

# *Introduction to FPGA Design*



A Quick Guide to using  
Xilinx ISE to synthesise and download  
VHDL code onto an FPGA

(Integrated System Environment)

## **Introduction**

The Xilinx ISE (Integrated System Environment) is a complete design environment for the design and programming of CPLD (Programmable Logic Devices) and FPGA (Field Programmable Gate Array) devices from Xilinx®. Xilinx sells a range of devices with different capabilities and device densities. Some of these devices contain over 10 million gate equivalents (a gate equivalent is usually defined as a 2-input AND function). The device used in the Engineering Lab is either a Spartan 3 device with a 200,000 gate capacity, which is large enough for most designs or the larger Spartan 6 XC6SLX16. We use Modelsim® for the design and simulation of circuits and systems using VHDL and then the Xilinx ISE suite for synthesising the VHDL code into logic gates, placing and routing these logic gates on the target device and programming the target device. The ISE can also be used for logic design and simulation using VHDL. For anyone interesting in finding out more, a free version of this software (called the Xilinx WebPack) is available from the Xilinx website [www.xilinx.com](http://www.xilinx.com).

In this experiment the Xilinx ISE tools will be used to:

- Import a completed VHDL design from your Modelsim Project Directory
- Import the User Constraints for the project using the User Constraint (.ucf) file provided.
- Synthesise the design (convert the VHDL code into logic functions)
- Place and Route the Design (optimise the logic and place and route it on the FPGA fabric)
- Generate the programming file for the design (generate a .bit file containing the configuration data for the FPGA chip).

The last part of the design process involves programing/configuring the target device (downloading the .bit file onto the FPGA) but will not be carried out in this session. The sections “Configure Target Device” on page 17 and “Testing the Design” on page 23 refer to this last part (the hardware configuration). You do not need them for this assignment but you can read them to get an idea of the procedure of programming the FPGA.

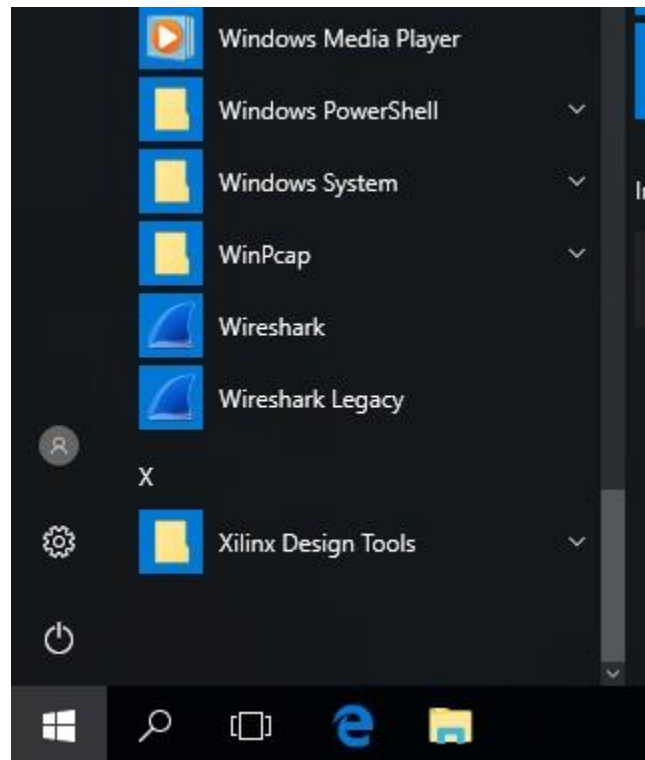
If there are no problems with your VHDL design this should be a straightforward process. However, problems in the design can result in errors during the implementation stages. If there are problems consult the demonstrator who will be able to help you resolve them.

## **Getting Started: Creating a New Project**

To start the Xilinx Program find

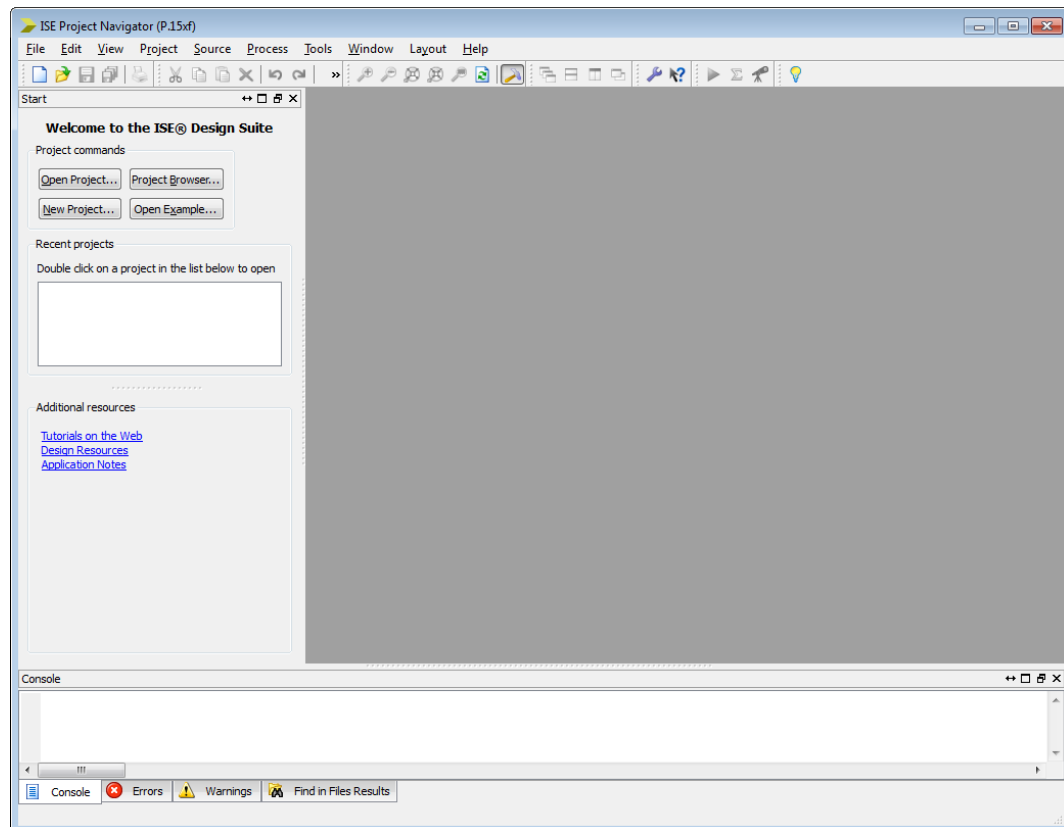
**Xilinx Design Design Tools > 64-bit Project Navigator**

in the program menu (see Figure 1).

