

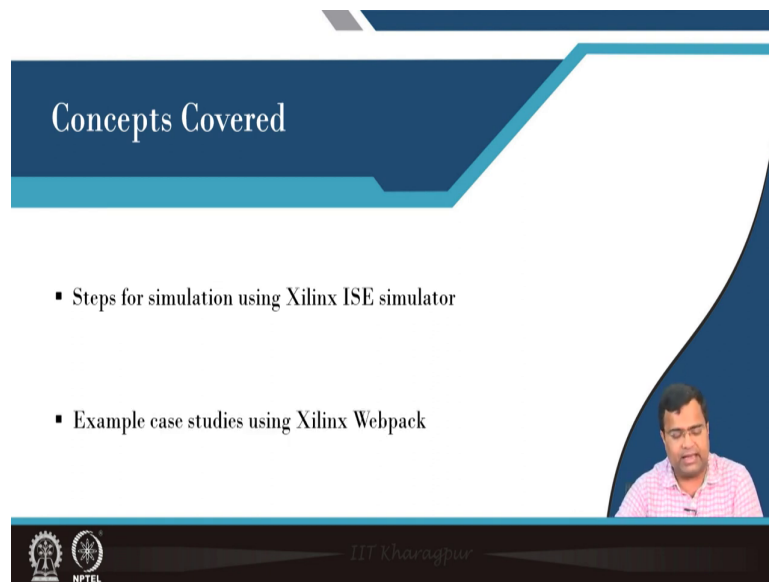
Digital Control in Switched Mode Power Converters and FPGA - based Prototyping
Prof. Santanu Kapat
Department of Electrical Engineering
Indian Institute of Technology, Kharagpur

Module - 07
Introduction to Verilog and Simulation Using Xilinx Webpack
Lecture - 65
Simulation of Verilog-HDL-based Design using Xilinx Webpack - I

Welcome. So, in this lecture we are going to simulate, we are going to first discuss because, how to simulate you know your Verilog code using the Xilinx ISE simulator. Again, I am saying this is not part of the exam as well as the assignment, even if you do not need to install the software. But, it is just for the interested participant who can simulate.

So, this lecture is divided into two parts; in part I, I will be showing some screenshots of how to do that and in part II will be actually going to Verilog Xilinx ISE software and we will write some code there.

(Refer Slide Time: 00:58)



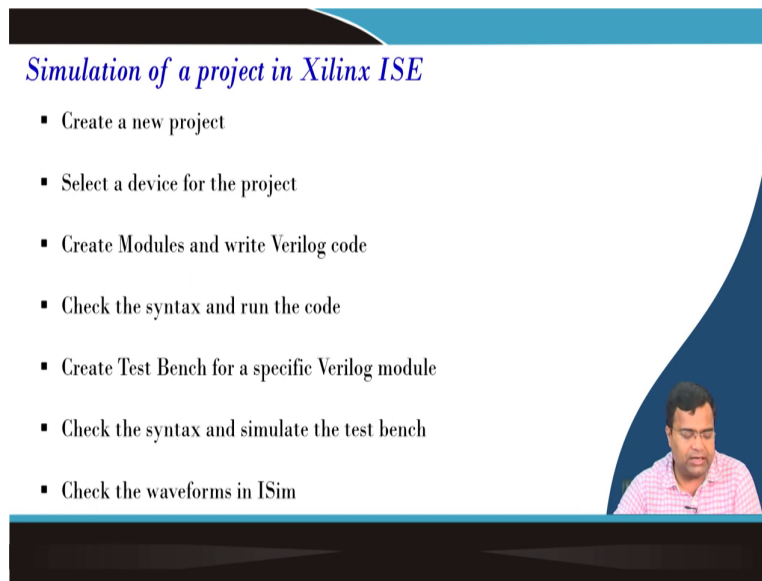
Concepts Covered

- Steps for simulation using Xilinx ISE simulator
- Example case studies using Xilinx Webpack

The slide features a dark blue header with the title 'Concepts Covered' in white. Below the header, there is a white area containing a bulleted list of two items. In the bottom right corner of the slide, there is a small video inset showing a man in a pink shirt. At the bottom of the slide, there is a black footer containing the logos of IIT Kharagpur and NPTEL, and the text 'IIT Kharagpur'.

So, here we will first show the step for simulating using the Xilinx ISE simulator and some example case studies using Xilinx webpack wave pack.

