

**Architectural Design of Digital Integrated Circuits**  
**Prof. Indranil Hatai**  
**School of VLSI Technology**  
**Indian Institute of Engineering Science and Technology, Shibpur, Howrah**

**Lecture - 02**  
**Introduction (Contd.)**

Welcome back to the course on Architectural Design of ICs. So, today again we will continue the introduction part ok.

(Refer Slide Time: 00:26)



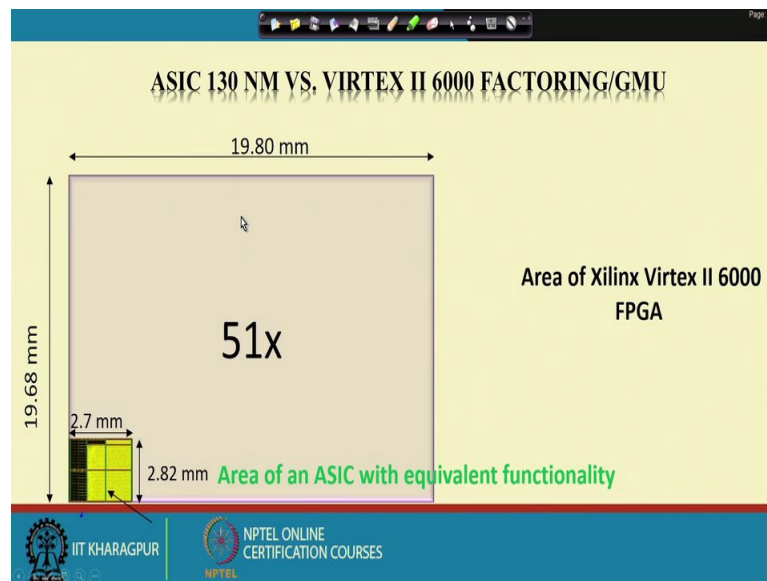
So, in the previous lecture I discussed about different design styles of VLSI design or to build one integrated circuit. So, at that time I said that semi custom design, full custom design, then system on chip and then programmable logic design. So, what are the; that means, difference between this a programmable device and ASIC is that; ASICs are basically It gives you high performance, it consumes low power and this also low cost whenever you are producing or whenever you are designing for high volume application.

And this FPGAs is that, this is the off the shelf; that means, this is already pre built. So, you can just buy it from market and then you can use it for your particular system design and then it comes with low development cost, and it also gives you the flexibility of; that means, the primary objective of this FPGA or programmable device is that, you can achieve the re configurability option; that means, you have the scope or you have the;

that means, chance to reuse those particular hardware for your design; that means, whenever you are designing that particular hardware or particular system.

And this also has another advantage is that, that shorter time to market also you can achieve if you are using FPGA. But here you the thing is that you are reducing the performance in compare with ASIC.

(Refer Slide Time: 02:09)



So, if you see just this is one just example of this the difference between this ASIC and FPGA. If you see area wise this, this particular sorry this particular area, this is basically the area for 1 plus Xilinx, Virtex 2; Virtex 2 6000 series FPGA. So, this much of area you require whereas, the with equivalent functionality or the equivalent get count if you follow the ASIC design style. So, at that time you require only this particular area ok.

So; that means, this is 51 x times higher; if you follow this FPGA design style at that time you are consuming 51 x times higher area, than if you follow the same design in ASIC design style. So, if you are following more area means it consumes more power and the more area to; that means, it requires more cost or more power hungry device or power hungry system whenever you are following FPGA in compare with ASIC ok. So, this is just 1 example of that.

