Exploring FPGAs For Design Of Digital Hardware

Sayak Haldar

sayakhaldar@ymail.com

Shuvam Bosana

shuvambosana0705@gmail.com

Subham Banerjee

subhambansphs@gmail.com

Students of

Indian Institute of Engineering Science and Technology, Shibpur

Abstract

This is the continuation of our last semester project. Here, we are going to show you how to implement a hardware design using FPGA with VHDL.

Contents

1 Introduction:

The focus on hardware in our country is not at par with that on software or other coding related subjects. India has come out with flying colors in the race to obtain independence and recognition in the field of Software and Information Sciences. But India, till now has not progressed much in hardware. As a result it has imported the required IC's from various other developed nations for industrial purposes and also for lab tests.
Likewise we have found little stress in hardware in our curriculum till the 4th semester. An extended portion of the Digital Logic course or a wing of the Computer Architecture and Organization course can easily be introduced in the curriculum. It can be an elective course as well. We have carried out several experiments using an FPGA in this semester as part of our Mini Project and we feel that this can easily be incorporated somewhere inside the hardware curriculum of our department in the 4th semester. Carrying out experiments of a similar nature as the Digital Logic or Computer Architecture Lab using an FPGA or a CPLD, programmed by some hardware description language like VHDL was our motive in this project. We have used the Xilinx Spartan-3 FPGA kit and have compiled all VHDL programs in the Xilinx ISE 10.1. We have created data sheets for laboratory experiments using this FPGA which might come in handy if a course like which we are talking about really comes into the big picture in our department.

2 Why FPGA?

Now, the question is obvious. Why people are going to use FPGA when there are thousands of ICs available at the market? So, here we are going to provide a list of advantages of using FPGAs for lab purpose:-

- The full form of FPGA is Field Programmable Gate Array- an integrated circuit with large no. of logic gates. It is initially unprogrammed. Users can program it with the help of Hardware Description Language(HDL) at their workfield. And with the help of a single FPGA any type of circuits(from low to high circuit complexity can be designed). So, A single FPGA board serves as a substitute to all tools and ICs used in an experiment

- The users need not to be concerned about the interconnections any more. Since this setup will negate the traditional wire chip breadboard concept and hence relieve us in the lab from entangled wires.

- Debugging will be easier. No need of checking the interconnections(wires) while the circuit produces a faulty behaviour.(Though several issues are to be concerned. Like:-FPGA I/O overflow issue etc)
- No need to concern over the short lab time period. Since the interfacce softwares are open-source and easy to install. So, Students can write the programs at their home and can carry the written programs to the lab using pendrives. Also the written codes are machine independent(computer machine independent). Hence a "Platform Independent" environment can be offered.
- Work can be saved from time to time. So, no need to design the code again from the very beginning. The existing code(already written) can be utilised.
- More number of circuits with various levels of complexity can be achieved.
- Concepts will be more clear. Since the students have to know the underlying implementation of ICs for designing the circuit now. Through this process, students will also come to know about the modern techniques adopted by leading hardware companies for designing circuits.

3 Steps of a typical FPGA based design:

- Architectural design
- Choice of language(VHDL or Verilong)
- Editing programs
- Compiling programs
- Synthesizing programs
- Configure target device
- Download the written programs in FPGA via JTAG port
- Documenting programs

4 Our Work With FPGA:

4.1 The FPGA We Choosed:

Our work is based on the Xilinx Spartan-3(family name) XC3S200(device name) FPGA model. It consists of the following things:-

System Gates	200k
Logic Cells	4,320
Block RAM bits	216k
Distributed RAM bits	30k
Max Single ended I/O	173
Max Differential I/O	76
Availability	2H03

4.2 The Interface Software we Choosed:

We used Xilinx ISE 10.1 as the interface software. Xilinx ISE is a useful tool produced by Xilinx for:-

- Synthesis and analysis of HDL designs
- Enabling the developer to synthesize their designs
- Performing timing analysis
- Examining RTL diagrams
- Simulating a design's reaction to different stimuli
- Configuring the target device

and we used the 10.1 version of it.

4.3 The Hardware Description Language We Choosed:

We choose Very High Speed Integrated Circuit Hardware Description Language(or VHDL) for our work.

4.4 Implementing A simple Digital Design in Xilinx FPGA With The Help Of Xilinx ISE Designing Tool:

We will discuss the steps for two common OS used by the programmers.
1.Windows Operating System 2. Linux based Operating System Ubuntu.

4.4.1 Step-1:Installation Guide

For Windows Operating System

Click the setup.exe from the installer folder named "Xilinx" and then do the rest.

For Ubuntu Operating System

- Copy the installer file to Ubuntu Home(of root directory).
- Open the terminal
- Type the following commands one by one
  - cd Home(Change the directory to Home)
  - cd Xilinx
  - cd bin
  - cd lin or cd lin64
    - cd lin(if your Ubuntu machine is of 32 bit)
    - cd lin64(if you Ubuntu machine of 64 bit. Note that:- Xilinx ISE 10.1 is not perfectly compatiable with ubuntu 64 bit. But the latest versions of Xilinx ISE are also perfectly compatiable with ubuntu)
  - ls
    - Check that if both setup and _setup exists in that folder or not.
  - sudo chmod 777 setup
    - Then the user will be asked for sudo password
  - sudo chmod 777_setup
  - ./setup and then do the rest.

4.4.2 Step-2:Starting The ISE Software:

For Windows Operating System

To start ISE, double-click the desktop icon.

Or,

Start ISE from the start menu by selecting:-

Start→ All Programs→ Xilinx ISE 10.1 → Project Navigator

For Ubuntu Operating System

- Open Home.(User must be logged in as root user)
- Open ISE
- Open bin
- Open either lin or lin64(based on the bitrate of the OS)
- Click on ise and the Project Navigator will open.

4.4.3 Step-3:Create A New Project

Create a new ISE project which will target the FPGA device on the Spartan-3 Startup Kit demo board.

1. Select File → New Project... The New Project Wizard appears.
2. Give a project name and mention the location properly and also set the top-level source type as HDL.

   Note: Here, we first choose the project location as ISEprojects/BESU-programs then mention the Project Name FULL_ADDER. So a folder named "FULL_ADDER" will be automatically created in the mentioned path

3. Click next to move the device properties page.
4. Fill in the properties in the table as shown below.
   - Product Category: All
   - Family: Spartan3(family name could be different. Users are advised to check the FPGA model they are using. Otherwise they will face problems at the time of downloading the .bit file to the FPGA kit)
   - Device:XC3S200(This could be different too. Users are advised to check it from the FPGA model they are using)
   - Package:FT256(this could be different too)
   - Speed:-4(this can be set to either -4 or -5. Setting the value as -5 will not cause any problem at the time of downloading the .bit file to the FPGA kit)
   - Top-Level Source Type:HDL

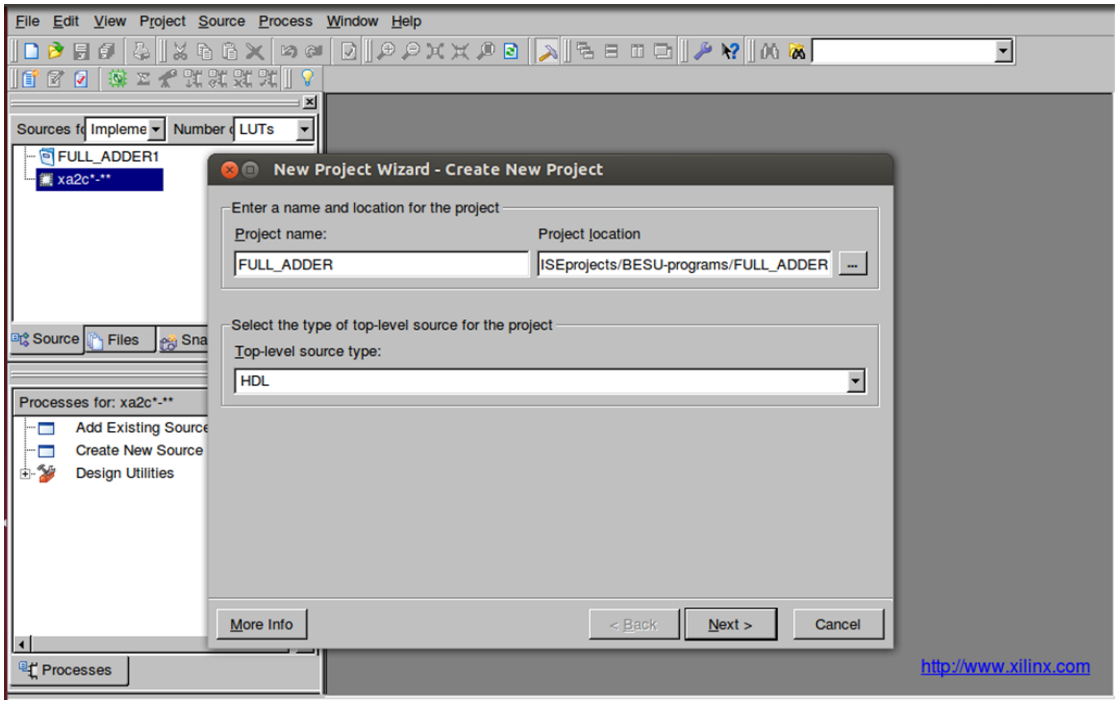

Figure 1: New Project wizard-Create New Project(Project Name and Project Location)

- Synthesis Tool:XST(VHDL/Verilog)
- Preferred Language:VHDL(according to our project. If the user choose the Verilog as the preferred language then he/she has to write it according to the syntaxes of the Verilog language. Users are advised to follow the qst tutotial of the ISE version they are using)
- Verify that Enable Enhanced Design Summary is selected.

