MEHRAN UNIVERSITY OF ENGINEERING AND TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

FPGA Based Digital Design (ES-373)
Batch: 20ES (6th Semester)

Lab Experiment No. 2 & 3
Xilinx ISE WebPACK Toolset for basic development on Digilent FPGA Boards

Name _____	**Roll #** **20ES062**
Signature of Lab Tutor _____	**Date** _____

RUBRICS:

Performance Metric	TEAMWORK	PARTICIPATION	CONDUCTING EXPERIMENT	USE OF MODERN ENGINEERIN GTOOLS	DATA ANALYSIS	CALCULATION AND CODING	OBSERVATION /PROGRAM RESULTS	**Total Score**
	0.5	**0.5**	**1**	**1**		**1**	**1**	**05**
Obtained								

OBJECTIVE(S)
The purpose of this lab is to:

#	*Topic*	**# Of Lectures**	**CLO**	**Taxonomy level**
1	Familiarize students with the Xilinx ISE environment for FPGA development.	3	4,5	P3, A4
2	Set up a project in Xilinx ISE, perform basic design entry, synthesis, and implement designs on the Digilent FPGA boards.			

OUTCOME(S)

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	**PL-5**: Modern Tool Usage
b. An ability to communicate effectively	**PLO10**: Communication

LAB REQUIREMENTS:

- Standard PC with Xilinx ISE Design Suite 12.3 or latest, Software installed.
- Digilent Adept Software.
- Xilinx ISE compatible board such as NEXYS4, NEXYS3, NEXYS2, OR BASYS2

DISCUSSION:

This lab aims to enhance students' Software skills, rather than programming skills. This lab will focus on design aspects of very basic logic using Xilinx ISE Software and work through exercises to gain more knowledge.

First, install Xilinx ISE WebPACK on PC or laptop. This lab is based on version 14.2. It is available as a free download from *www.xilinx.com*.

DESIGNING PROCEDURE:

1. DESIGN ENTRY

i. Invoke Xilinx ISE Design Suite 14.2 Software

Double click the Xilinx ISE Design Suite icon (figure 1) present on your desktop, Check with YOUR instructor if not available.

Figure 1: Xilinx ISE Design Suite

A screen like that of **figure 2** will be displayed.

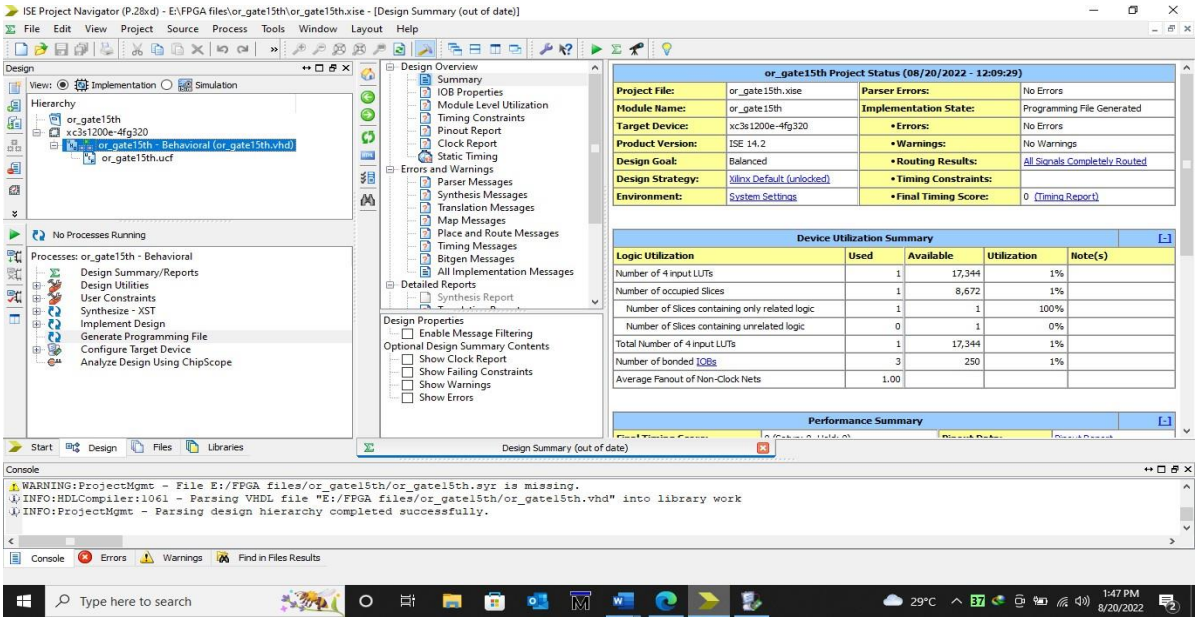


Figure 2: Xilinx ISE Design Suite **Project Navigator**

ii. Creating a New Project

To create a new project using the New Project Wizard, do the following:

From Project Navigator, select **File > New Project**.

The New Project Wizard appears, as shown in figure 3.

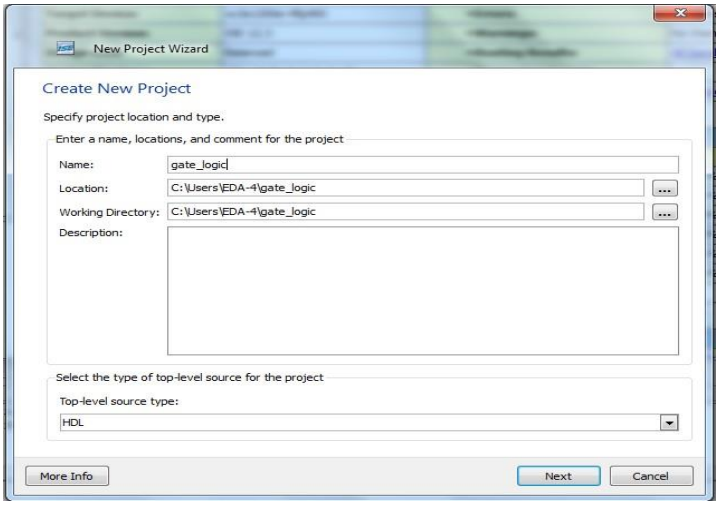


Figure 3*:* New Project Wizard**—Create New Project**

iii. Enter your desired Project name in the Name field.

iv. In the Location field, browse to the directory in which you installed the project.

v. Verify that **HDL** is selected as the Top-Level Source Type and click **Next**.

vi. In the project settings, you will need to select details that pertain to your FPGA on the board. The ***Nexys2*** is being used for this example.

Choose the following settings (**Figure 4**) as shown below:

- Product Category: **All**
- Family: **Spartan3E**
- Device: **XC3S1200E/ XC3S500E**
- Package: **FG320**
- Speed Grade: **-4**
- Top-Level Source Type: **HDL**
- Synthesis Tool: **XST (VHDL/Verilog)**
- Simulator: **ISim (VHDL/Verilog)**
- Preferred Language: **VHDL**
- Property Specification in Project File: **Store All Values**
- VHDL Source Analysis Standard: **VHDL-93/VHDL-2000X**

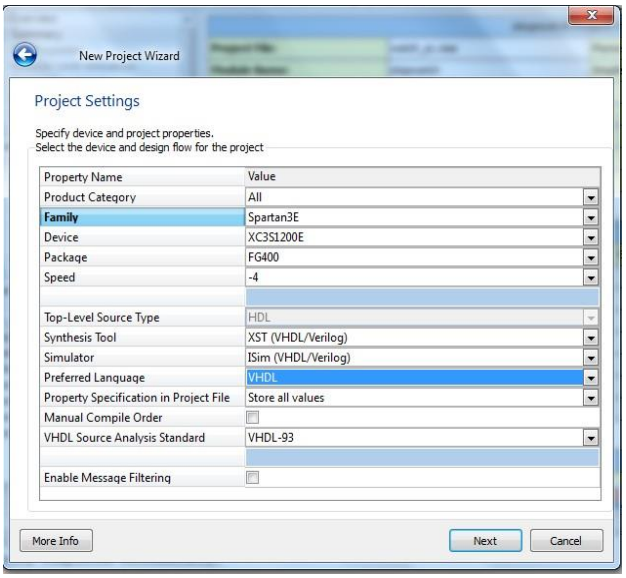


Figure 4*:* New Project Wizard**—Project Setting**

✓ **For Nexys 4:**
- Family: Artix-7
- Device: XC7A100T
- Package: CSG324
- Speed: -1

✓ **For Nexys 3:**
- Family: Spartan 6
- Device: XC6SLX16
- Package: CSG324
- Speed: -2

✓ **For Nexys 2-1200:**
- Family: Spartan-3E
- Device: XC3S1600E
- Package: FG320
- Speed: -4

✓ **For Nexys 2-500:**
- Family: Spartan-3E
- Device: XC3S500E
- Package: FG320
- Speed: -4

✓ **For Basys 2:**
- Family: Spartan-3E
- Device: XC3S100E
- Package: CP132
- Speed: -4

These settings can be done any time, but if settled once will not change automatically until user disturbances.

After Fulfill the Setting Click **Next,** the following Project summary window (**Figure 5**) will appear.

vii. Click **Finish** (**Figure 5**) to complete the project creation and begin the design process.