(Refer Slide Time: 01:07)



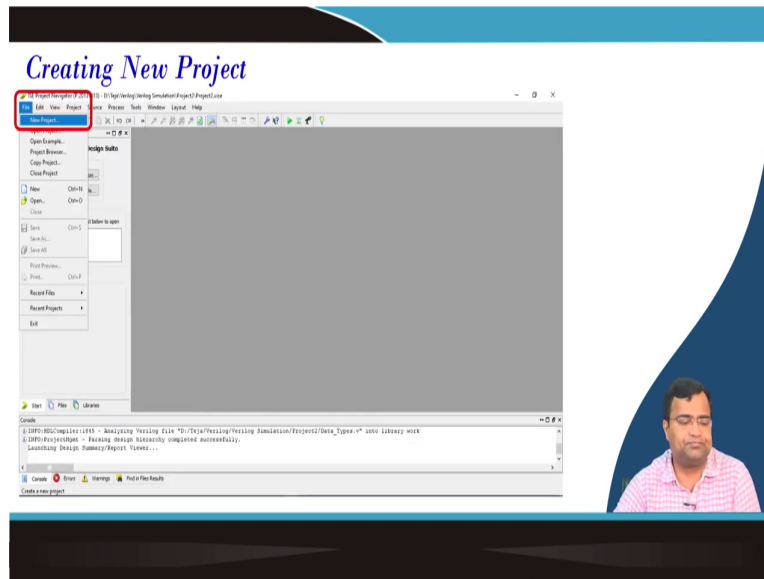
Simulation of a project in Xilinx ISE

- Create a new project
- Select a device for the project
- Create Modules and write Verilog code
- Check the syntax and run the code
- Create Test Bench for a specific Verilog module
- Check the syntax and simulate the test bench
- Check the waveforms in ISim

So, here once you want to simulate in Xilinx ISE. First of all, I am assuming that you know those who are interested you know this for optional Xilinx, you can implement can install it on your computer. And, you know you can check on the web how to install Xilinx ISE. So, we are assuming that you have already installed it.

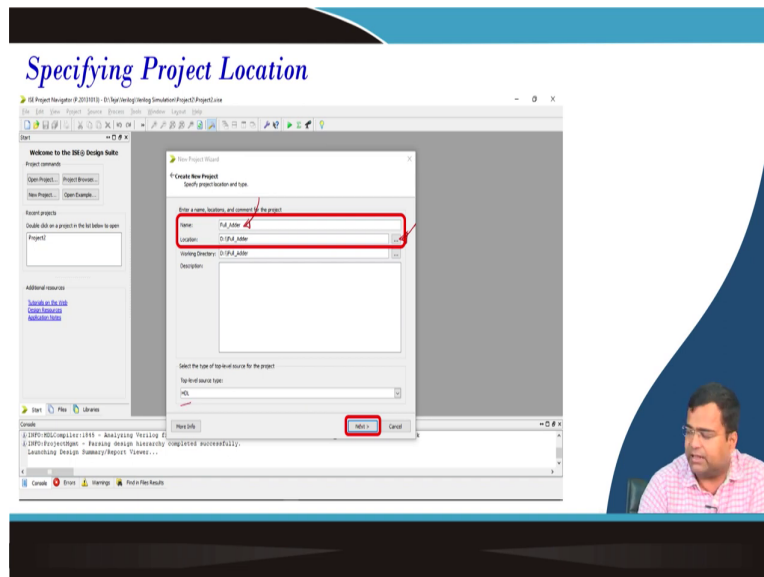
Now, you can create a project. The first steps are this, you need to create a project then you need to select a device. And, we will be showing what device you are going to use for this hardware demonstration the Xilinx device for this course. Then, we will create some modules and write some Verilog code. Then, check the syntax and run the code and create a test bench for a specific Verilog module. And, we need to check the syntax and simulate the test bench. And, we need to check how to see the waveform using ISim.

(Refer Slide Time: 02:01)



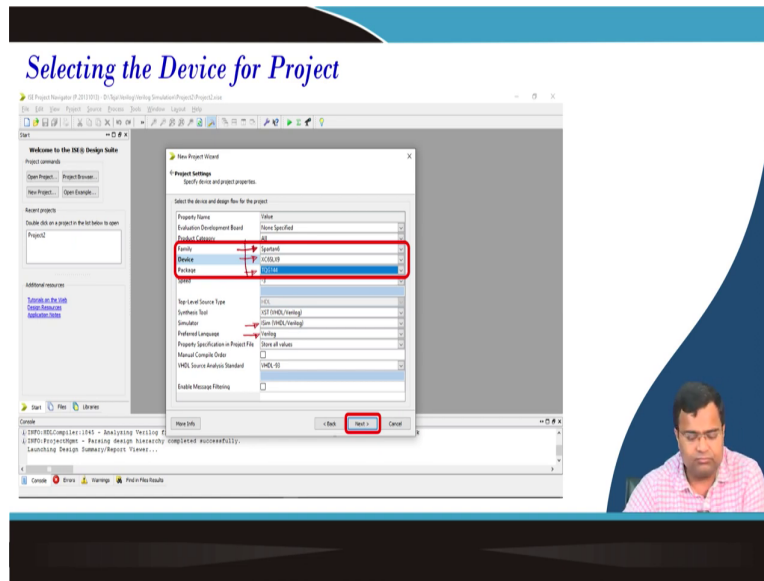
So, this is the first when you install the software. You can see if you go to that, if you click, if you open the software under the file you will create a New Project ok.