Leave the default values in the remaining fields.

A sample table from our project:-

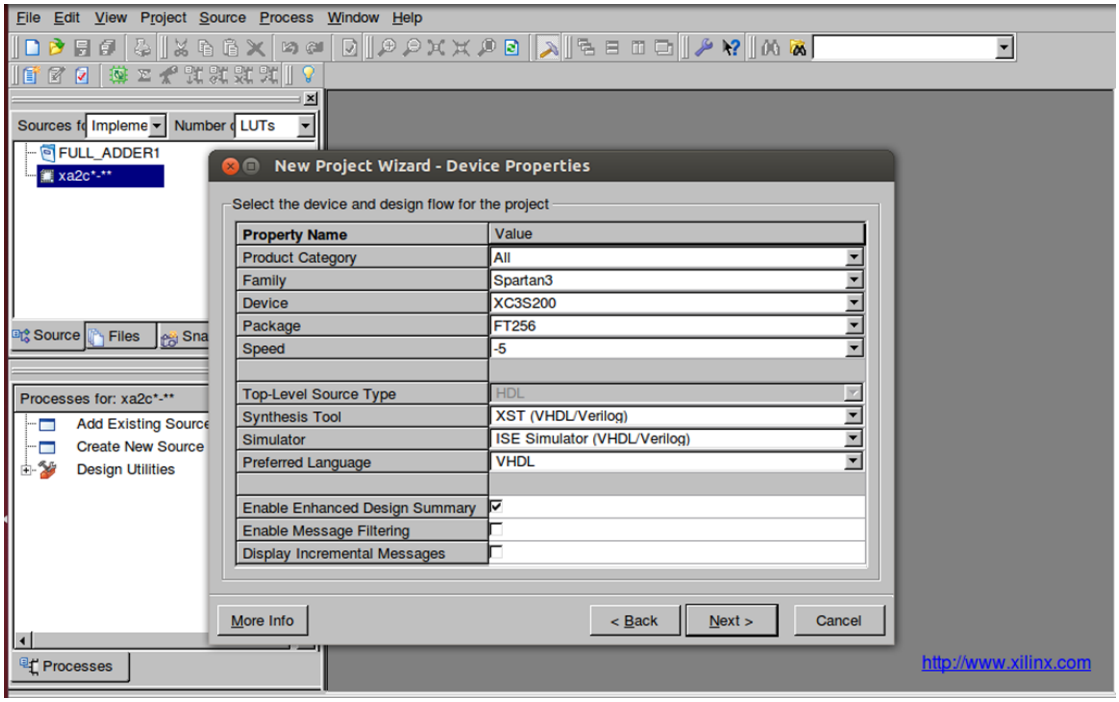

Figure 2: New Project Wizard-Device Properties

5. Click Next to proceed to the Create New Source window in the New Project Wizard.

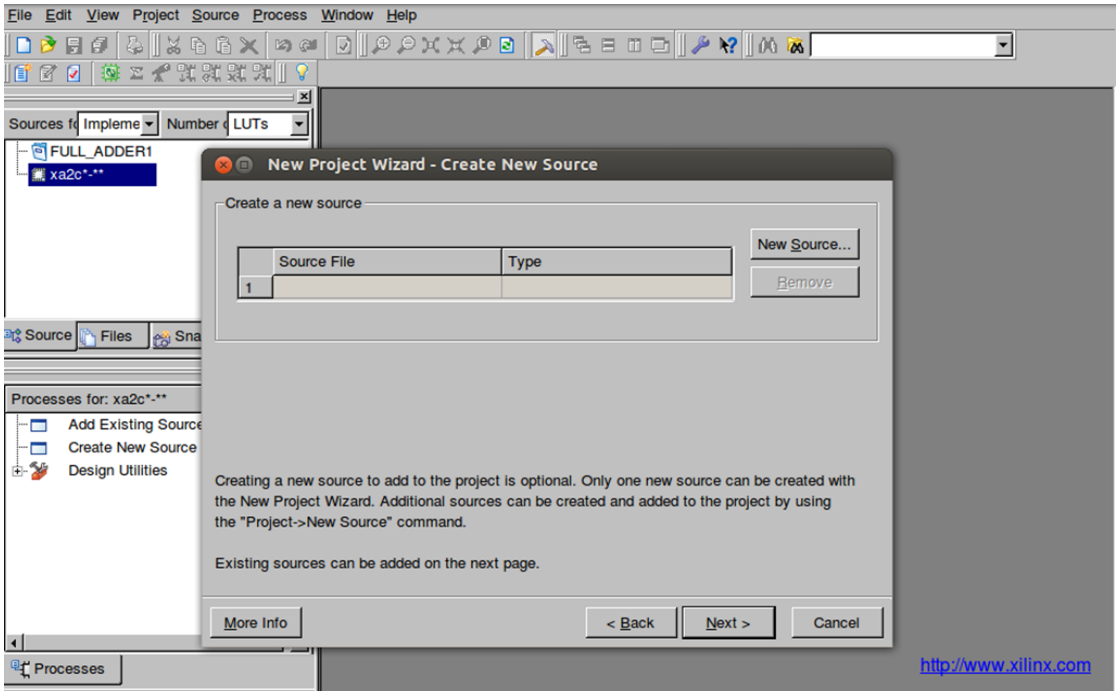

Figure 3: New Project Wizard-Create New Source

Note: If there is **only one source** to be created, then user can create it by clicking the New Source option. Then the following box will appear. Choose the right option from there(like in the picture it can be seen that we are going to create a VHDL Module.So, we select the option) and enter a name for it. Otherwise follow the process mentioned by us

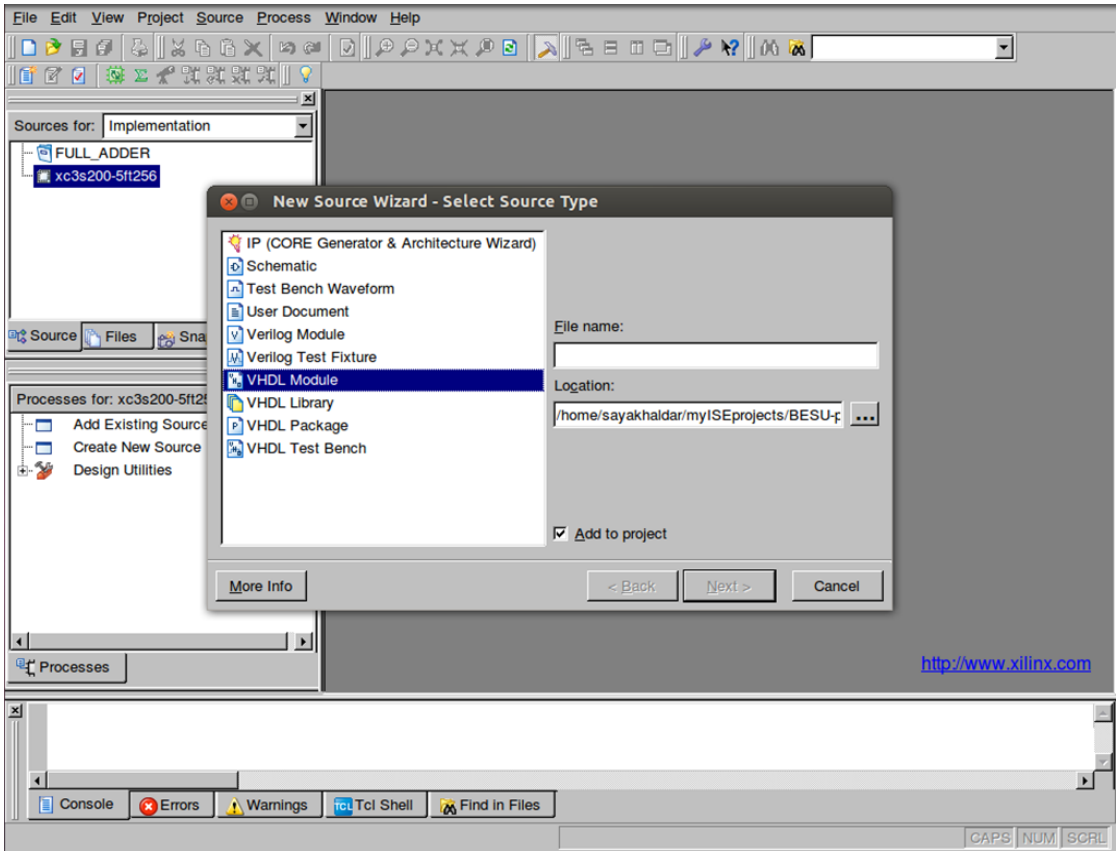

Figure 4: New Source Wizard-New Source Type

6. click next to proceed. The following box will appear.

