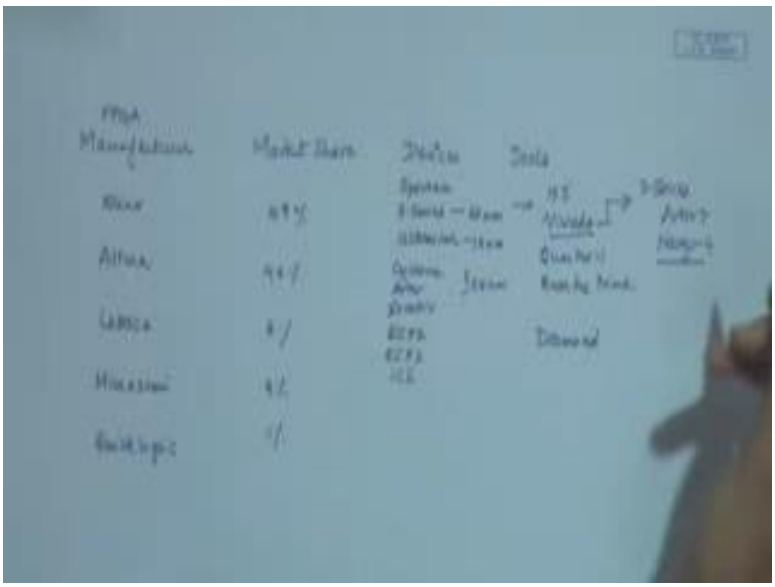
Embedded Systems Design
Prof. N. Vidya
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 12
Tutorial – IV

Hello all. In this tutorial we will learn about basic digital circuits on FPGA board using FPGA design tools. And till now we have studied how to write a Verilog code and how to simulate that code and test the functionality of the design. So, before going on to actually design the circuit in Verilog and using the tools, I would like to tell you something about FPGA manufacturers that are there currently in the industry. So, there are almost 5 major manufacturers in FPGA currently.

(Refer Slide Time: 01:09)



So, those are Xilinx, Altera and Lattice these are the 3 major contributors of FPGA market currently. So, Xilinx share almost 49 percent of the market FPGA market; Altera shares almost 40 percent of the market; lattice has 6 percent of the market and there are two other FPGA manufacturers: Micro semi and Quick logic they share very small amount like 4 percent and 1 percent of market share. So, most of the FPGA's are produced by Xilinx and Altera that are currently used in the industry and I will just quickly list down few of the families that they each of these manufacturers produce.

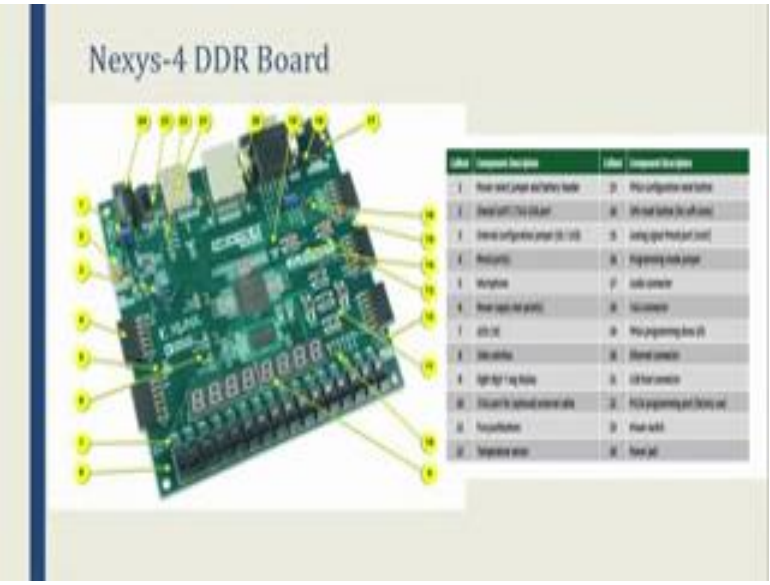
So, in Xilinx we have Spartan series; we have currently we have 7 series FPGs and the most recent one is the ultra scale FPGA's that has come up. So, these are 7 series FPGA's at 28 nanometer technology and some of the ultra scale FPGA's are of 18 nanometer technology. So, these are the most recent devices that have been that is there in the market currently and in Altera we have something called Cyclone family, and Artix family and Stratix family. So, these are 28 nanometer technologies and Altera has been recently acquired by Intel. So, this is Altera devices and in lattice we have something called ECP 2, ECP 3, I series a kind of devices and these are the basic application devices that we find in industry.

So, correspondingly each manufacturers have their own design tools, wherein we use a tools to port the design onto the board onto the devices that they provide. So, Xilinx has ISE design tool, and recently they have come up with Vivado for the 7 series FPGA's and ultra ha had quarter's 2 and recently they have upgraded it to quarters prime; and lattice they have diamond as a tool. So, these are the tool names they are basically do the Verilog design help us to implement the Verilog design on the board.

So, we will study in this tutorial how to use the way Vivado tool and one of the 7 series FPGA's specifically Artix 7 family, which is used in a development board called Nexys 4. So, we will use Vivado design tool to on the development board Nexys 4 to create some design digital dish circuit.

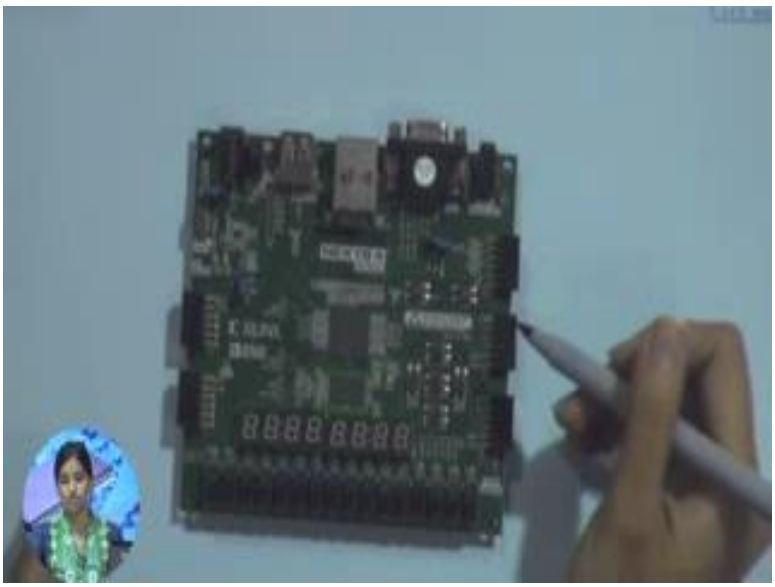
Student: (Refer Time: 04:28).

