

Digital Systems Design with PLDs and FPGAs
Kuruvilla Varghese
Department of Electronic Systems Engineering
Indian Institute of Science – Bangalore

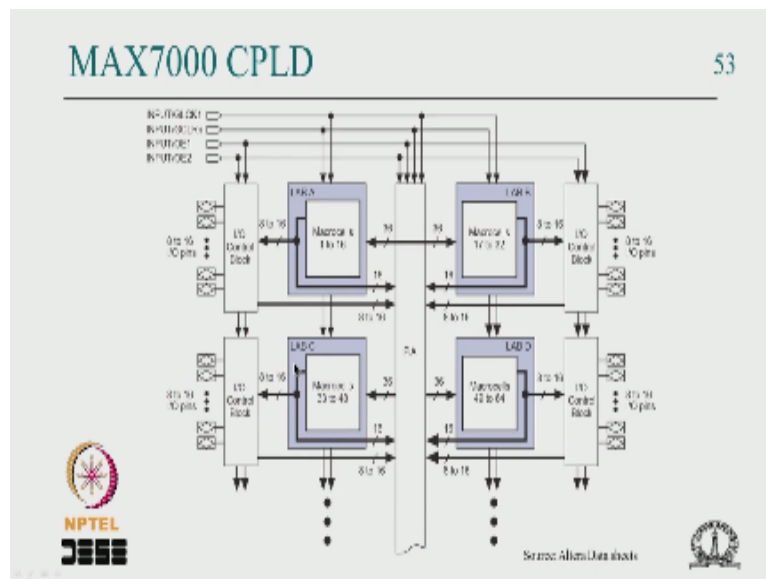
Lecture-35
FPGA Introduction

So welcome to just lecture on field programmable gate arrays and the course it is a design with PLDs and FPGAs. In the last lecture we have looked at the architecture of complex PLD essentially we have looked at the logic clock the interconnect switch architecture in the logic clock we have looked at the product term array, product term allocate, the macrocell consists of flip flops with certain configurable options and we have look at a given a VHDL code how it is synthesized and kind of place and fitted at synthesizer circuit within the CPLD ok.

And we also have seen the timing model of fit and the I/O block and the various applications and some kind of features of CPLDs and today we are going to look at much more popular class of devices called field Programmable Gate array and before that we just left the previous lecture slides, so that we can contrast the FPGAs of field Programmable Gate array is with this CPLDs complex Programmable Logic devices.

And then why the FPGAs are much more popular much more used than CPLD, we will be able to understand if you briefly look at the CPLD part and go to the FPGA, so let us turn to the slides of the last lecture.

(Refer Slide Time: 02:21)



So we had looked at Max7000 PLD which is from Altera as an example and we saw the CPLDs are kind of hierarchical PLD which consists of simple PLD blocks which is interconnected to a switch. So these are 16 macrocell that means kind of and/or with Programmable flip flop such 16 outputs.

And you know here it is shown 4 blocks and there is a interconnection and the AND gates input goes from the output of this few switch like you can see 36 lines goes there. So you can imagine 36 vertical lines and its complement and there are 16 output that means 16 flip flop sections are there that can be connected to the 16 IO box or can be feedback into the switch as input ok.

So also you have I/O block those pins can act as input as well as output input pins can come as input to the switch. So all these you know that as output feedback or the input of 4 section come as a input and output of this which goes as a input to the AND gate input ok. So essentially will take 4 you know 4 kind of blocks like this 4 macro 4, macrocell section of 4 logical block.

Then what you have as a switch size is you know see here 32×4 that is 128 inputs and 36×144 output ok, it is a huge switch and that is one disadvantage the PLD. So you cannot do a large you know very large kind of device where lot of macro blocks are there or macrocell are there and we also have seen their clocks which is going to flip flop it is not going there we can use input.

Similarly output unable to the enable the tri state you know dedicated output enabled or if it is not being used, it can be used as input.

(Refer Slide Time: 04:53)

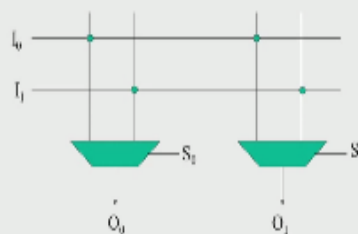
- Cross bar switch satisfying the connectivity requirements discussed before.
- $N \times N$ cross bar can be implemented using, N , N -to-1 Multiplexers.
- Interconnection between blocks using just one switch.
- Simple timing, fast.
- Cross bar don't scale well



So we have seen that this is a cross bar and crossbar is nothing.

(Refer Slide Time: 04:56)

2 x 2 Crossbar



- $N \times N$ crossbar requires N , N to 1 Multiplexers



But a 2×2 crossbar will have to 2 to 1 multiplexer ah to choose any of the input to output, so you can imagine in N/N that requires N , N to 1 multiplexers, so it is very huge in area and it cannot scale very much and there is 1 issue which the field Programmable Gate array shows ok.

(Refer Slide Time: 05:23)

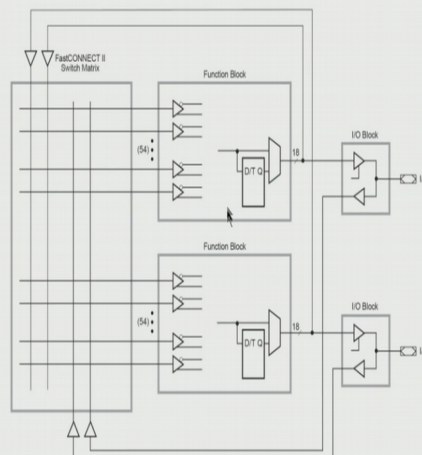
Table 1. MAX 7000 Device Features

Feature	EPM7032	EPM7064	EPM7096	EPM7128E	EPM7160E	EPM7192E	EPM7256E
Usable gates	600	1,250	1,800	2,500	3,200	3,750	5,000
Macrocells	32	64	96	128	160	192	256
Logic array blocks	2	4	6	8	10	12	16
Maximum user I/O pins	36	68	76	100	104	124	164
t_{PD} (ns)	6	6	7.5	7.5	10	12	12
t_{SU} (ns)	5	5	6	6	7	7	7
t_{FSU} (ns)	2.5	2.5	3	3	3	3	3
t_{CO1} (ns)	4	4	4.5	4.5	5	6	6
f_{MAX} (MHz)	151.5	151.5	125.0	125.0	100.0	90.9	90.9



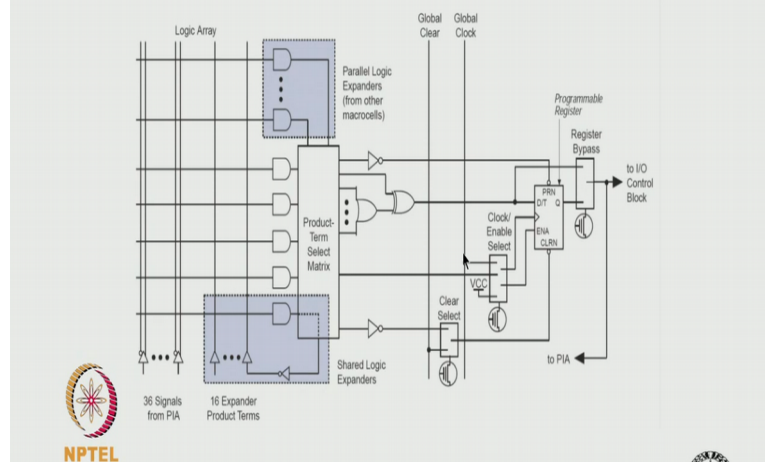
And we had a look at ok.