(Refer Slide Time: 02:16)



So, in the New Project, if you go this is a screenshot. Then, it will ask you know you have to write something; that means, you can you have to select a folder where you want to keep this project. So, you can create your folder. Then, you have to name you to have to keep the name of this project. So, here the project name is Full Adder ok. Then, you go to the next and do not change anything, this HDL top level should be there.

(Refer Slide Time: 02:41)

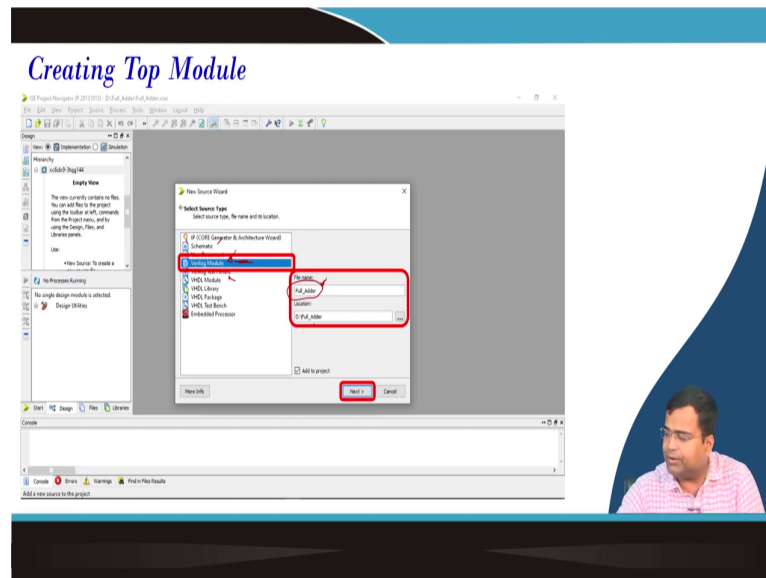


Then, the next stage will go for this stage where it will ask for the device. So, this thing you do not have to bother with, but here in our course we will be using Spartan6 and the particular device ID is this; that means, XC6SLX9 and the package is TQG1 double 1 double 4. And, then we have to specify the preferred language as the Verilog and the simulator ISim ok.

There can be model sim, the third-party software, but we are preferring that whatever Xilinx software is there ISim, that will be used. Then, everything else you keep as it is by default, only you have to specify the Verilog and the ISim and this particular device detail.

So, then it will ask for a new source; that means, it will create an ISE-type structure, here with a project name. Then, in this ISE what do you want to implement? Because, as if you are making an ISE, a virtual ISE will have several pins depending upon what exactly you want to implement and how many IO ports are there.

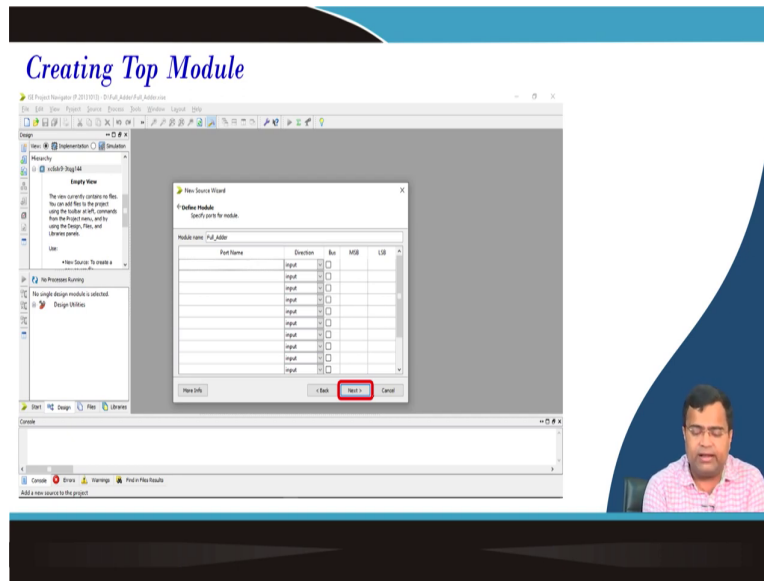
(Refer Slide Time: 04:22)



Then, you have to first add a New Source; that means, we want to write a Verilog code, New Source. Under this source, there are many options; Schematic, then we have to add the Verilog Module that is it. And, if you want to test, there is a test module also there, which will come next. So, under the Verilog module, this module will ask for the Verilog File name.