(Refer Slide Time: 04:38)



Yes it is it comes, zinc series also come in ultra scale. So, this is a basic Nexys 4 board how it looks like. So, I have got the board, you may want to look at it.

(Refer Slide Time: 04:51)



So, this is a basic FPGA board Nexys 4 board; the main component of this board is a FPGA Artix, 7 FPGA it is an Artix 700 t device, will come to know about it more in later slides. So, I will give what are the basic interfaces and the peripherals that are there in this device to. So, main component of this board is a Artix 7 FPGA's, since it is an FPGA development board and we have the power on this board using two interfaces first

is the adapter that is there here and there is a USB micro USB at power. So, these are two methods that we can use to power on the board, I will be using the USB method to power on the word I will show it and late later how does. So, we can select onto which power source we have to choose using this jumper.

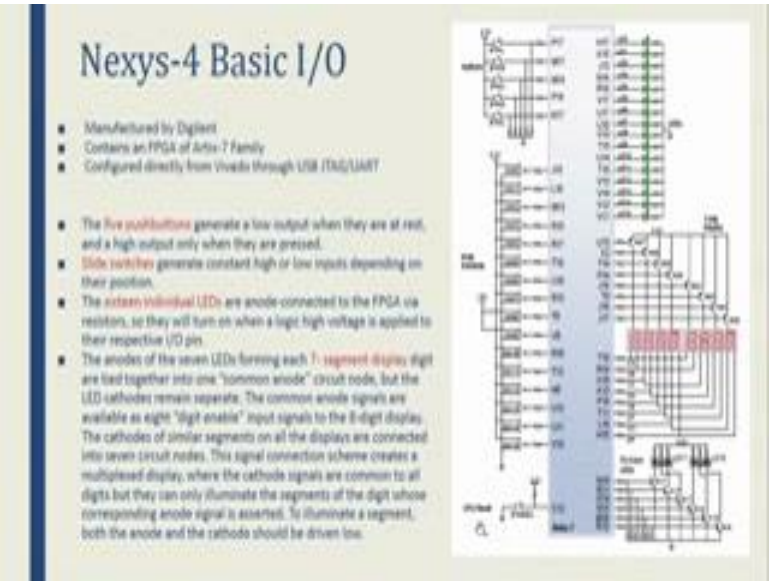
So, we have we now know how to power on the board, now we have to program the FPGA in order to make it work like a circuit that we want to design. So, there are different modes using which we can program the FPGA. So, they are basically four kinds of mode; first is the USB through USB j tag we can program the FPGA, second is through micro SD card. So, we have a micro SD card slot here in this FPGA. So, through the programming file present in the micro SD card we can program the FPGA, and there is an on board SP flash on this board through which we can program the FPGA and that is a non volatile flash memory, so we can even if the power is off we can in the next power on we can load the whatever programming file was stored in the flash we can load that again, without really have to program manually. So, that kind of programming approach is used when the board goes in an application.

And we also have USB HID board, which also can be used to program the FPGA. And this USB port can be used to interface with any USB mouse or keyboard that we use in computers. So, this is a basic programming. So, we now know how to power the FPGA, how to program the FPGA with the programming file. Now we will see what are the memories that we have in this board? So, this board is called Nexys 4 DDR, because it has a DDR s d ram and it is 128 megabyte DDR ram, it also has a serial flash apart from the s p i flash that I talked about 16 MB, it has a Ethernet r j 45 connector which can be used as a Ethernet interface, which allows 10 to 100 mbps Ethernet connection, we have a VGA port, there is an XADC pins that are there in this FPGA.

So, this FPGA supports analog can inputs. So, for that we have these p modes on board and the through these p modes we can directly access that unlock inputs that are there in the FPGA on this particular FPGA. Also in this board we have some other peripherals like accelerators, mono audio output, switches these are the basic iOS consists of the LED’s switch buttons and this 7 segment display. So, we will come to that in the next slide so.

Now I have shown the basic Nexys 4 DDR components and its interfaces how we power it on how we program the FPGA.

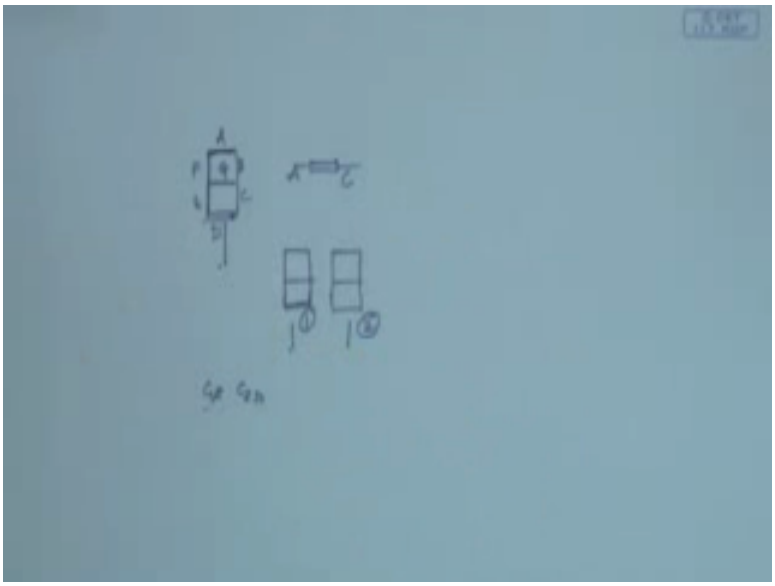
(Refer Slide Time: 08:57)



So, we will now see how what are the basic iOS of this Nexys 4. So, in this slide you can see as I have told we have button switches LED's 7 segment display and their reset button. So, if you see there is a CPU reset and there is a program reset buttons if you press this reset button whatever you have programmed into the FPGA it gets reset you have to program it again or it will load from the non volatile memory whatever it is low what was there in that.