(Refer Slide Time: 05:24)



This is that which shown for clarity, we have seen that the feedback thing has coming as input and the input itself is the I/O pin input is coming as input to the AND gate and the outputs are going to the logic clock as input the AND gate ok.

(Refer Slide Time: 05:46)

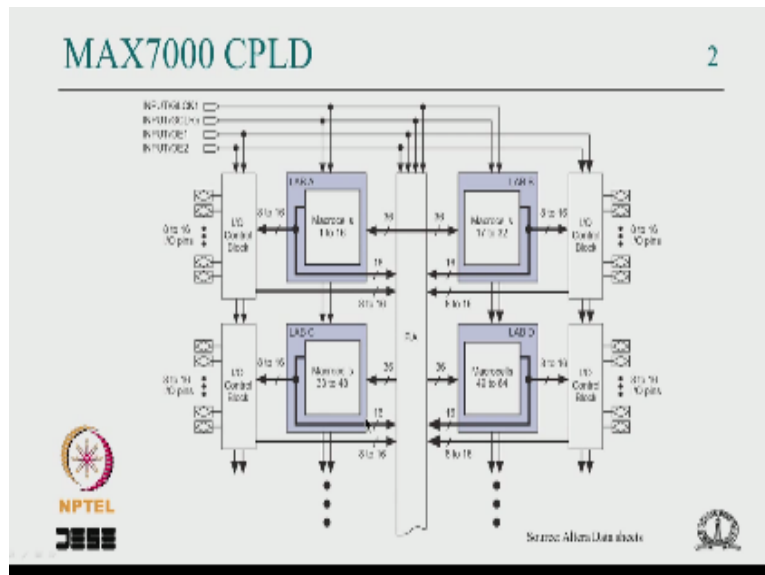


And these are the these was a kind of architecture we have in this Max7000 5 AND gates which can be connected to OR gate, XOR gate as a product term clock, clock enable or the you know synchronous reset and so on ok and one of the AND gate is feedback into the array so there are 16 section, 16 AND gates has expanded terms which can be used to cascade ok.

And in one section and it is not use that can be used by the next section and so on ok. And once again the problem remains here this AND gates are very wide there are lot of inputs, you know there are 36 variables and its complement can be connected to AND gate which is very rarely needed in you know real life cases.

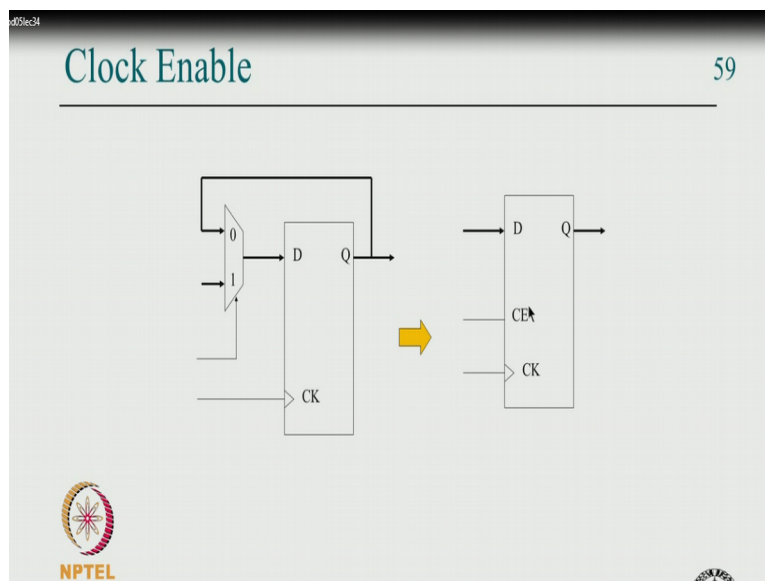
So such a structure will you know make you know 36 signal even make the switches the Programmable interconnected array or crossbar very big which does not make it scalable. So these two things are the one which is sorted out in FPGAs where in you do not have such very wide product terms and you do not have a huge single crossbar you know instead of having a central switch which is distributed.

(Refer Slide Time: 07:21)



You know across the device ok we will see the architectural, so if you look at the this particular CPLD you can see the maximum number of the logic clocks in this device of 16, but if you start more than that then the huge area will be occupied by this crossbar and so on, so that is addressed in FPGA.

(Refer Slide Time: 07:52)



And we have seen the flip flop as a cross you know re-circulating much built-in if you can you can use 1 kind of clock enable one level of clock enable for controlling the registers.

(Refer Slide Time: 08:06)

- 5 PT's per Macrocell
- D/T Flip-flop. Flip flop can be bypassed for combinational output.
- XOR Gate: Polarity control, PT optimization, Comparator, Arithmetic circuits, Parity
- PT set, PT Reset,
- Global Clock, PT Clock

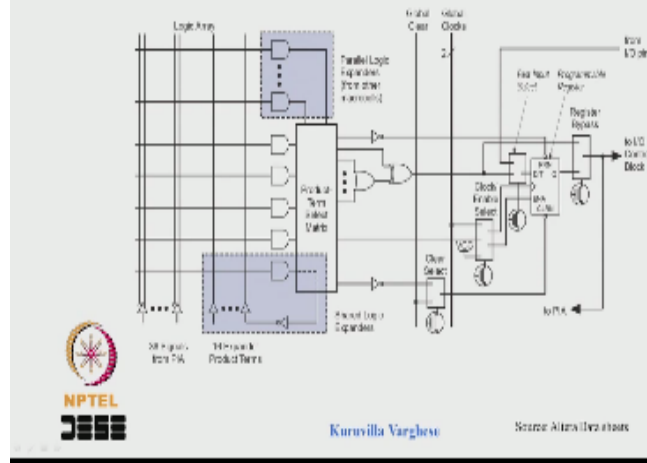
PT Clock enable



And just we have looked at it and we said some devices some current some devices allowed direct connection from the I/O pin directly to flip flop which can be used for synchronisation, we have seen the metastability when there are multiple clock domain one would like to synchronise input signal, so this can be used for fast synchronisation. Otherwise just come to the crossbar coming here go through all the way for synchronizing which will make the setup time very huge ok.

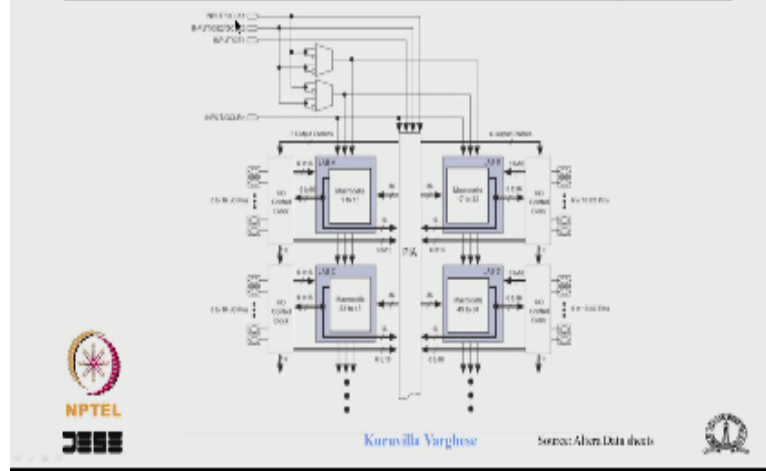
(Refer Slide Time: 08:44)

MAX7000S Macrocell Fast Input



And many signals may not satisfy that setup time, so that is avoided by this particular fast input which is shown in the picture like this.