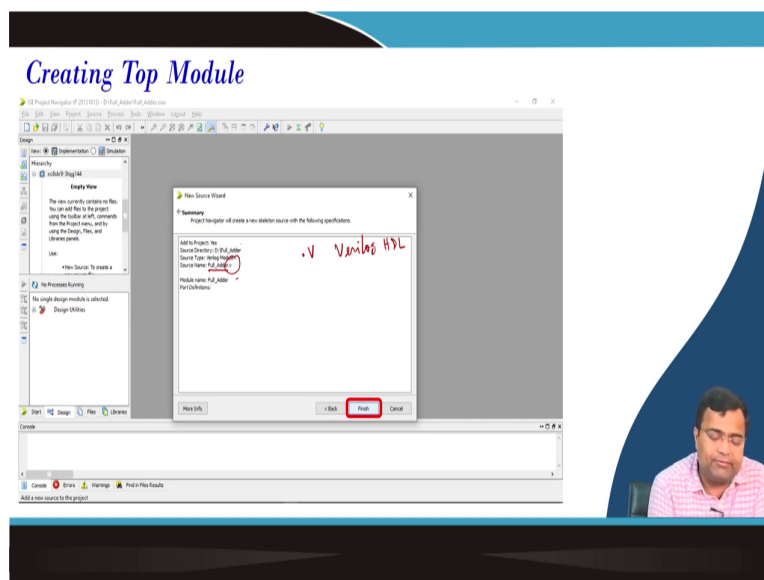
So, again you are making a Full Adder as the File name and it is also under the Location of Full Adder which is a project name. So, next remember the File name can be different. I mean it need not be the same as your project name. But, whatever File name will make, the Verilog dot v file will be created by that name. So, you should not change; that means, whatever you are doing should be consistent.

(Refer Slide Time: 05:06)



Then, if you go Next, it will ask how many input-output ports you want. But, I think you do not need to do anything, because you will be declaring the port inside the Verilog code. So, you do not leave it is and then go Next.

(Refer Slide Time: 05:21)



If you go Next, then it will ask if can you will see that your file name is Full underscore Adder dot v. So, it is a dot v file, dot v which is a Verilog file, Verilog extension is dot v HDL ok. And, the module name is Full Adder ok and you are also keeping the directory which is D drive in Full Adder in this case.

(Refer Slide Time: 07:24)

Writing Verilog Code for Top Module

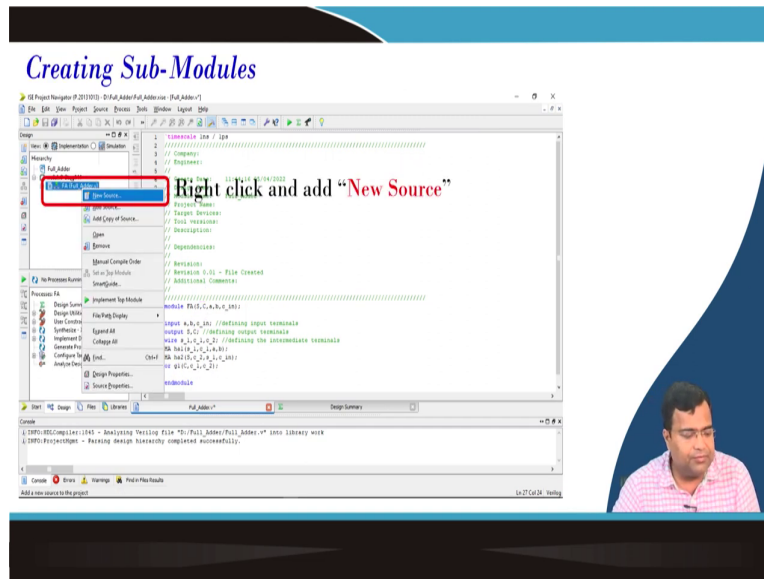
```
1 //*****  
2 //*****  
3 // Company:  
4 // Engineer:  
5 // Create Date: 11/04/16 09:14:2022  
6 // Device Name:  
7 // Module Name: Full_Adder  
8 // Target Device:  
9 // Tool Version:  
10 // Description:  
11 // Dependencies:  
12 //*****  
13 //*****  
14 // Revision:  
15 // Revision 0.01 - File Created  
16 // Additional Comments:  
17 //*****  
18 //*****  
19 module Full_C_a_b_c_in;  
20  
21  
22  
23 input a,b,cin //defining input terminals  
24 output S,Cout //defining output terminals  
25 wire s1,c1,c2 //defining the intermediate terminals  
26 HA HA1(a,b,cin) //defining the first half adder  
27 HA HA2(s1,c1,cin) //defining the second half adder  
28 OR OR1(s1,c1,c2) //defining the OR gate  
29 S = OR1.out1;  
30 Cout = OR1.out2;  
31 endmodule
```

module FA(S, C, a, b, C-out)

Next, now you fill up your Verilog code; that means, if you want to write a Full Adder, first you can see the summer, sum block. You know if it is not visible, I am writing here module, then it is the name as the Full Adder. So, here the module name is again different because we have a Full Adder dot v file. But, it is better to maintain the same name rather than name another name. So, we can use the Full Adder here, then S, C, a, b, c in.