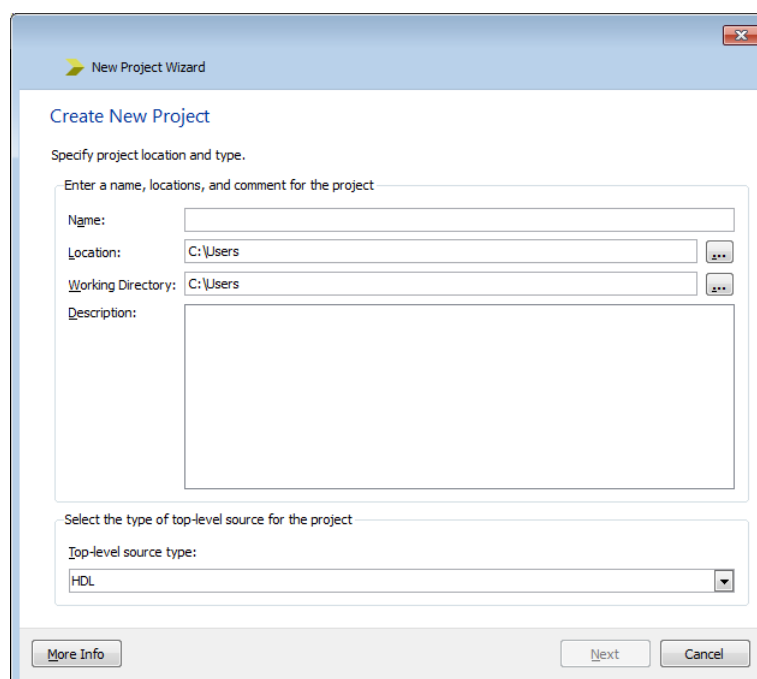


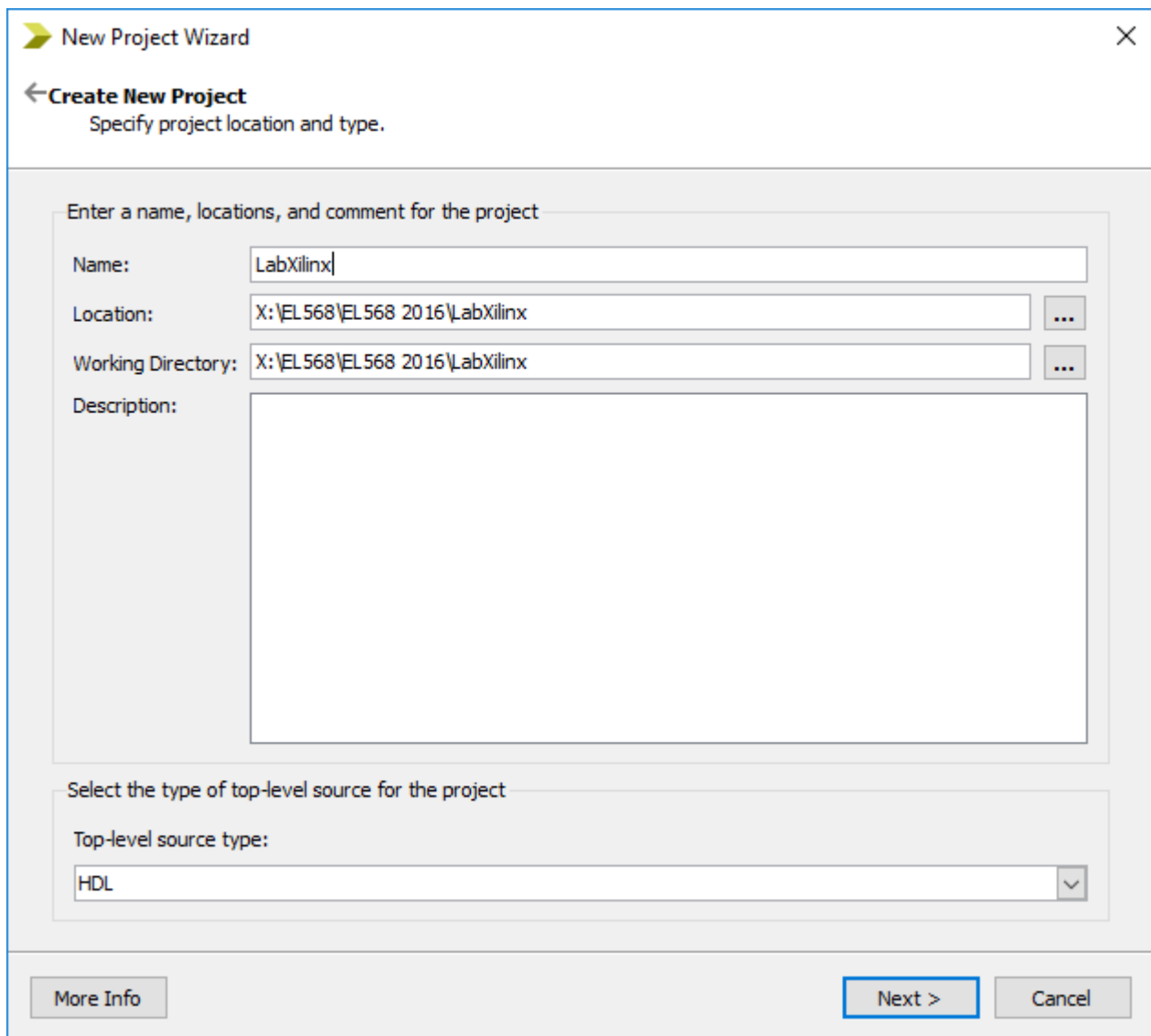
**Figure 1 Initiating Xilinx ISE**

The default window should be of the form shown in Figure 2.

**Figure 2 Default ISE Window**

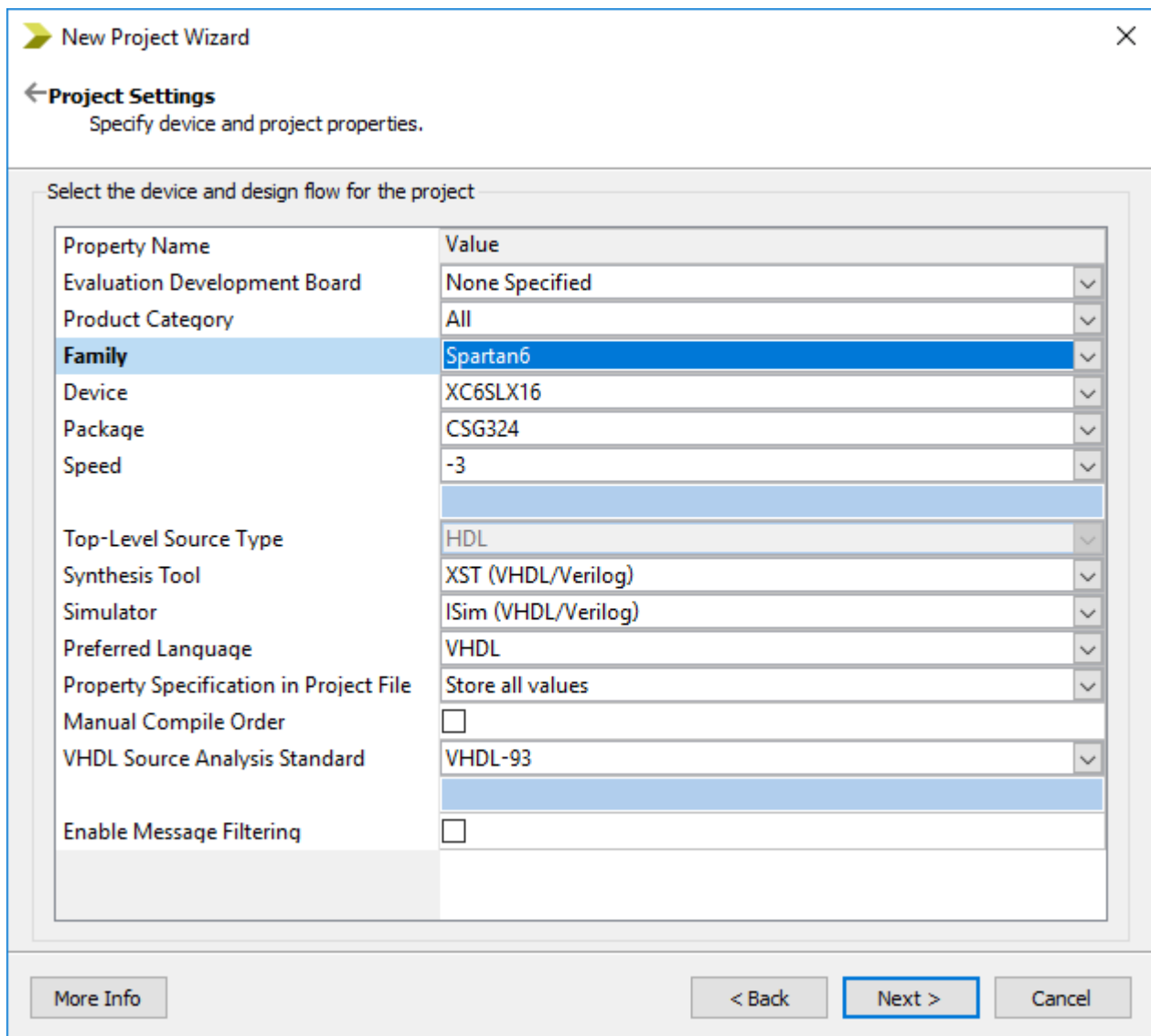
To begin a new project choose **New Project** from the menu. The following dialog box will appear which will need to be filled in with the project details, including the project directory which you should set up on your X:drive. (See Figures 3 and 4)

**Figure 3 Default Dialog Box**

The image shows the 'New Project Wizard' dialog box in Xilinx ISE. The title bar says 'New Project Wizard' with a close button. Below the title bar, there's a back arrow and the text 'Create New Project' followed by 'Specify project location and type.' The main area is divided into two sections. The first section is titled 'Enter a name, locations, and comment for the project'. It contains four fields: 'Name:' with the text 'LabXilinx', 'Location:' with the text 'X:\EL568\EL568 2016\LabXilinx' and a browse button (...), 'Working Directory:' with the text 'X:\EL568\EL568 2016\LabXilinx' and a browse button (...), and 'Description:' with a large empty text area. The second section is titled 'Select the type of top-level source for the project'. It contains a 'Top-level source type:' label and a dropdown menu currently showing 'HDL'. At the bottom, there are three buttons: 'More Info', 'Next >' (which is highlighted with a blue border), and 'Cancel'.

**Figure 4 Enter project name and project location**

The next dialog box to appear (see Figure 5) will require you to enter specific data relating to the project and the device you will be programming. There are two types of development boards in the Simulation Lab. Your board is a Nexys3 board which uses a Spartan6 XC6SLX16 in a CSG324 package with a speed grade of -3 (an indication of the maximum speed of the device, not particularly important in this experiment). Make sure you have entered the correct project details as shown for the Spartan6 device in Figure 5. Make sure to change the preferred language to VHDL.