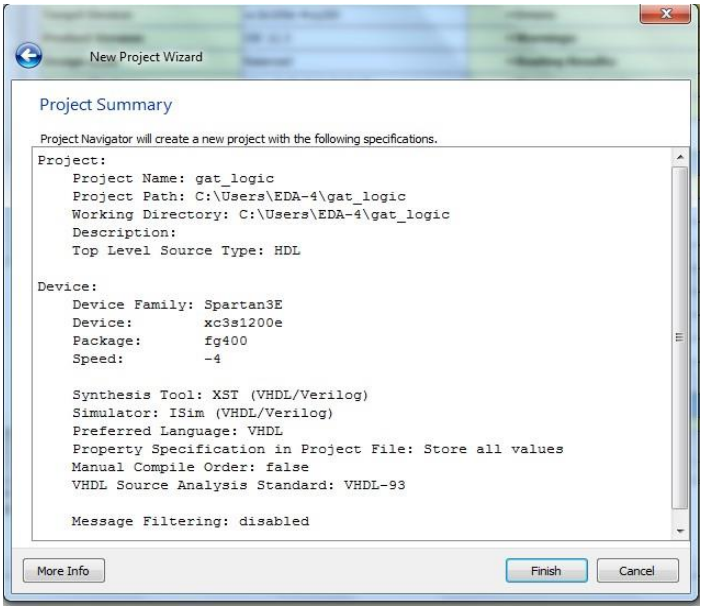


Figure 5*:* New Project Wizard- **Project Summary**

viii. **Creating** a New **VHDL module** Source file; Go to **Project** and click on **New Source**, select **VHDL Module** and Enter the name "**and_gate**" as shown in **figure 6** and click next you will get a window as shown in **figure 7.**

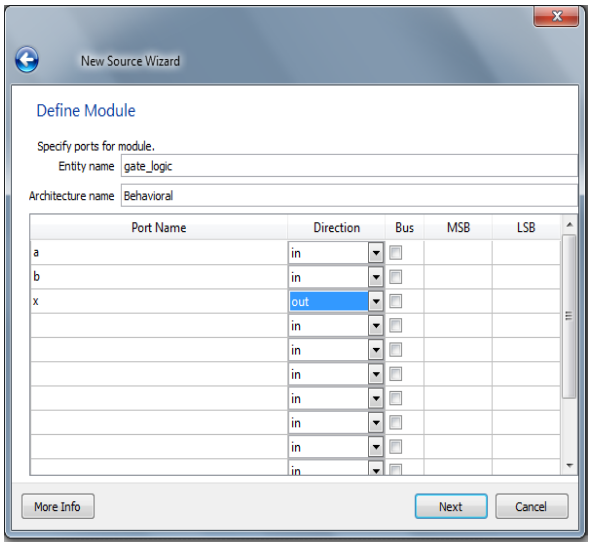


Figure 6*:* New Source Wizard- **Source Type**

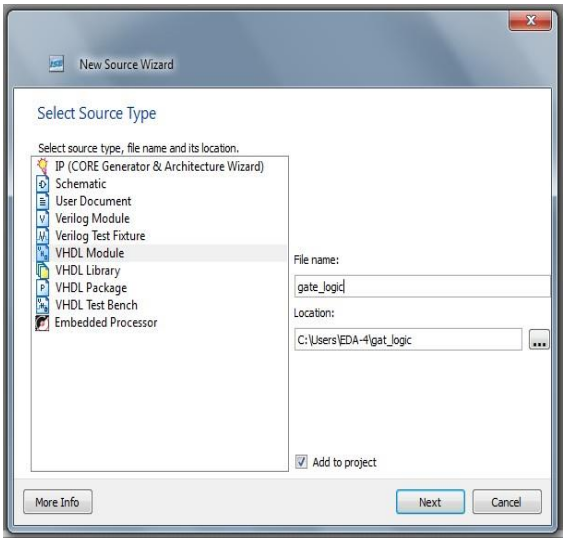


Figure 7*:* New Source Wizard- **Specify ports**

ix. Enter the ports for your module (**figure 7**). This is strictly optional and saves you the typing of creating the VHDL entity manually. Whether you fill this out or not, the resultant text file will be totally under your control for later editing.

x. After entering the ports click on **Next,** the following summary window (**Figure 8**) will appear.

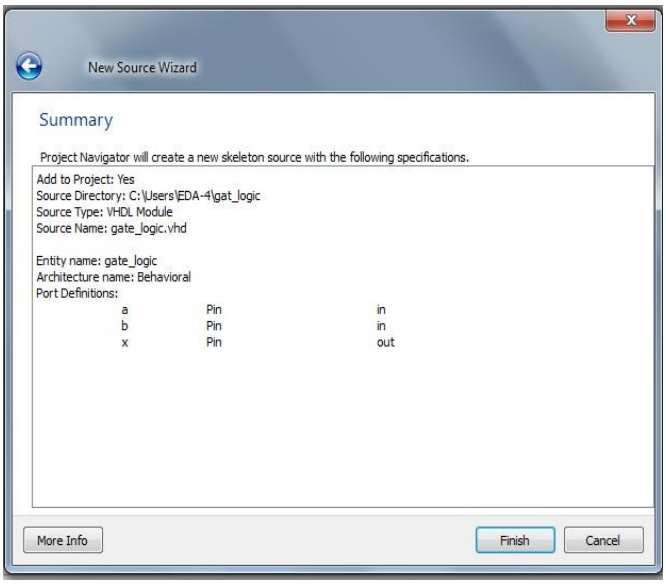


Figure 8*:* New Source Wizard- **Summary**

xi. Click **Finish** (**figure 8**) to open the empty VHDL file in the ISE Text Editor (**figure 9**). In the ISE Text Editor, the ports are already declared in the VHDL file, and some of the basic file structure is already in place. Keywords are displayed in blue, comments in green, and values are black. The file is color-coded to enhance readability and help you recognize typographical errors.