(Refer Slide Time: 08:56)



You know this MAX700S and we have seen an example of VHDL code.

(Refer Slide Time: 09:01)

Fitting: VHDL Code 65

```

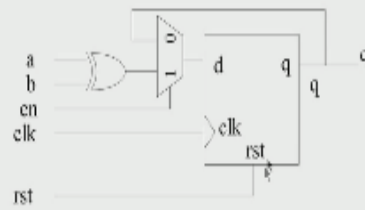
process (clk, rst)
begin
if (rst = '1') then q <= '0';
elsif (clk'event and clk = '1') then
  if (en = '1') then
    q <= a xor b;
  end if;
end if;
end process;

```

NPTEL JEE Kuravilla Varghese

How it is synthesized.

(Refer Slide Time: 09:02)

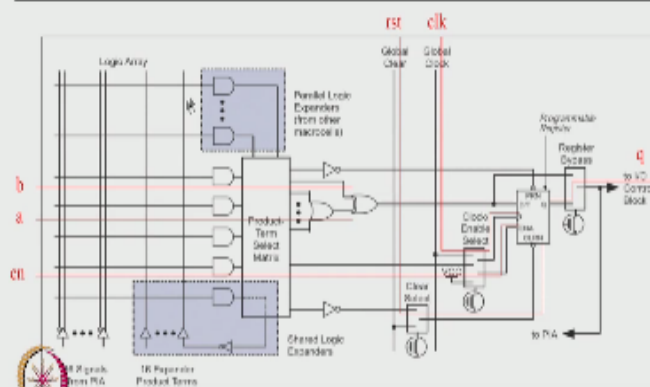


Kunivilla Varghese



And how it is really map into the MAX7000 CPLD.

(Refer Slide Time: 09:10)



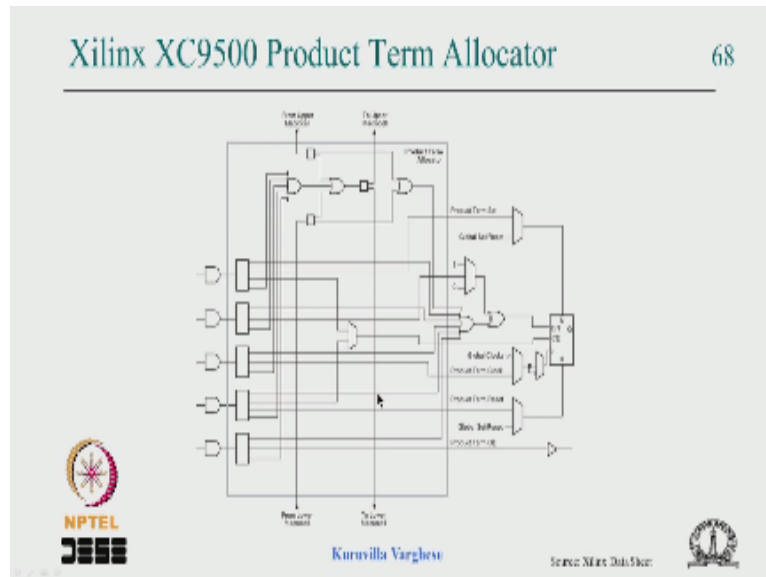
Kunivilla Varghese

Source: Altera DevTools



And this shows the product term allocator of XC9500.

(Refer Slide Time: 09:16)



We can see that how a single AND gate is connected to OR, XOR, or the clock or reset and so on and this shows that the product term steering part if it is not used by the top or bottom macrocell or this can be combined with the previous one and can be used and so on okay.

(Refer Slide Time: 09:36)

Product Term Allocator

69

- Demultiplexers let the PT's to be allocated for various functions like XOR input, PT clock, PT clock enable, PT Set, PT Reset and PT Steering.
- 5 input OR gate with associated steering circuit allows the unused PT's in one section (combined with ones from section above/below) to be steered to OR gate of the section above or below.



Kuruvilla Varghese

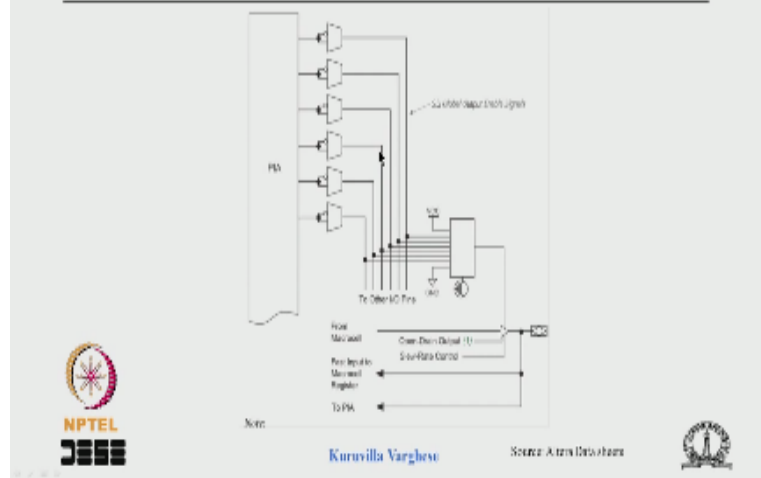


So lot of possibilities there.

(Refer Slide Time: 09:42)

MAX7000S I/O Control Block

70

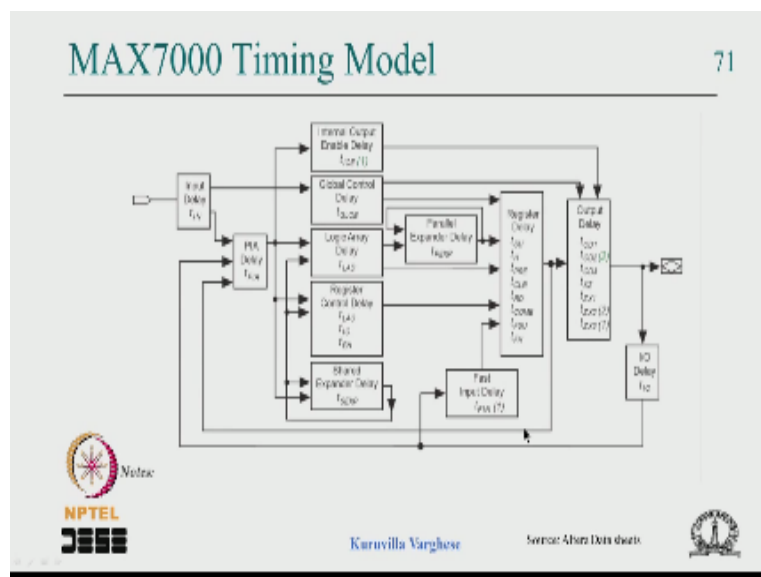


And this shows I/O configuration you have tri state gate enable which can be permanently enable disable or you can use various signal from the crossbar to choose to enable that okay. So that add to the flexibility.

(Refer Slide Time: 09:56)

MAX7000 Timing Model

71



And we have seen the timing model which is very simple because between any logic there is only 1, 1 crossbar connection and this all are specific to the macrocell ok. So it is the timing analysis symbol and the end to end time is quite fast compared to probably in FPGA , but this device since it is not very popularly used it may not be fabricated latest device technology that extent it may be slow like if you have a 22 NM technology is the latest CPLD.



Latest FPGA will be built on that but the CPLD may not be built on that because a volume same may not be that high, so that is why I did not built on that.