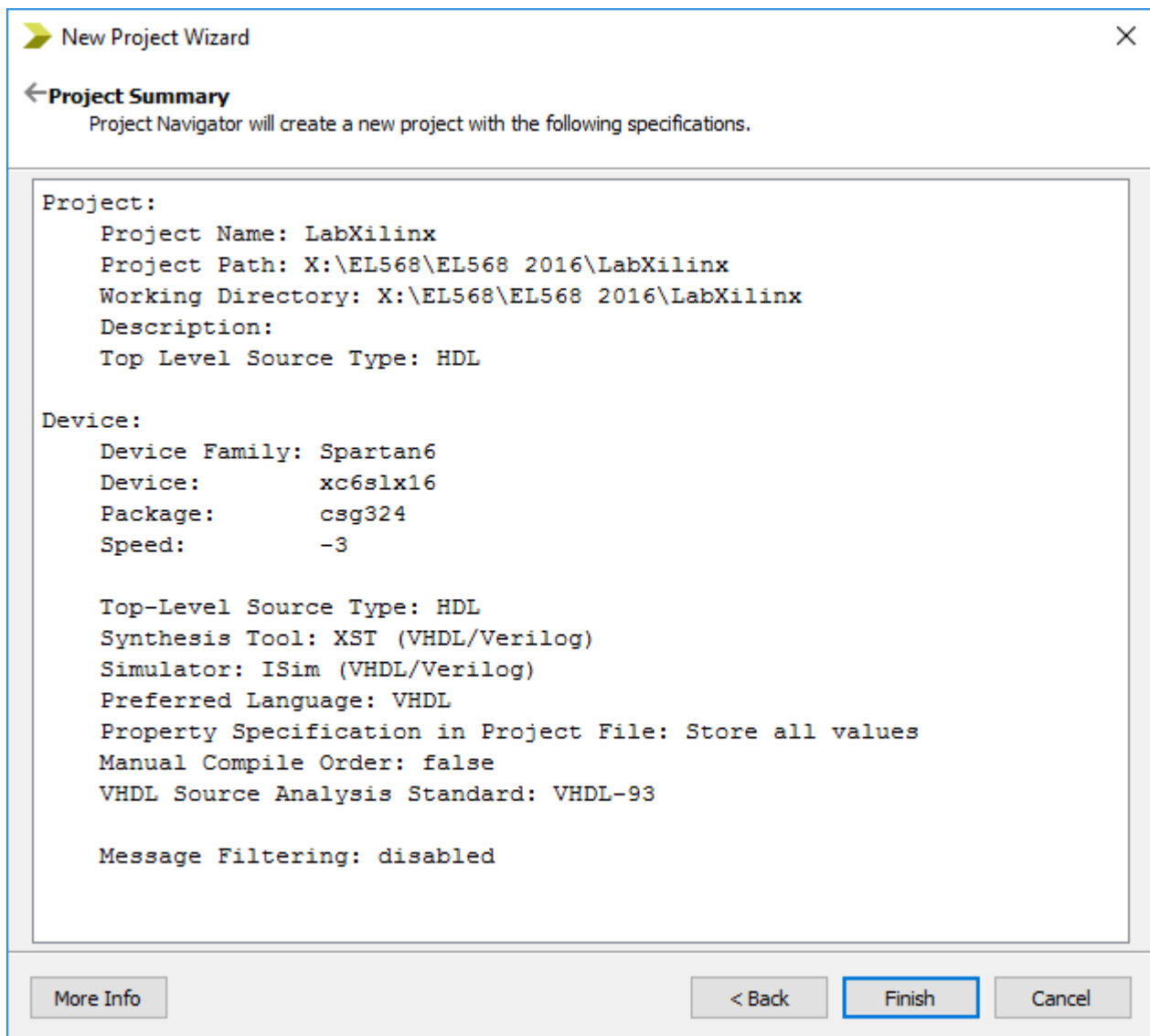


The image shows the 'New Project Wizard' window in Xilinx ISE, specifically the 'Project Settings' step. The window title is 'New Project Wizard' with a close button (X) in the top right corner. Below the title bar, there is a back arrow and the text 'Project Settings' followed by 'Specify device and project properties.' The main area is titled 'Select the device and design flow for the project'. It contains a table with two columns: 'Property Name' and 'Value'. The 'Family' property is highlighted in blue. The 'Value' column for 'Family' shows 'Spartan6' selected from a dropdown menu. Other properties include 'Evaluation Development Board' (None Specified), 'Product Category' (All), 'Device' (XC6SLX16), 'Package' (CSG324), 'Speed' (-3), '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), 'Manual Compile Order' (checkbox), 'VHDL Source Analysis Standard' (VHDL-93), and 'Enable Message Filtering' (checkbox). At the bottom, there are three buttons: 'More Info', '< Back', and 'Next >' (which is highlighted with a blue border), and a 'Cancel' button.

Property Name	Value
Evaluation Development Board	None Specified
Product Category	All
<b>Family</b>	<b>Spartan6</b>
Device	XC6SLX16
Package	CSG324
Speed	-3
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
Manual Compile Order	<input type="checkbox"/>
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	<input type="checkbox"/>

**Figure 5 Device Characteristics**

After clicking next the Project Summary window in Figure 6 should appear. Click on Finish.



**Figure 6 Project Summary**

The project window should now look as shown in Figure 7

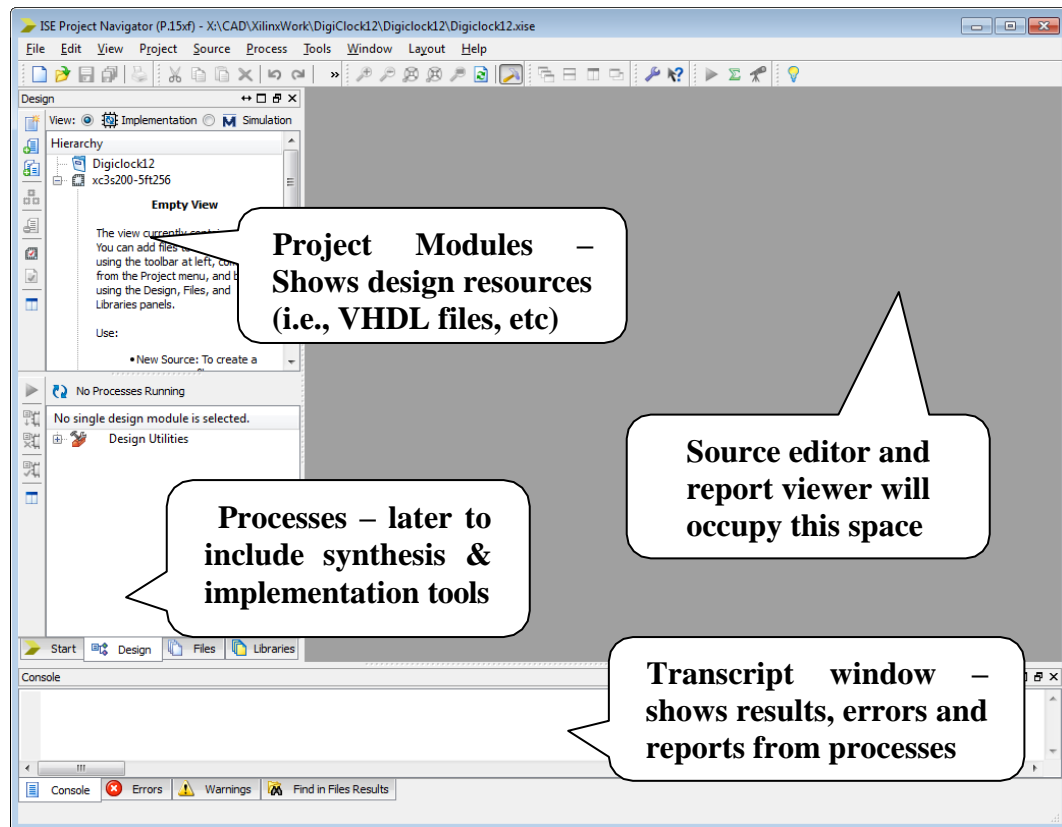
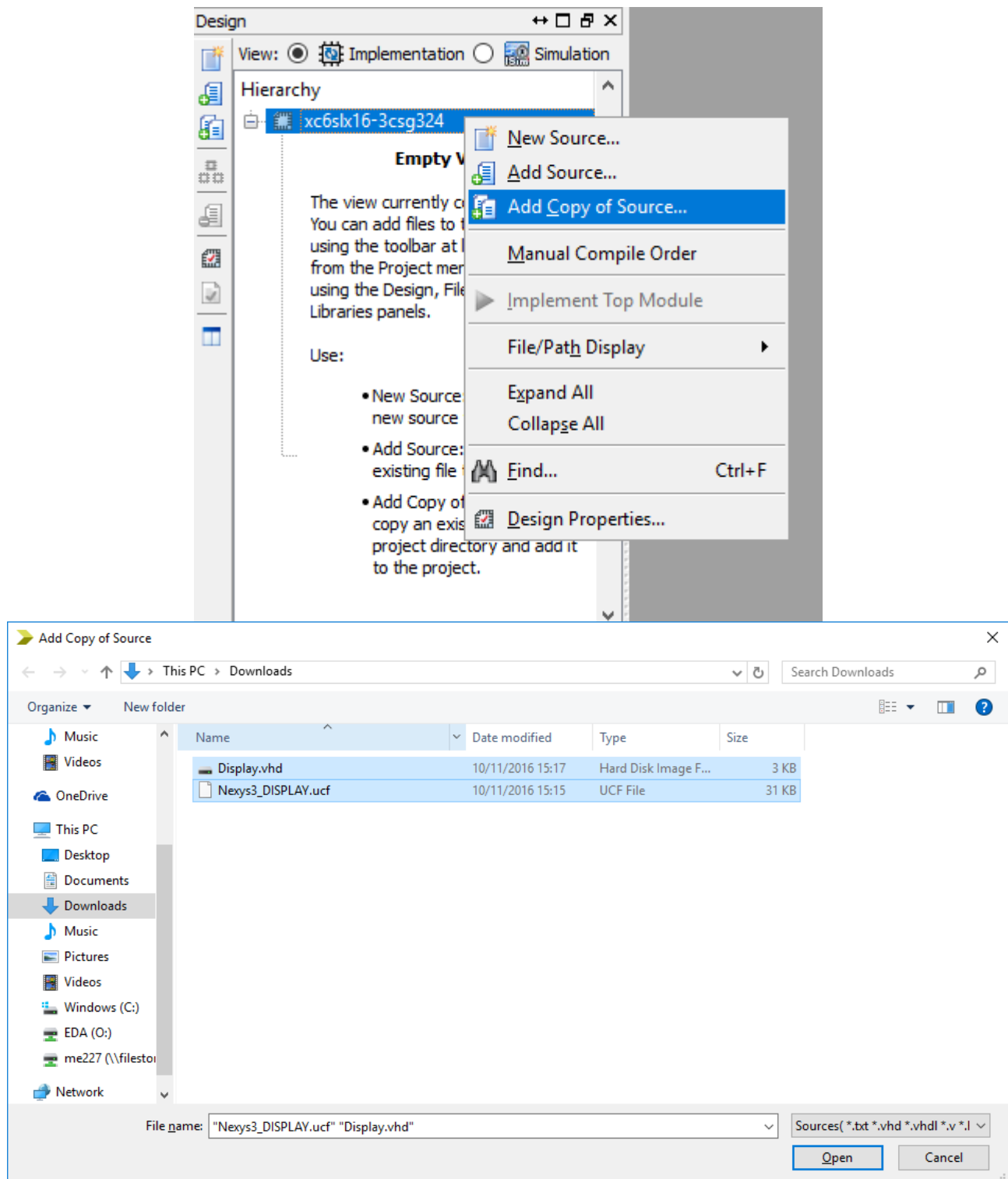