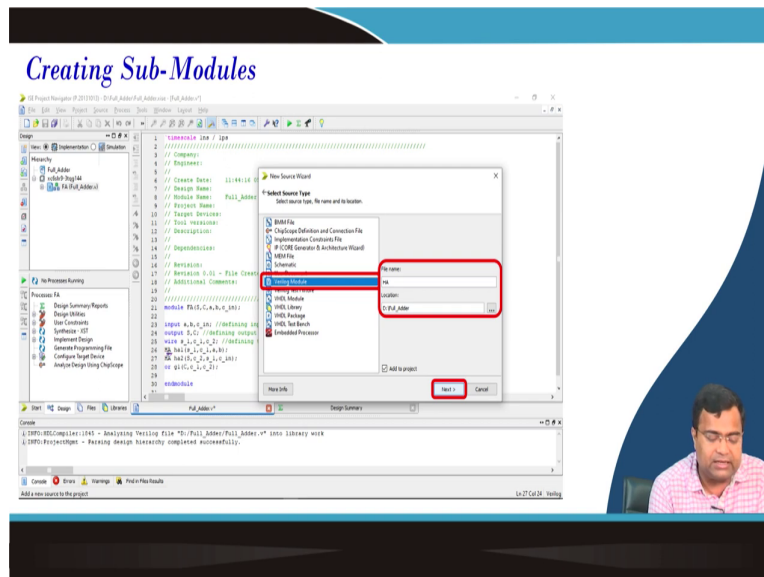
Then, we have to define input as a b c in, the output as S and C. Then, we know there are errors there are wire connections between the S bar C bar, c 2, and then Half Adder, we need to take call 2, instantaneous 2 Half Adder and we have to connect accordingly. And, then it will go to or gate to generate the carryout. And, then that is the end of the module.

(Refer Slide Time: 08:40)



Once, you write this code because we have discussed about this Verilog code; then you have to create a New Source under this Full Adder.

(Refer Slide Time: 08:46)



There you can have a Half Adder module because we have called Half Adder, but not you have not yet designed it. So, under this Half Adder, you have to create a Half Adder module name. Again, Verilog Module, the name is Half Adder. It must be the same as the module that we have instantiated here.

(Refer Slide Time: 09:04)

Creating Sub-Modules

The screenshot shows the Xilinx ISE Project Navigator interface. A 'New Source Wizard' dialog box is open, titled 'New Source Wizard' with the subtitle 'Specify ports for module.' The dialog contains a table with the following columns: 'Port Name', 'Direction', 'Bus', 'MSB', and 'LSB'. The 'Port Name' column is populated with 'input'. The 'Direction' column has checkboxes for 'input' and 'output'. The 'Bus', 'MSB', and 'LSB' columns are empty. The 'Module name' is 'ha'. The 'OK' button is highlighted with a red box.

(Refer Slide Time: 09:05)

Creating Sub-Modules

The screenshot shows the Xilinx ISE Project Navigator interface. A 'New Source Wizard' dialog box is open, titled 'New Source Wizard' with the subtitle 'Specify ports for module.' The dialog contains a table with the following columns: 'Port Name', 'Direction', 'Bus', 'MSB', and 'LSB'. The 'Port Name' column is populated with 'input'. The 'Direction' column has checkboxes for 'input' and 'output'. The 'Bus', 'MSB', and 'LSB' columns are empty. The 'Module name' is 'ha'. The 'OK' button is highlighted with a red box.

Then, again we have to create a Half Adder dot v file.

(Refer Slide Time: 09:09)

Creating Sub-Modules

Write the Verilog code for Half Adder

```
1 //***** 2SA / 2SA *****
2 //*****
3 // Company:
4 // Engineer:
5 // Design Date: 11/09/13 09:04:2022
6 // Design Name:
7 // Module Name: HA
8 // Project Name:
9 // Target Devices:
10 // Tool Versions:
11 // Description:
12 //
13 //
14 // Dependencies:
15 //
16 //
17 // Revision:
18 // Revision 0.01 - File Created
19 // Additional Comments:
20 //
21 module HA;
22 //
23 //
24 //*****
25 endmodule
```

And, it will generate this Half Adder another module.