(Refer Slide Time: 03:34)

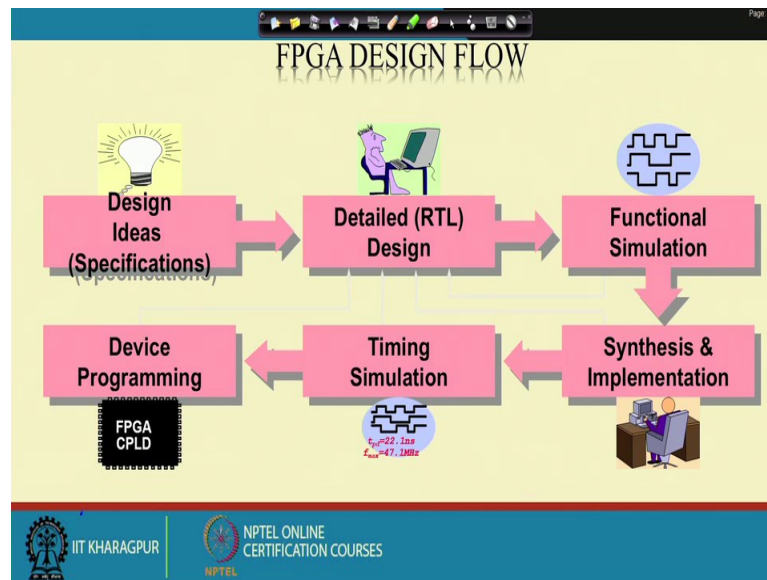
Government Developed	Commercially Developed
Ada based	C based
Strongly Type Cast	Mildly Type Cast
Difficult to learn	Easy to Learn
More Powerful	Less Powerful

So, here one size is that regarding this VHDL versus Verilog; what is VHDL and what is Verilog? So, I am I am telling you about designing 1 system or this VLSI design passes through this particular phase, this particular phase, but how you can describe the circuit. So, whenever you are using one tool using this automation tool whenever you are designing your system so at that time the system definition language should be there ok.

So, that is why there are hardware description language; that means, it is there are basically specific language for that by which if you use those language, you can describe the hardware or the tool will know that this is this language this hardware description language. So, this means that it is just like; that means, if your you can speak with Tamil. So, at that time you will be know only Tamil; if I speak to in Tamil. So, at that time you will understand what I am saying. If you are that Marathi; so, at that time if I speak in Marathi at that time only you will be get to know that what I am saying.

So, it is something like that that, whenever I am describing the hardware. So, at that time I need to describe the hardware using its particular language, which it can recognize what I am trying to say or what I am trying to command tool to do. So, this VHDL and Verilog these are the very commonly used language for describing the hardware. So, this is the just the difference between VHDL and Verilog both are very popular any of these VHDL or Verilog nowadays people are basically using system C or system verilog. So, any of these language you know and then you can use for your particular system design.

(Refer Slide Time: 05:48)



So, then we will follow this design FPGA design flow or this not specific to a FPGA design this is just if I say that VLSI design flow ok. So, VLSI design flow in VLSI design flow there are little bit difference between this FPGA design flow, and this ASIC design flow. So, in FPGA design flow if I say, it has to start with this ideas this design ideas or this specification then you have to know about the detail design or this RTL description of that particular things RTL means Register Transfer Logic then you have to follow the functional simulation, then after functional simulation you have to follow the synthesis and implementation option. So, after synthesis and implementation option you have to follow the timing simulation.

So, once timing simulation is done then you can go for device programming as you are following this FPGA design flow. So, here the thing is that this particular things is will be differed if I follow the ASIC design flow at the time I do not have any prebuilt device. So, I that I can go for device programming; after timing simulation what you have to do? There these things you have to do it manually; that means, you have to build your own ICs.

So, at that time you have to build; that means, you have to go for this map; that means, placement, there floor planning placement, then routing and you after all these you just build the geometric description on this (Refer Time: 07:38) file, and then it is you send it to the foundry, the corresponding chip will be fabricated on that particular foundry, and

you will get your corresponding hardware for your system whatever; that means, I have started with the specification of there is design ideas. So, once it has been comes from the foundry you will get your idea that into reality ok. So, now, I will just brief about all these what are that what that this particular phase means.

(Refer Slide Time: 08:11)

The slide is titled "DESIGN SPECIFICATION" and features a list of design considerations. A video inset shows a man in a purple shirt speaking. The slide also includes logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION CC.