Figure 7 Updated Project Navigator Window

## Adding Source Files

It is now time to add the project source files you generated in Modelsim during the Design phase of the project. In the project navigator window find **Project > Add Copy of Source** from the drop-down menus, or right-click the project as shown in Figures 8 and 9. **DO NOT** add the TB files as they are not used to synthesise the design.



**Figure 8 Adding existing VHDL Sources**

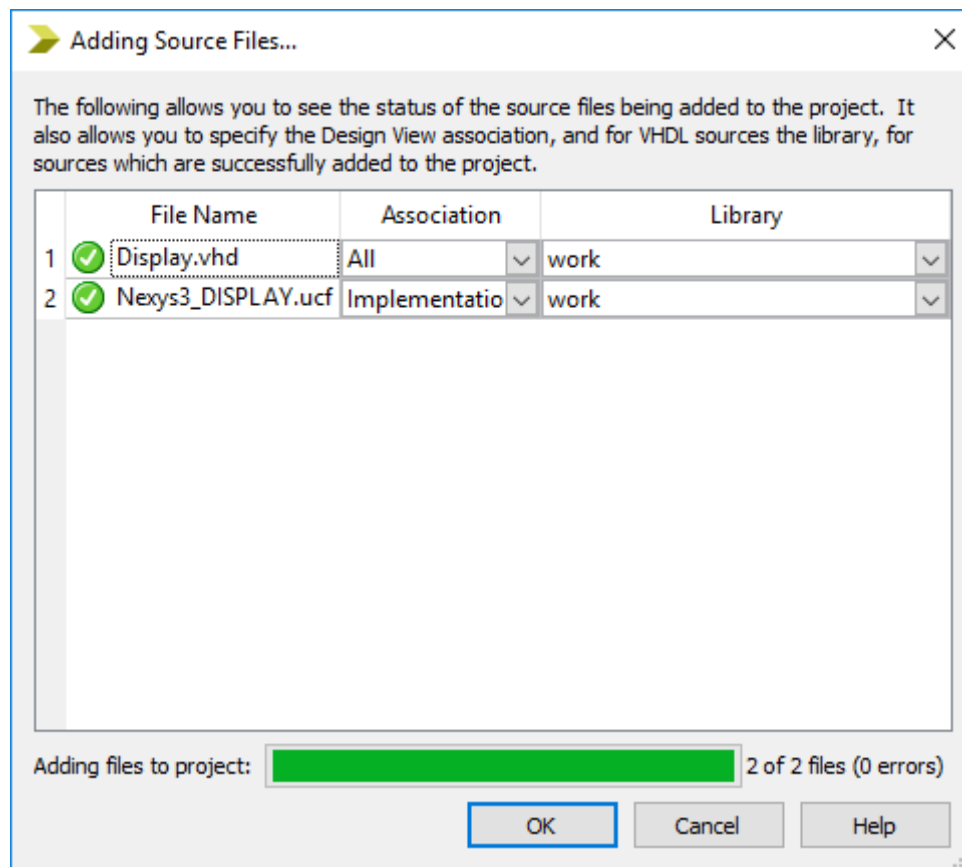


Figure 9 Add Source Dialog Box containing your .vhd files.

Click on OK to load these files into the project. The project navigator window should now look similar to that shown in Figure 10. Note the changes.

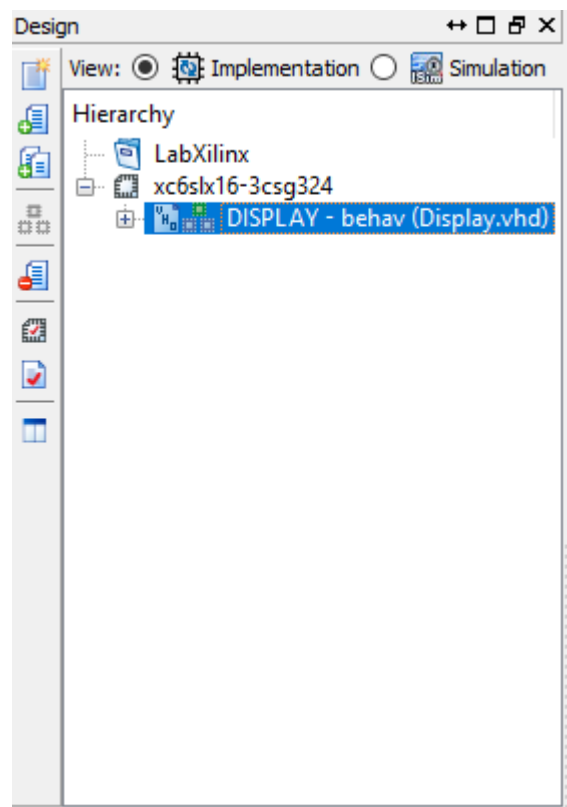


Figure 10 Imported File Summary

Click the “+” to expand the project (Figure 11)

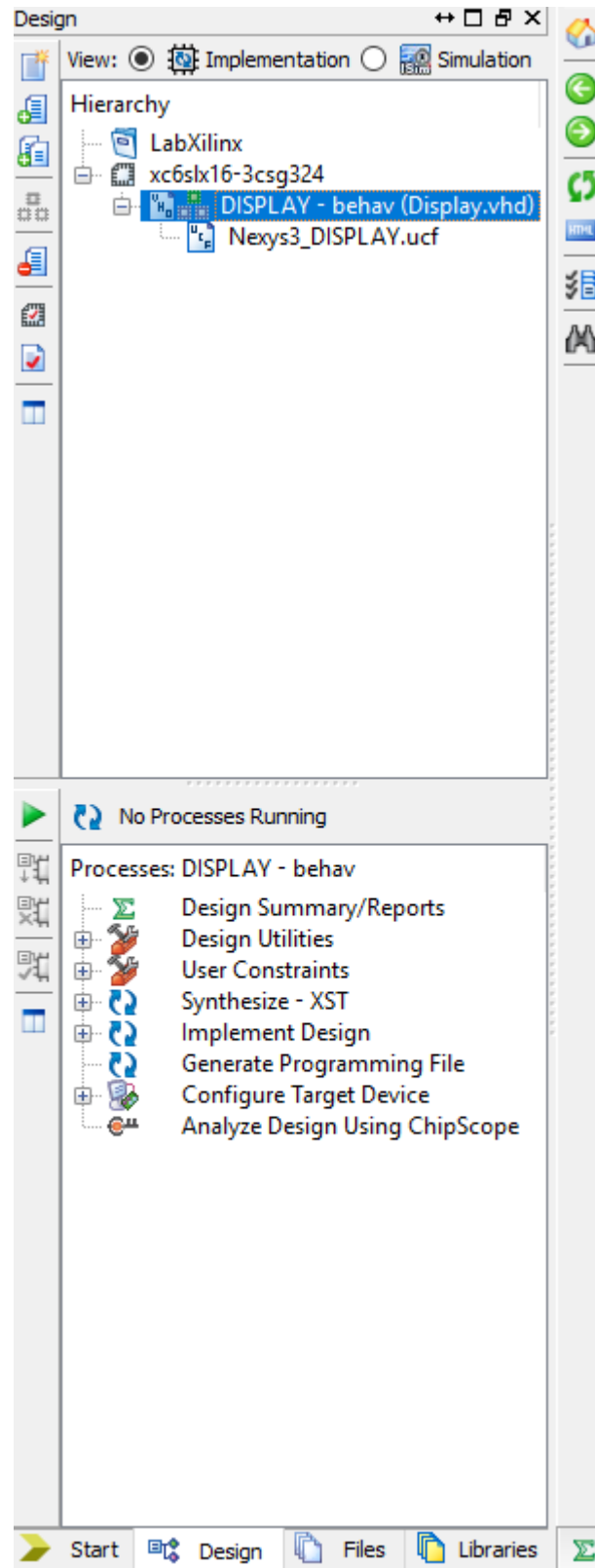


Figure 11 Imported Files

### The User Constraint File (.ucf)

The next stage is to look at the User Constraint File. This file is called `Nexys3_Display.ucf` and is a text file containing information about how and which pins of the Xilinx device are connected

on the PCB. Clearly in this project we need to define the connections to your design. The Nexys 3 FPGA Board Reference Manual (available on Moodle) provides further details about pin connections on this board that you might find useful for other projects.