So, we have 5 push buttons as you can see here and they give high output only when they are pressed, they are connected as shown in this figure. So, these are the push buttons and these are the pins in the FPGA where it is connected; we have slides which 16 slide switches, so it depends upon what position it is. So, according to the position it will be 0 or 1 correspondingly the FPGA input or FPGA input will be 0 or 1, also we have 16 LED's corresponding to this each switches here. So, same way whenever there is a high logic the LED's glow up. So, I will like to explain a little bit on the 7 segment display here. So, how the 7 segment display, how we can use 7 segment display.

(Refer Slide Time: 10:38)



So let me draw basic 7 segment display. So, this is a basic one digit of the 7 segment display and each cell each digit is composed of 7 LED's. So, I will name them A B C. So, each LED's have an anode and a cathode connection. So, you can consider it like this. So, how these are connected is for so these are suppose these are 2 digits. So, each digit anode of this digit are connected together. So, these are all A B C D E F G are all LED's. So, the anode connections are all shorted and the cathode connections are taken out for from each of these and they are shorted; hope it is clear for you. So, all the anode connection of a particular digit is shorted, and all the individual cathode like CA, C 1 A, C 2 A or like if this is the first digit and this is the second digit all these are sorry the all these are shorted hope it is clear.

Student: (Refer Time: 12:14).

So, what happens is from the FPGA if we drive low to both of these that particular segment gets eliminated. So, that is how we control the 7 segment display.

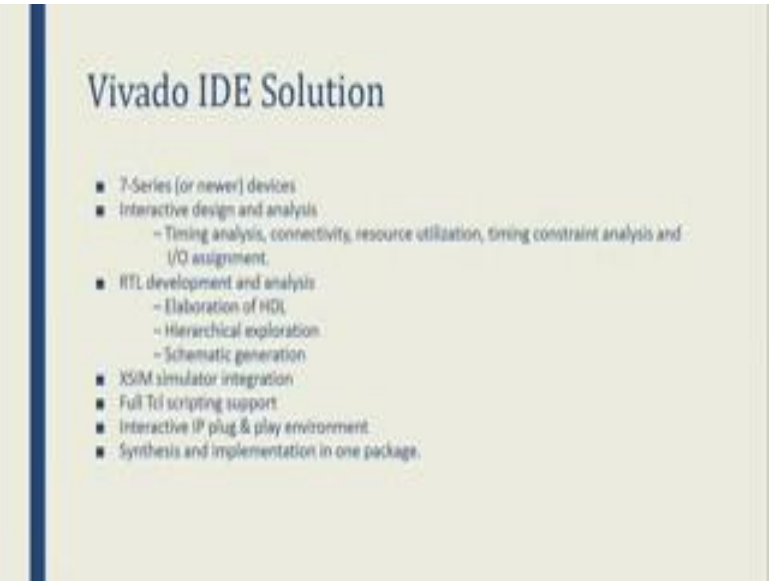
So, this is about the basic iOS. So, in this tutorial I will try to use some of the basic iOS to show you how a very basic circuit is being generated program and use in the book.

(Refer Slide Time: 12:49)

So, let us move on to next. So, I would like to describe few things about Artix 7 FPGA's. So, the FPGA's that is used in this particular board is XC 7 a 100 t 7 specifies the series and 100 t specify the number of logic cells that are used. So, Xilinx wet have their 7 series FPGA's and particularly Artix 7, which is a low cost and low power volume family; this is used for low cost and low power applications and they also have vertex 7 and Kintex 7 family this depends on.

So, the family, different families have different applications and their performance and power consumption differ. So now, I will talk about the Artix 7 series that we are using in this particular board. So, we have 100 k logic cells and we have around 15 k logic slices, each slice have 4 6 input and 8 flip flop. So, we will see how this particularly we can see how this composition of this architecture we can see in the tool. So, these are the basic Artix 7 features which you can find in the data sheet of the device also. So, I have put the references in the last light. So, you can check it out.

(Refer Slide Time: 14:18)

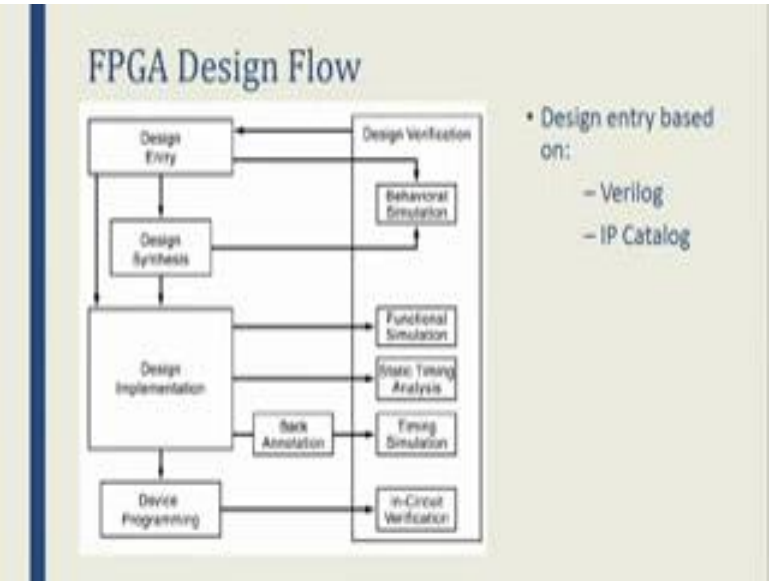


Let us move on to Vivado I d e solution. So, Vivado was introduced by Xilinx for 7 series FPGA's; they have an interactive design, it is an interactive design tool which is used for many things like timing analysis, resource utilization iOS assignment etcetera; it is also used for RTL development and analysis; there is a simulator built in simulator Xsim which is used in Vivado sorry. So, also this in Vivado they support TCL scripting. So, we can directly use command line to launch some of the 2 id solutions like some of the tools or to synthesize or implement the design.

So, we will see how we can do that later; also we in Vivado they provide IP catalog, which can be used to integrate readymade IP's that are available in the industry in your current design. So, everything is included in one particular package. So, that is why it is a Vivado IDE solution.