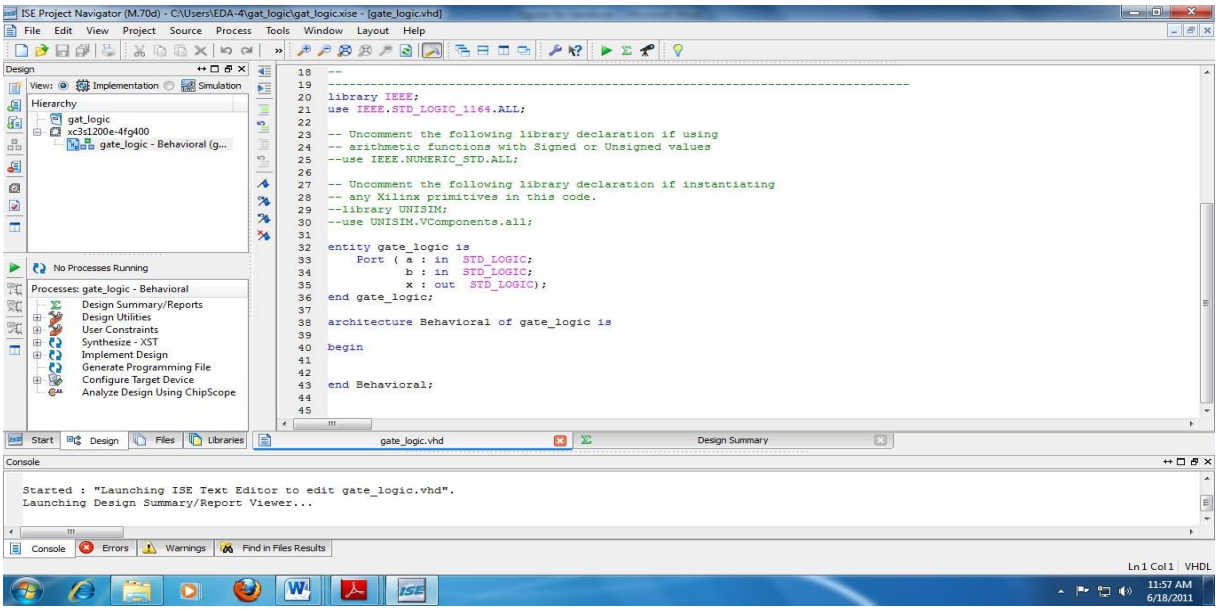


Figure 9*:* VHDL File in ISE Text Editor

xii. Write down the code for NAND gate as "**x<= a NAND b;**" after beginning in the VHDL File in ISE Text Editor **Figure 9**.
The whole VHDL code for NAND Gate is look like as under:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY gate_logic IS
PORT ( a: IN std_logic;
       b: IN std_logic;
```

x:OUT std_logic);
END gate_logic;

Architecture behavioral OF gate_logic IS
BEGIN
X<= a NAND b;
END gate_logic;

2. Synthesis:

Synthesis is a process by which an abstract form of desired circuit behavior (typically register transfer level (RTL)) is turned into a design implementation in terms of logic gates. Synthesis is one aspect of electronic design automation. With Xilinx ISE Design Suite user can convert the HDL (hardware description language) or other kinds of designs created with Design entry tool into the Gate level design for any FPGA family.

- To synthesize the design using Synthesis Tool, double click on the **Synthesize Design** option in the **Processes window**. (**Figure 10**)

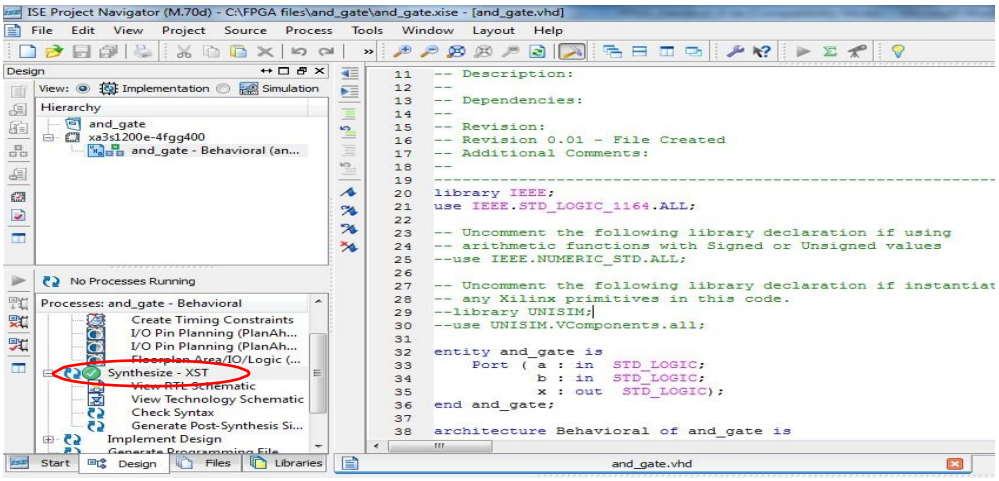


Figure 10: Synthesize the Design

- The schematic diagram of the synthesized VHDL code can be viewed by double clicking View RTL Schematic under Synthesize-XST menu in the Process Window. By double clicking it opens the top-level module showing only input(s) and output(s) as shown below (**figure 11**)

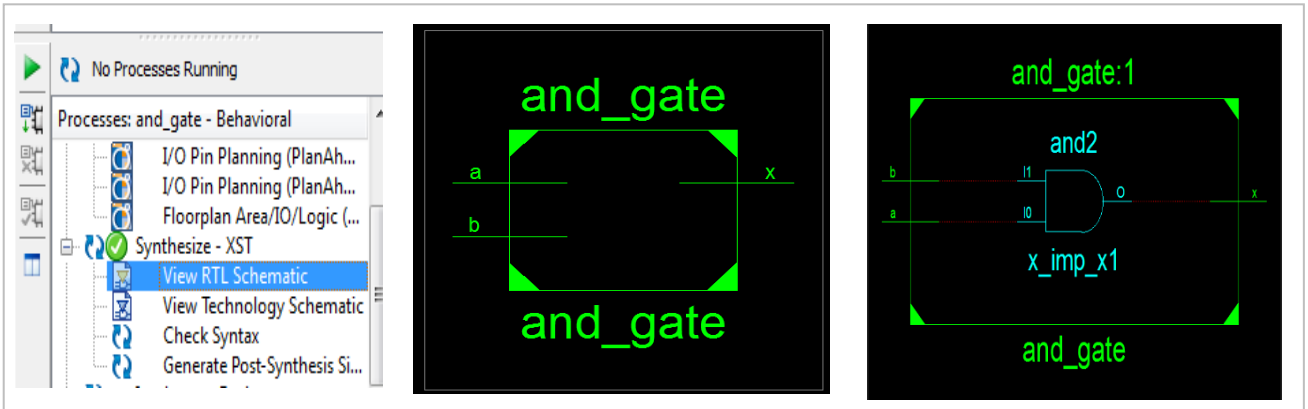


Figure 11: **RTL** Schematic Symbol

- Similarly, you can view the Technology Schematic Symbol by double clicking **View Technology Schematic** under Synthesize-XST menu in the Process Window. By double clicking it opens the top-level module showing the type of input(s) and output(s) as shown below (**figure 12**)

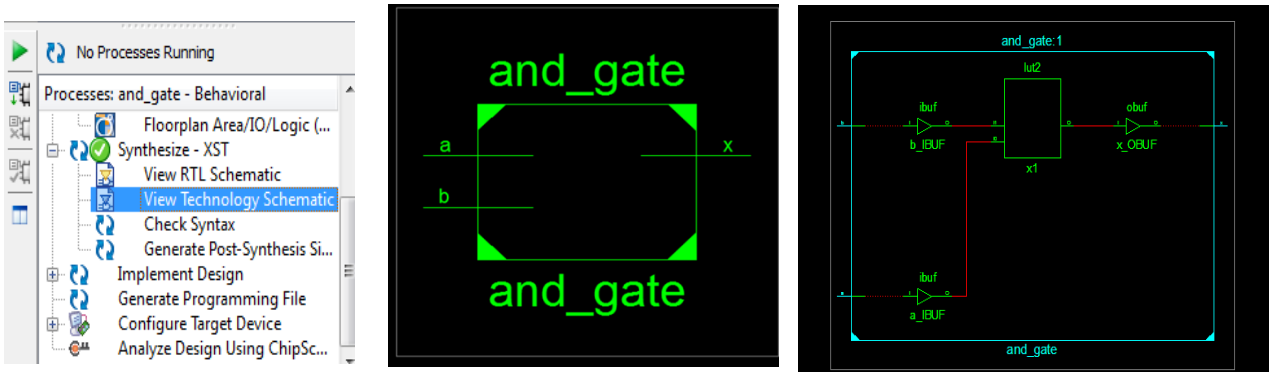


Figure 12: Technology Schematic Symbol

3. Implementation

To implement the design, double click the **Implement design** option in the **Processes window**. It will go through steps like **Translate, Map and Place & Route**. If any of these steps could not be done or done with errors, it will place a "**X"** mark in front of that, otherwise a tick mark will be placed after each of them to indicate the successful completion. If everything is done successfully, a tick mark will be placed before the **Implement Design** option. If there are warnings, warning symbol will be placed in front of the option indicating that there are some warnings. One can look at the warnings or errors in the **Console** window present at the bottom of the Navigator window. Every time the design file is saved; all these marks disappear asking for a fresh compilation.
Before implement the design you must create the User Constraint File **UCF**.

➢ User Constraint File (UCF):

To test the design on the NEXYS2 Board, the inputs need to be connected to the switches/buttons on the board and the outputs need to be connected to the onboard LED's. To create the constraint file, ensure that the implementation radio button is selected and your VHDL module is highlighted.
In the processes window, expand User Constraints and double click on I/O Pin Planning (Plan Ahead) Post Synthesis **(figure 13-a)**. Answer "Yes" when asked if you want to create the UCF file. This will create the constraint file but also open the Plan Ahead application. Wait for the Plan Ahead to fully open, and then close it. You will now see the .ucf file in your hierarchy **(figure 13-b)**.

