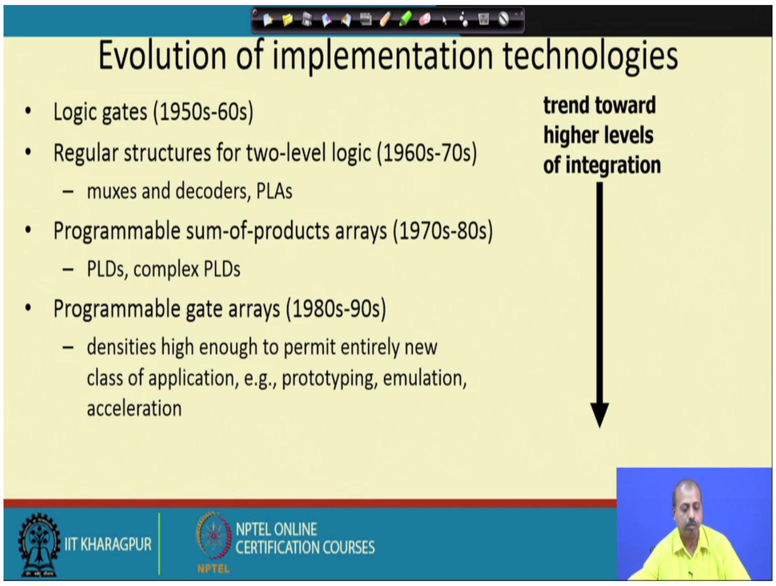
Digital Circuits
Prof. Santanu Chattopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture – 44
FPGA

The family of logic devices next we will look into another very prominent structure that you will find in today's digital design field, which is known as field programmable gate array or in short FPGA. So, this as the name suggests so it is field programmable means the programming of the device can be done at the customer end. So, just like we can program a PROM that is programmable read only memory or say PLA, so here also we can do the programming. And the programming can be done at your end that that you can just buy the device and connect some you have to have some sort of burner and through that burnerso the control program can be loaded on to this FPGA chip. And then the FPGA chip will be customized to your particular application.

So, it does not need detailed semiconductor manufacturing phases, so the last phase that is the final connection of the module so that can be done at the customer point. So, that drastically reduces the design turnaround time, so that is why it has become very popular. And also it is many cases, so it is reprogrammable so you can change the design. So, if there is an error so you can very easily correct it or if there is some modification or upgradation of the design. So, those things can be carried out very easily.

(Refer Slide Time: 01:46)



Evolution of implementation technologies

- Logic gates (1950s-60s)
- Regular structures for two-level logic (1960s-70s)
  - muxes and decoders, PLAs
- Programmable sum-of-products arrays (1970s-80s)
  - PLDs, complex PLDs
- Programmable gate arrays (1980s-90s)
  - densities high enough to permit entirely new class of application, e.g., prototyping, emulation, acceleration

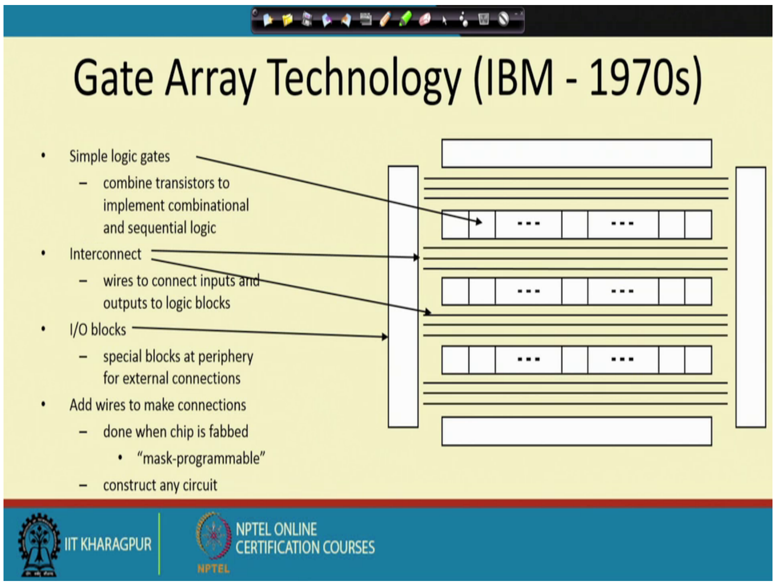
trend toward higher levels of integration

So, if we look into the evolution of this implementation technologies, so started with logic gates in 1950 and 60 then there was regular structures for 2 level logic gate 1960 70. So, 2 level logic means so you have got an and level and an or level, so this to a PLA type of structures then from there it evolved to more complex programmable devices which are known as PLDs programmable logic devices or CPLDs complex programmable logic devices that was 1970 and 80.

Then came the programmable gate array so programmable gate arrays means you have got an array of gates of similar type and then you can just connect them in some fashion to get the overall functionality implemented. So, in programmable gate array so density is high enough to permit entirely new class of application for example prototyping emulation acceleration. So entire design entire hardware design you can prototype on to this programmable gate arrays and you can get the thing done. So, you can have a basic idea about whether your design is going to work or not by doing a proper simulation or emulation of the system.