Student: (Refer Time: 15:32).

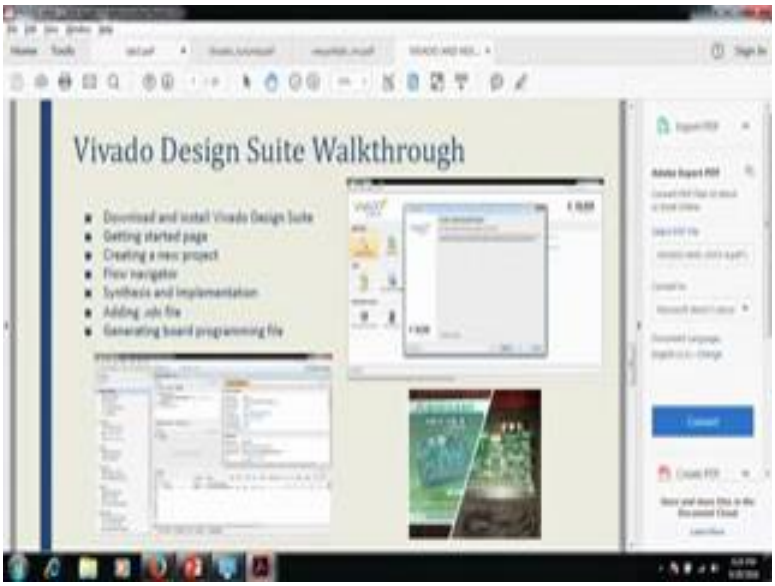
(Refer Slide Time: 15:39)



Yes we can use model sim also. So, before we actually go into the tool and I give you a detail, I would like to recall what we had studied about the FPGA design flow. So, we have first the design entry, here in this tutorial will be using design entry based on Verilog and IP catalog, then the design is synthesized before the synthesize we do a functional verification or behavioral verification of the circuit that we have written in Verilog, that is called behavioral simulation using Xsim tool, and then we implement the design where we do translate, map and place and route of the design; once the implemented design is ready we convert it into a binary form which is called bit stream and then we program it onto the board.

So, this is the basic flow and we also do function simulation, static timing analysis and timing simulation, once the implementation is done to check whether the performance meets our specification, this is the basic of the FPGA design flow which is followed in Xilinx tools mostly and in mostly other tools also.

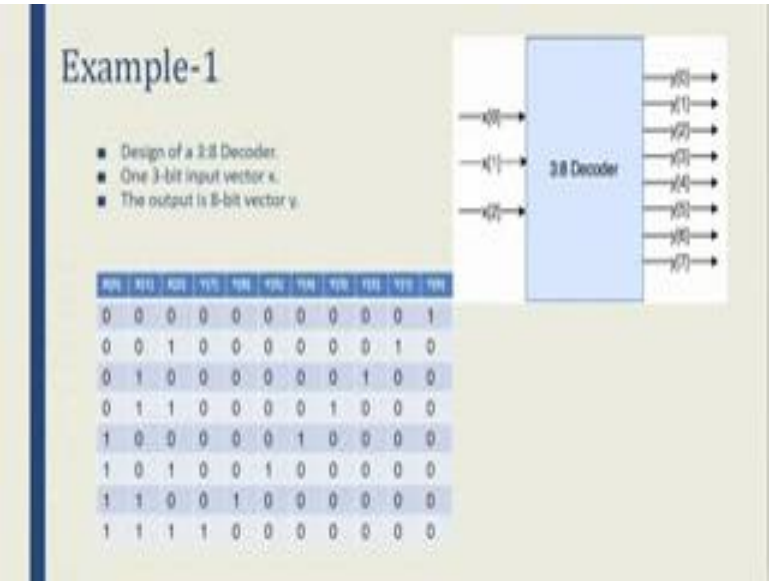
(Refer Slide Time: 16:54)



So, we will now see how we go about designing using a Vivado ID tool. So, first what you should be doing is you should be downloading the web edition Vivado that is available online in Xilinx's website, and you have to first create an account so that you get a web edition license.

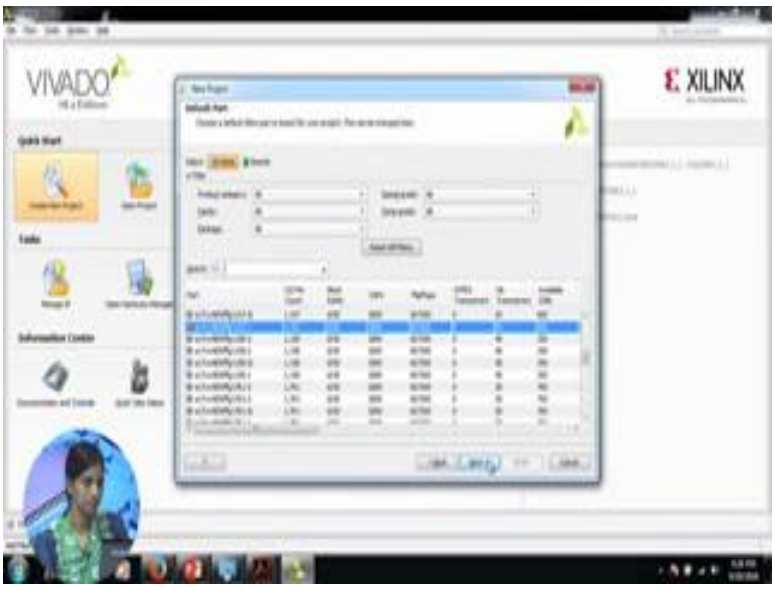
So, you download the software install the license and you have a working Vivado tool for your testing purpose. Some of the features are only available through licensed version like not free, but most of it is free in the web edition for our understanding purpose. So, let us I will just walk open it; before this I would like to explain some more things.

(Refer Slide Time: 17:54)



So, I will be showing a project based on this design. So, will design a 3 is to 8 Decoder which has 3 inputs and 8 outputs and you can see the truth table this is what the circuit behaves as. So, if there is a 0 if X is 0 output will get Y 0 as 1. So, this is correspondingly you know the 3 is to 8 Decoder functionality.