- Single Click on and gate.ucf file from within Project Navigator, and then Select "Edit Constraints (Text)" from the Process window as below **(figure 13-b**).

- Then you can edit the constraints as shown below **(figure 14**). To figure out the correct pin locations, consult the NEXYS2 manual. Even simpler, you may notice that most pins of interest are printed on the board surface.

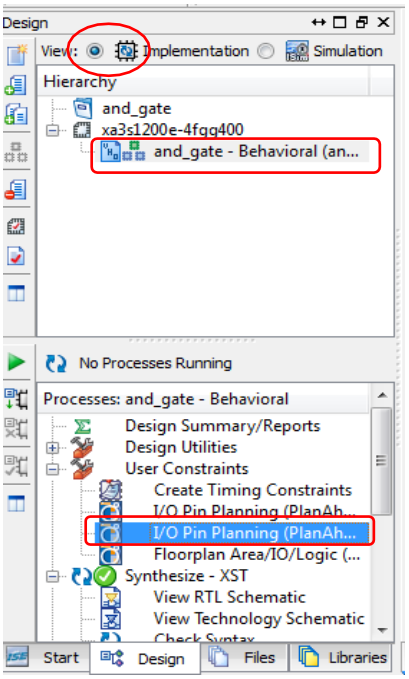


Figure 13-a: Creating **U**ser **C**onstraints **F**ile

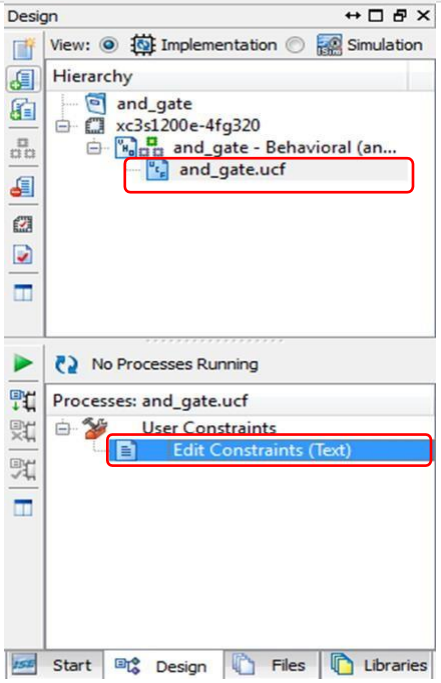


Figure 13-b: Edit UCF

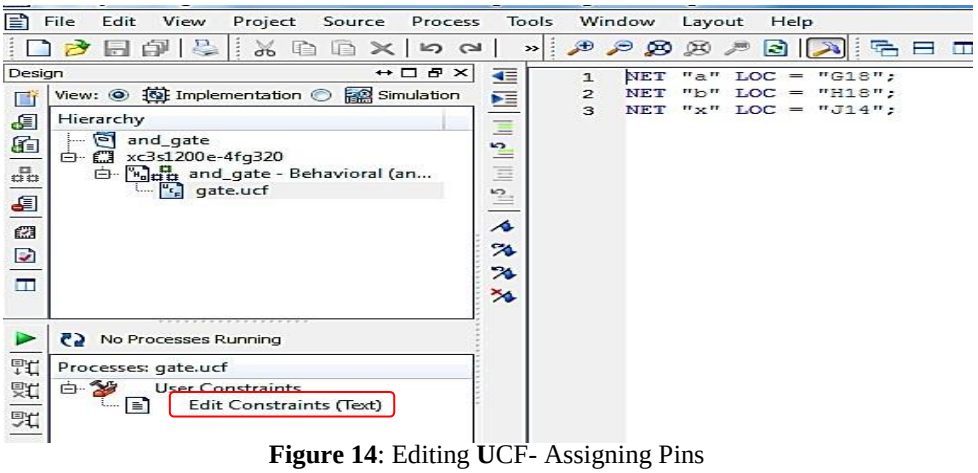


Figure 14: Editing **U**CF- Assigning Pins

- After assigning the pins to inputs and outputs of our design save Ucf file. Now clicking once on your top-level design in the Sources Pane, followed by a double click on "Implement Design" option in the **Processes window**. It will go through steps like **Translate, Map and Place & Route** as shown in **figure 15**.

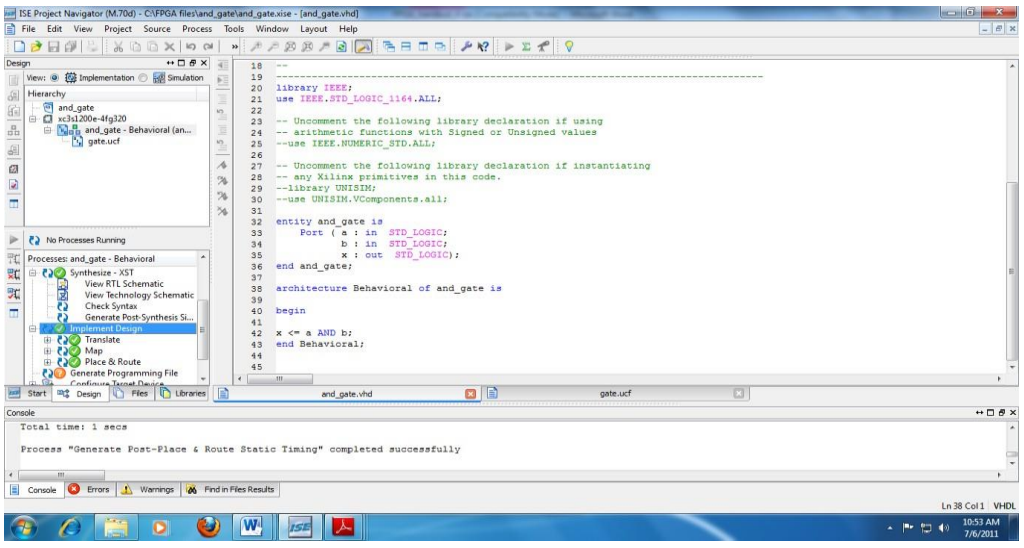


Figure 15: Implement the Design

- The next step in the process is to have the Xilinx ISE create a programming file of the design. This is done by clicking once on your top-level design in the Sources Pane, followed by a double click on **"Generate Programming File"** in the process window **(figure 16).** Now the programming file (bitstream file) of extension ".bit" has been generated, it is time to load your design into the FPGA, and test it out. This can be done in many ways depending on your available equipment and preferences.

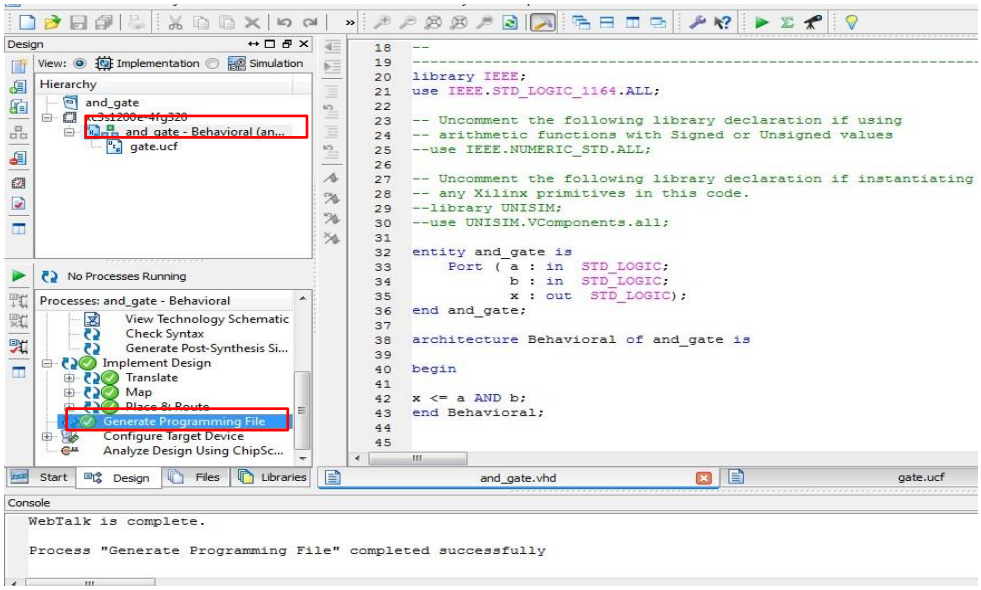


Figure 16: Generate the Programming File

4. Programming the Device through Digilent Adept

Once the programming file (bit stream file) is generated, the file must be downloaded to the **NEXYS2 Spartan3E** device. This is done by using another application, Adept Software provided by Digilent.

- Connect the demo board to the PC using the USB extension cable. Connect the USB Type connector to PC and Mini-AB end to the demo board.
- **Open the Digilent Adept.** Double click the Digilent Adept icon (figure 1) present on your desktop, Check with YOUR instructor if not available.
- The following window **(figure 17)** will appear. If the board is working properly, it should show the following information which is highlighted by the box. If it is not showing, there might be problems with the board and/or the cable.

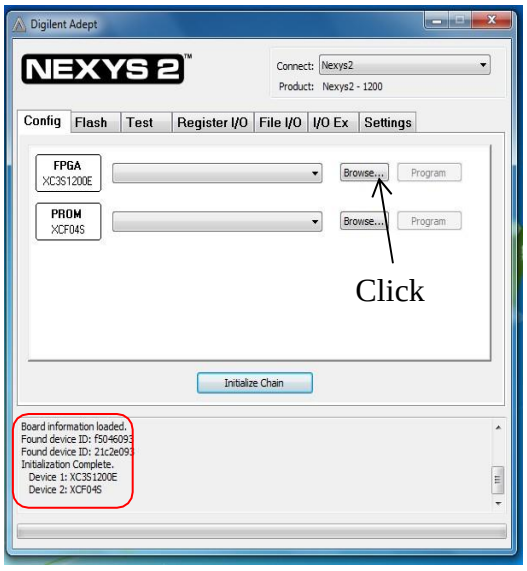


Figure 17: Adept Opening screen after connecting FPGA board

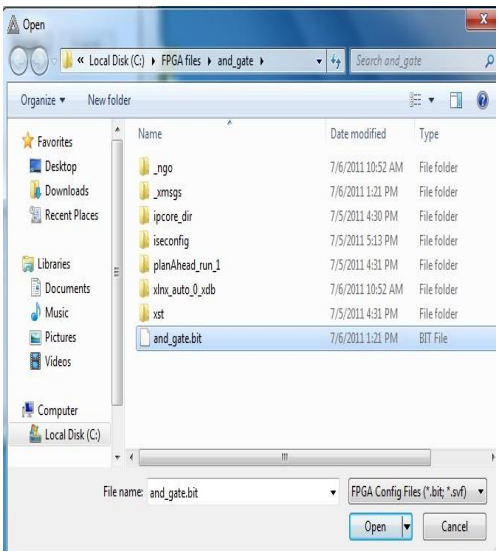


Figure 18: Choosing the bit stream file

- To download the program, click on Browse command in the first Row which says FPGA (XC3S1200E). Browse to the project folder and choose the corresponding bit file as shown below **figure 18** and click open.

- Once this is done, a warning window will pop up if the clock is set to CCLK which can be closed by clicking Yes as shown below **(Figure 19)**. Or it can be fixed by setting Clock source to JTAG Clock in the Synthesize setting in Xilinx ISE.

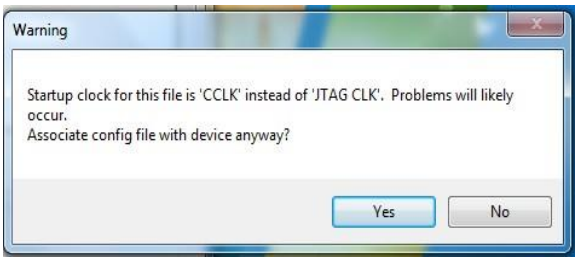


Figure 19: Warning about **CCLK** and **JTAG CLK**

- Now click on the Program button **(figure 20)** to program the FPGA and if it successfully programming the following information should show up in the status window **(figure 21)**. Ensure that the "Programming Successful" message appears in the message window.