(Refer Slide Time: 09:13)

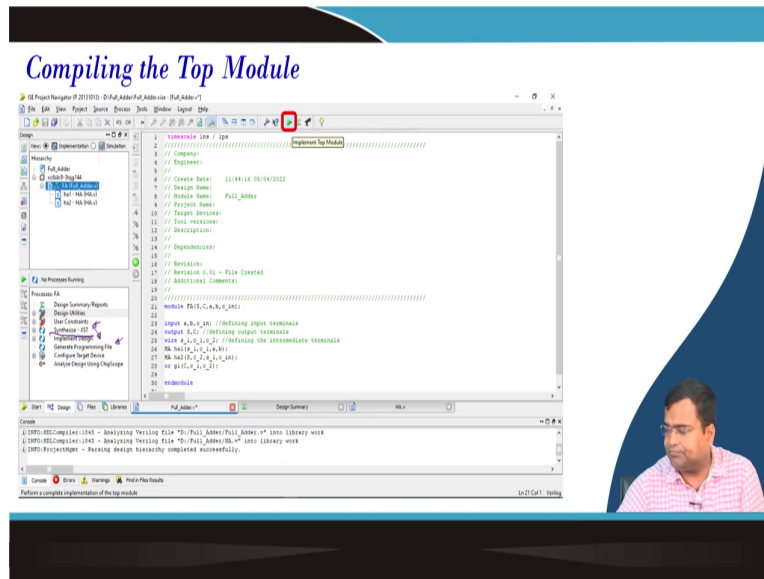
Writing Verilog Code for Sub-Modules

One "Top Module" and two "Sub Modules" are created

```
1 //***** 2SA / 2SA *****
2 //*****
3 // Company:
4 // Engineer:
5 // Design Date: 11/09/13 09:04:2022
6 // Design Name:
7 // Module Name: HA
8 // Project Name:
9 // Target Devices:
10 // Tool Versions:
11 // Description:
12 //
13 //
14 // Dependencies:
15 //
16 //
17 // Revision:
18 // Revision 0.01 - File Created
19 // Additional Comments:
20 //
21 module HA;
22 //
23 //
24 //*****
25 endmodule
```

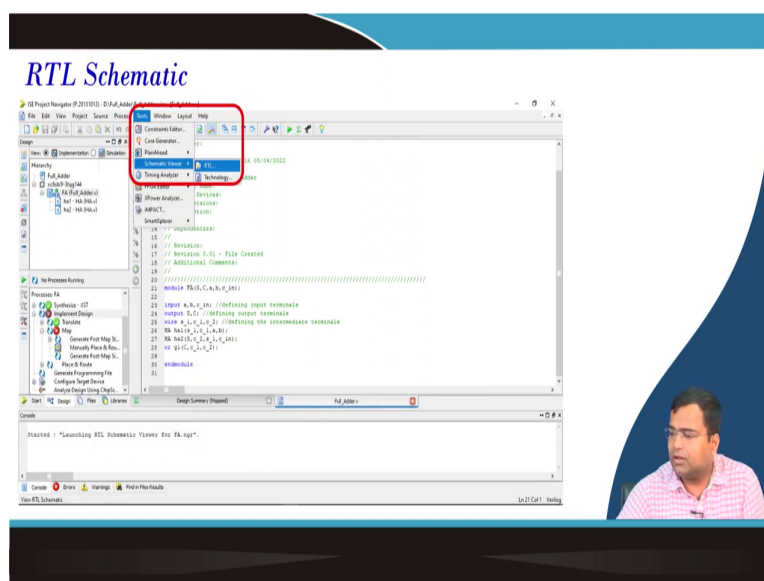
This Sub Module comes under this Sub Module and they are calling. So, Half Adder module you have a sum out, carry out, a, and b. So, you can write in terms of xor gate and that we have discussed earlier. So that means, two Half Adder now are called instantiate which is why they are coming under one Full Adder.

(Refer Slide Time: 09:33)



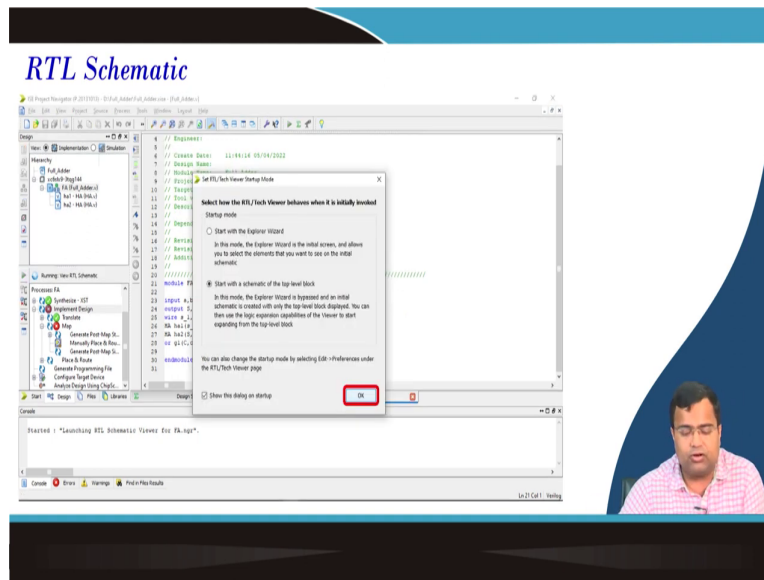
Once, you are done, then you go to implement the top module. You do not need to implement it because you are not using Verilog FPGA implementation right now. You can end up with synthesis because synthesis will have an RTL block and beyond the RTL block it will go for implementation, where it will generate you know, it will map into the target FPGA device. And, final it will generate the bit file which I will be showing when we go to the hardware demonstration. But, here you need to know the RTL logic.