So, if you as you go from this logic gates towards this lower side programmable gate arrays. So, these higher levels of integration can be seen so more and more number of more and more number of gates can be clapped on to one device, so you can as a result you can you can implement more complex designs. So, if we just look into the gate array, so that was gate array so it was.

(Refer Slide Time: 03:43)



Gate Array Technology (IBM - 1970s)

- Simple logic gates
  - combine transistors to implement combinational and sequential logic
- Interconnect
  - wires to connect inputs and outputs to logic blocks
- I/O blocks
  - special blocks at periphery for external connections
- Add wires to make connections
  - done when chip is fabbed
    - "mask-programmable"
  - construct any circuit

So, this is gate array technology so it came from IBM in 1970. So, the overall structure is similar to this, so we have got a say simple logic gates. So, like this so these are logic gates implemented in some layer and then there are some interconnects running in between.

So, this interconnect so they are running between the layers, so between the we can say between the rows of these logic elements and then these logic elements they are implementing some function. So, if you want to establish a connection from here to here say for example, so you have to make a connection like this and then at this point you have to make a connection like this.

So, what I mean is so maybe output of this module has to be fed to the input of this module, so you can use this interconnect to do the connection like this. So, this way this gate arrays can be combined through these interconnects to implement complex logic. Apart from that there are some I O blocks. So, from normally this I O comes from the outside world. So, for driving these logic gates you need some inputs and those inputs are given through this input block and outputs are taken from the output block and as so to have protection against over voltage or to have a higher drive current etc. This integral this I O blocks are designed in a much to be are design in such a fashion that they are capable of driving more load or they are capable of withstanding noise margin sufficiently, so that way this I O blocks are design.

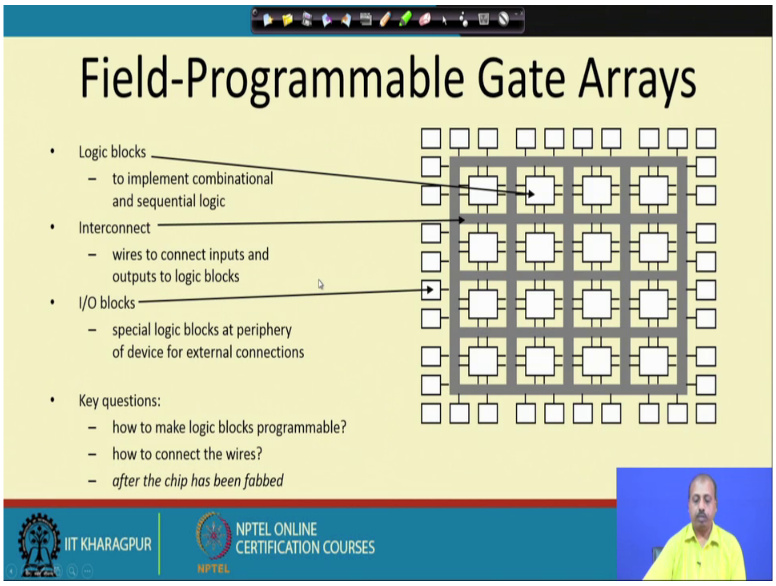
So, they are special blocks at the periphery our for external connections, then we can add wires to make connections. So, as I was showing there so you can have some extra wire connected and between these. So, these wires are these vertical wires are not shown here, but so there are connections from this and all those connections are programmable.

So, if you need to establish a connection then you need to burn out some of the points and as a result the connections will get established and the output of 1 block will be connected to input of another block. So, that way complex logic can be implemented. So, as a result so this is done when the chip is fabricated.

So, we can or the mask it is called mask programmable because, we can w put a mask for this type of connection and then we can do the connection or it may be. So, this is mask programmable, because this is not even, not yet in the customer side. So, this has to be done the final stage of connection has to be done at the fabrication house itself, but before the design has arrived to the fabrication house so the remaining part is fabricated.

So, only when the final connection have, when the design comes and the connections are determined, so the final metallization stage is carried out and it establishes the connection between the logic elements. So, it can be used for constructing any circuit.

(Refer Slide Time: 06:57)



Field-Programmable Gate Arrays

- Logic blocks
  - to implement combinational and sequential logic
- Interconnect
  - wires to connect inputs and outputs to logic blocks
- I/O blocks
  - special logic blocks at periphery of device for external connections
- Key questions:
  - how to make logic blocks programmable?
  - how to connect the wires?
  - *after the chip has been fabbed*