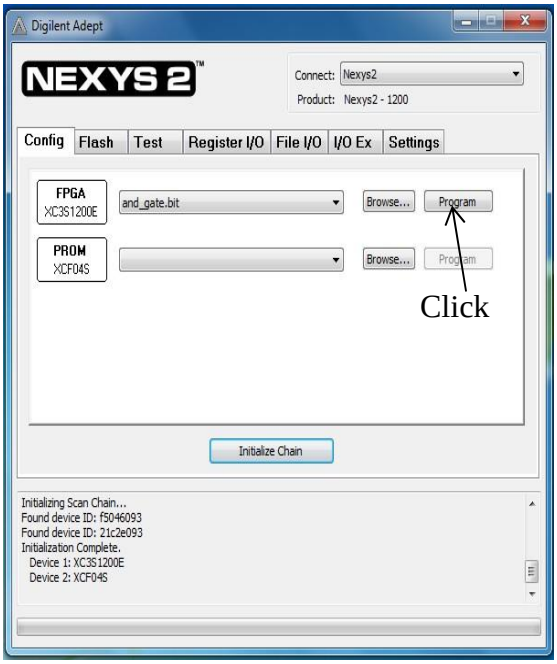


Figure 20: Programming

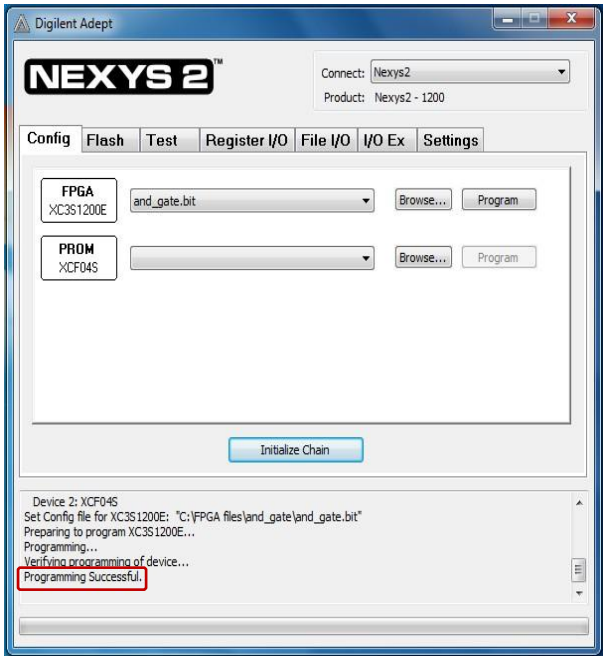


Figure 21: Showing the Programming Status

4. Programming the Device through Digilent iMpact

- You can start the programming process by double clicking Configure Target Device and ISE will launch yet another Xilinx tool called iMpact. A warning box appears complaining about "No iMpact project file exists…", so just click OK to launch iMpact as it will automatically read your existing project.

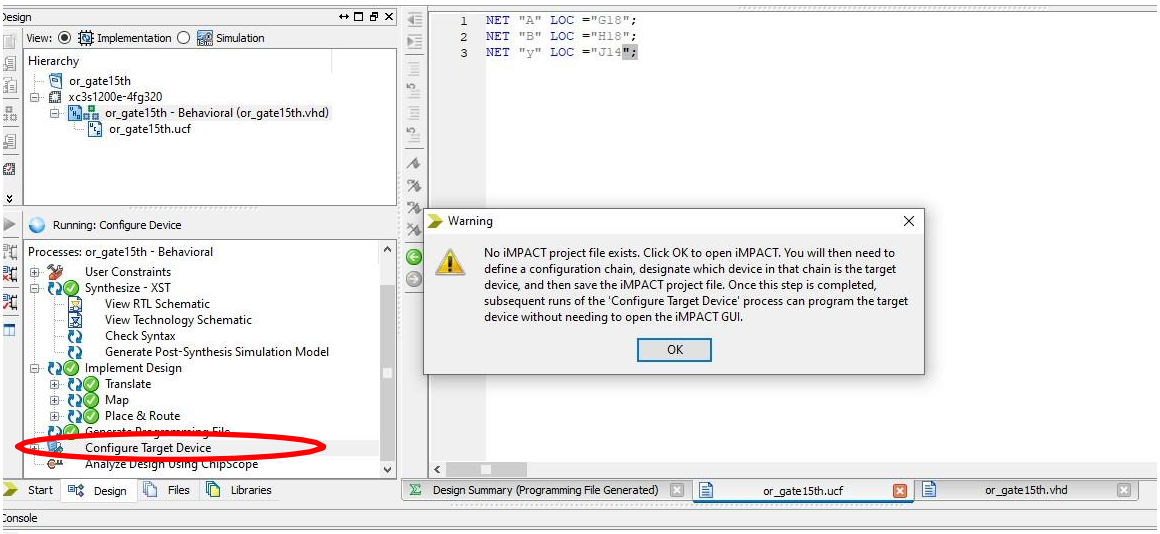


Figure 22: Opening iMpact tools-warning message

- In the ISE iMpact window, which again looks a lot like ISE, double click Boundary Scan in the top left pane. In the boundary scan windows in the main pane, where it says “Right click to Add Device or Initialize chain” right-click in the middle of the page and select Initialize Chain or just press *Ctrl + I*. This will ensure there is a good connection to your board and it can communicate with it.
-

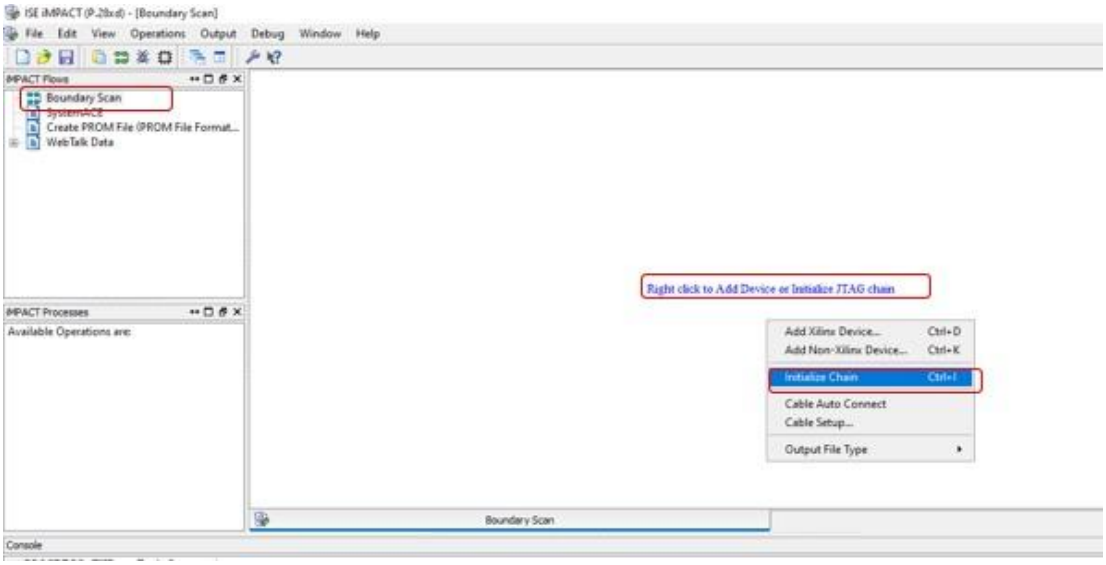


Figure 23: Boundary Scan and Initialize chain

- After iMpact verifies that the cable is connected it will prompt you to load your .bit file that you generated in previous step. Note that this file selection window doesn’t always default to your existing project so you may need to navigate to your project folder and locate the .bit file. This file is always named to your top module so in our case it is with extension **.bit.** Double click or select the bit file and click open, again make sure it is the right file.

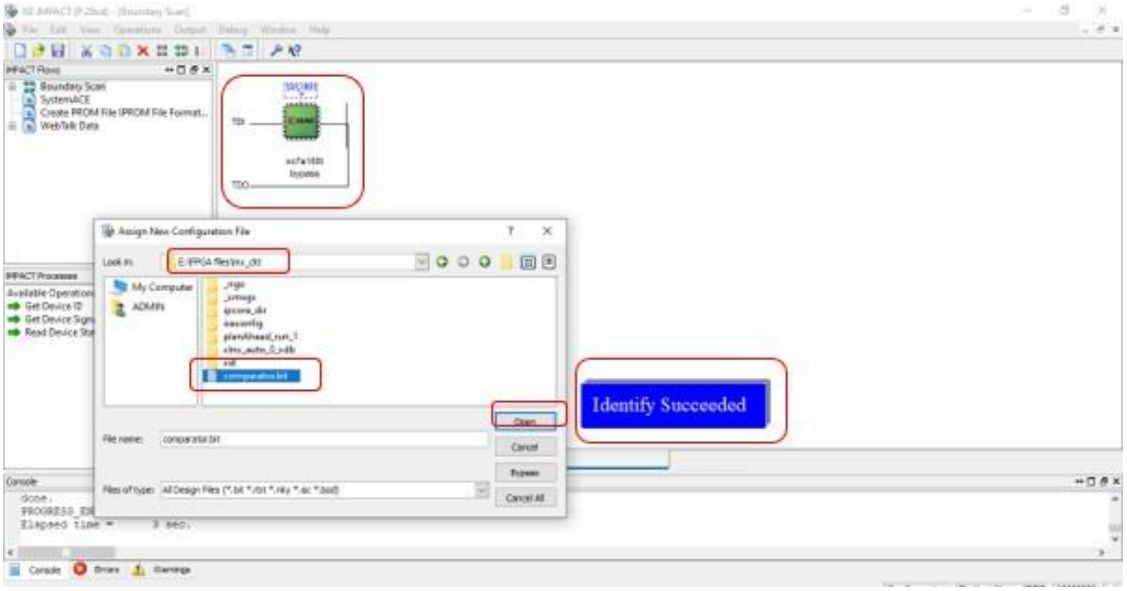


Figure 24: Browse the bit file

- After the bit file is read in, iMpact prompts you to attach a PROM controller, just click NO to skip this step since we're not putting anything in the Flash memory.

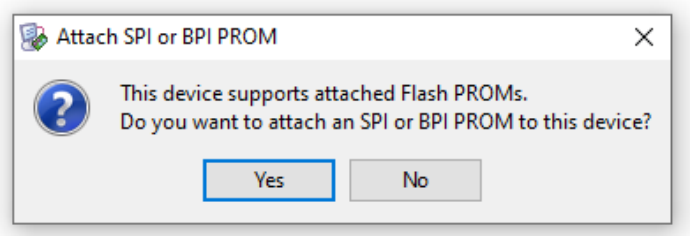


Figure 25: Warning for attach Flash PROMs

- In the next dialog box you would be verifying which device on the board you're targeting but in our case we only have the FPGA chip to program, so click Ok and the preparation for programming the board is complete.

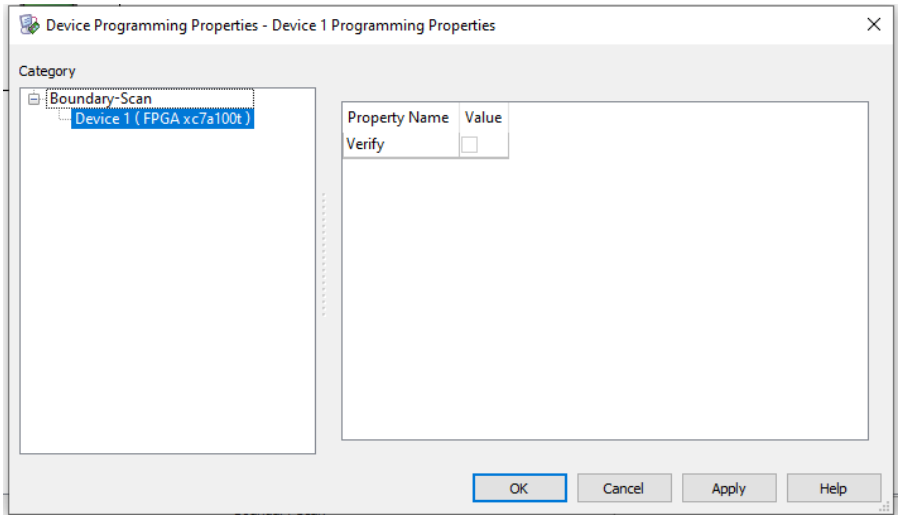


Figure 26: Verify the targeting Device

- All that is left is to right click on the green chip icon with the Xilinx logo in the main pane and click Program. After the communication bar finishes, your design is programmed to the **NEXYS4 DDR** FPGA board.

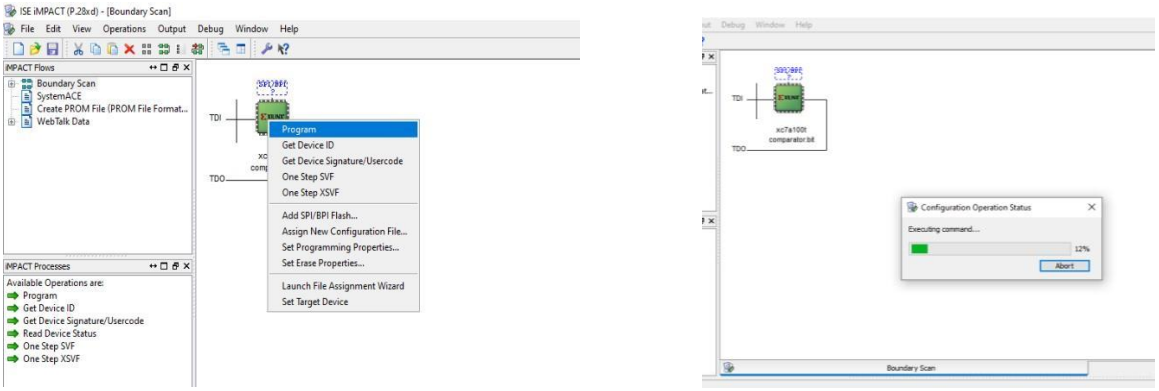


Figure 27: Program the FPGA Device

1. Write down the steps for creating *New Project*.

Open the software and go through the "File" tab and click on the new project.

Step :- 2 : to make the hardware and software like below

- Product category :All
- Family: Spartan 3E
- Device : XC3S1200E / XCS3S500E
- Package :- FG320
- Speed Grade:- -4
- Top level source :- HDL
- Synthesis Tool :- XST (VHDL /Verilog)
- Preferred Language :- VHDL
- Property specification in project file :- Store All values
  VHDL source Analysis standard :- VHDL-93/ VHDL -2000%

Step 3:- Click Next and Finish

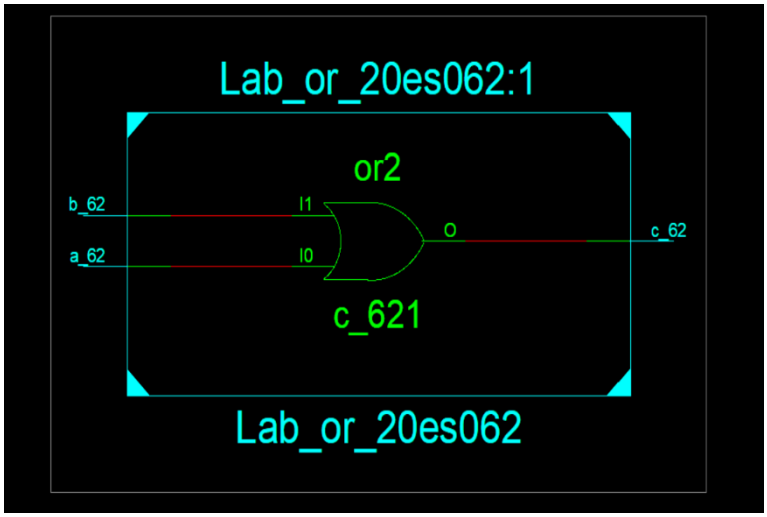
2. Write down the steps for creating New Source File (VHDL module).
   - Click on "project" Tab and New source
   - Select "VHDL Module" from options.
   - Select inputs and outputs .
   - Click next and finish option

Review Questions:

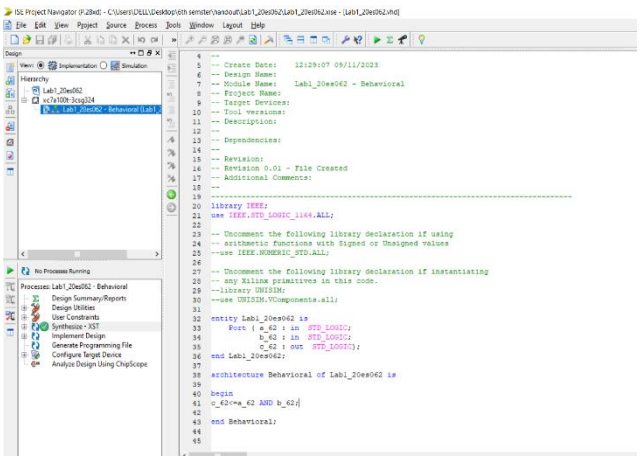
1. Repeat the same Exercise for All Logic Gates.

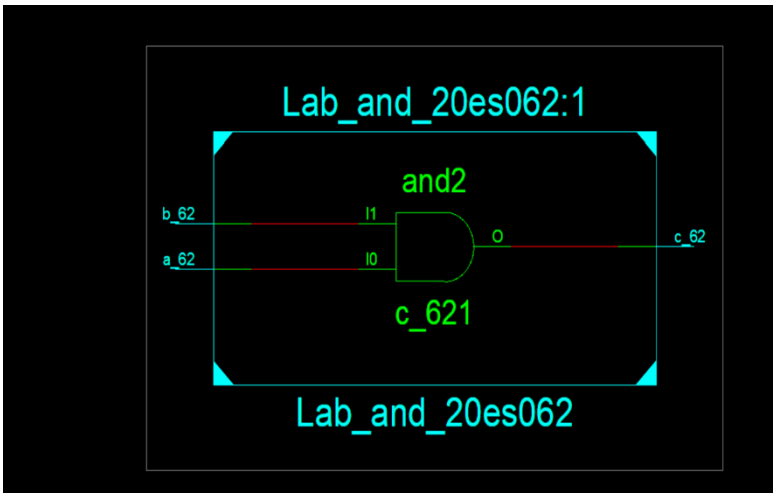
OR gate