- What are the main design considerations?
  - Design feasibility?
    - Performance
    - power consumption
    - cost
  - Design spec?
    - Written (Document)
      - Good starting point, but can be misinterpreted by design team
    - Executable (UML, C/C++, Behavioral VHDL, SystemVerilog)
      - Harder to understand, less room for misinterpretation
  - Implementation platform
    - FPGA/CPLD?
    - ASIC?
    - Which FPGA/CPLD vendor?
    - Which device family?
  - Development time?

So, this design specification it starts with design specification. So, what are the main design considerations? whenever I am considering one design specification? So, at that time what do I consider in my specification? So, the first thing is that this design feasibility; design feasibility is basically that basically comes with this performance, power consumption and cost.

So, performance power consumption and cost means that comes with design feasibility I have to check why I have to check for this particular thing? Performance it is not that I require or I expect that it will it will be on the higher side, which cannot feasible or which as a designer it is very hard to achieve that particular thing. Performance wise suppose if I say ; that means, ask you to design 110 Giga hertz speed processor. So, is it feasible or if I say that; that means, in nano watt power consumption processors I need. So, at that time I have to see or I have to check whether that thing is feasible or not; that means, as this is the starting point what will be my clock speed of the system, what will be the power consumption of the system, what will be the cost per volume of the system. So, all these things are as it has to be started initially.

So, whenever I will set the target, at that time I have to keep in my mind that is that feasible is it; that means, realizable. So, you have to check your the feasibility people as more greedy people we need; that means, as a as a person I need more of these to achieve, but we have to be within the limit ok. So, that is why whenever we are considering design specification at the time the design feasibility is the primary concern to be checked.

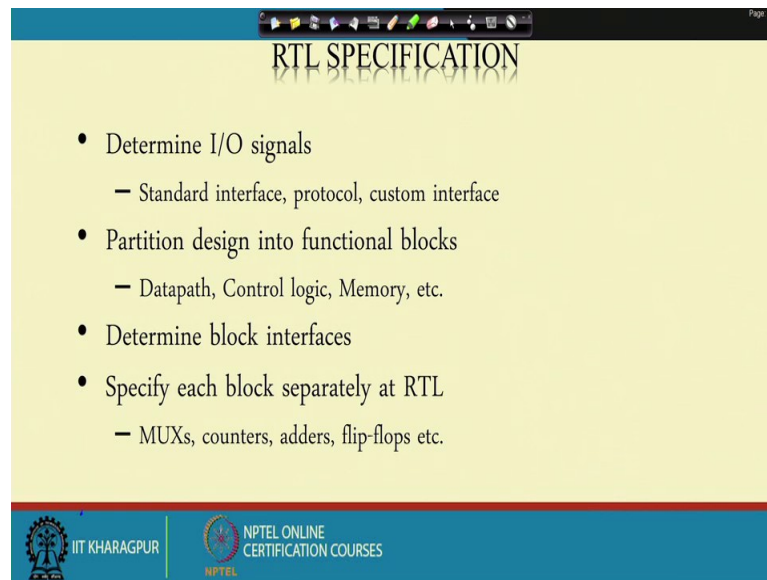
And then design spec. So, design specs spec should be written in a good starting point, but can be misinterpreted by design team ok. So, if I just; that means, there is specification is basically used in each of the space; that means, where from I am or what I am what is my target; specification means what is my I am setting my target. So, whenever I am working on that to achieve the target. So, every time I have to follow or I have to check what target I am having or what target I am to set, what I am basically trying to achieve ok?

So, that is why it should be documented in a good way so, that it has to be ah; that means, forwarded or it has to be known or it has to be access easily accessible to each of the members of your design team. So, it can be written in a execute executable manner with; that means, in UML or C C plus language. So, the problem with that is it is very harder to understand and less room, but less room for misinterpretation.

Then we have to select this implementation platform. Implementation platform means which design styles we will follow for implementing our design. So, basically the peoples are basically initial design phase, they validate their algorithm and the whole system in FPGA. So, once it has been validated, then you can go for the ASIC design or you can just can it to the foundry; that means, your design is you have checked, it is performing on finely or that means, you have achieved the corresponding specification then you go for the foundry ok.