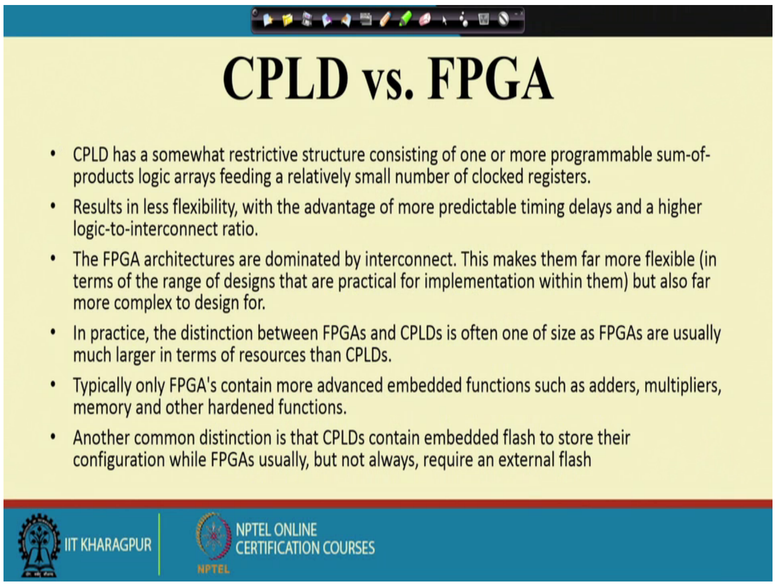
Next we will be looking into field programmable gate array, so in field programmable gate array so it is otherwise similar. So, we have got logic blocks to implement combinational and sequential logic. So, there are some logic gates and there are some flip flops which are part of this block logic block that can be used for implementing combinational and sequential logic. Then there are interconnects running both horizontally and vertically so these in the gray region.

So, they are for running interconnect and you can have some switches at this junction point, so these junction points are not shown explicitly. So, you can have some switch here and that switch will determine the connection from between the horizontal and vertical lines. So, we have got interconnects that wires to connect inputs and output logic blocks and the I O blocks for the input output operation. So, these are special logic blocks at periphery of device for external connection.

So, the question that we have is first of all how to make this logic blocks programmable. So, I said that you can there may be some sort some sort of logic elements there, but those logic elements may be programmed to have different combinational and sequential function implemented. So, how to make this logic blocks programmable, how to connect the wires like as I was telling that I need to connect the output of this logic block to the input of this logic block, so how do we establish the connection and naturally when we say it is field programmable.

So, this field means it has to be done after the chip has been given to the customer. So, at the customer side I we should be able to do this programming part. So, the design never goes to the fabrication house, so it is always design is with the customer only and then the getting on FPGA chip the customer should be able to program it, so that it gives the proper function.

(Refer Slide Time: 09:01)



CPLD vs. FPGA

- CPLD has a somewhat restrictive structure consisting of one or more programmable sum-of-products logic arrays feeding a relatively small number of clocked registers.
- Results in less flexibility, with the advantage of more predictable timing delays and a higher logic-to-interconnect ratio.
- The FPGA architectures are dominated by interconnect. This makes them far more flexible (in terms of the range of designs that are practical for implementation within them) but also far more complex to design for.
- In practice, the distinction between FPGAs and CPLDs is often one of size as FPGAs are usually much larger in terms of resources than CPLDs.
- Typically only FPGA's contain more advanced embedded functions such as adders, multipliers, memory and other hardened functions.
- Another common distinction is that CPLDs contain embedded flash to store their configuration while FPGAs usually, but not always, require an external flash

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, there are 2 types of logic devices that comes under this field programmable capability one is known as CPLD or complex programmable logic device. And another is called FPGA or field programmable gate array out of these 2 CPLD has a somewhat restrictive structure consisting of 1 or more programmable sum of products logic array feeding a relatively small number of clocked registers.

So, in CPLD the basic structure will be something like a 2 level structure. So, sum of product type of logic array will be there and or planes will be there and they will feed some number of clock flip flops so. And overall CPLD capacity wise it is less capacity is less compared to FPGA. So, as a result the CPLD it has got less flexibility with the advantage of more predictable timing delays and a higher logic to interconnect ratio.

So, it is less flexible, but at the same time you this timing delays that will occur is predictable. So, even if this logic is implemented in the programmable part, so the distance between those points or the lengths of the wires are predictable. So, as a result the timing delays are predictable and higher logic to interconnect ratio means you have a there are more number of logic elements compared to interconnect.

So, it is likely that we can push in more amount of more number of logic gates on to the CPLD structure. On the other hand these FPGA architectures they are dominated by interconnect. So, they have got large number of interconnects in them, so that you can establish connection between logic elements very easily logic elements are flip flops which is not the case for CPLD. So, this availability of interconnect now this makes them far more flexible in terms of the range of designs that are practical for implementation within them, but also for far more complex to design for.

So, that gives us the flexibility that you can have very wide range of designs implemented there, but at the same time it makes it very complex also. So you normally nobody does and FPGA design manually, so normally the cad tools are used for doing the design. In practice that distinction between FPGAs and CPLD it is often 1 of size as FPGAs are usually much higher in terms of resources than CPLDs. So, the total amount of logic elements that we have in FPGA total amount of interconnects that we have in FPGA, so they are much more compared to CPLDs. So, these FPGAs they are going to be much larger than CPLDs typically only FPGAs contain more advanced embedded functions such as adders multipliers memory and other hardened functions.