```
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity Lab1_20es062 is
    Port ( a_62 : in  STD_LOGIC;
           b_62 : in  STD_LOGIC;
           c_62 : out  STD_LOGIC);
end Lab1_20es062;

architecture Behavioral of Lab1_20es062 is

begin
c_62<=a_62 or b_62;

end Behavioral;
```



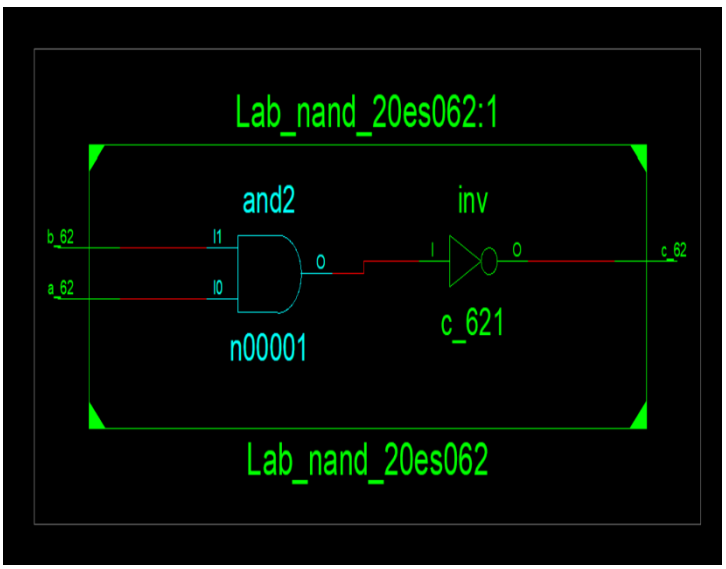
And gate





NAND gate

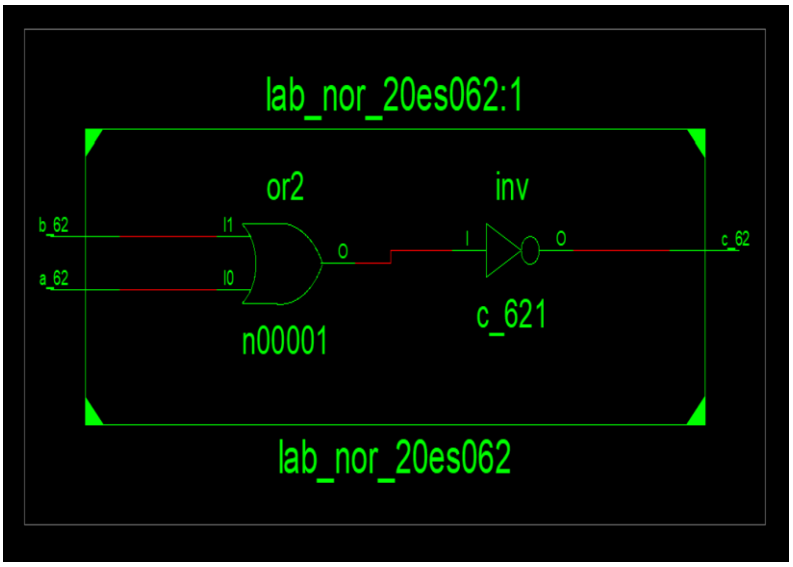
```
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity Lab1_20es062 is
    Port ( a_62 : in  STD_LOGIC;
           b_62 : in  STD_LOGIC;
           c_62 : out  STD_LOGIC);
end Lab1_20es062;

architecture Behavioral of Lab1_20es062 is

begin
c_62<=a_62 nand b_62;