(Refer Slide Time: 18:37)

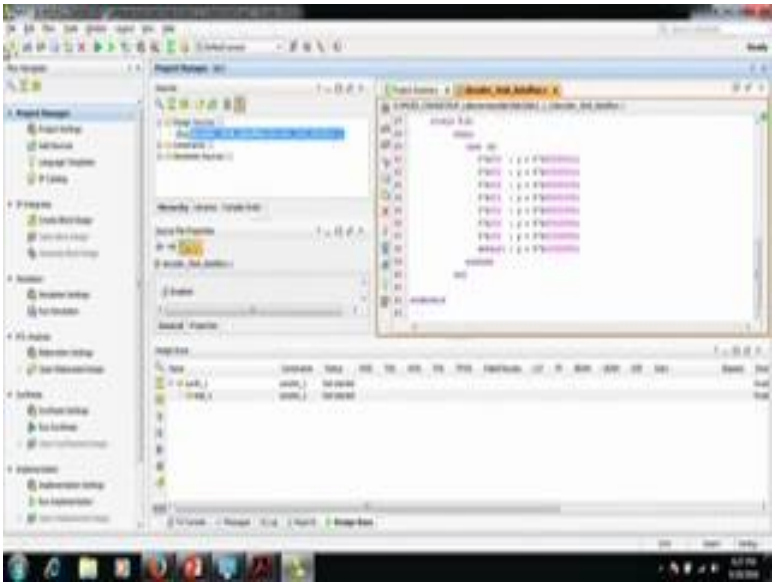


So, this is a starting opening page of the Vivado tool we create a new project first; you specify the project location where you want to make the project. So, I will do it here. So, we are making a RTL project we will select RTL project, we add the sources that we

need to add. So, I have already written a Verilog code for this, I will add directly from here. So, I will add this source if you have existing IP's that you want to reuse you can add it over here, I am just using only the decoder codes I am not adding anything here you can add constraints files also here.

So, in the Nexys four DDR side, Digilence side they have provided the exit X constraints file. I will tell you in detail what a constraint file is basically it gives a physical location and the timing constraint of the device. So, it is already provided by the board vender. So, I have added the. So, here we have a part selection thing, we have path and board. So, if we have our own custom board will have to specify the FPGA device family that we are using. So, if we have a board like as what we have. So, they have separate separately listed out the board. So, we do not have to specify each and everything separately.

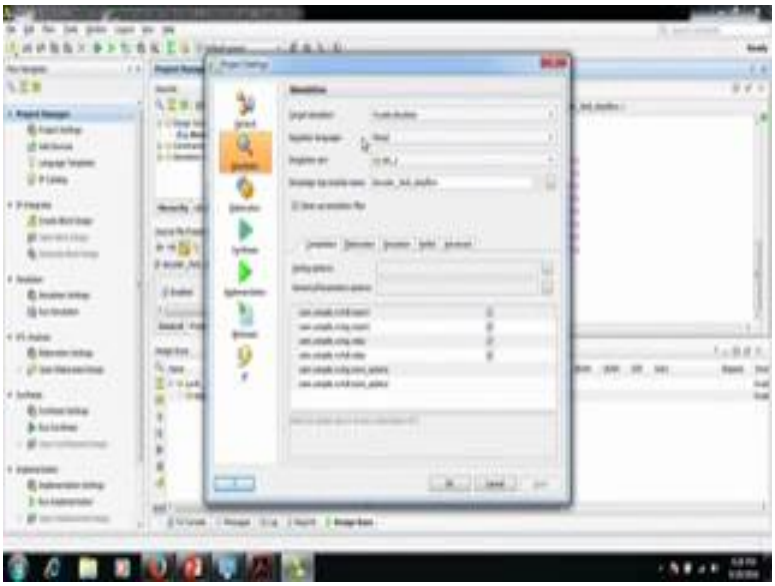
(Refer Slide Time: 20:57)



So, I select a Nexys four DDR board here. So, we are good to go, we are now creating a new project, this is a ID page when you open a project we have flow navigator which has all these things. So, I will go through it one by one. So, we have our codes that have written here which we have added. So, I will just open it in the editor. So, this is the editor and there is a code that I have written in Verilog.

(Refer Slide Time: 21:23)

(Refer Slide Time: 21:50)



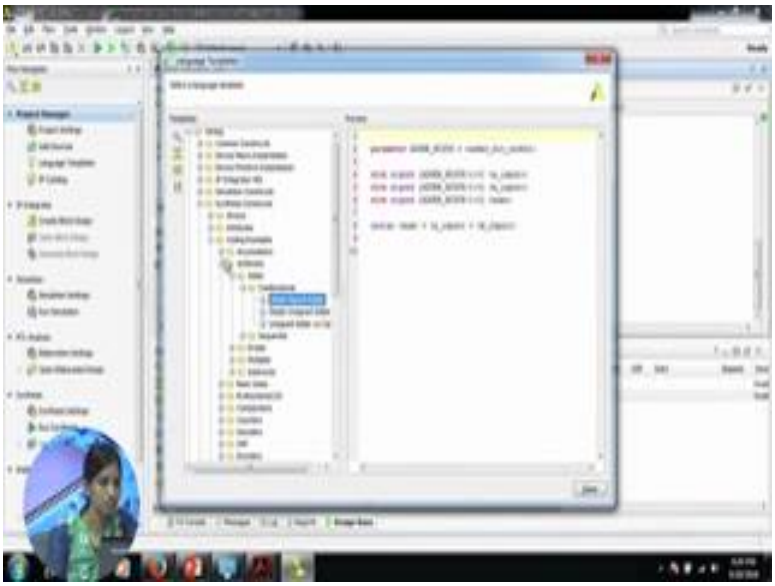
So let us see first what are the project settings that are there? So, we have mentioned which device we are using through the board setting. So, we are using Nexys four DDR and this particular device 100 t device and we are using a target language Verilog which we can change. So, we can have VHDL or Verilog here these are all default settings that we will keep it to a default; then we go to the simulation settings, which is a target simulator that we are using Vivado simulator and the simulator supports by default it supports mixed language single simulation, which means that we can simulate a design which has Verilog as well as VHDL files.