Double-click the file to look at the appropriate files contents, which should be in the form shown below for the Spartan6 device.

**Note a # refers to a comment in .ucf syntax**

```
## This file is based on a general .ucf for Nexys3 rev B board
## It has been edited for the EL568 display task
## To use it in a project:
## - remove or comment the lines corresponding to unused pins
## - rename the used signals according to the project

##Clock signal
Net "clk" LOC=V10 | IOSTANDARD=LVC MOS33;
Net "clk" TNM_NET = sys_clk_pin;
TIMESPEC TS_sys_clk_pin = PERIOD sys_clk_pin 100000 kHz;

## 7 segment display
Net "digit<7>" LOC = M13 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L61N, Sch name = DP
Net "digit<6>" LOC = T17 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L51P_M1DQ12, Sch name = CA
Net "digit<5>" LOC = T18 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L51N_M1DQ13, Sch name = CB
Net "digit<4>" LOC = U17 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L52P_M1DQ14, Sch name = CC
Net "digit<3>" LOC = U18 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L52N_M1DQ15, Sch name = CD
Net "digit<2>" LOC = M14 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L53P, Sch name = CE
Net "digit<1>" LOC = N14 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L53N_VREF, Sch name = CF
Net "digit<0>" LOC = L14 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L61P, Sch name = CG

Net "digen<0>" LOC = N16 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L50N_M1UDQSN, Sch name = AN0
Net "digen<1>" LOC= N15 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L50P_M1UDQS, Schname = AN1
Net "digen<2>" LOC = P18 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L49N_M1DQ11, Schname = AN2
Net "digen<3>" LOC = P17 | IOSTANDARD = LVC MOS33; #Bank = 1, pin name =
IO_L49P_M1DQ10, Schname = AN3

## Leds
Net "Leds<0>" LOC = U16 | IOSTANDARD = LVC MOS33; #Bank = 2, pin name =
IO_L2P_CMPCLK, Sch name = LD0
Net "Leds<1>" LOC = V16 | IOSTANDARD = LVC MOS33; #Bank = 2, pin name =
IO_L2N_CMPMOSI, Sch name = LD1
```

```
Net "Leds<2>" LOC = U15 | IOSTANDARD = LVCMOS33; #Bank = 2, pin name =
IO_L5P, Sch name = LD2
Net "Leds<3>" LOC = V15 | IOSTANDARD = LVCMOS33; #Bank = 2, pin name =
IO_L5N, Sch name = LD3
Net "Leds<4>" LOC = M11 | IOSTANDARD = LVCMOS33; #Bank = 2, pin name =
IO_L15P, Sch name = LD4
Net "Leds<5>" LOC = N11 | IOSTANDARD = LVCMOS33; #Bank = 2, pin name =
IO_L15N, Sch name = LD5
Net "Leds<6>" LOC = R11 | IOSTANDARD = LVCMOS33; #Bank = 2, pin name =
IO_L16P, Sch name = LD6
Net "Leds<7>" LOC = T11 | IOSTANDARD = LVCMOS33; #Bank = 2, pin name =
IO_L16N_VREF, Sch name = LD7

## Button
Net "rst" LOC = B8 | IOSTANDARD = LVCMOS33; #Bank = 0, pin name =
IO_L33P, Sch name = BTNS. Centre button = reset

##Rest of file has been commented out
```

Confirm that the **pin assignment and pin names correspond to your VHDL design.**

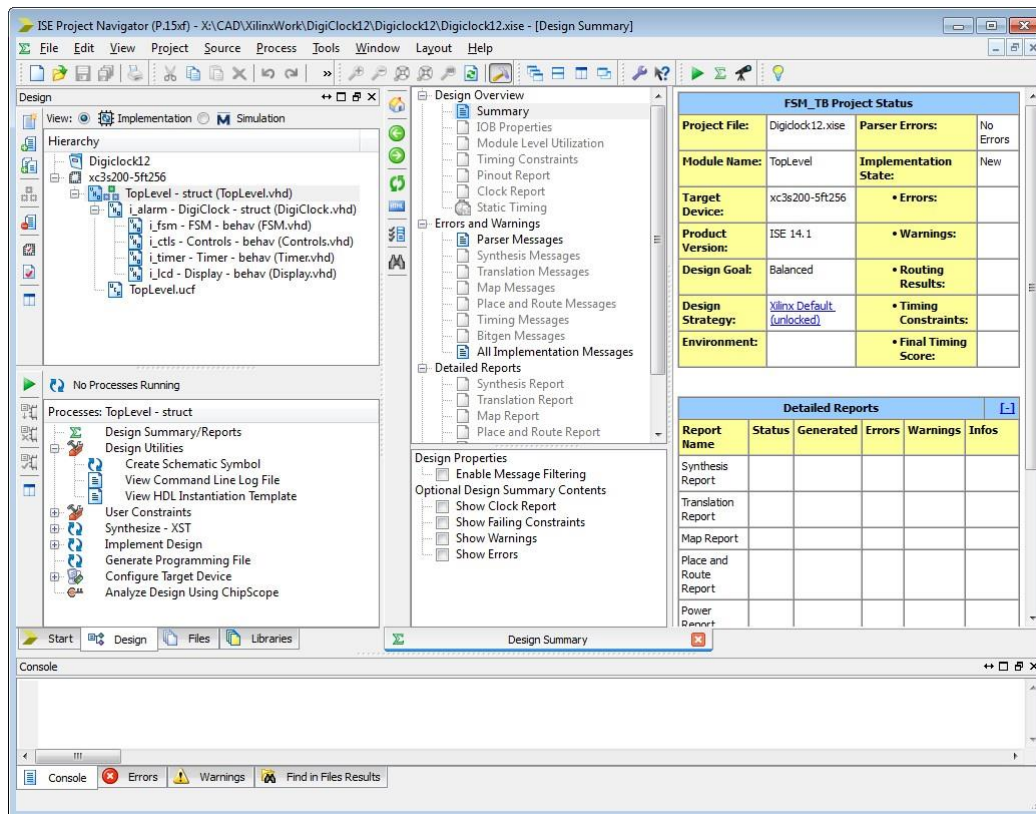
Now that the project details and files have been loaded into the design environment it is time to build the design. Listed in the processes window (see Figure 12) are the process operations that must be performed:

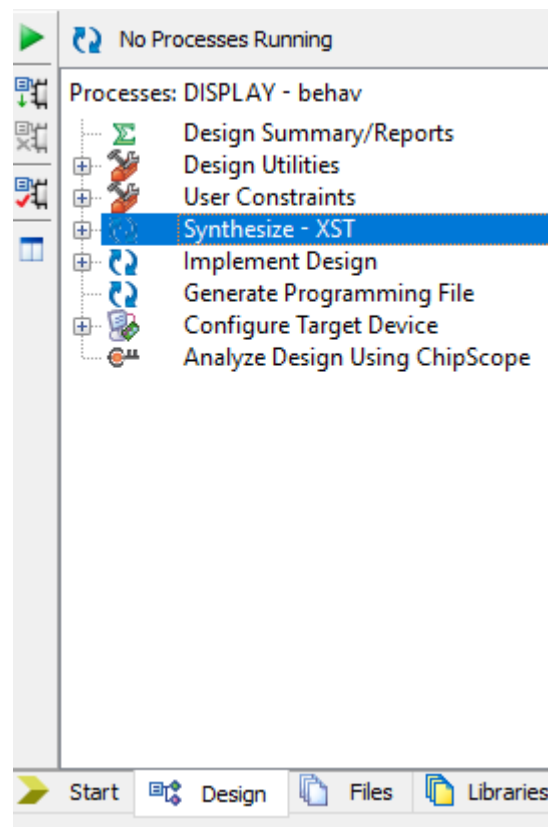
- Synthesise
- Implement Design
- Generate Programming File
- Configure Target Device.

We will now proceed through each of these stages, generating reports and files as we go. The reports contain detailed information about how the circuit is to be implemented and how much space (device resources) the design consumes. This is important information for the designer and also for finding any errors that may have occurred in the process. Although we will not be using this information (unless errors occur) it is useful to become familiar with the reports that are generated.

## Synthesising the Design

Double click on the synthesize –XST icon in the processes window (see Figure 12). The synthesis tool will take some time to complete. Note the updates in the report window and any errors or warnings that may have been generated in the console.