end Behavioral;
```



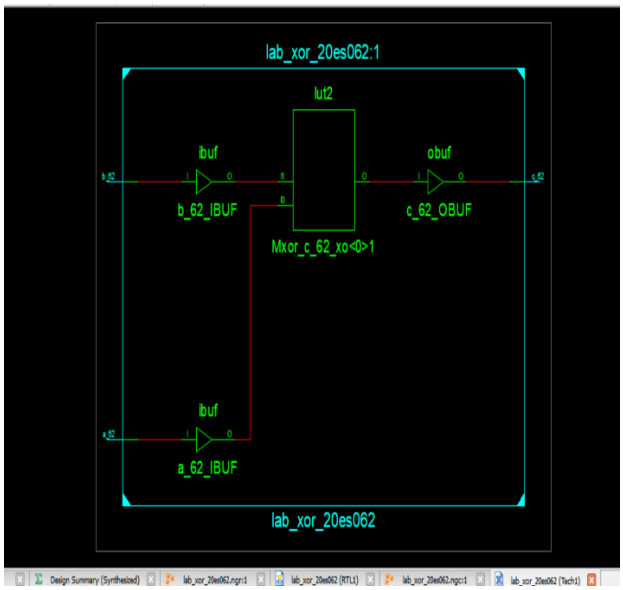
NOR gate

```
18  --
19  ----------------------------------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity Lab1_20es062 is
33      Port ( a_62 : in  STD_LOGIC;
34             b_62 : in  STD_LOGIC;
35             c_62 : out  STD_LOGIC);
36  end Lab1_20es062;
37
38  architecture Behavioral of Lab1_20es062 is
39
40  begin
41  c_62<=a_62 nor b_62;
42
43  end Behavioral;
44
45
```

Lab1_20es062.vhd | Design Summary (out of date)



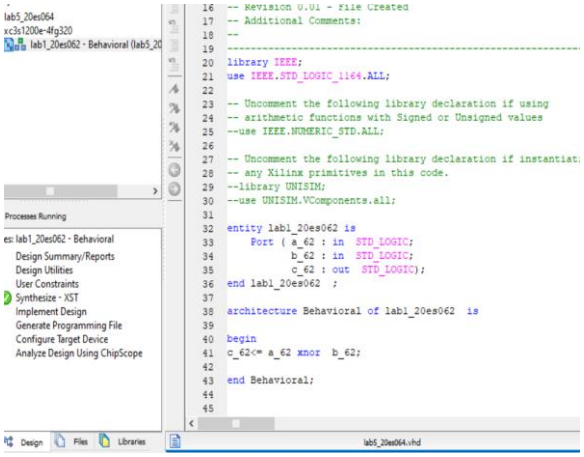
XOR GATE

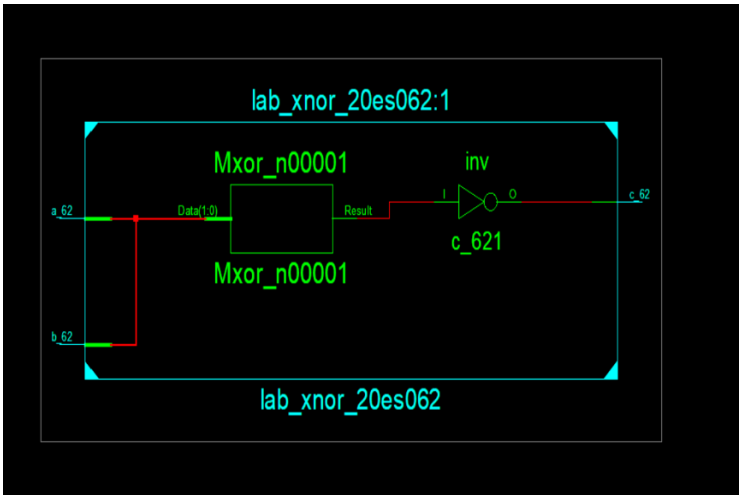



```
18  --
19  ----------------------------------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity Lab1_20es062 is
33      Port ( a_62 : in  STD_LOGIC;
34             b_62 : in  STD_LOGIC;
35             c_62 : out  STD_LOGIC);
36  end Lab1_20es062;
37
38  architecture Behavioral of Lab1_20es062 is
39
40  begin
41  c_62<=a_62 xor b_62;
42
43  end Behavioral;
44
45
```

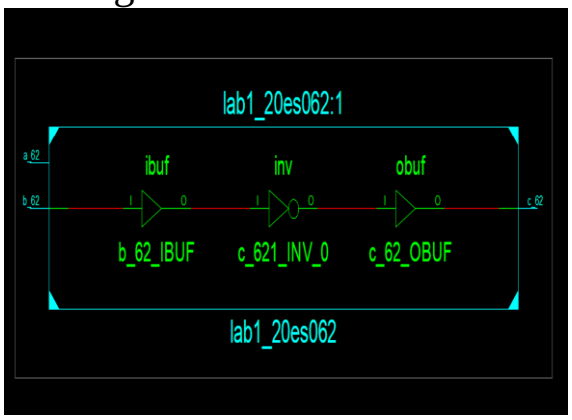
Lab1_20es062.vhd | Design Summary (out of date)

XNOR





NOT gate




```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity lab1_20es062 is
    Port ( a_62 : in  STD_LOGIC;
           b_62 : in  STD_LOGIC;
           c_62 : out  STD_LOGIC);
end lab1_20es062  ;

architecture Behavioral of lab1_20es062  is

begin
c_62<= not ( b_62) ;

end Behavioral;
```

Final assignment

- X= AB + C

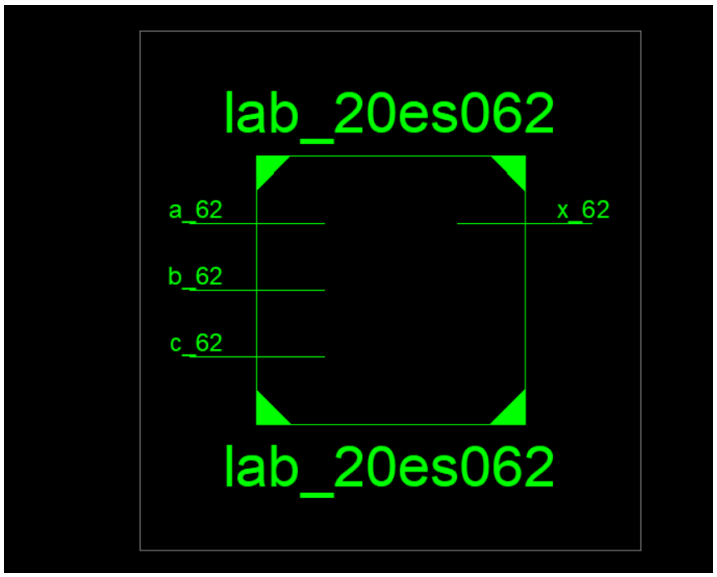
Code :-

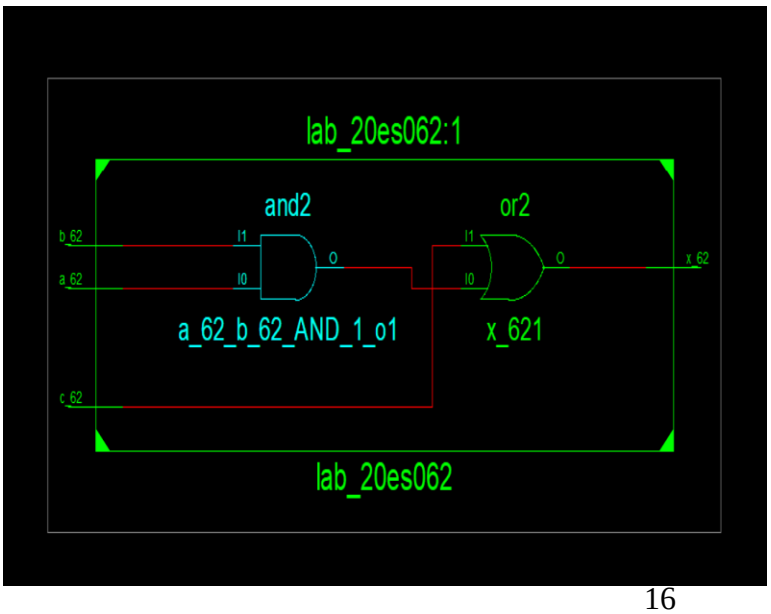
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity lab_20es062  is
    Port ( a_62 : in  STD_LOGIC;
           b_62 : in  STD_LOGIC;
           c_62 : in  STD_LOGIC;
           x_62 : out  STD_LOGIC);
end lab_20es062;

architecture Behavioral of lab_20es062 is

begin
 x_62<= (a_62 and b_62) or c_62;

end Behavioral;
```

UCF file

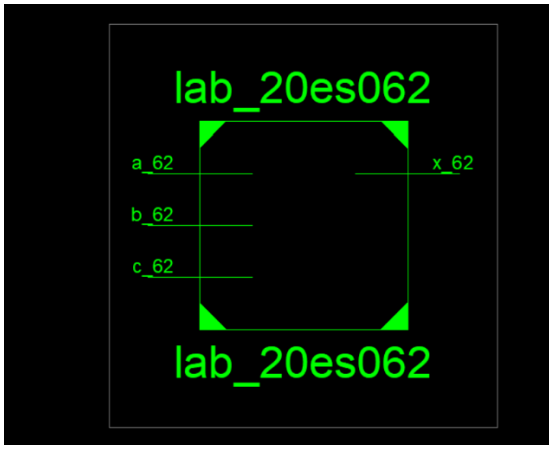
```
NET "A_62" = loc "J15" | IOSTANDARD = "LVCMOS33";
NET "B_62" = loc "L16" | IOSTANDARD = "LVCMOS33";
NET "C_62" = loc "M13" | IOSTANDARD = "LVCMOS33";
NET "X_62" = loc "H17" | IOSTANDARD = "LVCMOS33";
```