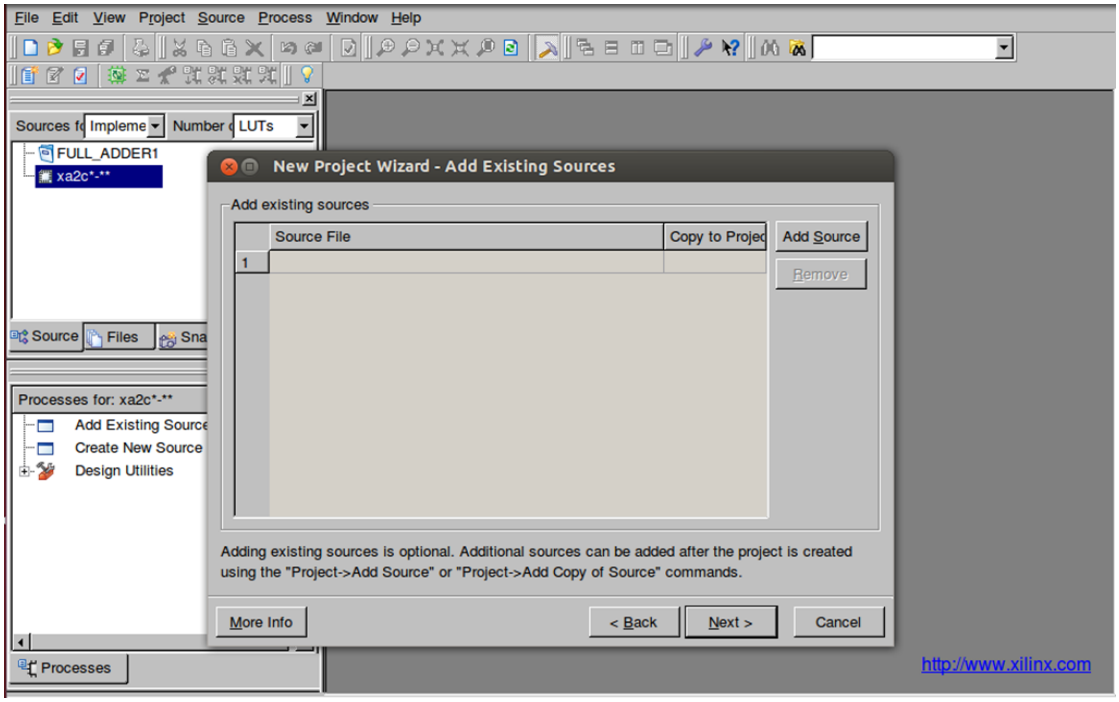

Figure 5: New Project Wizard-Add Existing Sources

Note: If users want to reuse some of the sources they have already created in some other projects, they can add it by clicking the Add Souce option.

7. Otherwise click next. The following box will appear showing the project summary of the project created.

8. click finish.

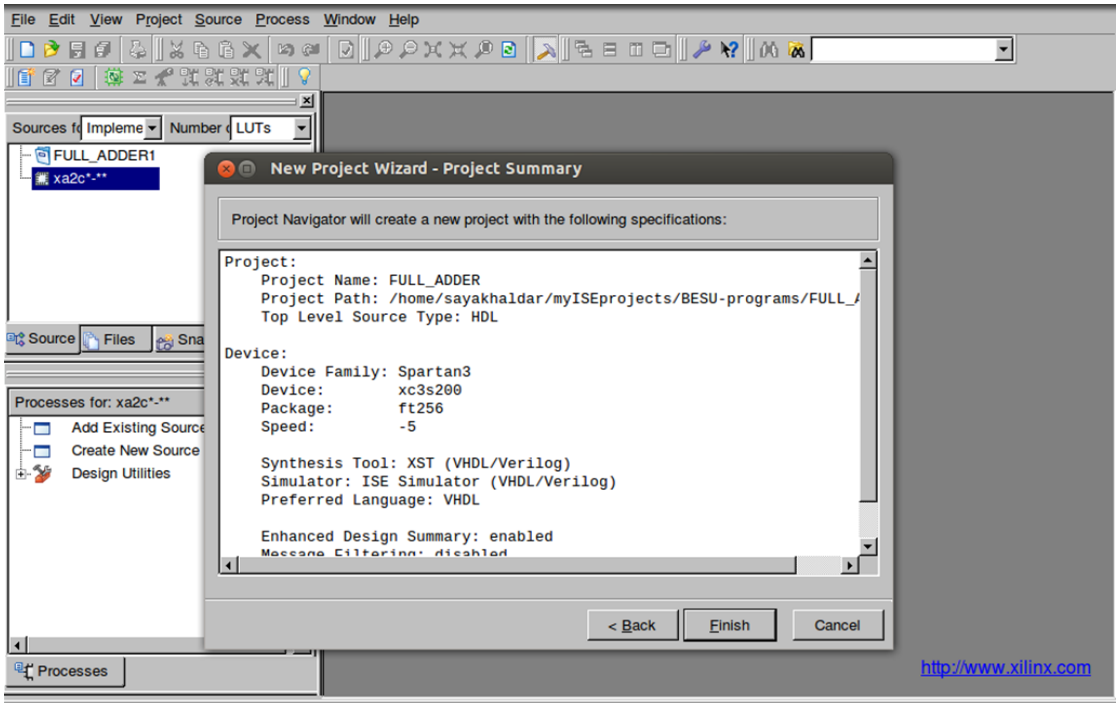

Figure 6: New Project Wizard-Project Summary

4.4.4 Step 4:Create an HDL Source

Uses will only learn the process of creating a VHDL source here. However, they can also create a Verilog source. In that case, the users are advised to check the qst tutorial of the corresponding version of the Xillinx ISE they are using. **Creating a VHDL Source**

Create a VHDL source file for the project as follows:

1. Click the New Source button in the New Project Wizard.
2. Select VHDL Module as the source type and enter a name.
3. Verify that the Add to project checkbox is selected.

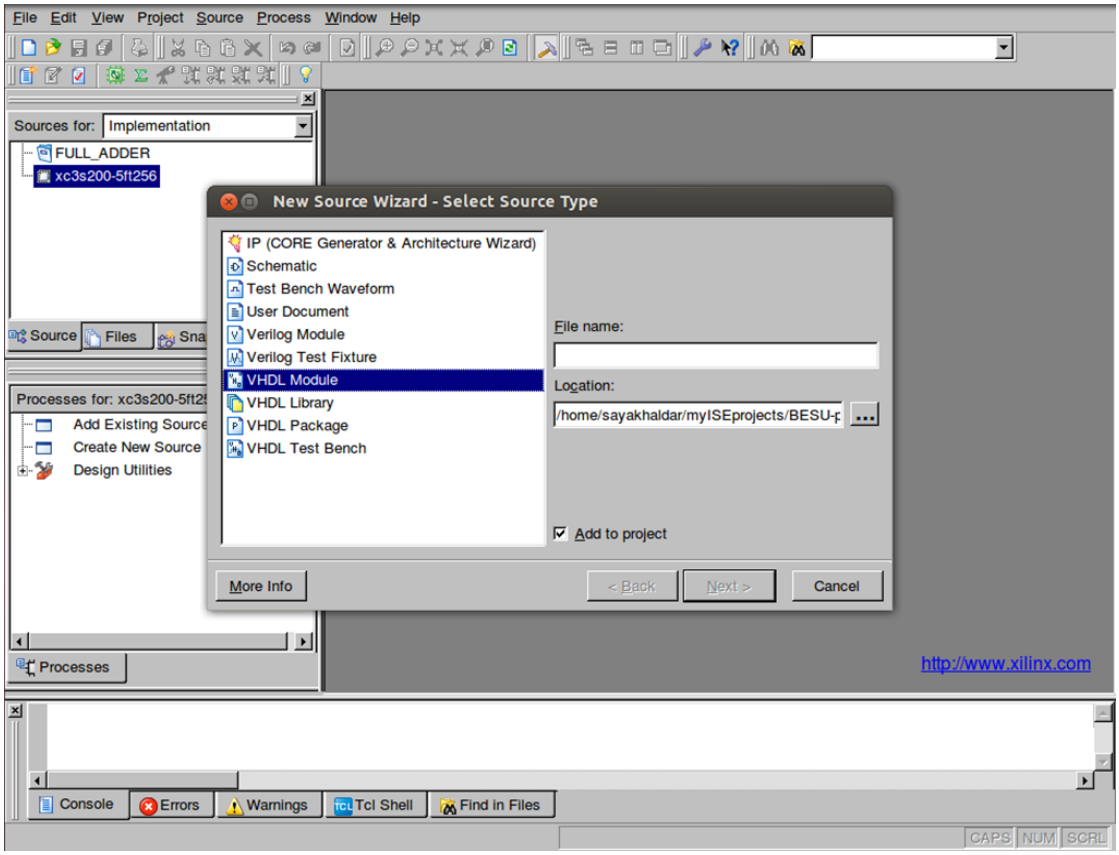

Figure 7: Creating A New VHDL Module From New Source Wizard-New Source Type

4. Click next.

5. The following Box will appear. Define the input and output pins properly.(A sample filled in box from our project for creating a full adder. You could create your own also)