(Refer Slide Time: 10:10)

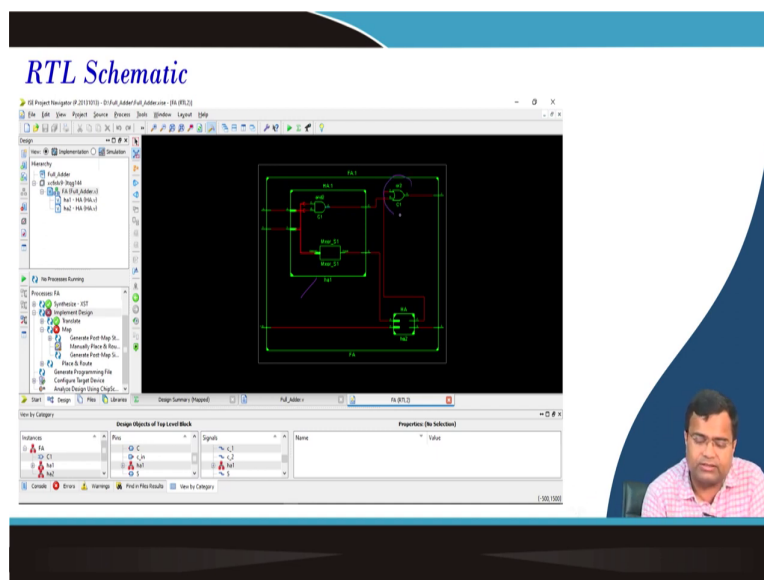


So that means, then you can see; that means if you implement it will show synthesis is done correctly, then translate. Then, you can see the schematic, there is a tool Schematic Viewer RTL.

(Refer Slide Time: 10:24)

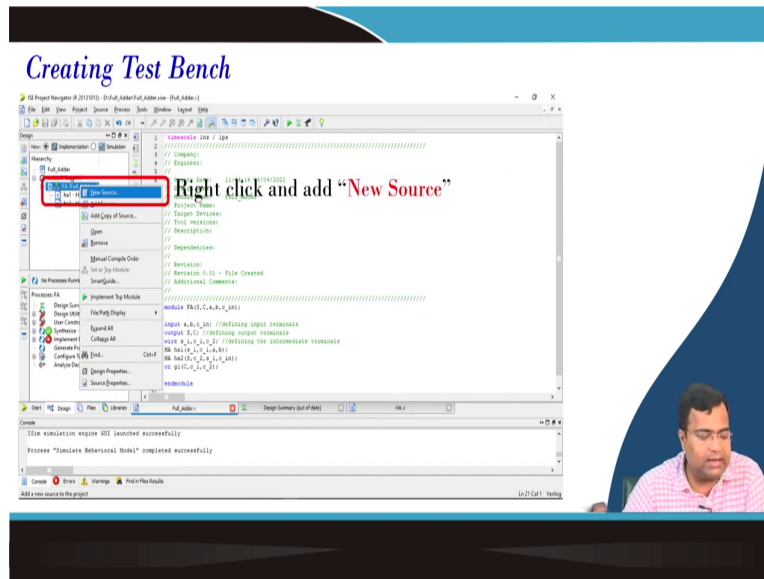


(Refer Slide Time: 10:25)