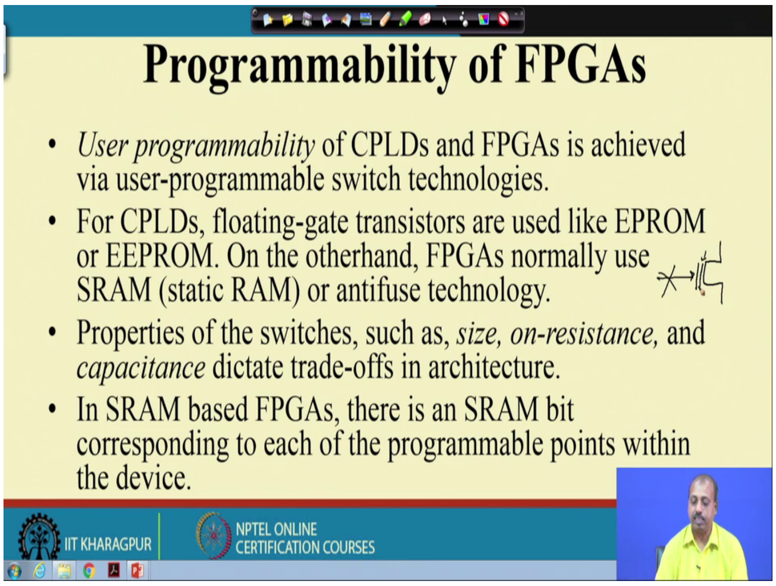
So, what I mean is that if you look into advanced FPGAs now, so you will find that there are built in adder available built in multipliers available and memories are also available because, if you are if you are trying to make a design. So, if you have trying to make a system so normally what we will need is apart from this logic element processing you will require some amount of memory and many a times we need this additions addition subtraction multiplication division type of operations.

So, they if we have built in modules for that, then the implementation will be much more efficient compared to the customer making the implementation by himself. So, that way FPGAs advanced FPGAs they will contain this adder multiplier memory, and other functions which are not available in case of CPLDs. Another common distinction is that CPLDs contain embedded flash to store their configuration while FPGAs usually, but not always required external flash.

So, CPLD devices they have got a flash memory within them, so as a result even if this power is even if it is switched off then thus the flash will contain the configuration program. So, when it is turned on so the CPLD functionality remains same; whereas, in case of FPGAs many a came in many FPGAs you will find that it will require an external flash. So, whenever you power on the FPGA chip or you give a reset so it will load the program from the external memory and then it will some external flash and they it will into an internal ram and then it will start operating in that fashion.

So, the point is that this embedded flash in CPLD that is good because, you do not have need this external flash; but at the same time it stops you from getting the reprogram ability. So, if I have got an external flash for this programming purpose, so I can just change the configuration program in the external flash. And we can get a new functionality realized by the same FPGA chip which is not true for CPLDs.

(Refer Slide Time: 14:08)



Programmability of FPGAs

- *User programmability* of CPLDs and FPGAs is achieved via user-programmable switch technologies.
- For CPLDs, floating-gate transistors are used like EPROM or EEPROM. On the otherhand, FPGAs normally use SRAM (static RAM) or antifuse technology.
- Properties of the switches, such as, *size, on-resistance,* and *capacitance* dictate trade-offs in architecture.
- In SRAM based FPGAs, there is an SRAM bit corresponding to each of the programmable points within the device.

So, if you look into this program programmability feature of FPGAs. So, user programmability of cp CPLDs and FPGAs achieved via user programmable switch technologies. So, both CPLD and FPGAs they use some sort of switch technology, for CPLDs we have got floating gate transistors like EPROM or EEPROM. So, floating gate transistors you know that it is these are transistors so if you apply a control voltage then the gate will appear.

So, as a result the communication device will be turned on and if you remove the voltage then this or if you apply some negative voltage or to are to remove that gate terminal. So, basically this is a transistor so this is. So, you know that this is a transistor. Now in this is a normal gate now there is another gate in case of CPLD and this floating gate thing. So, you have got another transistor another gate that will appear, so when we apply the voltage this additional gate appears as device gets turned on even if you remove this voltage now this gate will remain. So, as is the so this CPLD or FPGA using this protein gate technology so it will remain programmed in the same status.

Now, to remove this remove this floating gate you have to apply some other voltage and then this will be going out. So, these are this CPLDs they used normally this floating gate philosophy of this like EPROM or EEPROM FPGAs they normally use static ram or anti fuse technology. Of course, there are FPGAs that use a floating gate as well, but many of them use this static ram or this anti fuse technology.