6. Enter the proper architecture name for the VHDL module(Dataflow,Behavioral or Structural)

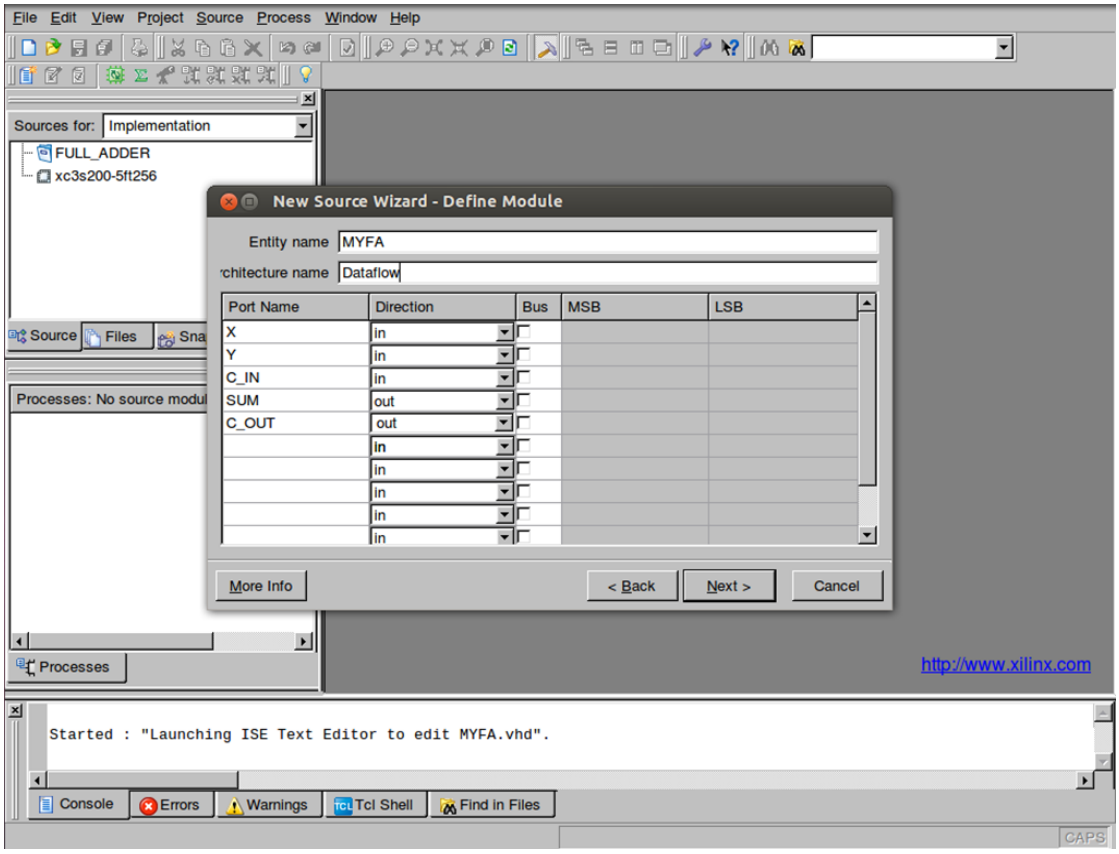

Figure 8: Declaring The Input And Output Pins For The VHDL Socurce From New Source Wizard-Define Module

7. Click next.

8. New Source Wizard-Summary box will appear. It will show the summary of the VHDL source created.

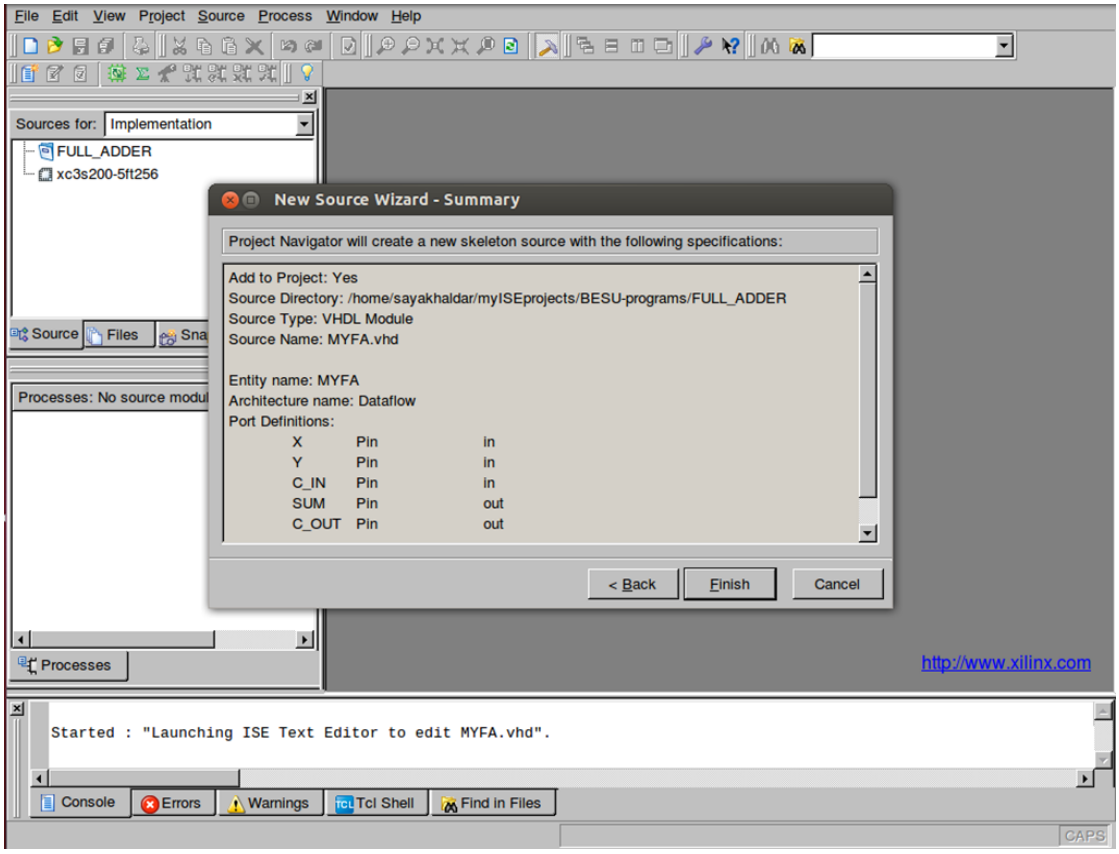

Figure 9: Displaying The Source Summary Of The Created VHDL Source

9. Click finish.

10. The source file containing the entity/architecture pair displays in the Workspace, and the MYFA(according to our project,otherwise it will be the VHDL module defined by the user) displays in the Source tab, as shown below:

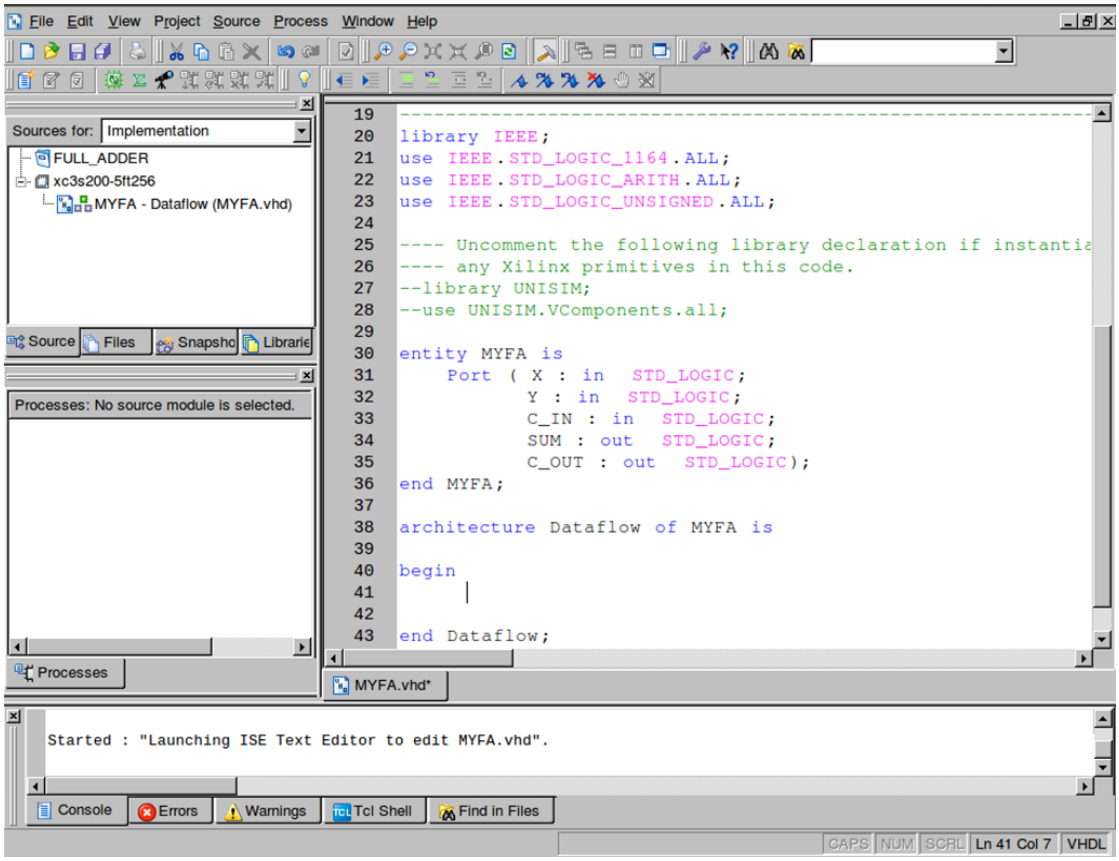

Figure 10: MYFA.vhd Initially

11. Define the corresponding architecture(according to the architecture name) after the statement **"begin"** which comes after:-