**Figure 12 Initiating the synthesis tool**

Note that when synthesis is finished there are some other options available for further analysis of the logic produced (click + to see them). These are:

**View RTL Schematic**

**View Technology Schematic**

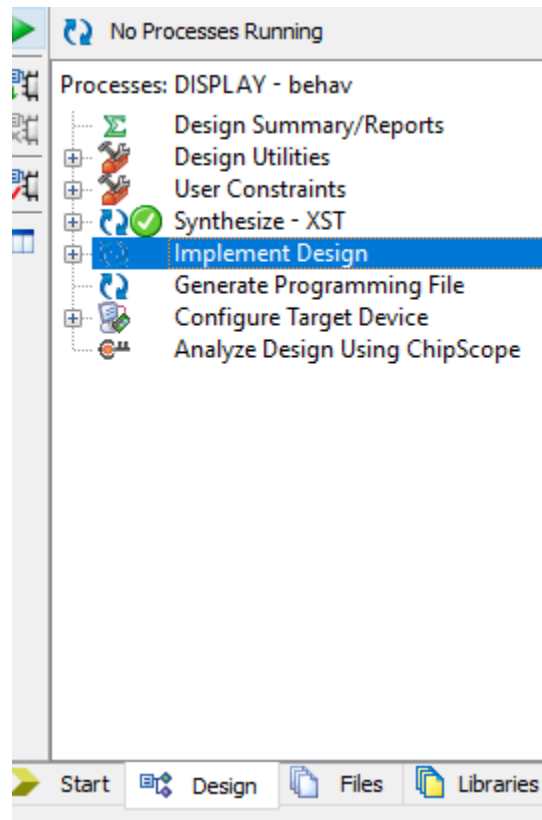
**Check Syntax**

**Generate Post Synthesis Simulation**

All of these are beyond the scope of this experiment but if you are interested, do have a look at them as they show the different steps that the synthesis procedure takes to translate a high level VHDL design into low-level logic primitives.

## **Implementing the Design**

The next stage is to perform place and route of the design. Double click on the Implement Design in the Processes Window (Figure 13).



**Figure 13 Implement Design**

Note the changes in the Design Window. This shows some statistics associated with the implementation process such as how many (and which types) of resources have been used to implement the design. This data is also stored in separate files in the project directory. They are important to help evaluate any errors and warnings that have occurred and to assess the system performance. Advanced users can often improve performance by setting different options in the implementation phase (this is not necessary in this experiment). Make a note of any warnings generated and the effect they may have on your design.

## **Generating the Programming File**

On successful completion of the implementation process, the next step is to generate the programming file which is used to configure the programmable logic on the FPGA device. Double click on the **Generate Programming File** icon and wait for the process to complete. The process will generate a `display.bit` file which will be located in your project directory.



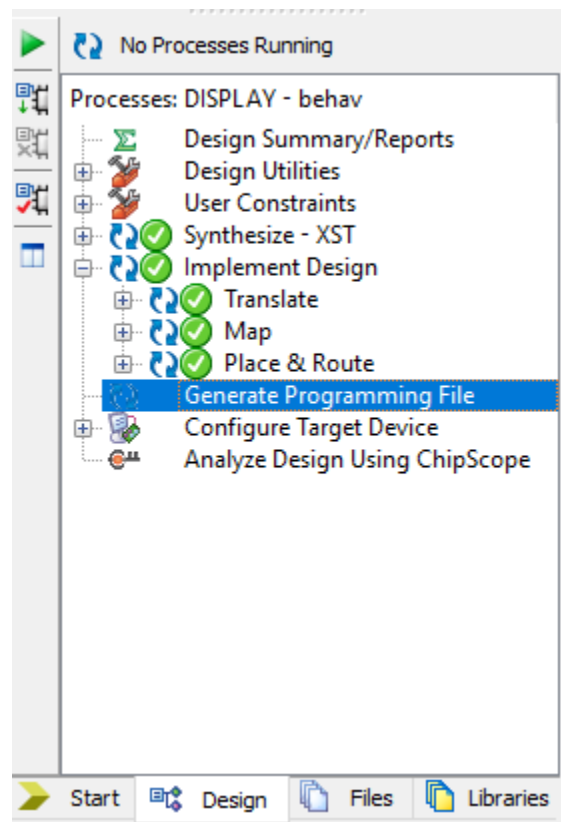


Figure 14 Generate Programme File

## Configure Target Device

Once the programming file has been generated the last phase is to configure the device. This is the last process in the processes window.

For the **Nexys3** board the configuration is directly through the USB cable as shown in Figure 15. If you are unsure ask the demonstrator.

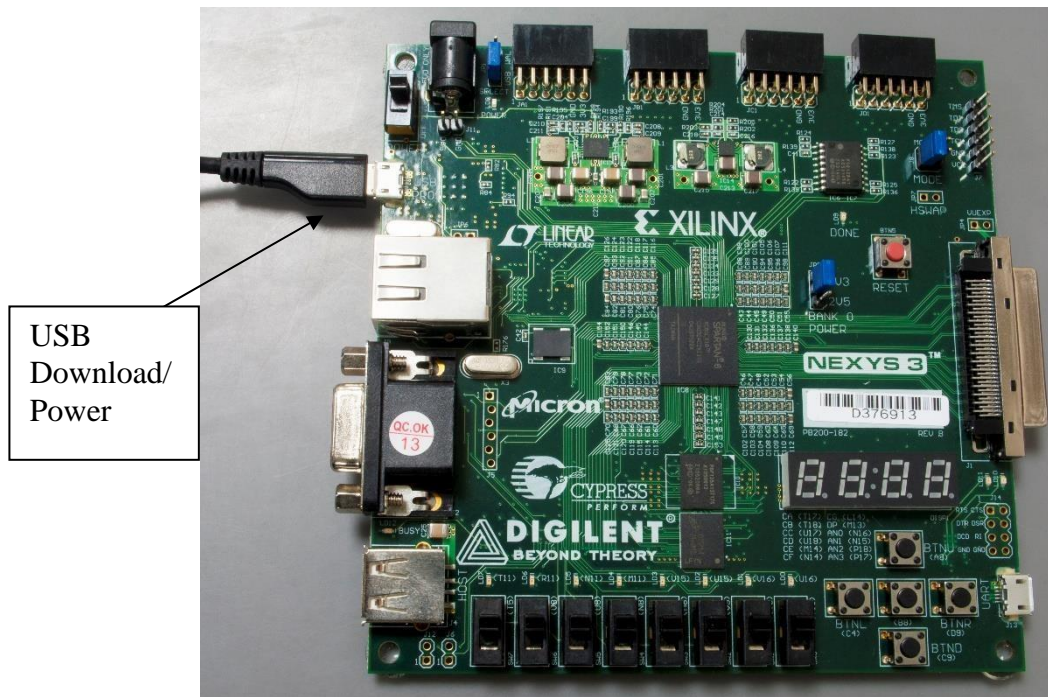


Figure 15 The Nexys3 Target Board

Now double click on the **Configure Target Device** icon in the Navigator Window. The screen similar to that shown in Figure 16 will appear. Click on OK. Sometimes a Dialog Box appears asking if you want to provide feedback to Xilinx. Click **no**. The dialog box in Figure 17 then appears. Double click on boundary scan in the left window. Now right click in the main window and select Initialise Chain. A dialog box shown in Figure 18 asks if you want to assign configuration files to your design. Click on yes and locate the .bit file that you have generated in the previous step. In this case it is the file **display.bit** which is stored in your project directory. Find it and click OK.

After clicking “open”, you will be asked if you wish to program the Flash PROM. Click “No” (Figure 19). Now click OK in the final properties window which will appear.

Move the mouse over the device (which should be highlighted green) and click on the right mouse button. The menu shown in Figure 20 appears. Select the program option. The program download now begins (this may take several seconds). Eventually the response shown in Figure 21 appears to indicate that the programming is complete and successful. If programming is not successful ask the demonstrator for assistance.