So, then the another type another; that means, things which you will be covered in the design specification that is the development time. That means, what is the time for developing the corresponding system hardware ok. So, the; that means, from starting of your system and to the from concept to the reality how much time you take, that will also be mentioned in the design specification.

(Refer Slide Time: 13:18)



The slide is titled "RTL SPECIFICATION" and contains a bulleted list of four main steps. The first step is "Determine I/O signals" with a sub-point "Standard interface, protocol, custom interface". The second step is "Partition design into functional blocks" with a sub-point "Datapath, Control logic, Memory, etc.". The third step is "Determine block interfaces". The fourth step is "Specify each block separately at RTL" with a sub-point "MUXs, counters, adders, flip-flops etc.". At the bottom of the slide, there are logos for IIT Kharagpur and NPTEL Online Certification Courses.

- Determine I/O signals
  - Standard interface, protocol, custom interface
- Partition design into functional blocks
  - Datapath, Control logic, Memory, etc.
- Determine block interfaces
- Specify each block separately at RTL
  - MUXs, counters, adders, flip-flops etc.

Then you have to build the RTL specification. So, in the RTL specification means one you have got the specification. So, at that time and you got the specification you have you have got the information, what you have to build or what hardware you have to build. So, everything you have already set. So, at that time you can you have to build the architecture for that ok. So, architecture for that means, not architecture before that you have to; that means, first you have to build the algorithm for that ok. So, so that, that particular algorithm will be used to build your corresponding hardware or corresponding system.

So, then though once this algorithm verification part is done. So, then you can go for that algorithm to the architecture mapping ok. So, this architecture algorithm to architecture mapping; architecture mapping means that will be described by using some of the block level representation so; that means, I need this particular component or this particular operators; operators means this nothing, but this adder subtractor multiplier these are the basic components which you used to manipulate or to implement or to describe one particular function ok.

So, then once this architecture is also finalized, then this architecture will be described using any VHDL or Verilog using any hardware description language ok. So, in RTL specification there should be this IO signals; IO signals means input output signals, then this partition design into functional blocks so; that means, in any particular systems you



have this control path and data path that is the part of the architecture where we will see most of the things; that means, here we will cover in the this particular course we will cover that things, that from this algorithm to the architecture ok.

So, how efficiently you can do? So, once you have build the suppose the algorithm team they have developed the one algorithm; so how efficiently you can map that particular algorithm into the architecture so that you can achieve more of the savings on the power as well as the area or you can improvise the speed to so, this that we will see in this particular course.

(Refer Slide Time: 16:20)

The slide is titled "DETAILED DESIGN" and contains the following content:

- Choose the design entry method
  - Schematic
    - Intuitive & easy to debug
    - Not portable
    - Poor designer productivity (gates/time)
  - HDL (Hardware Description Language), e.g. Verilog, VHDL, SystemC
    - Requires some experience, harder to debug
    - Descriptive & portable
    - Easy to modify
    - Greater productivity
  - Mixed HDL & schematic
- Interpret the specifications
- Manage the design hierarchy
  - Design partitioning
    - Chip partitioning
    - Logic partitioning
  - Use vendor-supplied IP libraries to reduce design time
  - Create & manage user-created libraries (circuits)

The slide also features the IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COL. logos at the bottom left, and a small icon of a person at a computer in the top right corner.

So, after that you have this detailed design, detailed design means you can you can chose in different two style that is in schematic or using HDL language any of these you can describe the corresponding; that means, once this architecture has been developed, then using this using any of this two schematic or HDL language HDL in language entry, any of this style you can chose for describing that particular architecture into the corresponding hardware.

Then, you have to do this manage the design hierarchy; design you for larger design you have to partition the design ok. So, partition you can do in chip partitioning or logic partitioning. So, this is the part of the FPGA design style, I will not much cover on this particular things So, this is the detailed design specification or this basically this



particular things you require whenever your designing in your following this FPGA design styles.