    architecture <architecture-name>of <entity name>is

12. If the user is following this guide step by step for creating the FULL_ADDER project of us then it will come after the statement **"begin"** which comes after:-

    architecture Dataflow of MYFA is

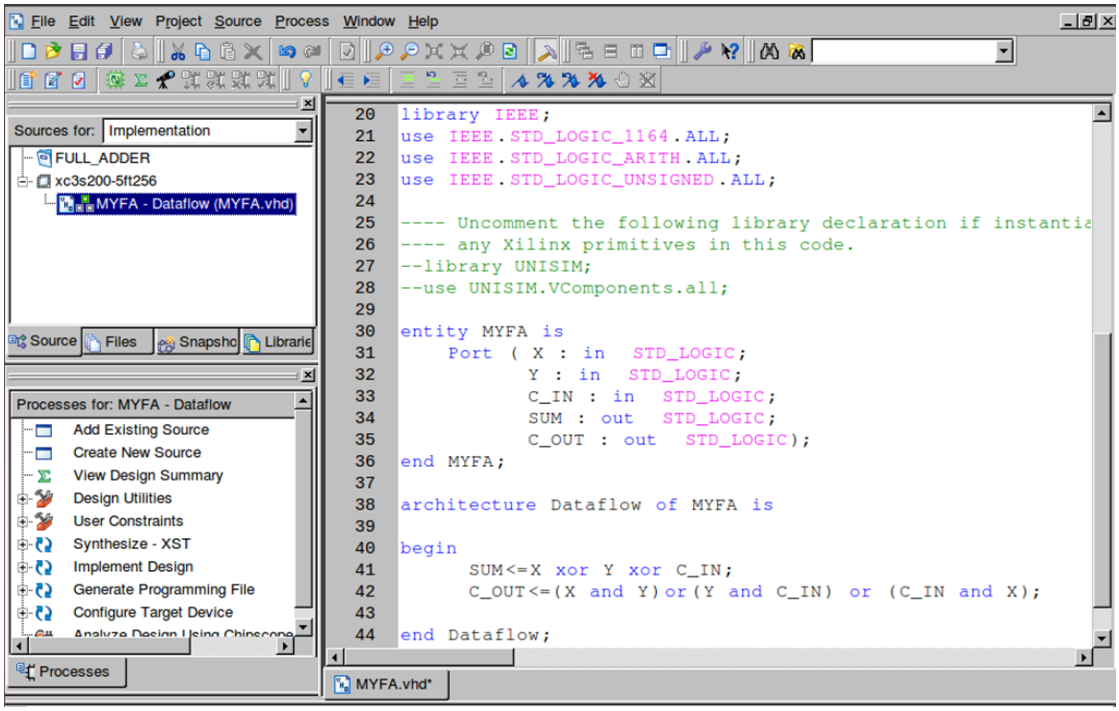

Figure 11: MYFA.vhd Finished

13. After writing the code, save it by clicking the save button from File→ Save.Otherwise click the Save or Save All button. It is just below the Edit option.

Figure 12: Saving The Project By Clicking The Save or Save All Button

4.4.5 Step 5:Checking the Syntax of the New Written Module

1. Verify that Implementation is selected from the drop-down list in the Sources window.

2. Select the MYFA design source(according to our project. Otherwise users have to select the design source written by them) in the Sources window to display the related processes in the Processes window.

3. Click the "+" next to the Synthesize-XST process to expand the process group.

4. Double-click the Check Syntax process.

> **Note:** You must correct any errors found in your source files. You can check for errors in the Console tab of the Transcript window. If you continue without valid syntax, you will not be able to simulate or synthesize your design

5. If there is no syntactic error in the code then after a while(after double Clicking the check syntax process) there will be a green tick sign beside the Check Syntax process option like the one in the following picture:-

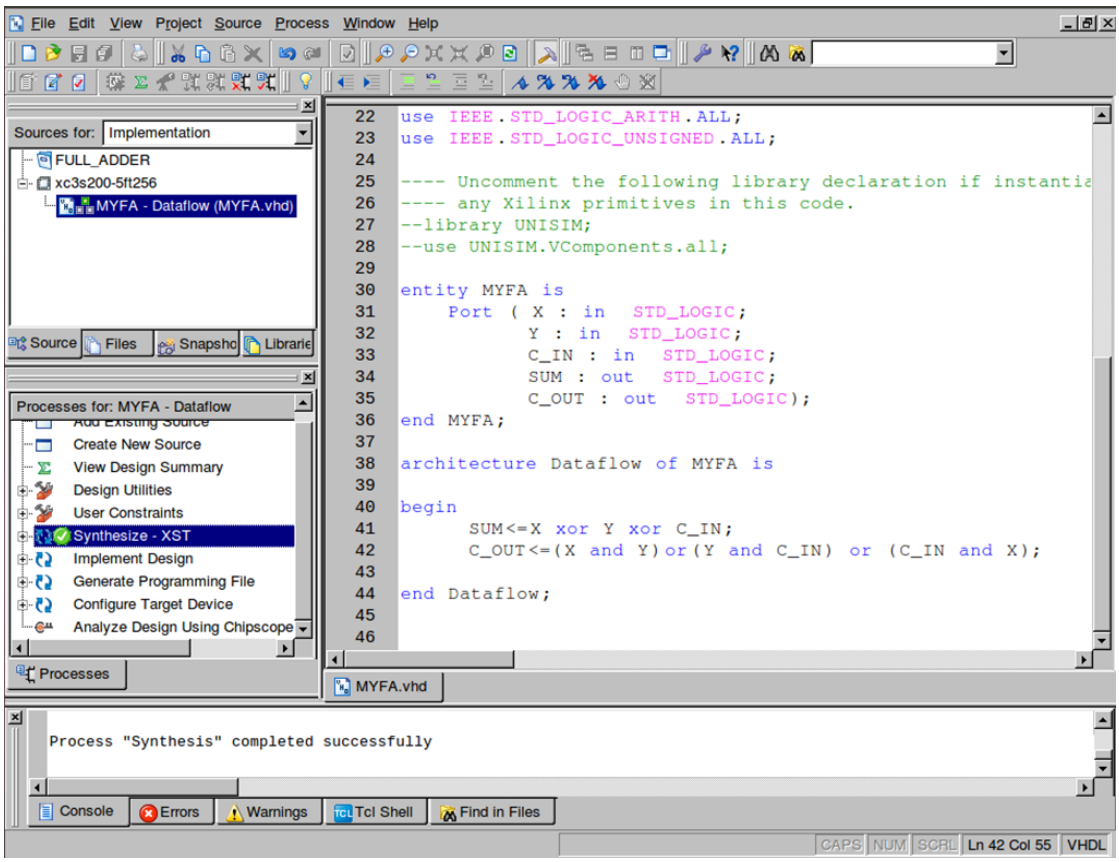

Figure 13: Synthesis Is Done Successfully

4.4.6 Step 6:Design Simulation

Verifying Functionality using Behavioral Simulation

Create a test bench waveform containing input stimulus you can use to verify the functionality of the MYFA module(according to our project). The test bench waveform is a graphical view of a test bench.

Note: This step is not needed for generating the .bit file and downloading it to the FPGA kit. This is just to check the functionality of the created module. Since a code can have both type of errors.1. Syntactic 2. Logical. Now, syntactic errors can be avoided by synthesizing the code. But to check if there is any logical error in the code or not this step is needed. If the user is very much confident about his/her code he/she can skip this step

Create the test bench waveform as follows(according to our project):

1. Select the MYHA HDL file in the Sources window.
2. Create a new test bench source by selecting Project→ New Source.
3. In the New Source Wizard, select Test Bench WaveForm as the source type, and type WAVE or WAVE_tbw(with file extension. Though the file extenstion is by default added if it is not mentioned by the user) in the File Name field.