(Refer Slide Time: 10:56)

CPLD vs FPGA 72

Features	PLD	FPGA
Logic	AND-OR	Mux / LUT / Gates
Register to Logic ratio	Small	Large
Timing	Simple	Complex
Architecture Variation	Small	Large
Programming Technology	Flash	Anti-Fuse, SRAM
Capacity	10 K	Few Million Logic Cells + Few MB RAM

Karruvilla Varghese



So we have seen the features of that basically AND/OR very wide AND/OR number of registers are small or timing is symbol architecture variation is more other the programming Technologies cash in the capacity is around 10 k. So what we're going to the FPGA, what FPGA addresses a is this Y decoding is removed a complex crossbar connection is distributed.

And one more thing is that it basically gives you lot of registers ok. So in a CPLD register the logic race is very small that means got huge AND gates number of AND gates are high but if you compared to the logic the amount of registers available in CPLD is less. So you cannot think of implementing some memory factors FIFO and RANZ and all that in a CPLD, but in FPGA.

(Refer Slide Time: 12:02)

- Small design comprising of counters, FSM, Small logic
- Examples
 - Memory controllers (DRAM controller)
 - Bus protocol translation (CPU Bus -> PCI Bus)
 - Optical encoders.
 - Small control circuit in Instrumentation, Power Electronics for Data Acquisition control, Small digital control circuits.



So we have seen some application where lot of FSM counters random logic is needed the CPL we can be used but otherwise it is tough to use the lack of the memory effects these kind of application like communication, Signal Processing, Cryptography and orders affected by those kind of in a lack of resources.

(Refer Slide Time: 12:31)

- Design which requires lot of registers and memory and complex designs cannot be implemented.
 - Signal processing architectures (Filters)
 - Complex arithmetic circuits
 - Communication circuits (Packet processing, Modems)
 - CODEC's
 - Cryptographic circuits



But maybe symbol Cryptography can be implemented in CPLD depends how much what kind of algorithm your implementing and how much is the keyword turn so on ok. So basically it independence on that but then I am sure some small cryptographic circuits how can be implemented in CPLD.

So let us come back to the our topic of today's lecture a field Programmable Gate array. So as I said what it address is a 300000 or 3 features in the CPLD. The CPLD we have seen there is

a crossbar which is very huge which is a central resource which cannot be scaled. So in FPGA this particular point is attacked and the interconnection architecture is distributed across the chip, that chip can scale well you know if you look at the other number of load excel in FPGA run into millions the case of complexes FPGA, which allows you allow the designer to even emulate the ASIC in FPGA.

Sometime the complex SOC required multiple FPGAs to emulate or to implement but nevertheless this distributed interconnection architecture is the one which enables the FPGAs to become complex in terms of the logic resources and memory resources ok and the second thing is that we are seen that in CPLD there are the product comes which is very wide which is not required in practice or real life.

So in FPGA you have the logic resources for the combination logic with smaller input, so that the area is not waste okay the resources are not wasted. The third thing the CPLDs suffers from lack of registers which is address very much in FPGA in fact the FPGA balance is a combinational logic with flip flop, so each logic clock will have kind of balance between the combinational logic and flip flop ok.

If you look back in CPLD maybe there are 5 product terms very wide product terms and 1 flip flop but here you will have a much more logic in terms of the inputs ok. There it was kind of 36 input, but in FPGA there may be 4 input or 5 input or 6 input, 7, 8 like that. Now I can say with all the combinations and all the variation is it start with number 4 and in a big logic blocks up to the 8 inputs that what is the FPGA targets.

And there are lot of flip flops ok and many times these logic combinational logic is combined say you start with 4 input combination circuit then you combine multiple of them to make 5, 6, 7 and so on ok. So you get lot of flip flops because if you go for you combine them till you have the flip flops that logical sources left out of their logic lot of flip flops available in FPGA .

So the register to logic ratio is very high in FPGA which allows you to implement a lot of communication, lot of given network functions and signal passing box and so on ok. So let us go to the slides of this field Programmable Gate array that is all almost we are coming to the end of the course, so let us see this.

(Refer Slide Time: 16:50)

The slide is titled "Topics" in the top left corner and has the number "25" in the top right corner. Below the title is a horizontal line. A bulleted list of topics is centered on the slide:

- FPGA Architecture (Xilinx, Altera, Actel)
- FPGA related Design issues
- FPGA related Timing issues
- Tool Flow
- FPGA Configuration
- SoPC
- Debugging
- Case Study

At the bottom of the slide, there are three logos: the NPTEL logo on the left, the JEE logo in the center, and a circular logo on the right. The name "Kurusilla Varghese" is written in small text at the bottom center.

So we are going to cover the topics we are going to cover in a kind of from a very broader perspective is that we will look at the FPGA architecture of Xilinx remain the I will concentrate on this Xilinx architecture and I will just show some Altera very briefly Altera and Actel to contrast and I will discuss FPGA to related design issues, FPGA related timing issues, the tools flow, how to configure FPGA, little bit about system on Programmable chip or Programmable So PC whatever various device and it call it so.

You can call it P shock which is the program system on chip that means the complete system like SOC built on an FPGA fabric, how to debug the FPGA design from case studies and so on ok. So that is the topic the case study is definitely need not be very much link to the FPGA, yes it is a link to the FPGA and the sense that we are going to implement that enough.

But this will kind of wind up and put everything together we have learnt till now the VHDL, the digital design advance digital design, the FPGA basically and all that will be tied together in the case study we are going to handle.

(Refer Slide Time: 18:21)

- ASIC, MPGA/Standard Cell, FPGA
- Volumes, NRE cost, Turn around time
- Array of logic resources with programmable interconnection.
- Logic resources (Combinational, Flip flops)
- Combinational: LUT, Multiplexers, Gates
- Programmable interconnections: SRAM, Flash, Anti-fuse
- Special Resources: PLL/DLL, RAMs, FIFOs, Memory Controllers, Network Interfaces, Processors



So the beginning of the course we have discuss the evolution of this devices ok. We have discuss the evolution of this PLDs in greater detail, but since we have discussed this at the beginning of the course I will just briefly mention that ASIC is something which is built from the scratch okay. The designers might use some kind of standard libraries to implement this basic libraries .

But more or less it is a huge kind of time it takes to cost which is called non recurring engineering cause, that means there is a one-time design cost a lot of idiot tools are required lot of the silence time is required and one need to set up foundry line to fabricate this all that is a onetime cause which has to be amortized once the ASIC you know it will goes up, because it is an electrochemical process.

And the beginning number of devices out of 100 will be maybe only 80% then as the processes tune you get 95, 98, 99 then you know it makes sense to make an ASIC. So that trouble with ASIC is that since more or less it is implemented from scratch, design from scratch, this cost NRE cost will make it viable only for huge volumes and it takes a one, one and half years from starting to end for a reasonable complexity ASIC to come to the production ok.

So this is 1 feature of ASIC, so there is a middle ground where these cells used are fixed by the foundry and they will give you libraries and you pick up the library and interconnect them, so only the interconnection is the unknown fact about these, the mask everything for

the standard architecture standard blocks are available with the foundry, other what designer once end of the design what goes is basically the interconnection of the standard blocks okay.

So it is quicker but the NRE cost is kind of medium or as ASIC so this will work for the medium volume, maybe in a lesser time it can design working with standard block and the performance will suffer because it may not be very much optimise at a complete system level. So there are design code mask Programmable Gate array, the array logic resources where in you know it is interconnection is programmed at the foundry.