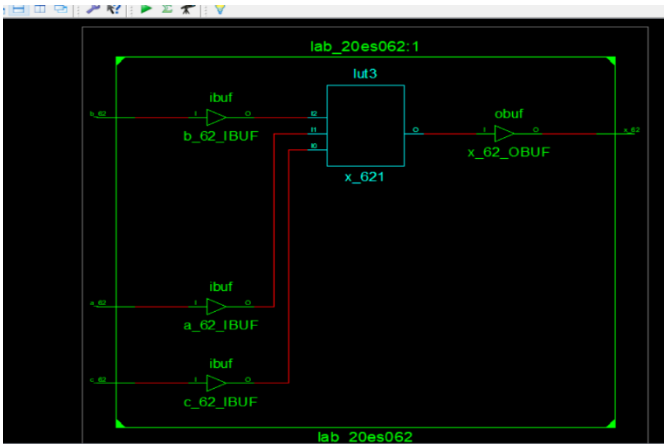
RTL





Schematic





- Y= B(A+C) + CD

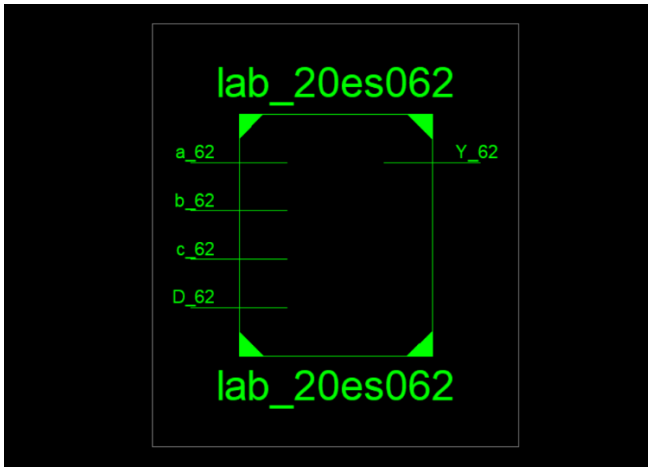
Code :-

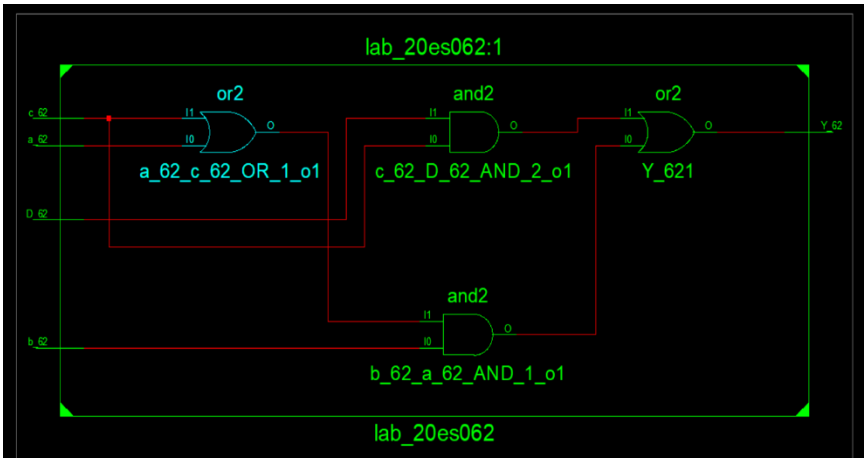
```
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity  lab_20es062  is
33      Port ( a_62 : in  STD_LOGIC;
34             b_62 : in  STD_LOGIC;
35             c_62 : in  STD_LOGIC;
36             D_62 : in  STD_LOGIC;
37             Y_62 : out  STD_LOGIC);
38  end  lab_20es062  ;
39
40  architecture Behavioral of  lab_20es062  is
41
42  begin
43  Y_62<= ( b_62 and (a_62 or c_62)) or ( c_62 and D_62);
44  end Behavioral;
```

UCF file

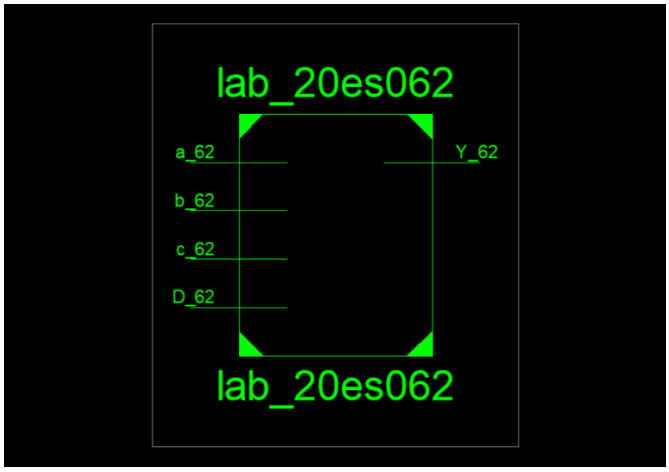
```
1  NET "A_62" = loc "J15" | IOSTANDARD = "LVCMOS33";
2  NET "B_62" = loc "L16" | IOSTANDARD = "LVCMOS33";
3  NET "C_62" = loc "M13" | IOSTANDARD = "LVCMOS33";
4  NET "D_62" = loc "R15" | IOSTANDARD = "LVCMOS33";
5  NET "Y_62" = loc "M13" | IOSTANDARD = "LVCMOS33";
```

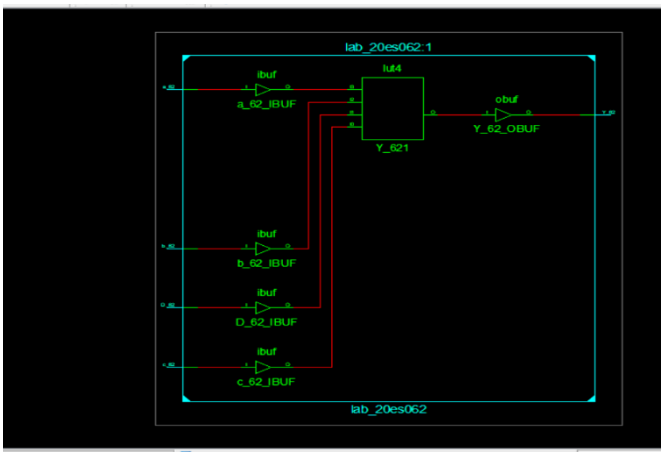
Y= B(A+C) + CD
RTL





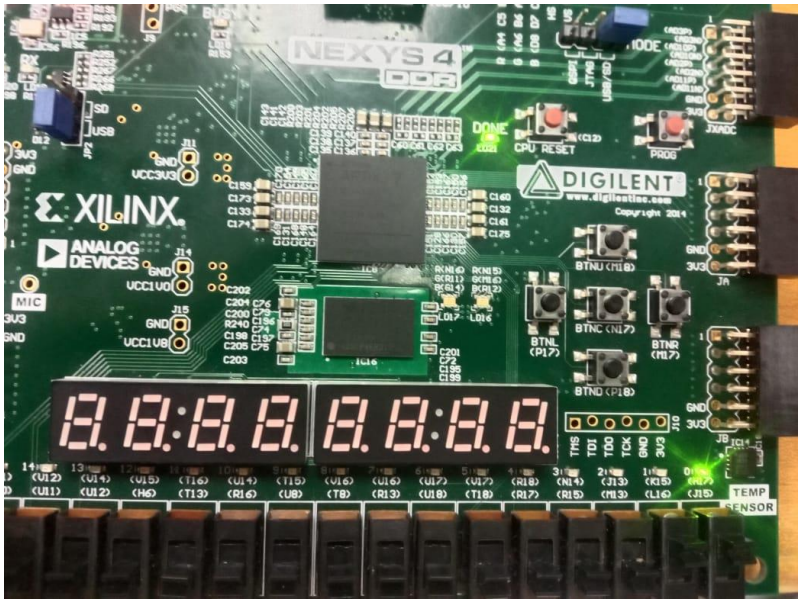
Schematic



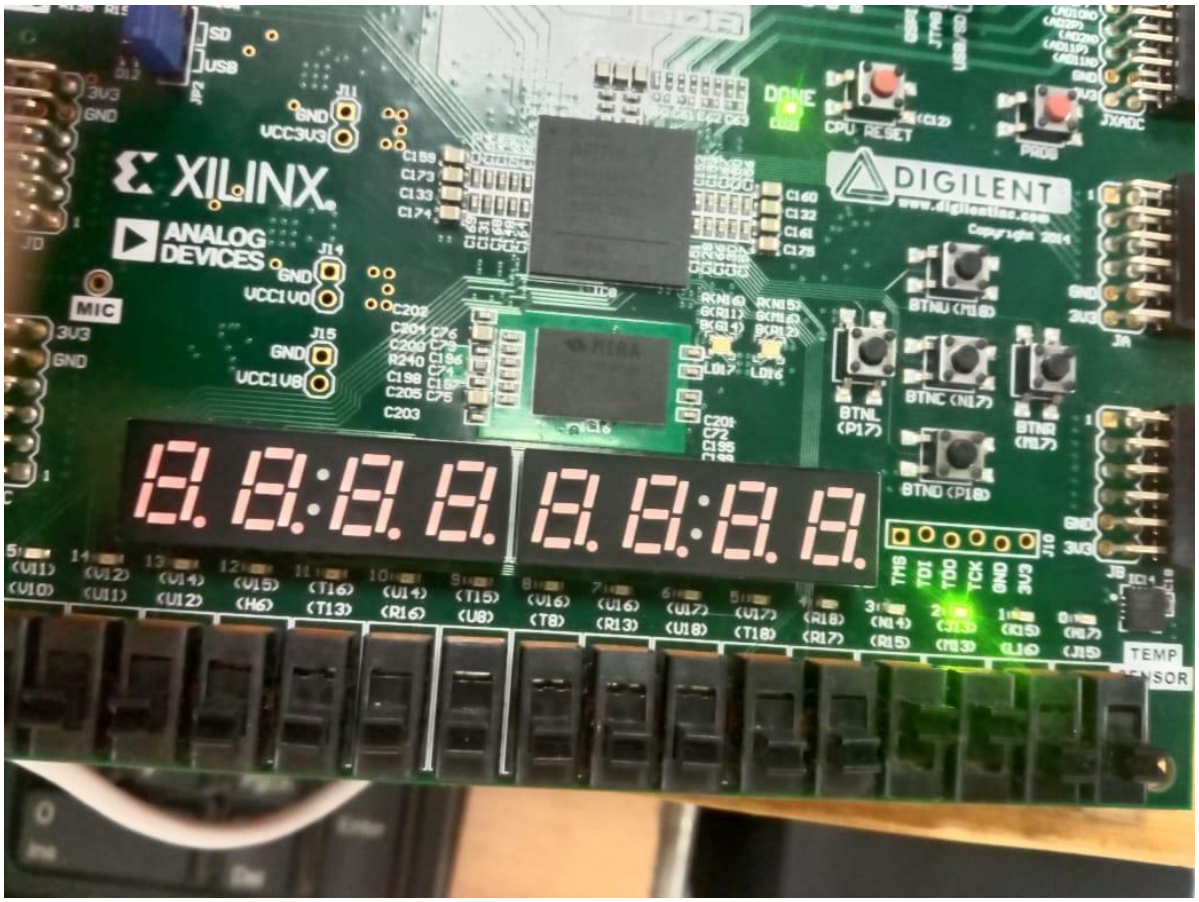


X= AB + C

Hardware result :-



Y= B(A+C) + CD


XILINX
ANALOG DEVICES
DIGILENT
CPU RESET
BTNU (M18)
BTNL (P17)
BTNC (N17)
BTNR (M17)
BTND (P18)
MIC
TEMP

Final Assignment:

3. Write down the steps for creating *New Project*.

Open the software and go through the “File” tab and click on the new project.

Step :- 2 : to make the hardware and software like below

- Product category :All
- Family: Spartan 3E
- Device : XC3S1200E / XCS3S500E
- Package :- FG320
- Speed Grade:- -4
- Top level source :- HDL
- Synthesis Tool :- XST (VHDL /Verilog)
- Preferred Language :- VHDL
- Property specification in project file :- Store All values
  VHDL source Analysis standard :- VHDL-93/ VHDL -2000%

Step 3:- Click Next and Finish