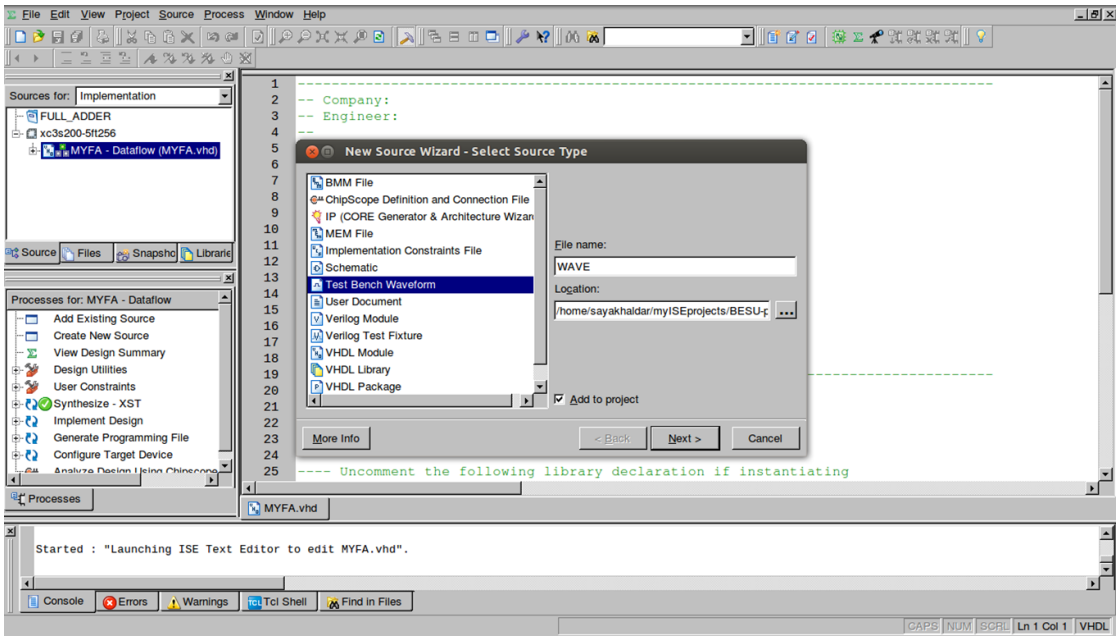

Figure 14: Creating A New Test Bench Waveform Source File

4. Click Next.

5. The Associated Source page shows that you are associating the test bench waveform with the source file. Here, the souce file is MYFA. Click next.

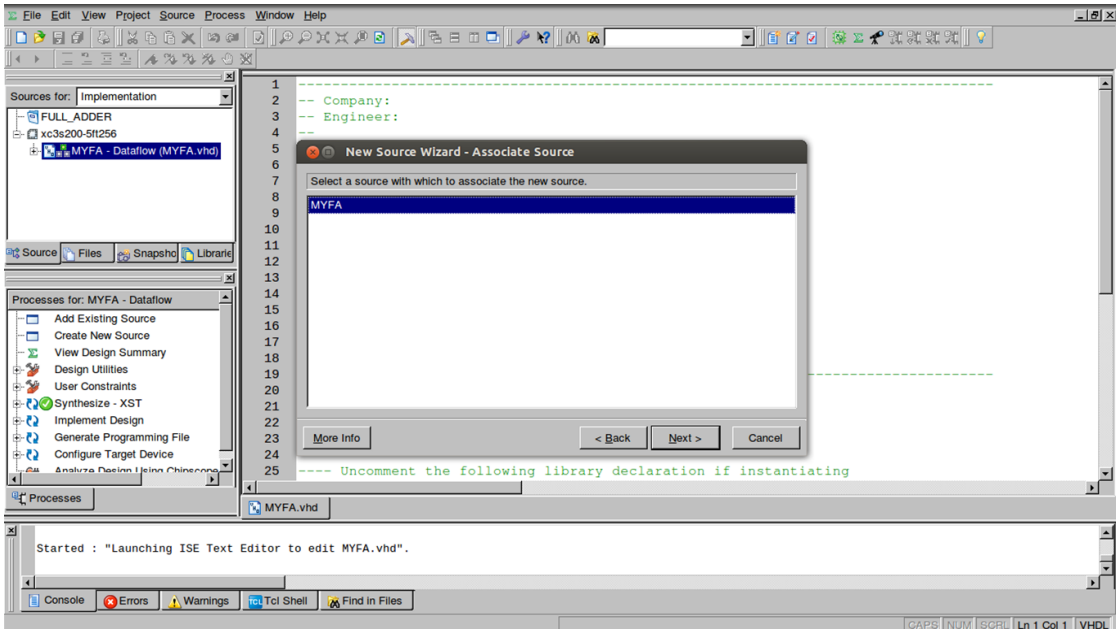

Figure 15: Choosing The Right Source File For The Test Bench Waveform

> **Note:** This step is not so important when the project only contains one source file. But if the project contains more than one source file then this step is very important.(Generally, we need more than one source file when we are following the Structural Architecture design of a circuit using VHDL