So we can say, so this works for medium volume and we can say the field Programmable Gate array is a field Programmable version of the foundry version of this gate array, this mass programmable gate array of standard cell the it is interconnection is kind of fabricated in the foundry, but here the interconnection is configured in the field okay, in a lab or in your workplace. So it is called field Programmable Gate array.

When you say filed it is not the electric field is it means the feel your working ok like the lab or something like that ok. If somebody is confused with that particular work ok definitely due to devices where transistor the field is kind of somehow involve with the programming but that is not the name you know men for basically said that is field Programmable Gate array.

So it involves then the array of some logic resources the interconnection with programmability which can be program in the field ok. So that is what is FPGAs about basically it is an array or matrix of logic resources with Programmable interconnection ok. So you have array of them ok, not like in a CPLD you can say array because it is the width is very less you know you have 2/2 or like 2 in the width and 4 in the depth and so on 4 or column wise and 2 row wise and so on.

But here which use array of logic resources with Programmable interconnection and if you look at the logic resources we need we know that we have to ultimately do some computation using data path which consists of combinational circuit and flip flops and the data path we are controlling by the finite state machine which also required combinational circuit for next state logic output logic and all that and flip flops.

So we can say the logical resources in any reasonable design was combination circuit and flip flop be data path or the state machine of controllers. So that is provided in the logical resources of FPGA in particular if you look at the combinational circuit the normal combination lookup table the maximum used is the lookup table. We will see what is lookup table.

We have discussed in a way in the case of CPLD in the evolution of CPLD, but we will have no kind of a review on that and some FPGAs is multiplexers as a combinational logic and you know that a 2 to 1 multiplexer if the select line connected to A then you have A bar and A is available or if you have a 4 to 1 multiplexer you have 2 select line to the select line A, B and depending on the input you get the output, over the output.

So I have multiplexers AND or so you form the minterm of the select lines available the minterms of that and the Programmable OR is achieved by connecting the various inputs as 0, 0, 1 wherever there is one that minterm is a part of the output you know output variable that is the multiplexer of course there are some FPGAs with use gates for implementing the combination logic.

So that is about the combinational logic and if you look at the Programmable interconnection these are the technologies used SRAM that is static RAM and it is a bit of an unfortunate mean though at a kind of top level the name is correct but my kind of mislead you in terms of the technology used for interconnection the flash transistor and how many more and the fuse okay and/or you can for the array it is Anti-fuse ok .

So here the Anti-fuse is a kind of opposite of a fuse you know that the fuse you normally pass a current to blow it but Anti-fuse is something does not have a connection by applying a voltage you make a connection okay. So that is Anti-fuse and FBPGAs as special resources like PLL which is phase lock loop, DLL delay lock loop which is not used nowadays mainly all the FPGAs use PLL phase locked loop which has advantage over DLL.

Maybe we will briefly look at it though there is no reason because the current FPGA use DLL then the memories RAMS the FIFOs the same RAM can be used as FIFO with additional logic memory controller sometime high and FPGAs need since it has lot of computation you know possibilities or option of implementing complex computation it needs lot of memory

and the DRAMs are used DDR3 can be in the face and in that case DDR3 controller implemented in FBGA resource maybe slow.

So there are hard kind of controllers in the fabric network in the phrases like the the kind of physical interface or the data link of the Ethernet and so on ok or at least the blocks the used in the prominent network interfaces, sometimes the Complex FPGA has processes built-in hard core processors built-in. So that you can implement a complete SOC solution within an FPGA ok.

So that is a kind of not self introduction to the FPGA its evolution what it contains it has evolved from basic standard cell or MPGA and FPGA. It consists of array logical resources with interconnect Programmable interconnections, the logical resources basically is combination circuit and flip flops and the combination circuit used it FPGA array from static RAM, so we lookup table then the multiplexers and gate.

The Programmable interconnection mainly uses SRAM technology flash transistors and the Anti-fuse technology and the special resources like DLL memories network interfaces memory controllers and so ok. So that is the FPGA then not sell, so let us look at the slide.

(Refer Slide Time: 29:26)

The slide is titled "Commercial FPGA's" and is numbered "27". It lists two main manufacturers: Xilinx and Altera. Under Xilinx, the products listed are Spartan-3, Spartan-6, Virtex-4, Virtex-5, Virtex-6, Artix-7, Kintex-7, Virtex-7, and Zynq. Under Altera, the products listed are Cyclone, Cyclone II, Cyclone III, Cyclone IV, Cyclone V, Arria II, Arria V, Stratix II, Stratix III, Stratix IV, and Stratix V. The slide also features logos for NPTEL, JEE, and Kuruvilla Varghese.

So if you look at the commercial FPGAs these are the two major manufacturers as a market who has the majority of the market the Xilinx and Altera. Xilinx has a low-cost devices call Spartan series, currently they have Spartan 3 and Spartan 6 though it is low cost these are

kind of still a lot can be implemented in this Spartan 6 in a complete if you want a complete SOC can be built in medium complexity SOC can be built in Spartan 6.

It is low cost but that does not mean it is low complexity of low performance in a lack some inbuilt processor and hard course devices but there is still powerful and they have the complex FPGA, the previous generation was called Virtex-4, 5 and 6 and current generation as Artix, Kintex, and Virtex or kind of volt share kind of architecture but they have certain additional resources which is used for you know which is used for the you know I/O requirement or Signal Processing and so on okay.

The Xilinx also have a new devices call Zynq which is nothing but a fabric like Virtex 7 Plus Dual Core and processes of building ok that is not built into the fabric of FPGA, it is in the same Silicon chip, so it is hard ARM processor at very high speed. So a complete software hardware solution high performance software hardware solution can be implemented in FPGA.

When it comes to Altera and they have the cyclone kind of logos devices again that does not mean it is low performance low complexity, they have Arria series and Stratix series which is you know which is the complex version of the these kind of the devices, so and they have a soft core processors called Cyclone can be implemented in this device and Xilinx core like microblaze cyclone means the design is available in terms of the Verilog or VHDL code. There is something out Picoblaze with microcontroller a bit and the microblaze soft core risk 32 bit processor similarly all to 32 bit with soft core processor.

(Refer Slide Time: 32:32)

- Actel

- Axcelerator (Antifuse)
- IGLOO, IGLOOE (Flash)
- ProASIC Plus (Flash)
- ProASIC3, ProASIC3E (Flash)
- RTAX (Radiation Tolerant, Anti-fuse)
- RTSX -SU (Radiation Tolerant, Anti-fuse)
- Smart Fusion, Smart Fusion 2 (ARM Cortex – M3)

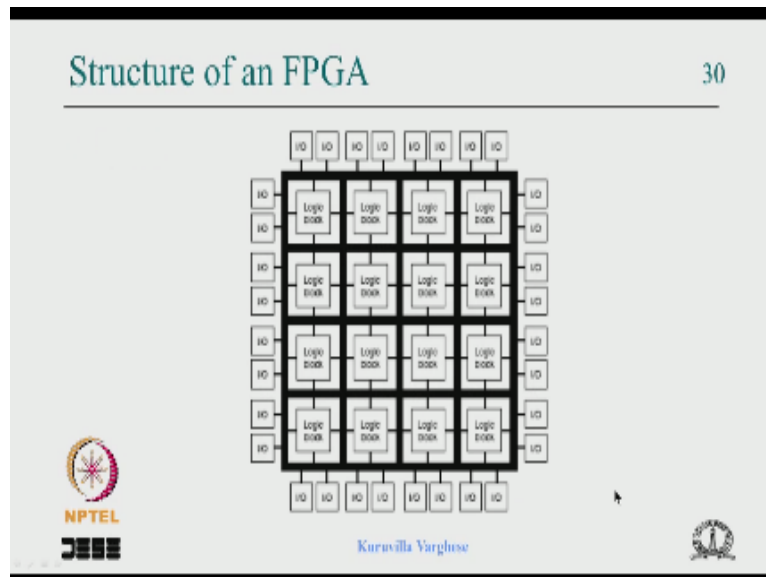


Actel as different all these are the static RAM based FPGAs and when it comes to Actel and the manufacturers who has devices like axcelerator and RTAX RTAX and Anti-fuse base and there are flash base PRoASIC you know smart with arm cortex M3 in a built-in hard core with this Actel. So Actel has certain line of Anti-fuse FPGAs which are quite useful in the space application because it kind of a non-volatile technologies a one-time Programmable devices.