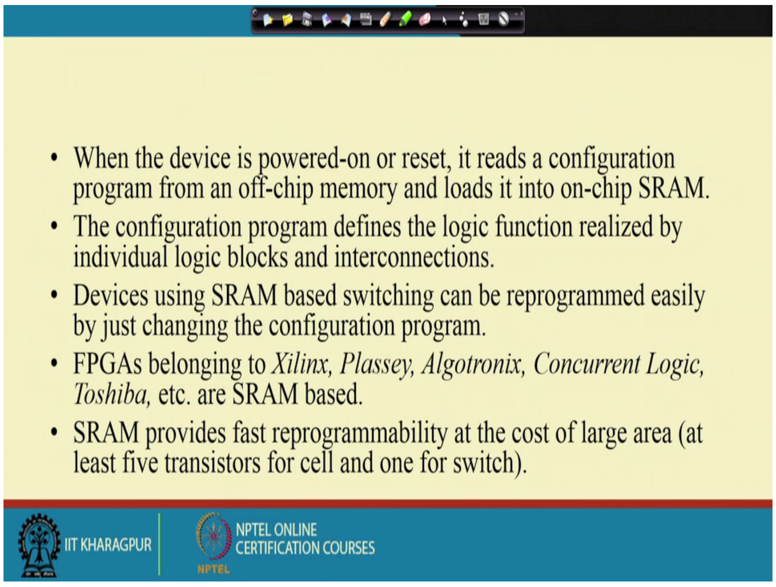
So, properties of the switches such as size on resistance and capacitance dictate tradeoff between the trade offs in the architecture. So, these switches they are coming as extra so we need them for programming the device, but they do not participate in forming the logic part of the of the of the design.

So, as a result they are overheads you can say so they are size on resistance size will determine the what is the amount of logic that you can put there on resistance will determine the delay in the communications array on resistance and capacitance they will detect the dictate the delay that you will have. In the design in SRAM based FPGAs there is an SRAM bit corresponding to each of the programmable points within the device.

So, device has got many programmable point like between to whenever we have got 2 interconnect, so those 2 interconnects are connected to each other or not, so that is controlled by 1 SRAM bit similarly. So, the programmable device or wherever you can think about programmability so it is implemented by 1 SRAM bit.

So, there are large number of SRAM bits and that actually constitutes the configuration program and let when this the FPGA SRAM based FPGA they are switched on, so we get this configuration program is loaded on to the FPGA. So, that all those programmable points are now programmed to have proper values and then the chip behaves accordingly. So we will look into this SRAM these programming techniques slightly later.

(Refer Slide Time: 17:39)



- When the device is powered-on or reset, it reads a configuration program from an off-chip memory and loads it into on-chip SRAM.
- The configuration program defines the logic function realized by individual logic blocks and interconnections.
- Devices using SRAM based switching can be reprogrammed easily by just changing the configuration program.
- FPGAs belonging to *Xilinx, Plassey, Algotronix, Concurrent Logic, Toshiba,* etc. are SRAM based.
- SRAM provides fast reprogrammability at the cost of large area (at least five transistors for cell and one for switch).

Now, so this is the thing that when the device is powered on or reset SRAM based FPGAs it reads configuration program from an off chip memory and loads it into on chip SRAM. So, the chip the FPGA chip will have got an SRAM which will have the configuration program.

So, when the chip is switched on or reset it is the configuration program will be loaded from off chip memory to the SRAM, the configuration program defines the logic function realized by individual logic blocks and interconnections. So, that will be determining the logic that is implemented by the logic blocks that we have and the interconnection, so that will determine the overall functionality of the FPGA chip.

Devices using SRAM based switching can be reprogrammed easily by just changing the configuration program, so that is very good, because if I normally what happens is that when you are in the early stage of you design so there will be designed bugs. So, if you go for directly chip fabrication then there is a possibility that many of those bugs are not detected at the time of design.

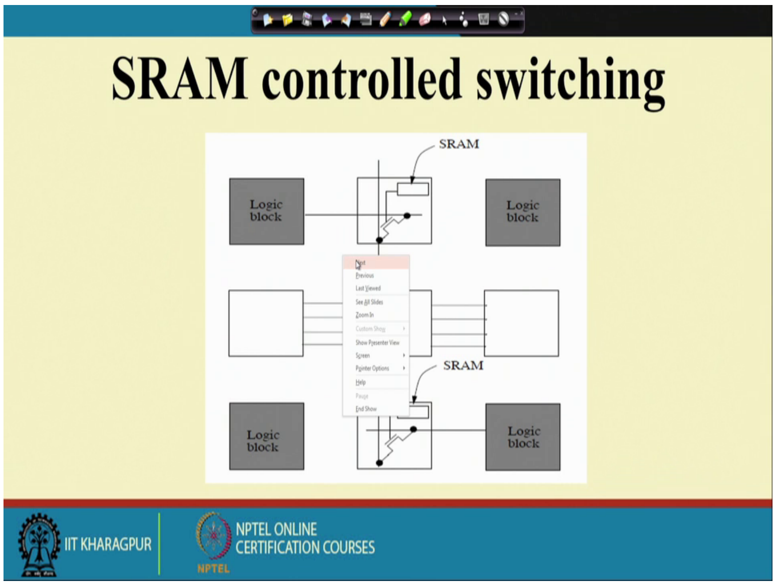
So, what we can do we can put this design onto an FPGA chip and we can check for the functionality whether the chip is working properly or not. If we find some bug there then nothing is lost because, I can just change the configuration program. So, I can correct my design as a result it will gives a new configuration program and that new configuration program can be loaded on to the again that SRAM based configuration memory and that gives us the corrected chip and also if you are going for some upgradation like you have got some functionality.