(Refer Slide Time: 23:13)

(Refer Slide Time: 22:19)

And these are all the basic settings which will keep for now as default settings, this is the elaboration which tells what kind of elaboration does a tool do; these are the synthesis set settings. So, we have different kind of settings that are possible there are more options. So, for now we will keep all this as default whatever it is there, and then similarly we have implementation settings which kind of bit stream do we need at the output and the IP core settings. So, we have seen the project settings now. So, similar what we saw in the new project phase like same thing we can add sources here, if later on in the design we find that we need to add a design add a source we can add a source here.

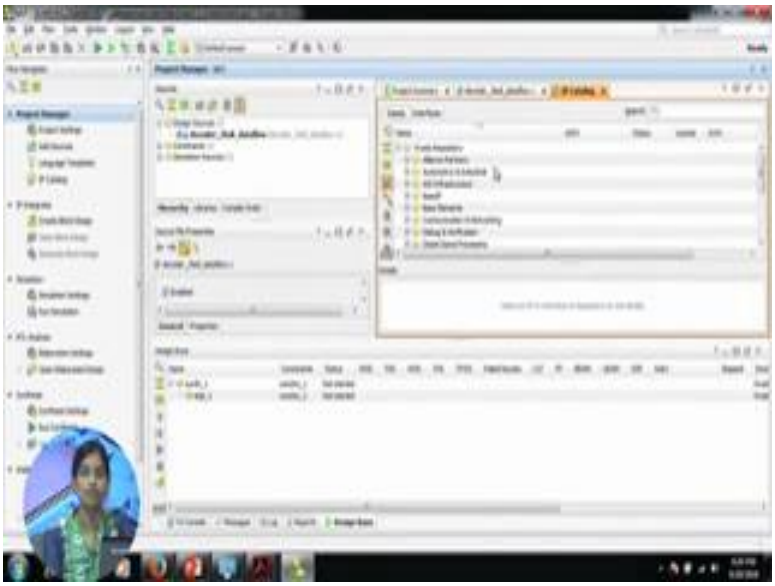
(Refer Slide Time: 23:05)



Language templates; so language template is a very good thing that I feel that the Xilinx people have added. So, we have different languages support with Vivado and they provide a basic template for everything that they have. So, they have some device specific instantiation templates and they have some language specific constraints. So, we studied in we have some Verilog constructs and all. So, they have given. So, let me start with some of the things that you already know like case and conditional assignment. So, they have given the basic syntax and construct here. So, that you can look it up whenever there is some error that you are facing. Similarly there are many things like you can check coding examples, like all the basic digital circuits, and their examples example course.

So, this is a basic adder design; these are basically the language constructs and you can look it up whenever you are free, you can like explore it more they have micro they are device specific instantiation templates also.

(Refer Slide Time: 24:33)



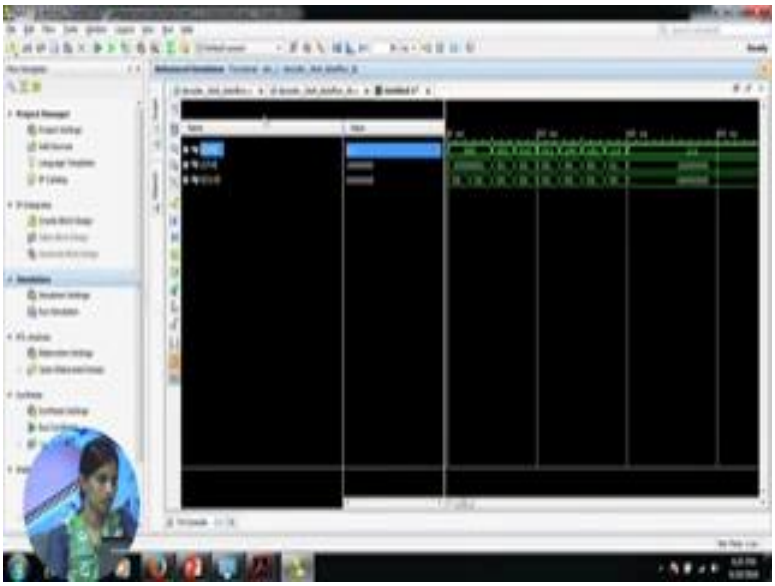
Next we will see IP catalog; IP catalogue is basically whatever is available from the Xilinx repositories Xilinx. So, these are all IP course that Xilinx provide for free with Vivado that we can instantiate in our project or in our design. So, we will be using one of it, if time permits I will try to explain or try to show a design using an embedded processor something called micro blaze. So, we can directly instantiate this processor as a black box in our device. So, that is IP catalogue.

Student: (Refer Time: 25:15).

It is a catalog of IP's intellectual properties. So, if we have suppose we have a decoder, and we have designed it to for different application like we have h 0.264 video d, video encoders and decoders. So, they can be used across application for different purposes. So, that and I in order that that a design can be used for different application we package it as an IP and people sell it.

So, that a new person do not have to write it again, so that is basically what IP catalogue contains such kind of IP's. So, here it is Xilinx specific because I am using Vivado and there are only IP's which Xilinx provides for free. So, we have already seen simulation settings here. So, let me add a decoder simulation file here, so that we can see whether our design is working properly.

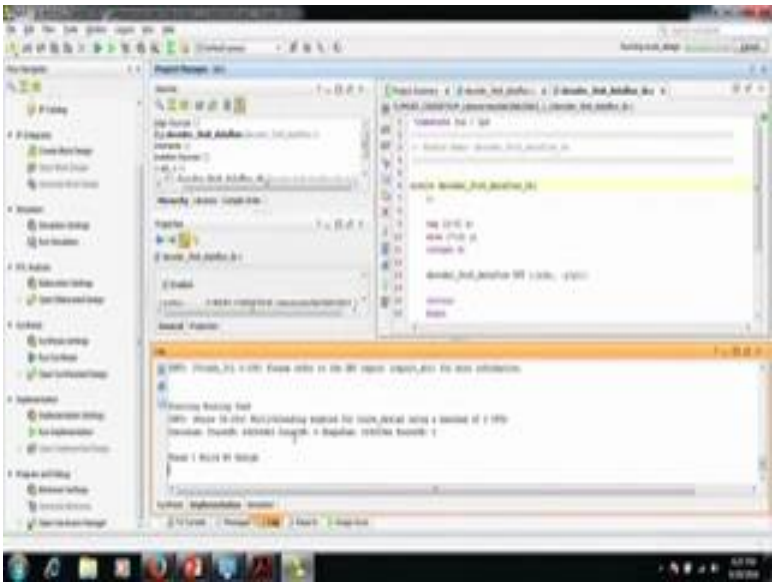
(Refer Slide Time: 26:32)