These devices is SRAM technologies are kind of even if its radiation has an program and send into the space because of radiation are these memories can get the RAM program up like a configuration memory can get corrupted but not in the case of these radiation hardened anti-fuse because it is permanently fused, but it has a great disadvantage saying that it is only one time Programmable once you program it, it cannot be raised and reuse.

So it is not very good for prototyping a like it like that but then you want to test the system during the development not a good idea to use a technologies may be it has tied in static RAM kind of FPGA then once it can be transferred to the Anti-fuse kind of technology.

(Refer Slide Time: 34:20)



So essentially in a natural this may not be a very accurate picture, but at least the previous generation of the FPGAs add this kind of architecture the current ones I cannot say that it is exactly this but then its good starting point and it is quite correct you know that there are additional details you know in a current FPGA. So let us look at the structure of an FPGA.

So basically have I/O pins around of course there are VCC the ground pins and all that 20 of them for the inner core and I/O pins normally the core works at a low voltage and the I/O pin work at much higher voltage. So this may be under 1 voltage but then when it comes to the I/O pin it may be 2.5 or 3.3 volt devices depending on the technology , otherwise there are I/O pins around.

And you can see that their logic blocks array of that ok, here it only shown 4/4 but in real life it can be my say huge number 32/32 or 1 digit/1 digit and so on and even higher sometime and these are kind of wires all around. So you can see that this may be more number of wires which is connected in a switch here. Again the wires all the wires are connected with switch here.

And this the bottom and left that the input the logic clock and so then that means there is a switch here, a switch here and this the top and the right hand side is output. So in principle how the design is done in an FPGA that the synthesise logic is placed across multiple blocks depending on some kind of concern and say an I/O pin can be connected to the input of a logic clock.

And that output can come here maybe can be taken to the input of this logic clock as well as you know comes all the way and this logic clock and this output can be may be taken to the input of this logic clock and so on. So basically this structure allows you lot of possibility of connection and you can see that is not a single switch like CPLD basically in the CPLDs there was the logic clock and huge switch.

But here these are kind of simpler blocks very small logic and flip flops and a lot of them and it is interconnected through a lot of wires and lot of switches which is distributed. So that you know it can be staying in a big you know you can have a bigger much bigger device than CPLD and so this may be there are you know when you talk about switch here, switch here you need not that you know all the wires from this switch run to the switch.

There could be wires running from the first switch to the third switch first switched to the fourth switch and all that. so there could be wires which is kind of single length of the single block or we can single length wires or double length wires or quad length for the wires running from one end of the device to the next end of the device. Otherwise this kind of wires, different length wires required because otherwise supposedly there is logic clock here the output of which need to go to the input of the logic clock at the end.

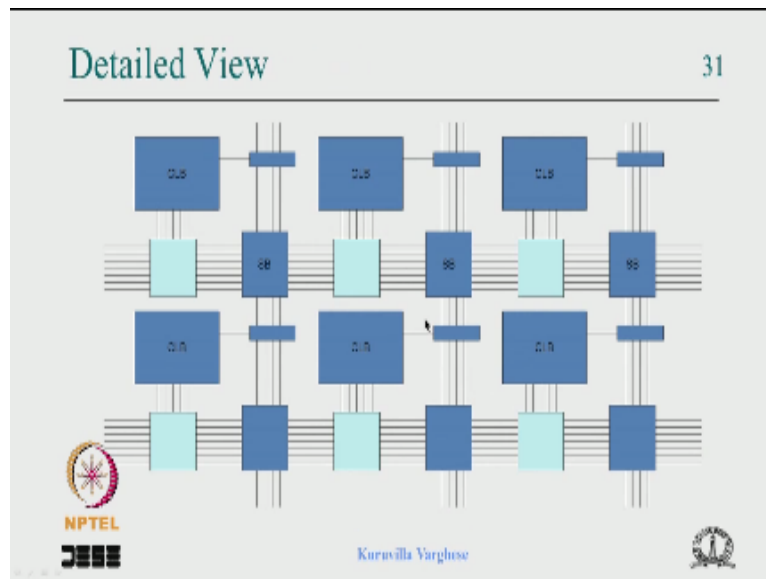
And if you connect by single end wires will incur lot of switches are required for interconnecting which you know raise is the area of the switches and it will incur lot of delays for it switched there will be delay and so end to end delay very large and the maximum frequency because we have seen in a data path the frequency achieve $t_{cq} + t_{com} + t_{setup}$ and here the combination delay will include the combination delay include all the wireless delays the interconnection delays.

And in FPGA if the lot of switches used these interconnection delays will be more than the logic delay. So that issue with FPGA interconnection delays which is much much larger than sometime more than the logic login depends where the logic is placed in within the chip maybe if you have some logic here and output is taken all the way to the right hand side of the FPGA which is very huge through lot of switches then it incur a lot of delay.

So that should be kept in mind the interconnect delay is one kind of disadvantage of this FPGA but that is at the cost of flexibility and cost of scalability of the device, this is fluted as

you know interconnection is what allows you to grow in size is but that add to the end to end delay when the circuits are place then returned within FPGA ok.

(Refer Slide Time: 40:12)



So this shows a little more detailed diagram, so this shows the CLD you can see here and these are the switch blocks all around 8 at the corner and these are wires we can see that this shows that the 4 wires are kind of connecting this switch to here and you can see there are may be 7 wires connecting horizontal and wires and these are the input to the logic clock and CLB means it means it is called configurable logic block ok that is one thing once again you should remember that not that the wires are Programmable or configurable what logic within the logic block is also configurable.

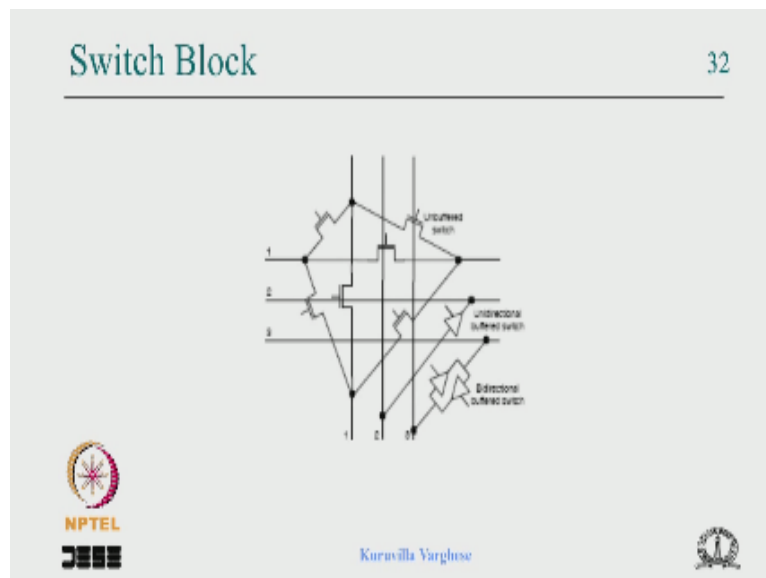
So there is not only programmability in terms of interconnection there is programmability within the logic block ok to make it very flexible. So when you say configure FPGA that means configuration all these connections as well as configuring the logic clock ok that should be kept in mind. So it is a configurable logic block can see that means that there are switches between this input wire and output wire and so on ok.