So, you want to add a new function then also these FPGAs are going to be helpful because, all that you need is to change the configuration program. Now FPGAs belonging to Xilinx Plassey Algotronix concurrent logic Toshiba, so they all use this SRAM based FPGAs SRAM based programming technologies. In fact, there are large numbers of vendors for this FPGAs. And these are some of them out of these Xilinx is the most popular because of it is capabilities as we will see later others are also there so they are also used in many applications, SRAM provides fast reprogram ability at the cost of large area.

So, reprogramme as I said the reprogram ability becomes very easy because, you can very easily change the configuration program. However, the area occupied is more because each SRAM based will require 5 transistors ok. So, that way if I have got say 10000 programmable points, so there will be 50000 transistors just needed for making the configuration memory so that makes it that makes it very costly anyway.

So, that is a trade off so if you are asking for flexibility we have to pay for the pay for that. And this is the payment that we have to do the amount of extra space that is needed extra area that is needed for the FPGAs.

(Refer Slide Time: 20:53)



SRAM controlled switching

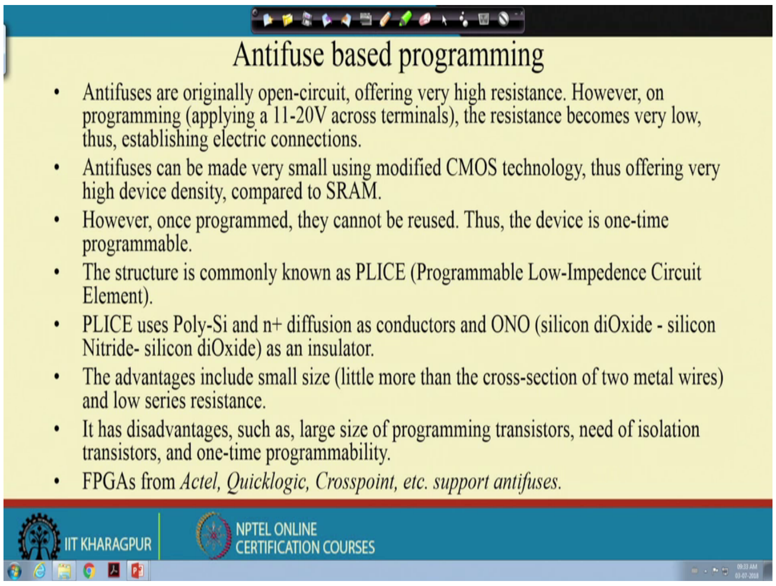
So, this diagram it shows how this SRAM can control the switching, so these are suppose some logic blocks within the FPGA and we want to establish a connection. So, output of this logic block should feed the input of this logic block.

So, how to do that so if these interconnect ok, so these interconnect comes to this is a switch box so this is a switch box. So, it has got this line and this line as 2 wire lines there and there is a programmable transistor here. So, this SRAM bit is going to control this transistor, so if this bit is equal to 1 in that case this transistor is on as a result a connection is established between this line and this line. So, with this the signal that is available that is given here will be available at this point.

Similarly, if this point is equal to 1 this SRAM bit is equal to 1. So, connection is established between this line and this line this transistor is turned on and these 2 lines are now shorted. So, as a result the output of this logic block will come to the input of this logic block so that will happen. And similarly whatever like here you see that there are 4 lines coming as input 4 lines going there out as output and similarly 4 more lines will come 4 more columns will come vertically and go out vertically.

So, between all these points so we have got this type of programmable transistors and by putting the proper SRAM bit equal to 1 so we can establish the connection. So, if we do not want the connection then the SRAM bit is 0. If we want the connection then the SRAM bit is equal to 1, this way we can use this controlled SRAM to control the switching.

(Refer Slide Time: 22:37)



Antifuse based programming

- Antifuses are originally open-circuit, offering very high resistance. However, on programming (applying a 11-20V across terminals), the resistance becomes very low, thus, establishing electric connections.
- Antifuses can be made very small using modified CMOS technology, thus offering very high device density, compared to SRAM.
- However, once programmed, they cannot be reused. Thus, the device is one-time programmable.
- The structure is commonly known as PLICE (Programmable Low-Impedence Circuit Element).
- PLICE uses Poly-Si and n+ diffusion as conductors and ONO (silicon diOxide - silicon Nitride- silicon diOxide) as an insulator.
- The advantages include small size (little more than the cross-section of two metal wires) and low series resistance.
- It has disadvantages, such as, large size of programming transistors, need of isolation transistors, and one-time programmability.
- FPGAs from *Actel, Quicklogic, Crosspoint, etc. support antifuses.*

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Major problem with this SRAM based control is that the area that you need ok. So, another type of programming technology that is popular with FPGAs is known as Antifuse based programming, so Antifuses they were or they are originally open circuit. So, it is like this so conceptually you can view it a like this, so these are the two points and in between we have got a Antifuse. So, this Antifuse on this is open normally it is open circuit and that as a result the resistance between these two points is very high resistance between these two points is very high.