So, I will add a simulation source. So, you can see it has come here TB. So, this is the basic test bench. So, let us run this and see here you can see that inside the test bench module we have called the design basic design and we are verifying it using some of the inputs. So, here I am giving I am increasing k every call. So, I am just giving 0 1 2 like that like I am just incrementing the input. So, if you see the simulator gives us exact output what the design has to give. So, we have provided 0 1 2 3 4 up until 7 as the input and the simulator tells us that the design that we have written using Verilog, gives us this particular output. So, so the default numeric value that you or the dyadic that you see is hexadecimal, I can convert it into binary or decimal and you can check, I think binary would be easier for you to understand.

So, these are different for different values according to the truth table that we had, we have the output values. So, this is a basic simulation how we do the simulation in Vivado.

(Refer Slide Time: 29:21)

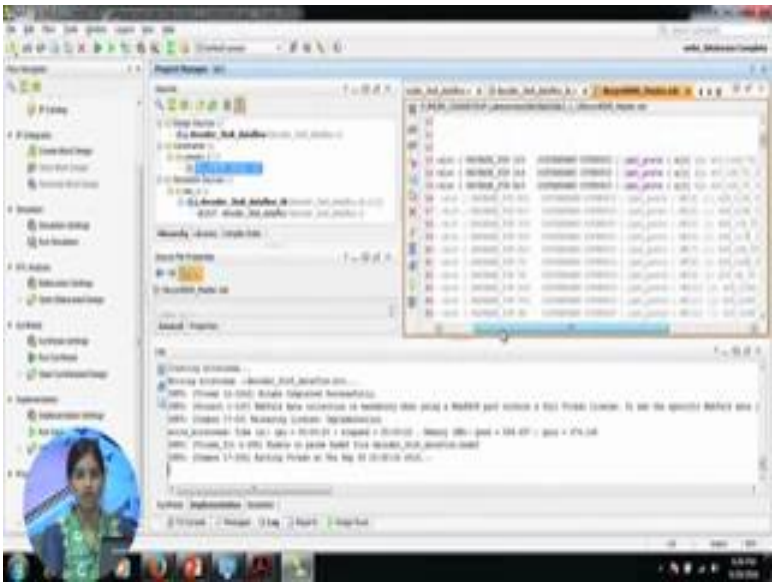


So, let us now synthesize the design, you can see that we have run the simulation synthesis is running and you can see the message lowered. So, now, when the synthesis done we will run the implementation here, it depends upon the machine that it is the tool is running, so it takes time.

Student: (Refer Time: 00:00).

No it is integrated inside the Vivado tool. So, the basic Vivado simulator is Xsim. So, will generate a bit stream, so you can see the project summary here actually, we have completed synthesis without any errors and implementation is like the bit stream generation is still running and we have got two warnings.

(Refer Slide Time: 31:45)



If we have add a synchronous design with lock, we would have got timing information also. So, currently they have not given anything.

So, we have successfully generated a bit stream. So, before we program the FPGA, we have not added a constrained file here. So, there is no way that the particular bit stream will have the information that which input or which output is connected to which port on the FPGA. So, that is why will have to add a (Refer Time: 32:47) I have already added.

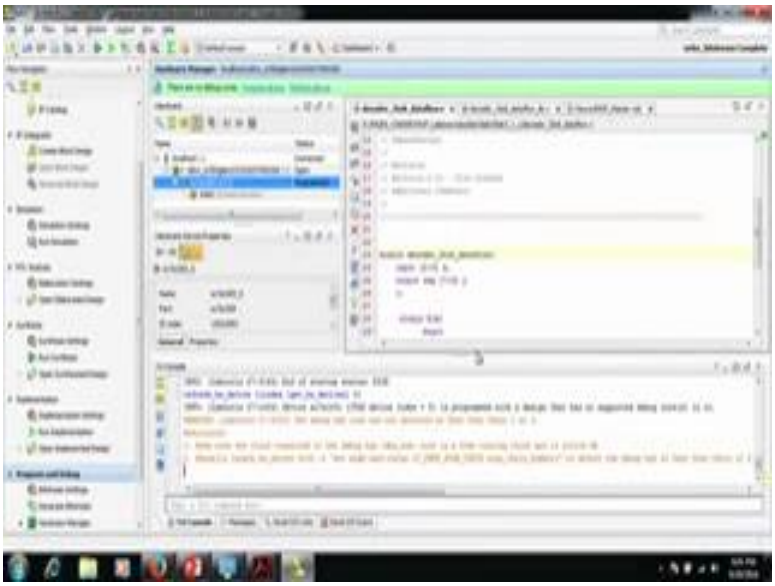
So, in the constraint file I will show you what the constraint file has, the constraint file has all the pins input and output pins of the FPGA, and I have I have connected the inputs X 3 bit inputs to the switches on this board to these switches on this board and the outputs the Y outputs eight bit outputs are connected to the these particular LED's. So, we can see with the change in these inputs we can see LED's different LED's clearly. So, I will connect this and we can see quickly.

(Refer Slide Time: 33:49)

So, this is the power LED which shows us the FPGA is on, now we will open the hard hardware manager will open the target since we have connected this will program it to j tag interface and will program it. So, you can see that the FPGA's program there is a done LED which shows that FPGA's program and if you reset this button the program file is erased completely. So, will have to program it again, so that is what this switch does. So, will program the device again, as you can see 0 0, the switch condition is 0 0 0 input of the decoder, and we have an LED which close.

So, that is the corresponding y output. So, I will just quickly show them basic three is to 8 decoder working using this board, so we have one first LED glowing, we have third LED glowing, 4 5 6 and 7. So, this is the 3 is to 8 decoder that is working on both.