6. The following box will appear with the source summary for the Test Bench Waveform.

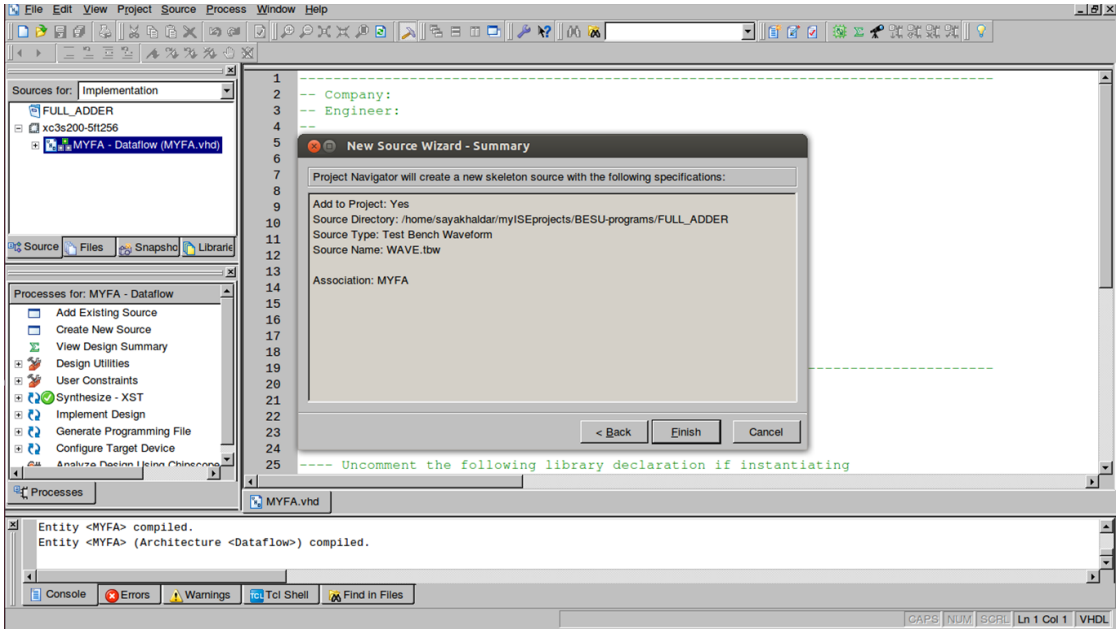

Figure 16: Displaying The Source Summary Of The Test Bench Waveform

7. Click Finish.

8. Now the following box will appear.

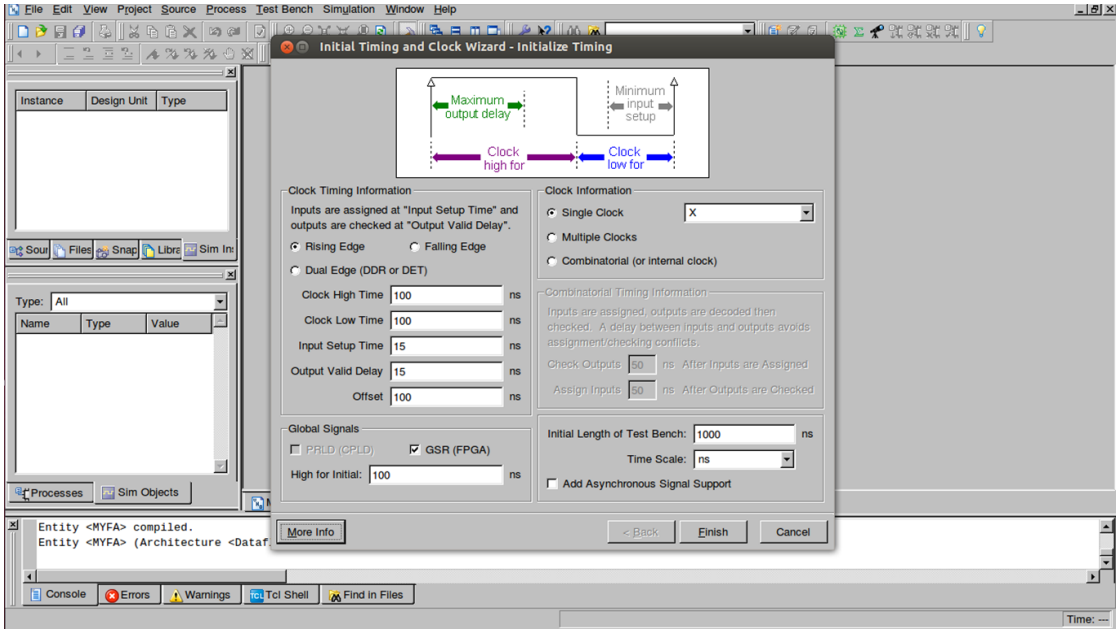

Figure 17: Setting The Necessary Things For Test Bench Waveform File

9. Click finish
10. The WAVE.tbw is generated now with the given details.

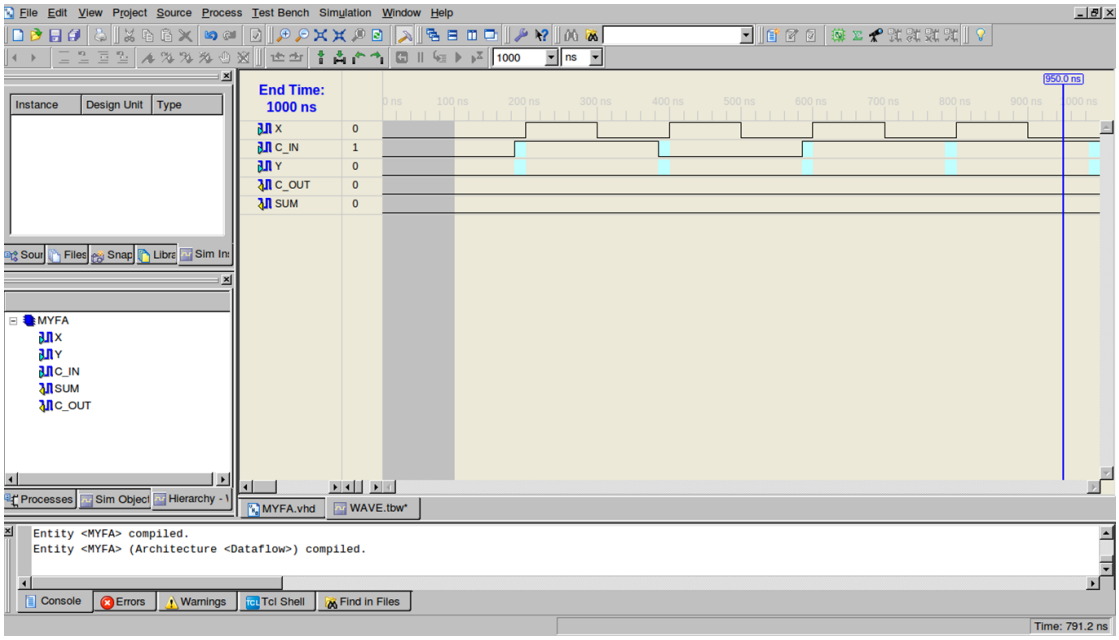

Figure 18: The WAVE.tbw Is Generated With The Given Details

11. Click on the blue shaded areas for making the input signals high or low.

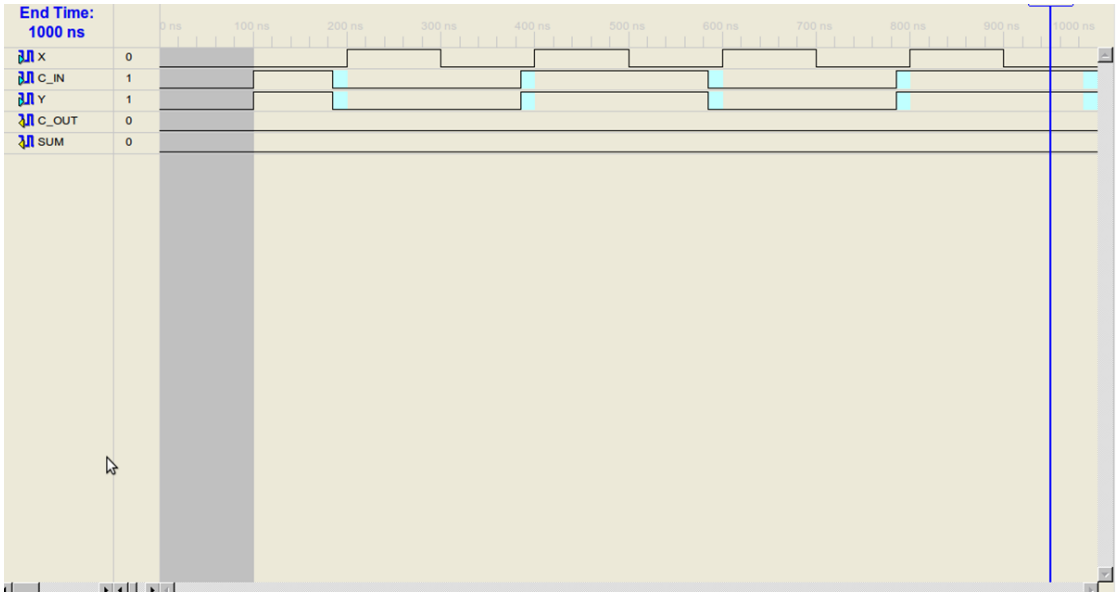

Figure 19: Modifying The WAVE.tbw File

12. Save the Test Bench Waveform File

13. In the Sources window, select the Behavioral Simulation view to see that the test bench waveform file is automatically added to your project.

14. Verify that Behavioral Simulation and WAVE_tbw are selected in the Sources window.

15. In the Processes tab, click the "+" to expand the Xilinx ISE Simulator process and double-click the Simulate Behavioral Model process. The ISE Simulator opens and runs the simulation to the end of the test bench.

16. To view your simulation results, select the Simulation tab and zoom in on the transitions.

4.4.7 Step 7:Creating A Implementation Constraints File

This is a very important step to implement the design using an FPGA. Here, we do the pin assignments for our design.

The following steps are needed to create and write a Implementation Constraint File.

1. In the New Source Wizard select Implementation Constraints File as the source type, and type the name. In our project it is named as FA.

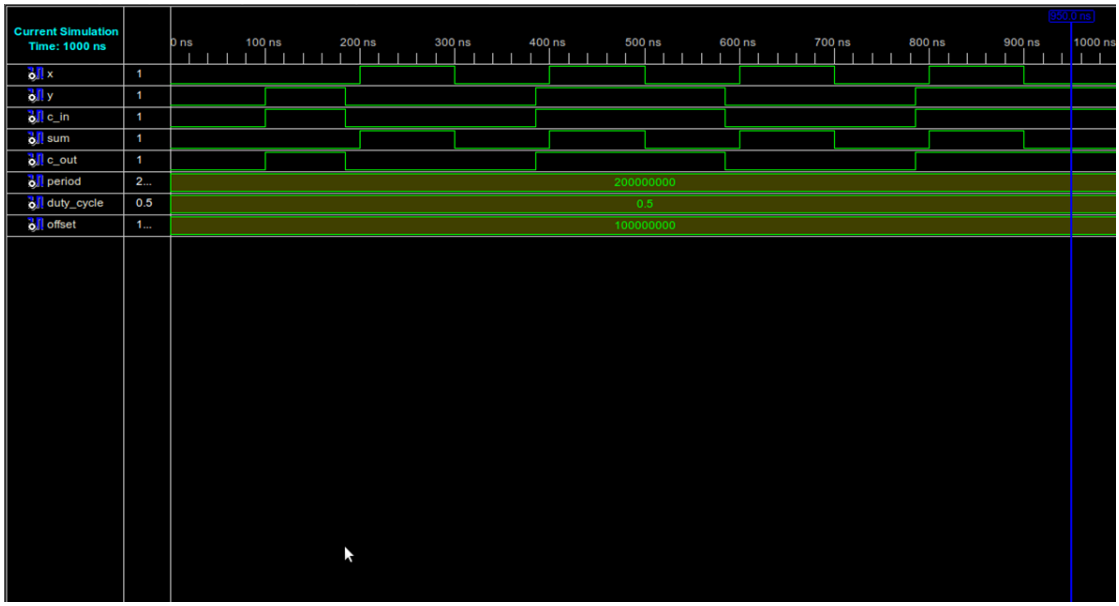

Figure 20: The Simulation Result

2. Go to the project directory and open the FA.ucf file.(In case of Windows OS, open it with notepad. In case of Ubuntu OS open it with text editor)

3. Do the pin assignments. Check the following picture:

   Note: Here,p21,p48,p80 are input pins of the FPGA model and p20,p46 are output pins of the FPGA model. The pin nos. will be different in different FPGA model. So. users are advised to check the pin nos. and change them if necessary from the hardcopy of the manual provided with the FPGA kit. Another thing is to be noted that the name of the input and output pins are not case sensetive

4. Save the file.

5. Make sure that the FPGA kit is connected to the work computer via JTAG port.

6. From the Processes tab, double click the Synthesize-XST option.

7. Assuming that the code is syntactic error free, Double click the Implement Design option

8. Then double click the Generating File Option. If this is done successfully then the corresponding .bit file will be generated(same name with the entity name).