Now, if we apply a programming voltage so 11 to 20 volt across the terminals then as if so this connection gets shortened ok. So, this resistance becomes very low and that establishes an electric connection and now even if you withdraw the voltage the programming voltage the connection remains ok. So, that that is how this is going to be useful. So, wherever you need the connections or programming so we have to apply proper voltage across the terminals. So, that the resistance becomes low the on resistance becomes low, so we have the connection gets established.

So, Antifuse is advantage that we have with Antifuse is that they can be made very small ok. So, they can be made very small using some modified CMOS technology thus offering very high device density compared to SRAM. So, SRAM SRAM is 6 transistor shall I said. So, compared to that these Antifuses can be made very small, so that is why you can accommodate large number of Antifuses within small chip area. So, that is why it is a very popular, but the difficulty with Antifuse is that once it is programmed they cannot be reused.

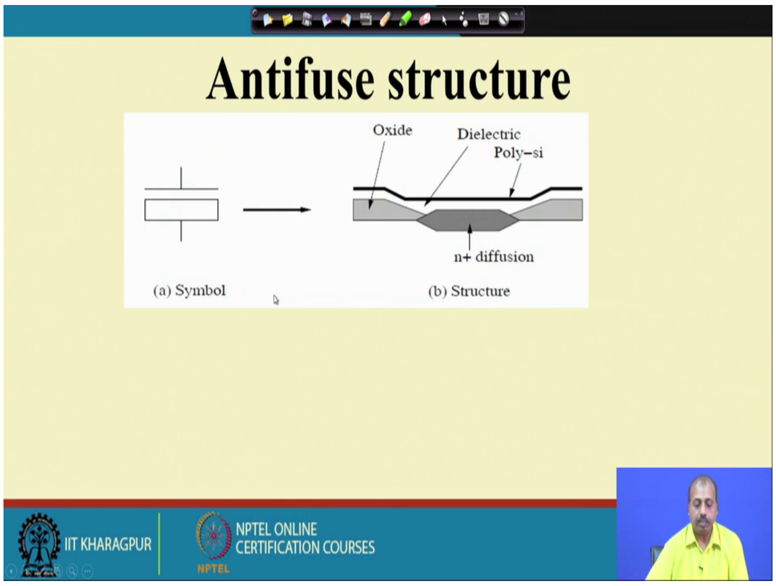
So, it is for 1 time use only so this cannot be reused so the device is 1 time programmable. So, we do not get the flexibility that I can use the same chip again and again and the other advantage that we had like say if there is a design error, so we can use we can correct the error and use the same FPGA chip for corrected design. So, that is not possible for Antifuse based FPGAs, but still they are popular because of their low cost and this size of the Antifuse being small.

So, this Antifuse structure so they are now commonly known as PLICE programmable low impedance circuit element so and this PLICE it uses as far as manufacturing is concerned it uses poly silicon and n plus diffusions as conductor and ONO that is silicon dioxide silicon nitride silicon dioxide as an insulator. So, the advantages that we have that includes small size little more than the cross section or 2 metal wires, so as that is why it is size is pretty small and it has got low series resistance. So, there is resistance of this Antifuses is very small when it is programmed at disadvantage.

So, large size of programming transistors so this 11 to 12 20 volt that I say it has to be applied. So, that volt has to be generated across the across those programming elements. So, so programming transistors become large and need of isolation transistor, so normally that 11 to 12 20 volt. So, if it comes across some other circuit element that circuit element may get damaged, so they need to be applied specifically to these programming terminals.

So, that way we need some isolation transistors and it is 1 time programmable, so that I have already said that it is you can program it only once. So, FPGAs from ACTEL quick logic cross point they support this Antifuses. So, these are some of these standard FPGAs that we have that support this Antifuse based programming.

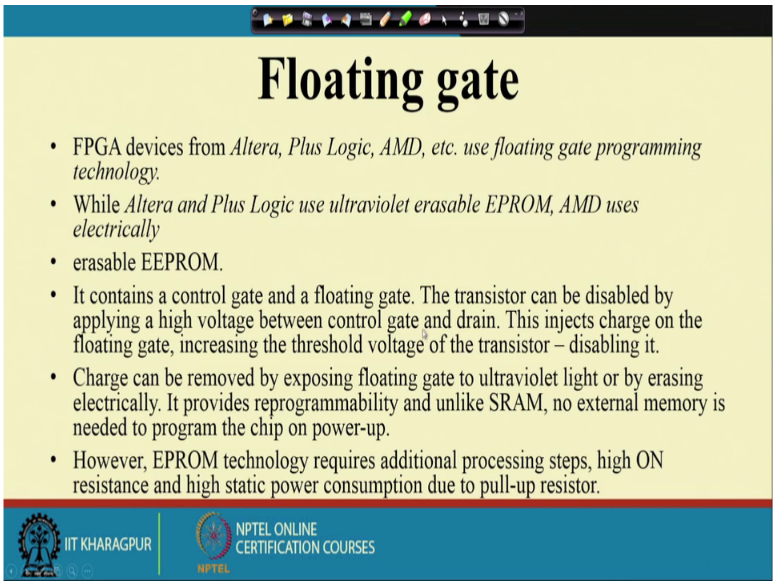
(Refer Slide Time: 26:56)


Antifuse structure

So, this is the antifuse structure so symbolically it is represented like this. So, where whereas, the structurally so you have got this oxide layer then there is a poly there is a n plus diffusion and this is another oxide layer and we have got this poly silicon layer.