(Refer Slide Time: 17:31)

The slide is titled "FUNCTIONAL SIMULATION" and contains the following bulleted list:

- Preparation for simulation
  - Generate simulation patterns
    - Waveform entry
    - HDL testbench
  - Generate simulation netlist
- Functional simulation
  - To verify the functionality of your design only
- Simulation results
  - Waveform display
  - Text output
  - Self-checking testbench
- Challenge
  - Sufficient & efficient test patterns

The slide also features the IIT KHARAGPUR logo on the left and the NPTEL ONLINE CERTIFICATION COURSES logo on the right.

Then you have to follow this functional simulation. So, why I need to know about this simulation or this functionality check checking of the particular system? So, means whenever I have described the system using any language. So, at that time I have to check suppose I describe or I have build one circuit for adder. So, by mistake if I do any mistake so, at that time whenever I have described at that time I have just missed somewhere it is now becoming one subtractor or just it deviates from the functionality of the adder. So, at that time I how do I know?

So, using the simulation; that means, I will put some of the test vector (Refer Time: 18:29) to it and then I will check output in adder if I say that I know 1 plus 1 it will be 2 ok. So, how do I know from the beginning I know that 1 plus 1 plus it will be 2. So, if it is not doing that thing or it if the result is not giving me 1 plus 1 plus is equals to two back. So, at if it if it gives me other value from 2. So, at that time I will check my design, that I and I will let you I will know that there is a problem in the particular whenever I have described the system; that means language or the hardware using language. So, again I will check; that means, the logic part I have to change ok. So, that is why we do functional simulation.

(Refer Slide Time: 19:19)

**HDL SYNTHESIS**

- Synthesis = Translation + Optimization
  - Translate HDL design files into gate-level netlist
  - Optimize according to your design constraints
    - Area constraints
    - Timing constraints
    - Power constraints
- Main challenges
  - Learn synthesizable coding style
  - Use proper design partitioning for synthesis
  - Specify reasonable design constraints
  - Use HDL synthesis tools efficiently

assign  $z=a\&b$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSE

And then this once this functional simulation is done; then we have to follow this HDL synthesis. So, HDL synthesis means I have described it using language now I have to after that synthesis means what? It has to from the HDL coding it has to map to the corresponding using some of the logic gates so; that means, you just see that first thing this translate HDL design files into the gate level net list.

And all this whenever I am this converting from logical to the gate level thing; that means, the logical expression from the gate level expression. So, at that time I have to I can achieve more of the optimization to achieve better performance in terms of area timing and power. So, that is why whenever you are optimizing your design at the time you can put some of the constant as area constraint, timing constant and power constant.

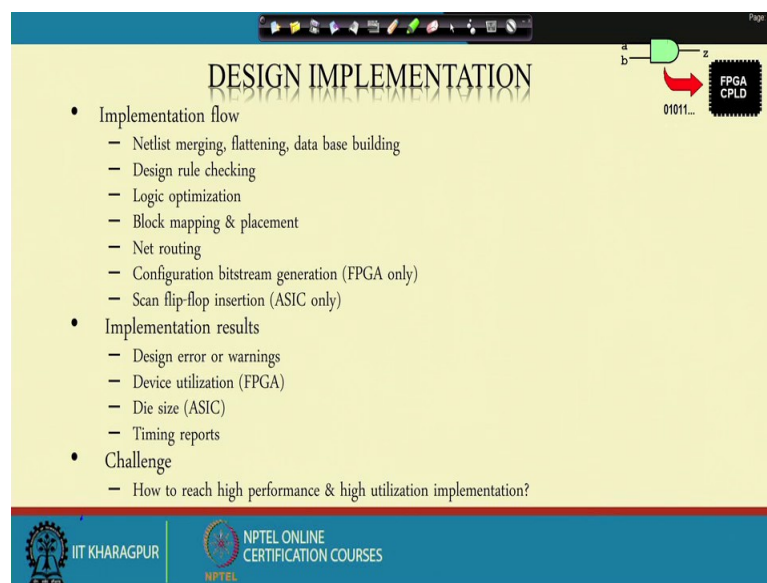
So, where from will get this constraints; that means, the information about this constant where from will get this information? From the specification you will get this information. So, from the specification means you I have already know that my targeted clock speed is 1 Giga hertz ok. So, that; that means, the time the maximum clock time which is required that is one nano second I have to achieve whenever I am designing the system ok.