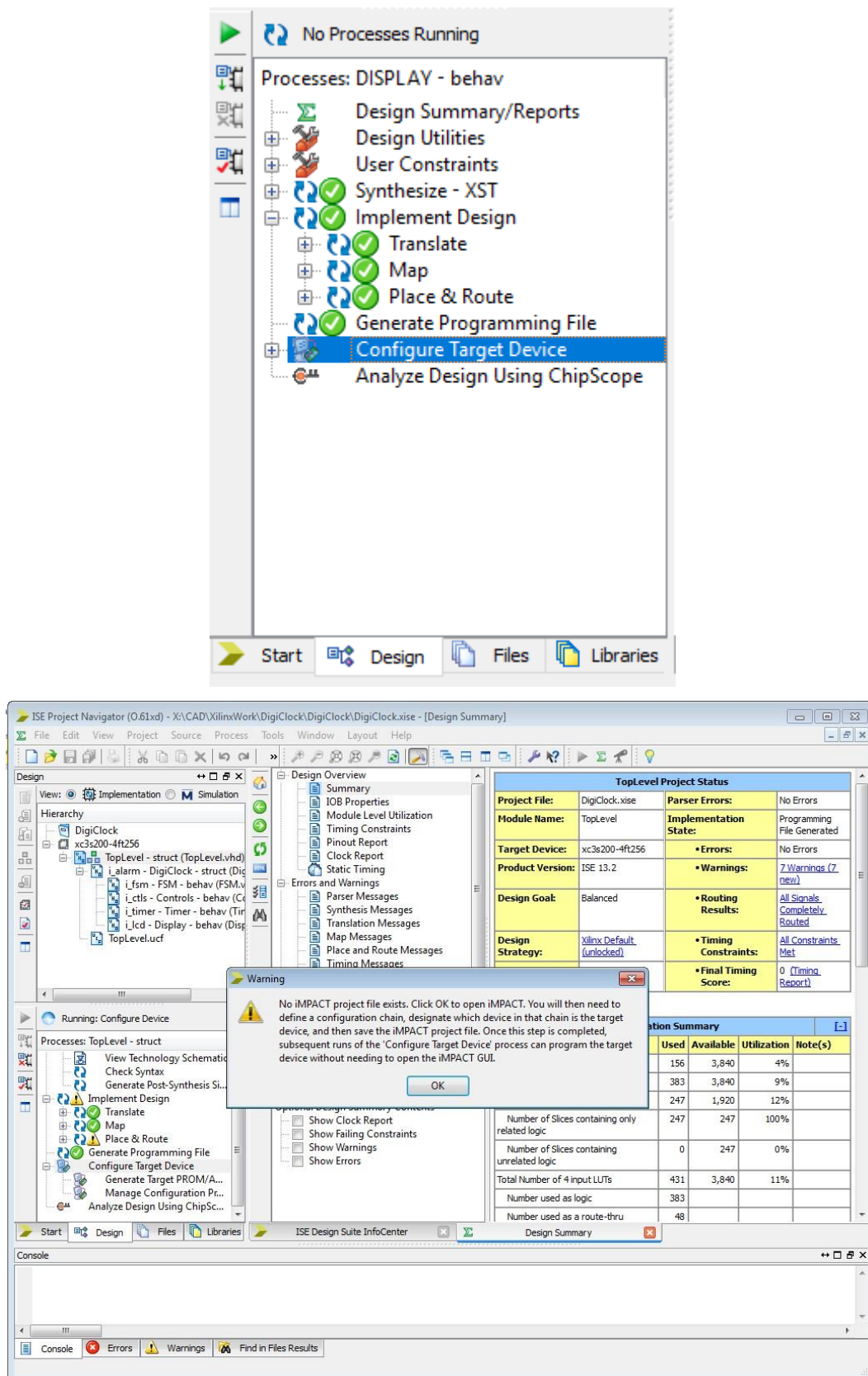
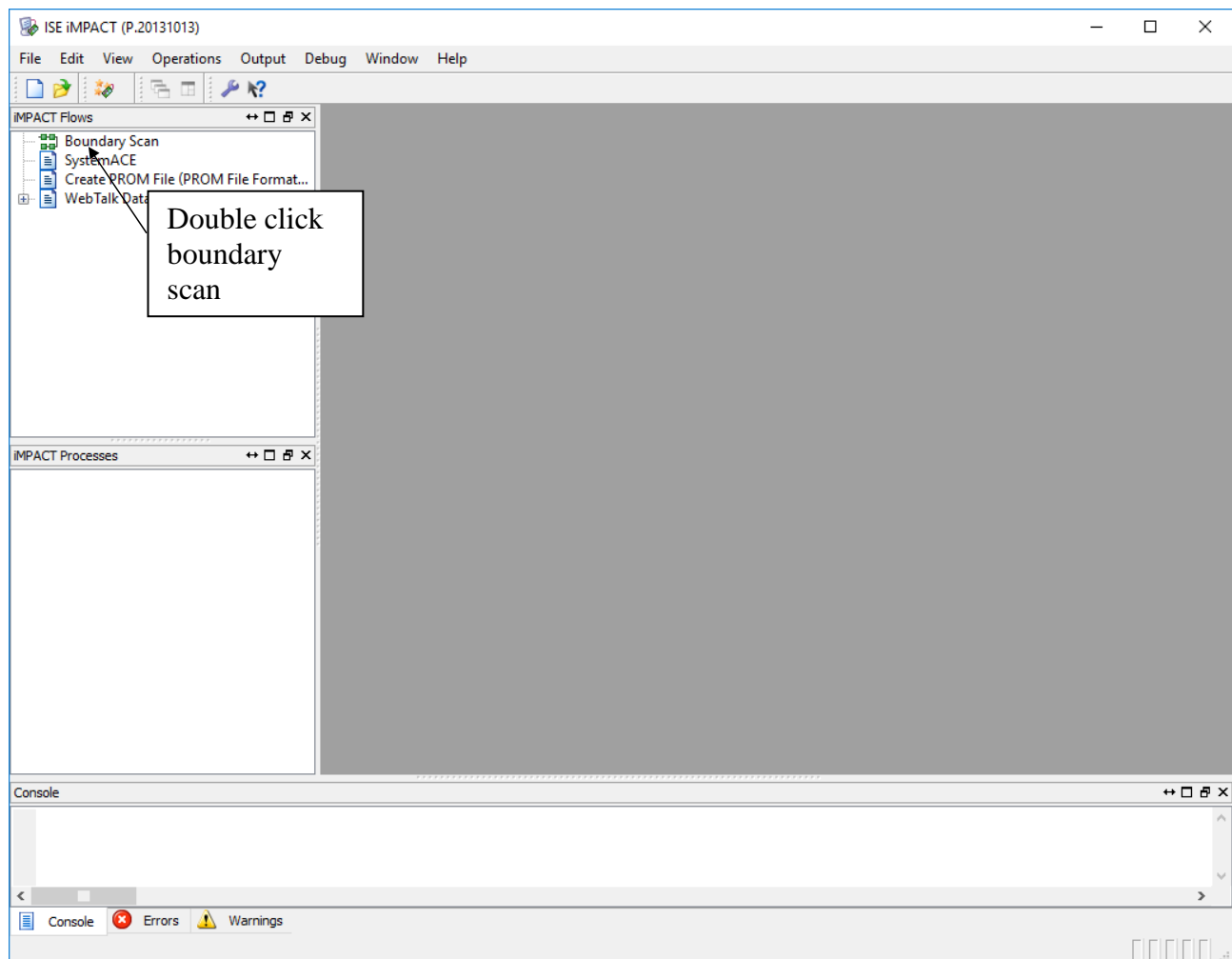
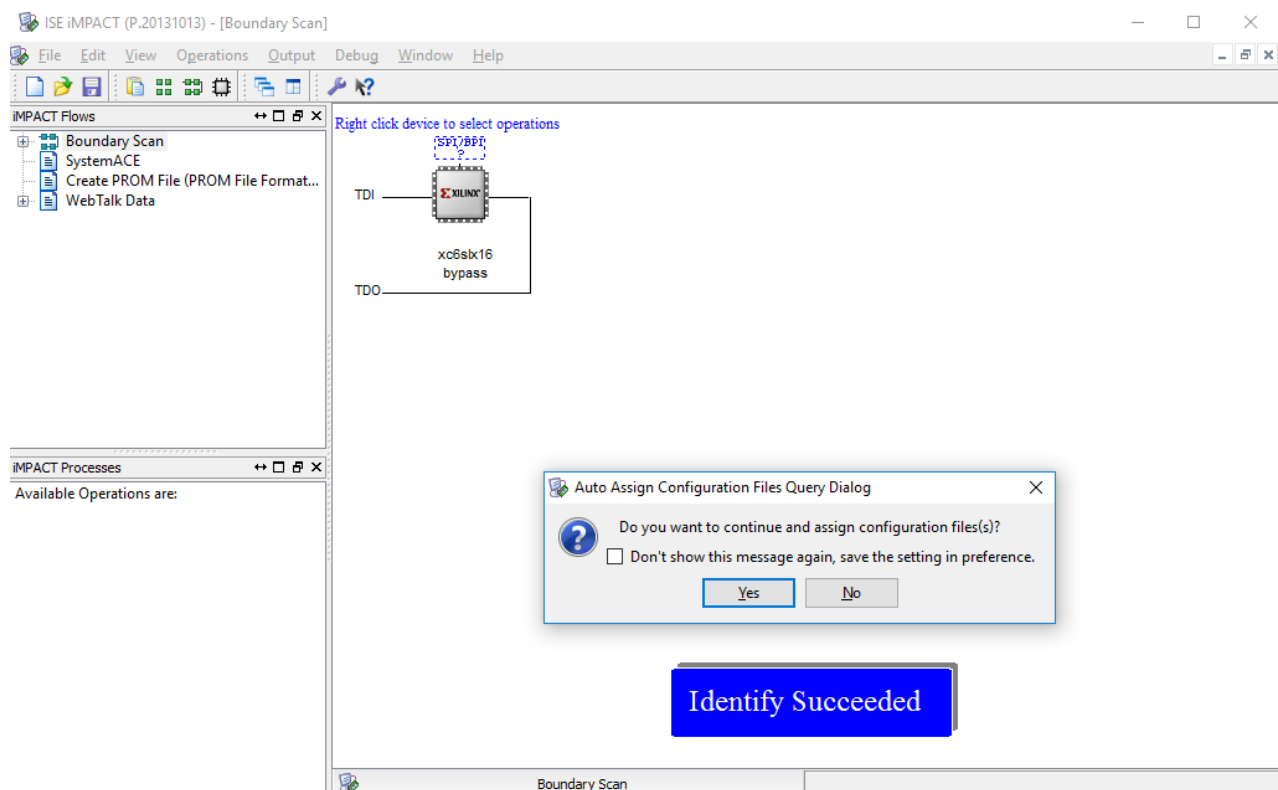


Figure 16 Configure Target Device using the Impact Programme.