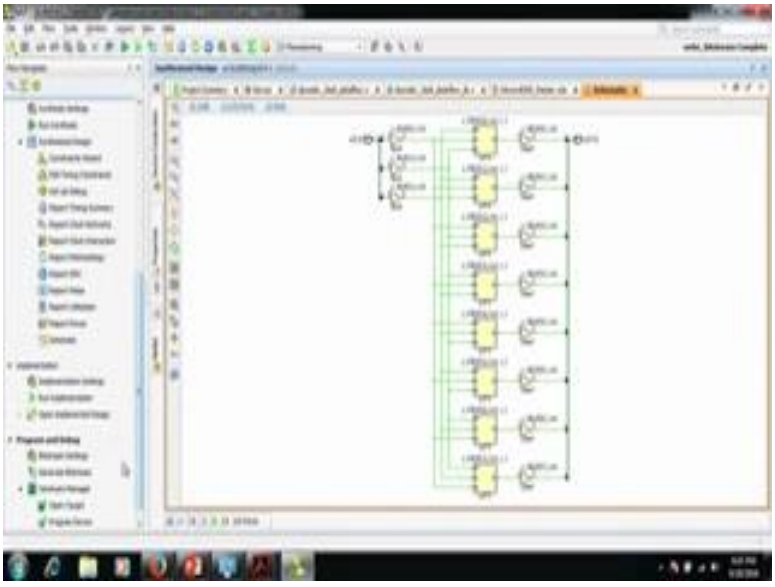
(Refer Slide Time: 35:34)



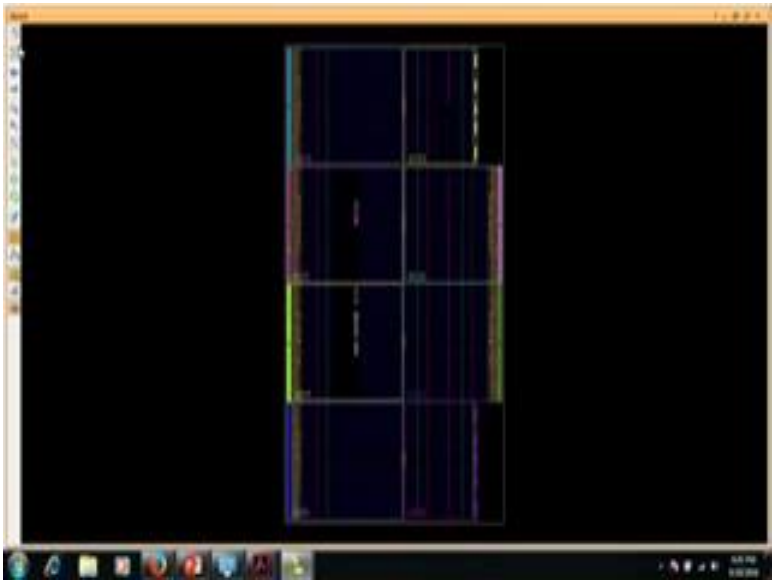
So, hope you have understood how to work with a board and how to put a digital circuit onto the Xilinx Artix 7 board using Vivado design. So, for all other boards and for all the other tools it is very similar, it is just that the naming differs and some of the conventions differ other than that the basic flow remains same.

So, I would like to show you how this particular design that we have ported on can be seen in this layout. So, let us see first the schematic of the design that we have synthesized.

(Refer Slide Time: 36:32)



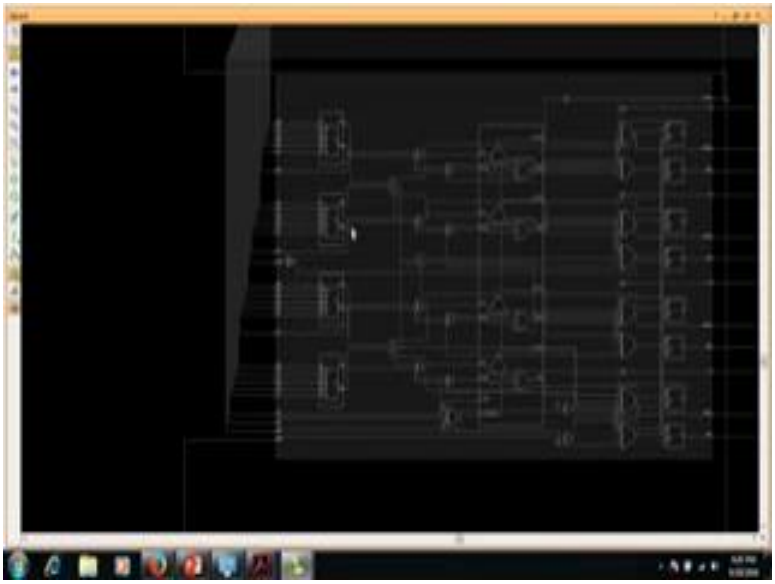
(Refer Slide Time: 36:48)



So, this is the schematic of the device we have some input buffers and output buffers and an l u t is used to design the logic; we can also see the device. So, this is how a particular Artix 7 FPGA looks like, and it has all these components. So, I will just. So, we have I O banks separate I O banks, and we have some 6 I O banks in this particular FPGA and all of these are coloured, you can see the clock regions these are the clock regions these are the different sides that are there and these are the ports and nets.

So, let us see how our input and output is mapped here, if you see here this portion, and I click there are some white dots that have been shown. So, those are the 3 input ports that we have be that has been used inside the board and if you see here it is very not that clear actually. So, these are the white dots that you see is a output ports that we have used and if I click this you can see the routing resources.

(Refer Slide Time: 38:57)



So, I will just show you the basic this is a basic FPGA fabric that we have; this is one slice we have 4 6 input l u t's and six flip flops, some mux's exert it. So, this is how a particular FPGA that you must have learned an organization of an FPGA is and these are the routing resources.

Student: (Refer Time: 39:36).

Yes, actually I will show just actually our design is very small. So, I am not able to locate it, in the next tutorial I will just; will see a bigger design in the next tutorials.

Student: (Refer Time: 40:33).

Yes. So, that is all for today's tutorial we have successfully ported a FPGA design onto the board.

Thank you.