The power if I said it will be let say 1 milli watt ok. So, 1 milli watt in the initially I know. So, that constraint I will put whenever I will do this particular synthesis so, that this optimization can be done while I am converting from this HDL language to the

corresponding gate level expression logical expression to the gate level expression. So, here that the main challenge is that to learn the synthesizable coding style, then use proper design partitioning for synthesis specify reasonable design constraints, and use HDL synthesis tool efficiently ok.

So, this is all about automated way. So, we will see in the automated way also you can do the optimization manually also you can do. So, manually you do the optimization and then you just let the tool to do the more optimization on your particular system describe. So, we will see this optimization manual optimization procedure for this logical expression to the sorry for from this HDL expression or this logical expression to the gate level expression ok. So, that we will see more how to do that in this particular course?

(Refer Slide Time: 22:50)



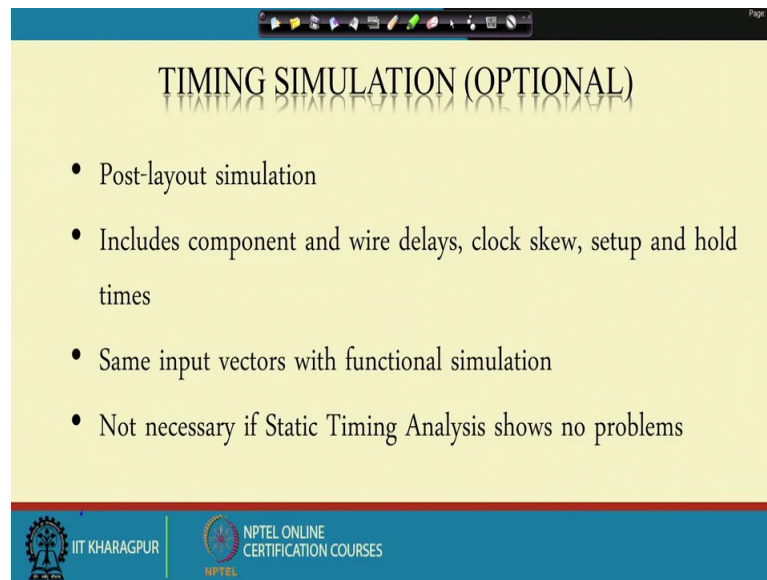
The slide is titled "DESIGN IMPLEMENTATION" and is part of a presentation (Page 7/7). It features a diagram in the top right corner showing a logic gate with inputs 'a' and 'b' and output 'z', connected to an "FPGA CPLD" device. The output 'z' is labeled with the binary value "01011...".

- Implementation flow
  - Netlist merging, flattening, data base building
  - Design rule checking
  - Logic optimization
  - Block mapping & placement
  - Net routing
  - Configuration bitstream generation (FPGA only)
  - Scan flip-flop insertion (ASIC only)
- Implementation results
  - Design error or warnings
  - Device utilization (FPGA)
  - Die size (ASIC)
  - Timing reports
- Challenge
  - How to reach high performance & high utilization implementation?

The slide footer includes the logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES.

So, then there is this design implementation, this implementation flow is there; that means, placement, routing, float planning, mapping all these things you have to do in a FPGA design style.

(Refer Slide Time: 23:03)



Page 7/7

## TIMING SIMULATION (OPTIONAL)

- Post-layout simulation
- Includes component and wire delays, clock skew, setup and hold times
- Same input vectors with functional simulation
- Not necessary if Static Timing Analysis shows no problems