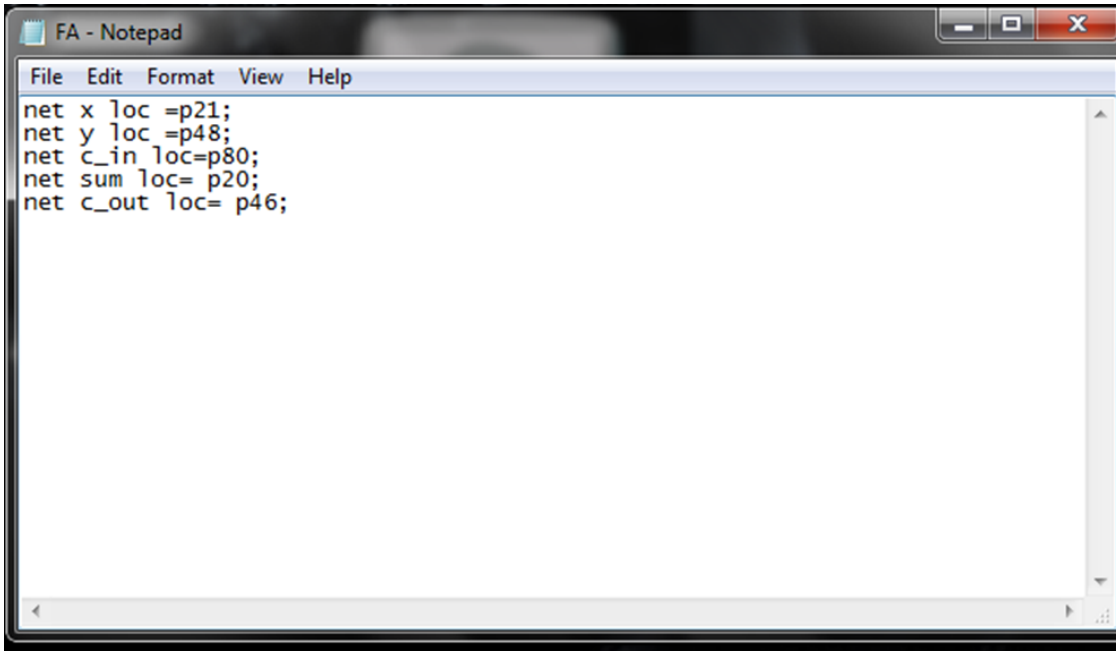

Figure 21: The Pin Assignments Are Done in .ucf File

9. Double click the Configure Target Device option
10. The Xilinx WebTalk Dialog box may open during this process. Click Decline.
11. iMPACT opens and the Configure Devices dialog box is displayed.
12. In the Welcome dialog box, select Configure devices using Boundary-Scan (JTAG).
13. Verify that Automatically connect to a cable and identify Boundary-Scan chain is selected.
14. Click Finish.
15. If you get a message saying that there are two devices found, click OK to continue. The devices connected to the JTAG chain on the board will be detected and displayed in the iMPACT window.
16. The Assign New Configuration File dialog box appears. To assign a configuration file to the xc3s200 device in the JTAG chain, select the FA.bit file and click Open
17. If you get a Warning message, click OK.

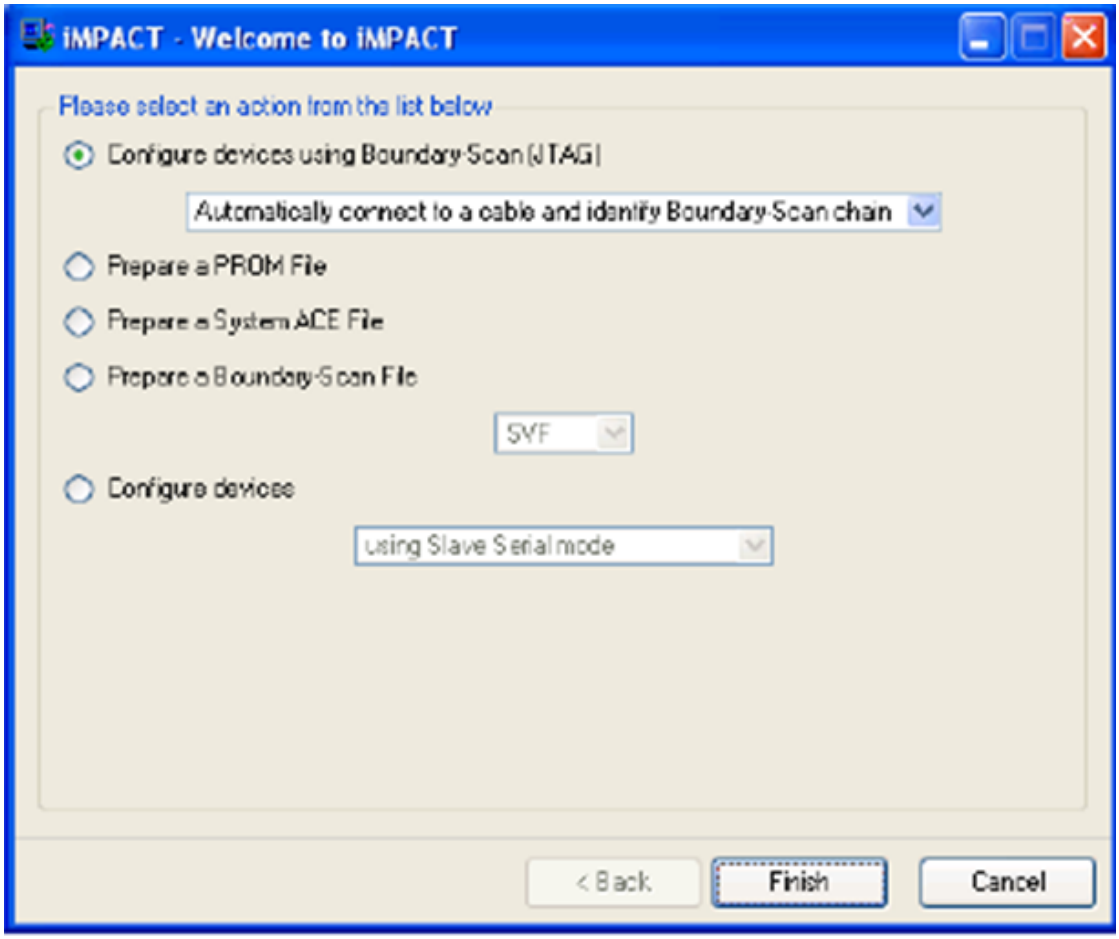

Figure 22: iMPACT Opens And The Configure Devices Dialog Box Is Displayed.

18. Select Bypass to skip any remaining devices.

19. Right-click on the xc3s200 device image, and select Program... The Programming Properties dialog box opens.

20. Click OK to program the device. When programming is complete, the Program Succeeded message is displayed.

Note: Steps will be different for different versions of FPGA as well as different version Xilinx ISE used.

5 Importance Of Doing FPGA Based Design:

Nowdays,FPGA based designing in needed in many fields. Applications of FPGAs include digital signal processing, software-defined radio, ASIC prototyping, medical imaging, computer vision, speech recognition, cryptography, bioinformatics, computer hardware emulation, radio astronomy, metal detection and a growing range of other areas.

A list of common FPGA applications is provided below.

- Aerospace and Defense (Avionics/DO-254, Communications, Missiles & Munitions, Secure Solutions, Space)
- ASIC Prototyping
- Audio (Connectivity Solutions, Portable Electronics, Radio, Digital Signal Processing (DSP))
- Automotive (High Resolution Video, Image Processing, Vehicle Networking and Connectivity, Automotive Infotainment)
- Broadcast (Real-Time Video Engine, EdgeQAM, Encoders, Displays, Switches and Routers)
- Consumer Electronics (Digital Displays, Digital Cameras, Multi-function Printers, Portable Electronics, Set-top Boxes)
- Distributed Monetary Systems (Transaction verification, BitCoin Mining, Data Center, Servers, Security, Routers, Switches, Gateways, Load Balancing)
- High Performance Computing (Servers, Super Computers, SIGINT Systems, High-end RADARS, High-end Beam Forming Systems, Data Mining Systems)
- Industrial(Industrial Imaging, Industrial Networking, Motor Control)
- Medical(Ultrasound, CT Scanner, MRI, X-ray, PET, Surgical Systems)
- Security (Industrial Imaging, Secure Solutions, Image Processing)
- Video & Image Processing(High Resolution Video, Video Over IP Gateway, Digital Displays, Industrial Imaging)

- Wired Communications(Optical Transport Networks, Network Processing, Connectivity Interfaces)
- Wireless Communications(Baseband, Connectivity Interfaces, Mobile Backhaul, Radio)

So, learning FPGA based hardware design could be very handful. This is the importance of this project.

6 References

- Xilinx ISE 10.1 quick start tutorial
- Xilinx ISE 10.1 in depth tutorial
- VHDL primer by Jan Van der Spiegel. University of Pennsylvania
- VHDL Programming by Example by Doughlas Perry
- www.wikipedia.org

7 Our Special Thanks To:

- Professor Amit Kumar Das, Department of Computer Science and Technology,IIEST,Shibpur
- Professor Ramanath Dutta, Department of Electronics & Communication Engineering,STCET,Khidirpur