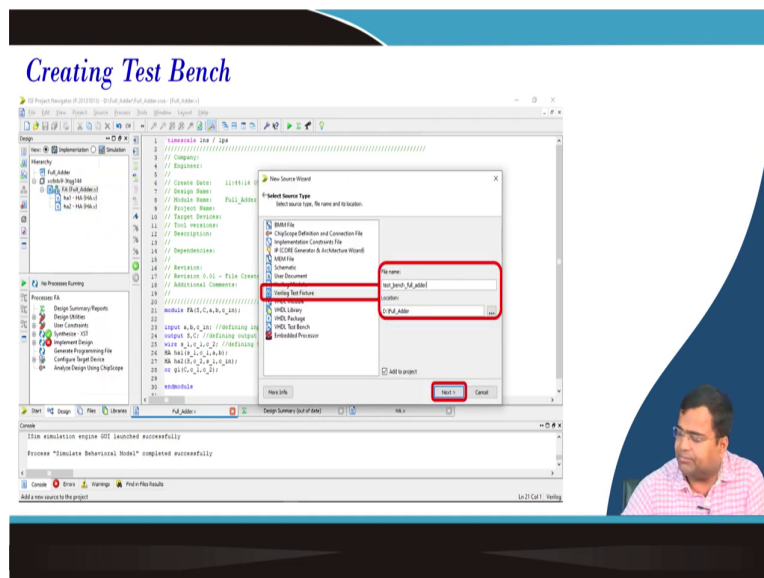
It will show that it is the Full Adder circuit which consists of 1 Half Adder here like. So, this is I think the Full Adder and Full Adder also have an external gate right? This is Half Adder 1, this is Half Adder 2. So, the first Half Adder will take the two inputs and the second Half Adder take the output of the first Half Adder and the carry-in.

(Refer Slide Time: 10:54)



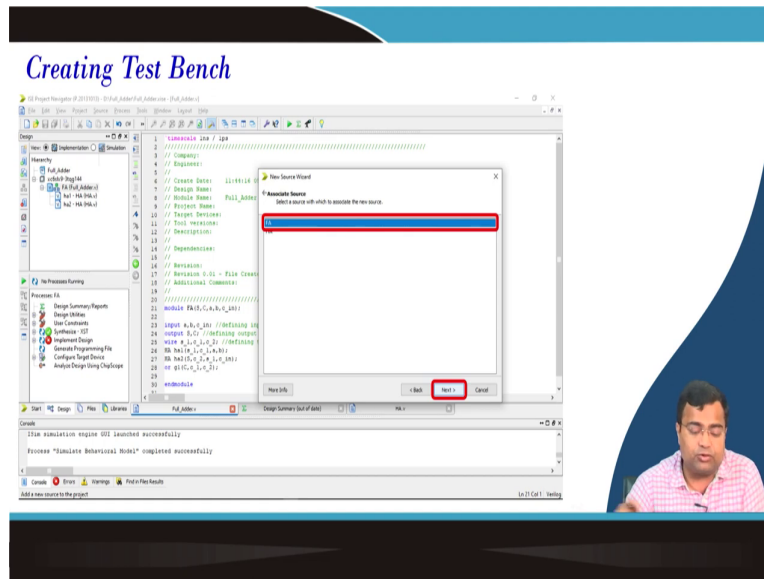
Then, you can add a New Source, if you want to test it.

(Refer Slide Time: 10:58)



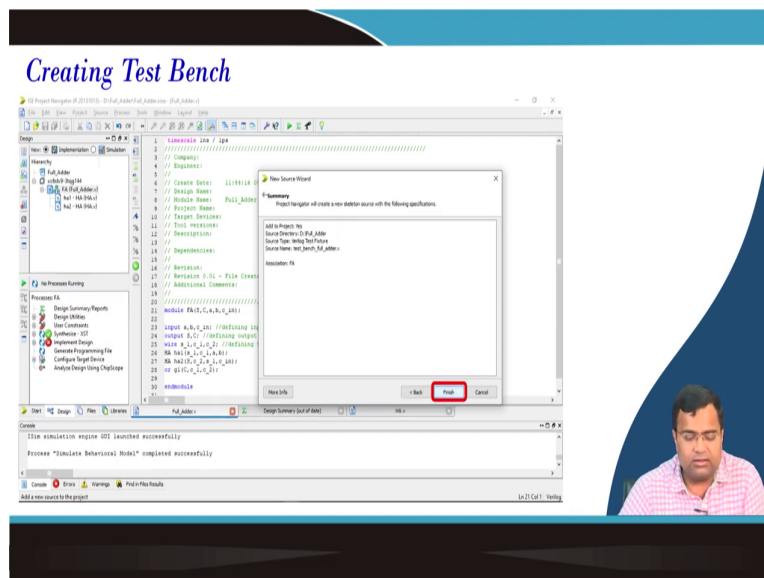
Go to the Verilog Test Fixture; that means, if you right-click here, New Source Test Fixture. Now what; that means, test bench for Full Adder? So, you need to make sure that you are testing the Full Adder, because of the Full Adder circuit you want to test. You can also selectively test the Half Adder circuit, customizable, no problem. So, you have to select which block you have to test. Then, go you have to create a test plan.

(Refer Slide Time: 11:25)



Then, it will associate with Half Adder or Full Adder.

(Refer Slide Time: 11:30)



So, if you make it Full Adder, then it will create a Test Bench circuit.

Thank you very much.