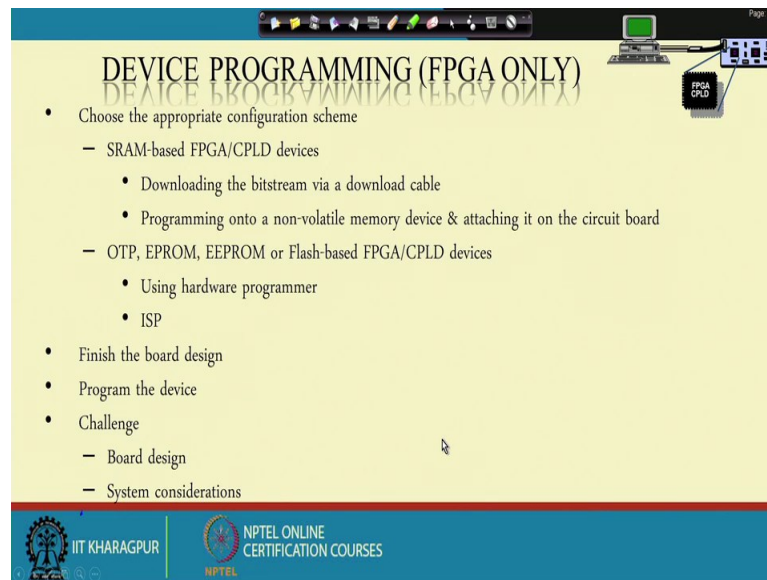
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, I will not cover that much on this particular things then. So, initially what I have done? I have done this functional simulation right. So, here what I have to do then after this synthesis, I have I have got to know about gate level net list I have to do this timing simulation. So, in timing simulation what is there? That means, in this particular simulation in the functionality simulation no gate delay or net delay is basically included it only checks whether for 1 I am getting two or not for particular array design ok.

But here you will in timing simulation here 1 plus 1 plus is you just put test vector to the adder then you; obviously, you will check whether you are getting 2, but after how much time of applying the input how much after how much time you are getting the output, that is also be included in the timing simulation ok. So, that is the main difference of the basic difference between this functional simulation and this timing simulation.

So, here you see this timing simulation it includes component and wire delays, clock skew setup and hold times though I have put the same vectors what I have put in the functional simulation. And not necessary if static time analysis shows no problem so; that means, whenever we are doing this timing simulation at that time this static time analysis is one of the very important parameters in this timing simulation.

(Refer Slide Time: 24:58)



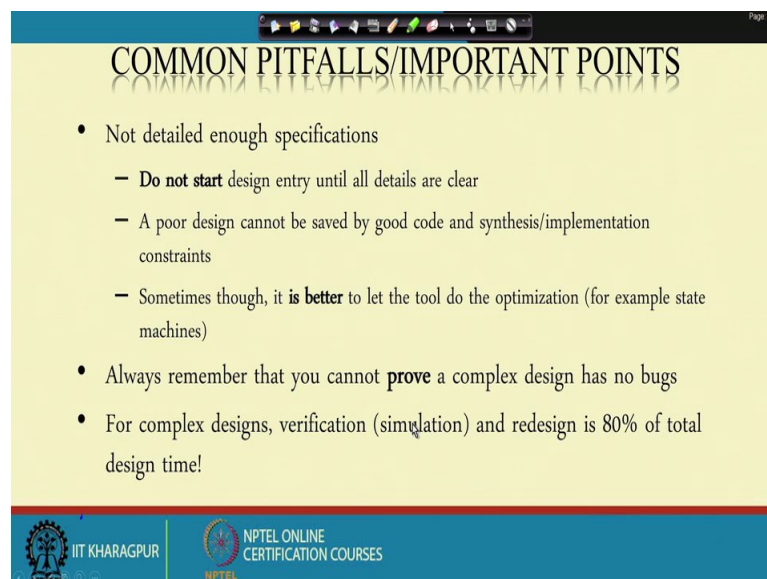
Slide titled "DEVICE PROGRAMMING (FPGA ONLY)". The slide content is as follows:

- Choose the appropriate configuration scheme
  - SRAM-based FPGA/CPLD devices
    - Downloading the bitstream via a download cable
    - Programming onto a non-volatile memory device & attaching it on the circuit board
  - OTP, EPROM, EEPROM or Flash-based FPGA/CPLD devices
    - Using hardware programmer
    - ISP
- Finish the board design
- Program the device
- Challenge
  - Board design
  - System considerations

The slide footer includes the IIT Kharagpur logo and the text "NPTEL ONLINE CERTIFICATION COURSES".