So, this is the dielectric that we have, so this dielectric if we apply a potential then this dielectric will break it will establish a connection between this poly silicon and this n plus diffusion. So, that is how the connection gets established and we get this Antifuse based structure.

(Refer Slide Time: 27:33)


Floating gate

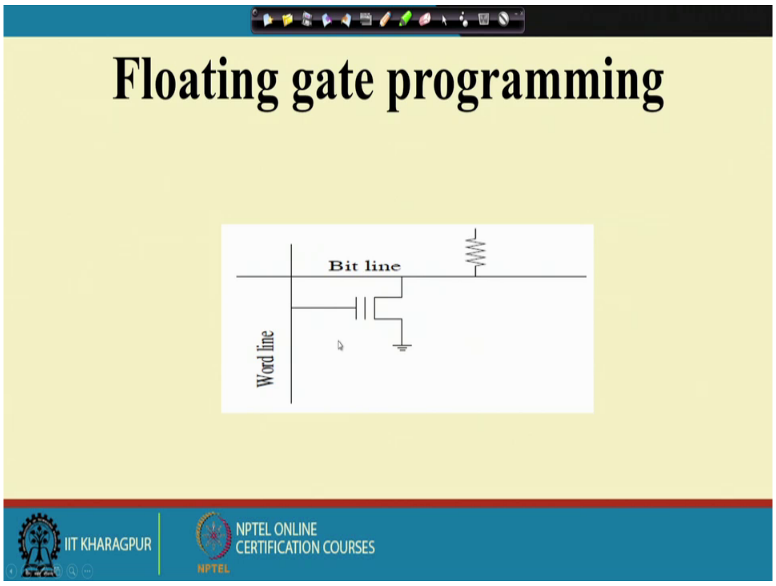
- FPGA devices from *Altera, Plus Logic, AMD, etc. use floating gate programming technology.*
- While *Altera and Plus Logic use ultraviolet erasable EPROM, AMD uses electrically*
- erasable EEPROM.
- It contains a control gate and a floating gate. The transistor can be disabled by applying a high voltage between control gate and drain. This injects charge on the floating gate, increasing the threshold voltage of the transistor – disabling it.
- Charge can be removed by exposing floating gate to ultraviolet light or by erasing electrically. It provides reprogrammability and unlike SRAM, no external memory is needed to program the chip on power-up.
- However, EPROM technology requires additional processing steps, high ON resistance and high static power consumption due to pull-up resistor.

Other programming technology that we have is the floating gate technology. So, FPGAs from Altera plus logic AMD etcetera they use this floating gate programming technology.

So, this Altera and plus logic they use ultraviolet erasable EPROM so and AMD uses electrically erasable EPROM. So, you know that there is EEPROM and EPROM. So, this Altera and plus logic they use EPROM technology and this AMD uses EEPROM technology. So, as I said that it contains a control gate and a floating gate a transistor can be disabled by applying a high voltage between the control gate, and drain these injects charge on the floating gate increasing the threshold voltage of the transistor and it will get disabled.

So, threshold voltage of the transistor if it is increased means to apply to turn on the transistor, so you have to apply more voltage.

(Refer Slide Time: 28:31)


Floating gate programming

So, this is the diagram so if you apply some voltage here then this floating gate appears, so as a result the threshold voltage of this transistor goes up. So, transistor becomes normally off sort of thing, whereas, if this is not there then the threshold voltage is low so then this you have got this transistor on.

So, this floating gate that we can insert and the charge can be removed by exposing the floating gate to ultraviolet light or by erasing it electrically it provides reprogram ability and unlike SRAM no external memory is needed for program to program the chip on power up. So, this is reprogrammable because, you can remove that floating gate by applying either ultraviolet light or this a high voltage, But unlike SRAM, so it does not need this off chip this reprogrammable off chip memory for hold the control program.

This EPROM technology it requires additional processing steps of course, the high on resistance and high static power consumption due to pull up resistor. So, this is the problem with EPROM technology, so that is there with the floating gate based this FPGAs. So, this is the floating gate programming I said so if you want to apply a particular if you apply a voltage then this gate will appear here and as a result this will be this threshold voltage will increase. Whereas, if you do not apply the voltage the threshold this floating gate will not be there, so threshold voltage will be low. So, the transistor will be treated as on so that way we can establish connection or not.