A similarly this shows an output of a logic block is coming to a switch here, it essentially means you have either you can connect this output this input this one this one or this one or maybe this it does not matter now you can connect it either way or sometime it is a single line to which is correct or there is a disconnect depending on you know the number of output and so on ok.

So it depends on the vendor and vendors does he know analyse lot of applications which is mapped into this kind of fabric and from the statistics day of judgement on how many wires to place and so on ok. So it is not that the number of wires and type of wires are fixed from device to device depending on the complexity that could be more or less wire is so on okay. So that should come from satisfied this cannot be blindly put maybe they started analysing the application.

Then put some earlier generation now they keep statistics of the current applications and when are the world to the next family of devices they kind of scale it appropriately this number was the type of wires and so on ok.

(Refer Slide Time: 42:56)



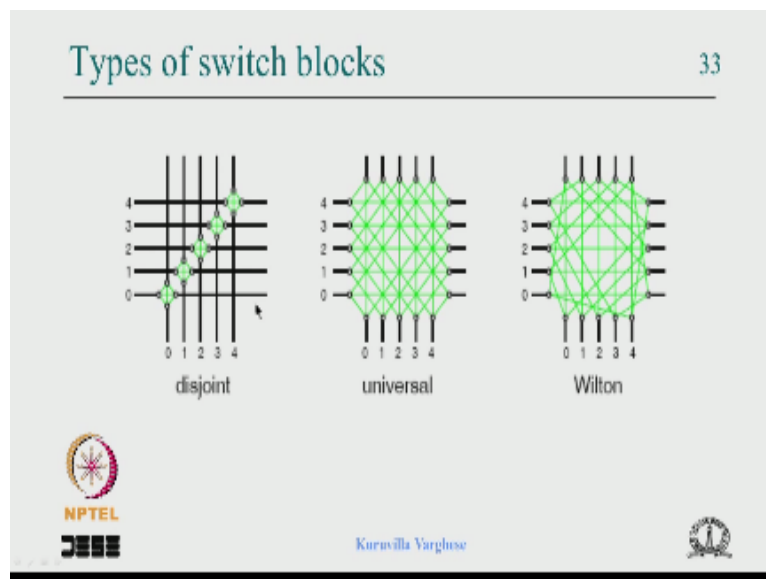
And if you look at the interconnection itself this is how it is interconnected switches which is shown here for your vertical wires and horizontal was so what is done is that between a vertical wire and cable wire you have a transistor when the gate is made on it is interconnected ok and you can have a uni-directional connection or a by direct connection depending on your name maybe its output is coming from the side and going to the input on the other side you need of uni-directional.

But if there is some kind of data flow signal flow in both direction then you need a bidirectional switch and now if u look or in principle if you have a 3 by 3 this can be connected not only to this it can be connected to this as well as does and you know maybe yes each of this line can be connected to sleep but sometime providing all possible

connection can make this very so many times is not that a single wire is connected to every other wire and so on okay.

Particularly there may not be any connection between the vertical and vertical why are there no need because we have seen the output is coming to all vertical lines. So normally the vertical lines are connected to some wires and but here when it comes to a switch box this path need to be connected to this vertical line to cut through ok so that connection is required that is we can see that it is shown here.

(Refer Slide Time: 44:50)



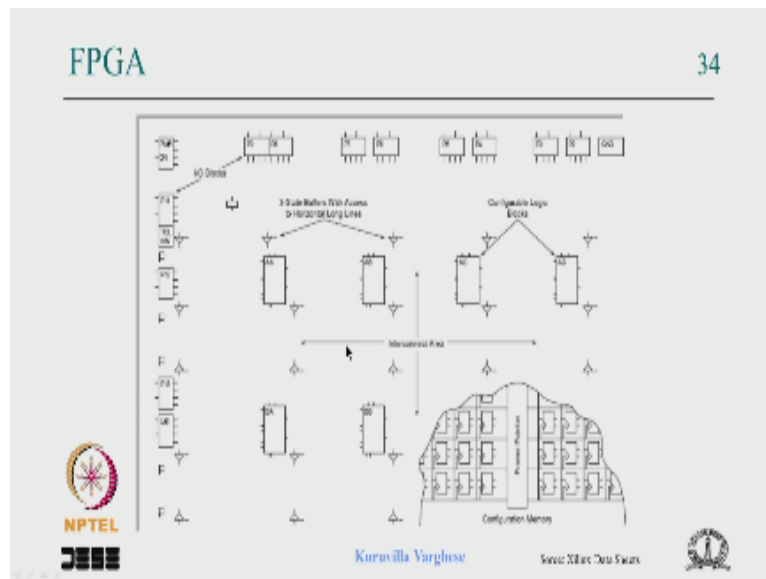
So there is a vertical connection here ok and you will connect in here there are various which architectures you can see that the this joint connection and where in one each wire though two wires are connected to wire and one over wire a slightly different mother of connecting again sliding different method of connecting. So you have the choice you know you may not be that this via need to be cut this wire maybe can be connected to the some of the wire.

We saying this case we can see that does the first wire is connected to the to the top most wire, but here though the first wire is convertible wire is connected to the bottom of original wire. So there are different architecture depending on that is a little bit of a back end detail of the device so as a design of we need not bother too much about it but I as a device if you are in FPGA design.

We designing the FPGA design then only it bother about it. So we are basically looking FPGA as a device to program our device design. So we need not come in we need to

understand but we are not going to the details of various interconnections and so on. So let us go to the next level.

(Refer Slide Time: 46:13)



So this is a very old slide from the designing data sheet. So that illustrates one thing and FPGA you would have seen I have discuss their a lot of combination circuit, flip flop, so there will be lookup table flip flops interconnection wires and switches and all, but then you know that these all these need to be configured like switches need to be kind of program of the logic block need to be configured and so on.

So there are lot of overhead configuration circuit that means is configuration pattern need to be stored somewhere at the power on these switches need to program and so on. So you should know that as if it is not just what the design of seize the designer mainly see only the logic resources flip flops and wires but there is an underlying configuration overhead, logic and the circuit ok.

So that makes a FPGA quite bit the wires and the switches and in the configuration overhead circuit we will make it a much bigger in be spread out. So you can expect at a row speed and the speed like ASIC or FPGA, but there are in FPGA exploit that it is a Programmable fabric and it is our it is in the control of designer to put you know however many computational unit with FPGA and maybe we will do very wide computation might choose to convert 64bit data or two 32bit data and do it parallelly.

Do it at a much higher rate that we shown to achieve performance, so FPGA lacks in row and walk in speed maybe in ASIC clock as a 1 zeta may be FPGA clock we say 200 megahertz or 400 megahertz at the more reasonable complexity design is put in do it, but by design techniques by kind of doing y data computation multiple course and so on we can replicate the number of computation unit you can do lot of computation and achieve the performance may be bigger than much much bigger than the ASIC.

