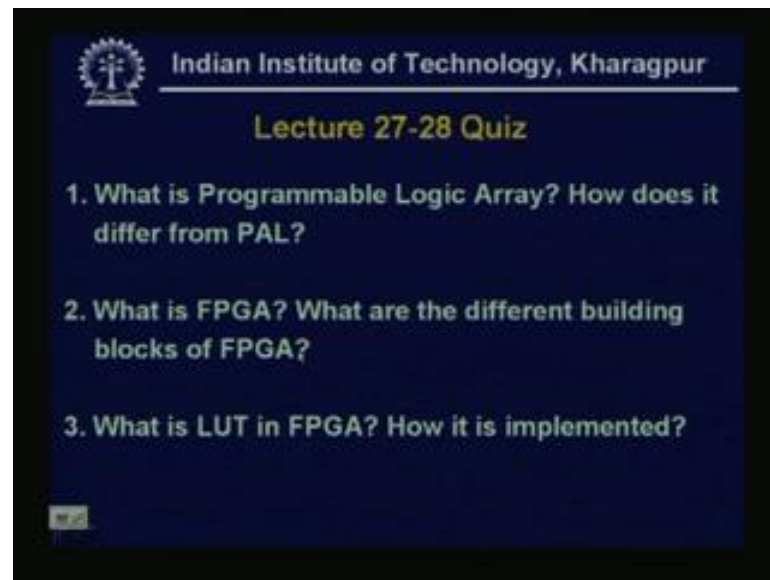
kept. And see here, these are actually configurable reconfigurable, so the different type of multipliers can also be implemented.

(Refer Slide Time: 55:01)



Now, this is another example of which has been taken from Altera family, that embedded DSP blocks, now if we see that, what are the different building blocks kept here. Say, see there are four 18 by 18 multipliers, just now we have seen, say these are adder subtractor accumulator and these are some summation circuits. So, this is overall embedded DSP blocks, again this can be treated as a basic building block for DSP type of FPGA. So, not only that multipliers or the adders are the building blocks, this is a together this can be again, so actually the design is a hierarchical.

(Refer Slide Time: 56:04)



So, we have seen the different type of programmable logic devices, today we have discuss the FPGA which is the state of the OR technique for VLSIs design and earlier we have read the basic things PLA, PAL etc. So, these are the lecture 27 and 28 quizzes, so mainly the, what is programmable logic array whether, how does it differ from PAL, and then, what is FPGA and the different building blocks of FPGA.

And another thing is that, how the programmable logic blocks or the LUT look-up table in a FPGA is implemented and what is LUT. So, these are the quiz questions from the last two lectures and we end the discussion of programmable logic devices here.

Thank you.

Digital System Design
Prof. D.Roychoudhury
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur

Lecture - 29
Design of Arithmetic Circuits

So, today's lecture is on design of arithmetic circuits. In this class we will read that how we can design the different type of adders, multipliers and other arithmetic units.

(Refer Slide Time: 57:41)

Indian Institute of Technology, Kharagpur

1-bit Adder

- Four possible operations:
 - $0+0=0$ ✓
 - $0+1=1$ ✓
 - $1+0=1$ ✓
 - $1+1=10$ ✓
- Require to output **sum** and **carry**.

Handwritten notes on the slide include: "Carry 0/1", "sum of (1+1)", and "Carry = 1".

Block diagram: An 'Adder' block with two inputs and two outputs labeled 'Sum' and 'Carry'.

As the adder is the most important and the mostly used digital circuits in the, in a digital system. So, first we will see that the different type of adder design and its performances and which one is suited for what type of design etc. So, first we start discussion on the adder design, as all of us know that, what is the truth table of 1 bit adder, so if we know that if it is a 1 bit mean, the bit can be either 0 or 1.

So, there are different of four possibilities, that either 0 is can be added with 0, 0 plus 0 is 0, 0 plus 1 is 1, 1 plus 0 is 1, in all these three cases, the carry is 0. Now, 1 plus 1 is 1 0, so if we see that binary, this is 1 1 1 and actually 1 0.