So, after that there is a option that you have to do this device programming; that means, as I am following FPGA. So, once everything is done timing simulation is working fine and then you can go for the device programming part.

(Refer Slide Time: 25:13)



Slide titled "COMMON PITFALLS/IMPORTANT POINTS". The slide content is as follows:

- Not detailed enough specifications
  - **Do not start** design entry until all details are clear
  - A poor design cannot be saved by good code and synthesis/implementation constraints
  - Sometimes though, it **is better** to let the tool do the optimization (for example state machines)
- Always remember that you cannot **prove** a complex design has no bugs
- For complex designs, verification (simulation) and redesign is 80% of total design time!

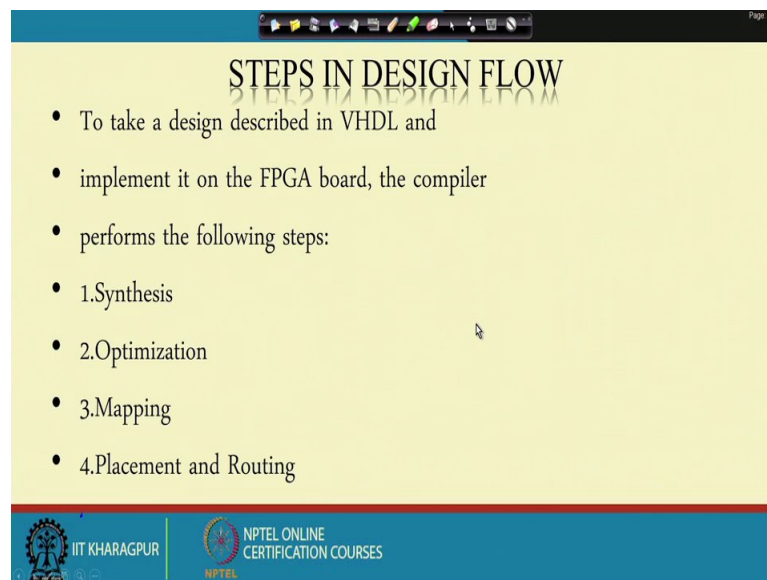
The slide footer includes the IIT Kharagpur logo and the text "NPTEL ONLINE CERTIFICATION COURSES".

So, the common pit falls or the important points you have to be keep in mind that is, not detailed enough specification, do not start design entry until all details are clear, a poor design cannot be saved by good code and synthesis implementation constant. Sometimes tough it is better to let the tool to do the optimization; that means, it is not that the

manual optimization is always better, sometimes you just let the tool to do that, and everything will be clear; that means, whenever your specification will be very much clear whenever you have started the system; that means, your target will be set properly and then you start working on that, to achieve this corresponding target.

And always remember that you cannot prove a complex design has no bug ok. And for complex design verification and redesign is 80 percent of total design time. So, these are the common things or the common; that means important points, which will be remembered or which will be every times whenever you are following each of the phase in your corresponding system design ok.

(Refer Slide Time: 26:36)



The slide is titled "STEPS IN DESIGN FLOW" and is presented on a yellow background. It contains a bulleted list of steps. The first bullet point is "To take a design described in VHDL and implement it on the FPGA board, the compiler performs the following steps:". This is followed by a numbered list: "1. Synthesis", "2. Optimization", "3. Mapping", and "4. Placement and Routing". The slide also features a footer with the IIT Kharagpur logo and the text "NPTEL ONLINE CERTIFICATION COURSES".

- To take a design described in VHDL and implement it on the FPGA board, the compiler performs the following steps:
  1. Synthesis
  2. Optimization
  3. Mapping
  4. Placement and Routing

So, the steps in this design flow is to you have to describe using HDL and then you have to choose the implementation platform, then this particular things synthesis optimization mapping, placement and routing. So, this is the steps which are involved in design flow ok.

So, for today this is it, and next day again we will see more on this particular course so.

Thank you all.