So there is another disadvantage of FPGA the power dissipation will be quite high compared to the ASIC because of the huge area huge interconnection and naturally the participation will be much higher and you may not because these are kind of Pre built the resources vary custom, low power design cannot be done in FPGA because the application is not variable done.

So kind of very aggressive low power design cannot be done and in an FPGA as in ASIC so 1 tree counts say the whole speed the participation the FPGA suppose but then definitely can it achieve high performance with lot of flexibility of programming reprogramming configuring if FPGA will be able to achieve. So let us move on to the to the slide.

(Refer Slide Time: 50:10)

The slide is titled "FPGA" in the top left corner and has the number "35" in the top right corner. Below the title is a horizontal line. The main content is a bulleted list of FPGA components:

- I/O Blocks (Tri-state output / Input, Synchronizing Flip-flops)
- Array of Configurable Logic Blocks
- Horizontal and Vertical wires with programmable switches in between
- Single length, Double length, Quad, Hex and Long lines
- Resources available to user

Below the list, there is a circular logo with a compass rose and the text "Resources for configuring programmable switches in the interconnect structures and Logic blocks". At the bottom left, there are logos for "NPTEL" and "JEE". At the bottom center, the name "Kurusilla Varghese" is written. At the bottom right, there is a small circular logo.

So essentially if you look at the FPGA you have I/O block and this we did not discuss the IO/ blocks always contain you see there are the tri state gate based output. So which can be used as input, so these I/O pins and there will be always a synchronizing flip flop, we have seen the need for synchronisation because this input is coming asynchronously to the clock use which usually will be the case because we will supply the clock to the FPGA.

Everything done will be clock by the clock, maybe the signal is coming from another chip, maybe come by another you know the clock. So this need to be synchronised, so there are synchronizing flip flops in the stage synchronising flip flop and I/O gate and it support multiple IO standards it working with 2.5 or 3.3 like low voltage CMOS or it suppose some kind of the voltage pattern which is ideal to the PCI Express or PCI and so on.

So all that is supported in an I/O pin, so that is I/O block, then you have array of configurable logic block we have said are the main thing to note is a not fix this configurable lot of configuration as possible within logic block then there a horizontal and vertical wires with Programmable switches in between, we have seen that you know there are the horizontal and vertical wires with Programmable switch.

Then the wires could be single end and double end, quad, and hex and long lines and there are resources available to the user basically the logic block the flip flop then the other resources for configuring the Programmable switch in the in the connectors and logic block of a lot of Programmable consecutive which program this logic block and program the interconnection of the wires ok. So that is in a nutshell it is an FPGA assemble FPGA.

In addition high level FPGAs will have as I said BSP blocks the hard core memory controller the network interfaces and so on. So we have covered the essentials of the field Programmable Gate array.

(Refer Slide Time: 52:52)

Programmable Connections 36

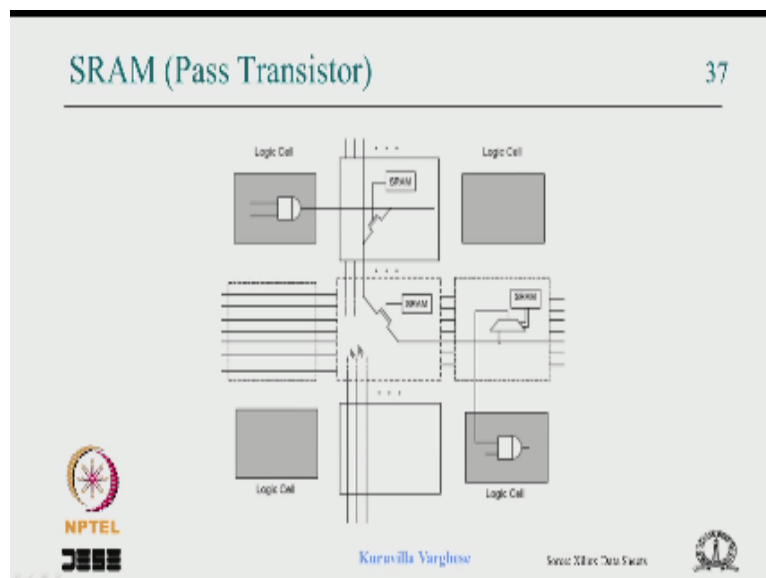
- SRAM (Pass Transistor)
- Flash
- Antifuse

NPTEL
JEE

Kurusilla Varghese

So let us look at this particular Programmable connections which is available in static RAM base FPGA as I said this SRAM is a bit of unfortunate name the interconnection technology used is a pass transistor. We will see why it is called as Ram the some FPGAs use flash transistor you have seen that in the case CPLD and some use a technology called Anti-fuse which is opposite of that of a fuse.

(Refer Slide Time: 53:30)



So let us look at the static RAM this shows a FPGA architecture which say 4 configurable logic clocks wires, horizontal wires, vertical wires and switch interconnecting the vertical wire and this shows that output of a logic cell is connected to the vertical wire, so the connection is an NMOS transistor connect that pass AO and so basically if this N transistor is one the output is connected to this input.

And similarly this vertical wire as a connection to the horizontal wire using an N transistor, so if the gate is is made high it connects ok, now the problem with this SRAM, FPGA is that you cannot remember it cannot is not the state of the gate of NMOS transistor is not stored permanently at every power on this need to be programmed ok. So if you look at what can hold a bit assemble bit is a flip flop ok.

So at the gate of each pass transistor is flip flop and now when we save a program we programming the flip flop as 1 or 0 ok. So that can be so there is a flip flop and see if you see that 2 to 1 mux you see here. There are a lot of lines lot of horizontal lines and you can have a kind of crossbar connection reset that N/N crossbar uses a n, n-1 multiplexer, so basically the multiplexer is one which is so this is a kind of 1 to 1 multiplexer.

But here you can think of same say clock 1 multiplexer which allows you to connect any of the 4 horizontal wires to be connected to the input of logic cell. So here there are 4 inputs and you have 2 select lines or there are 2 flip flops connected to the select line. So if you have the ability to program 0,0,0,1 1,0,1,1 and so on. So now so you can imagine wherever there is a switch there is a flip flop now you cannot program this flip flops and usually so what is that this flip flops are connected together assign static RAM or as a memory which is 8 bit or 16 bit.

In previous generation it was 8 bit y memory in the current generation it is 16 bit wide. So to program means you write this configuration memory which is SRAM but the real interconnect technologies pass transistor in a some type of device.

(Refer Slide Time: 56:38)

Pass Transistor with configuration cell 38

- Flip-Flop to store the switch status (4 Transistors)
- Write Transistor to write Configuration status
- Total: 6 Transistors
- Switches are organized as SRAM hence the name

So I think we are coming to the to the end of the lectures of before and I think we instead of going further because then we have quite a bit Programmable interconnection. So let us wind it up here. So what we have seen is basically general architecture of FPGA lot of I/O blocks configurable logic blocks and which is distributed across this area to Programmable wires switches in between which is again distributed not a single switch, it is like a array of switch with regular wires lead.

So the logic clock can be interconnected to that and when you look at the logic block itself it is a combinational circuit and flip flop. Combinational circuit are look up table multiplexers of gates, flip flops, then the interconnection technology is static RAM and if you are the flash

kind of devices and so that is what if essentially seen about FPGA we have seen the commercial Xilinx, Altera and Actel devices.

We have seen some of the familiar devices, so the last part was a Programmable interconnect we have seen the SRAM kind of technology. So in the next lecture will continue with this program to interconnect technology and go the architecture of the FPGA to understand in deep, so I wish you all the best and thank you.