**Figure 17 Main iMPACT window**

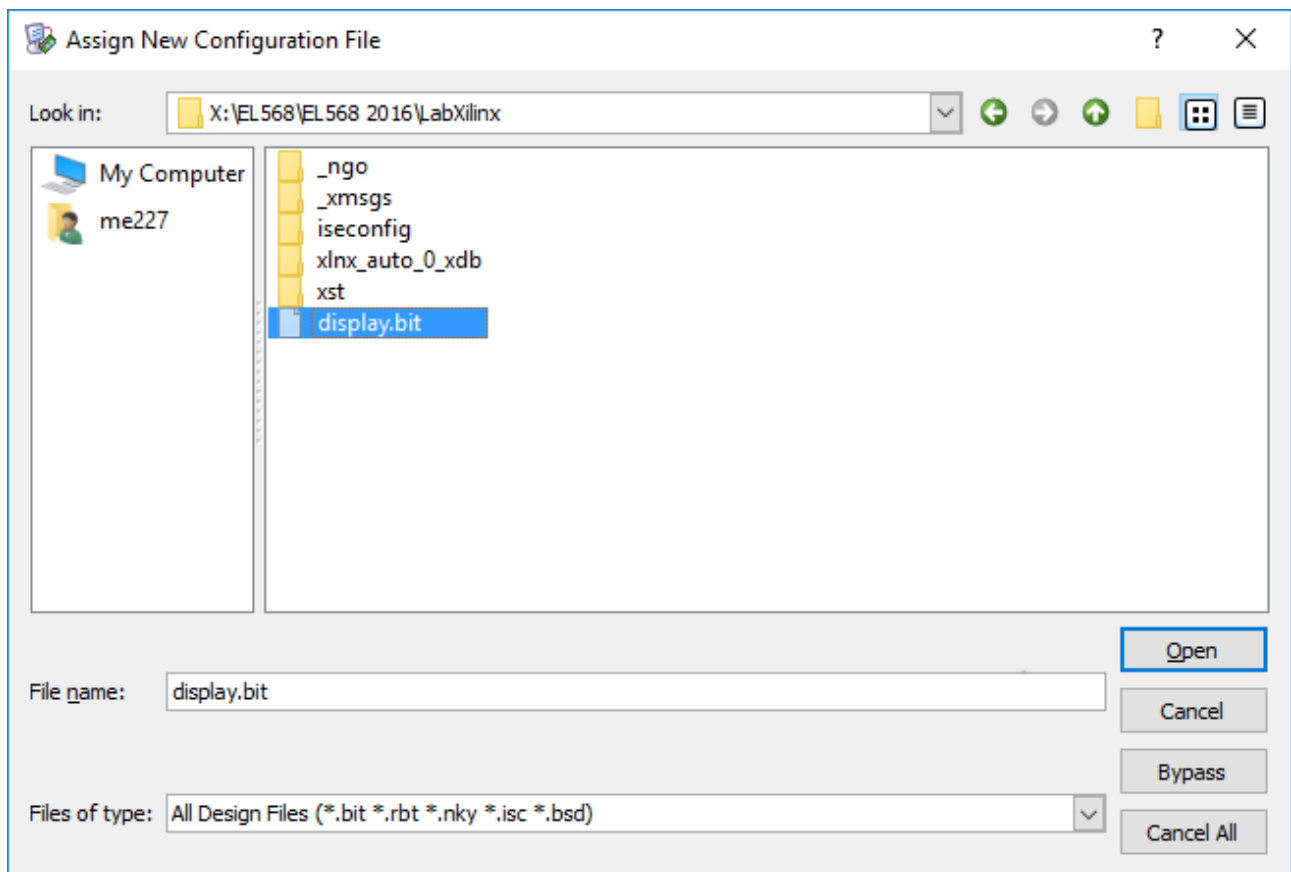


Figure 18 Defining the programme file for the Spartan device

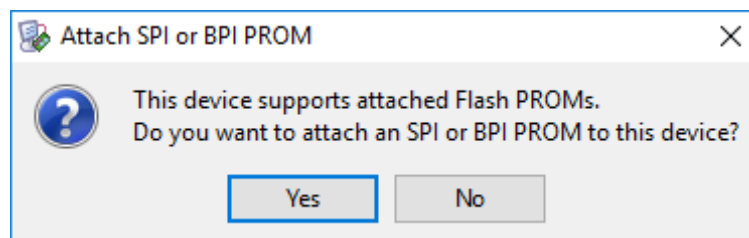


Figure 19 Flash PROM

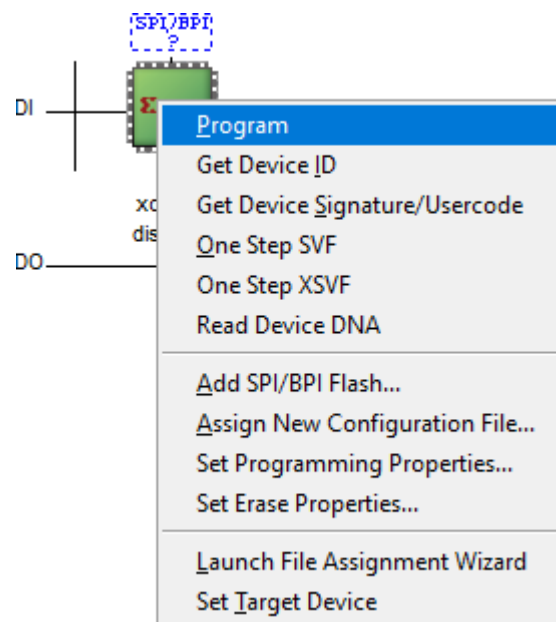
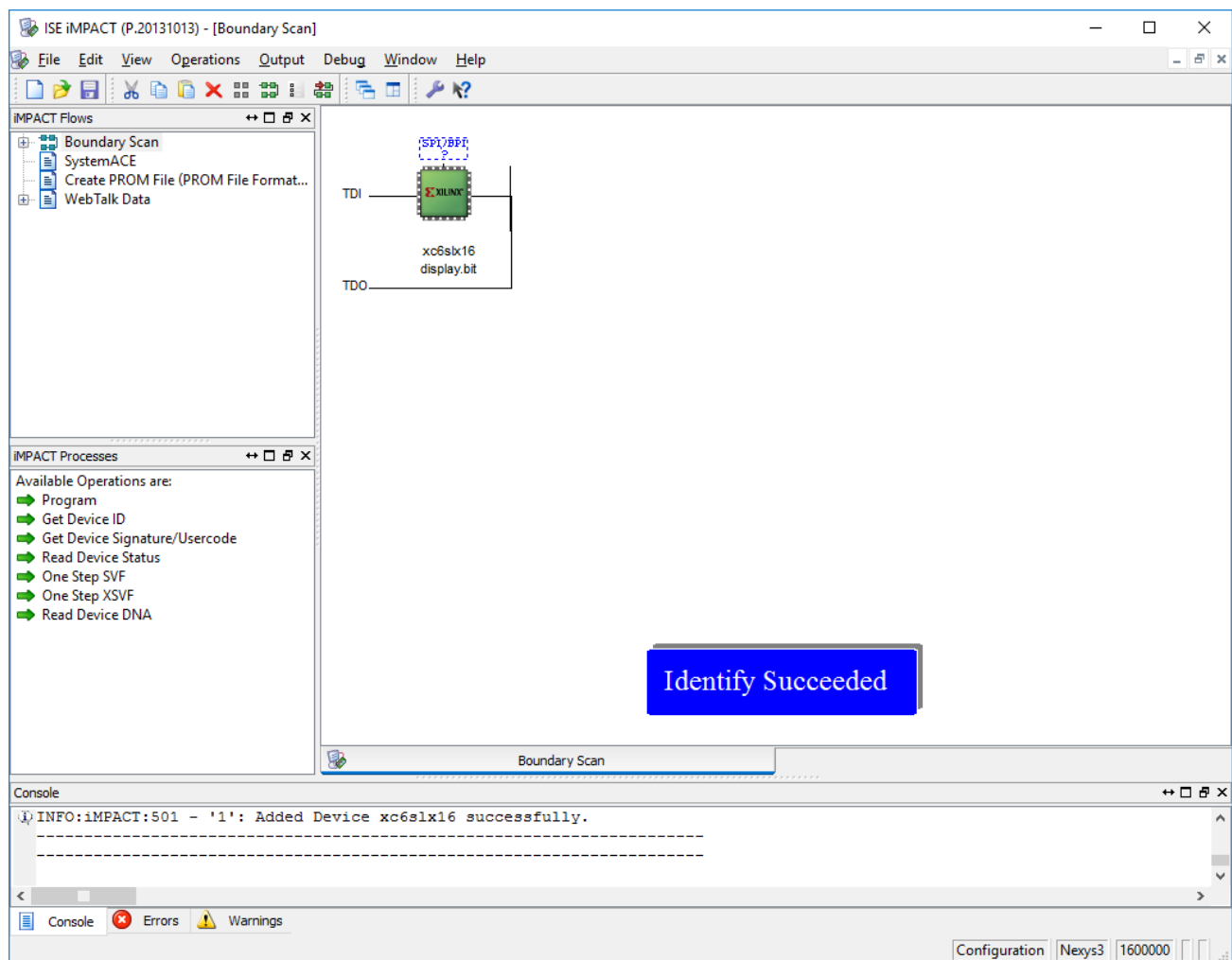
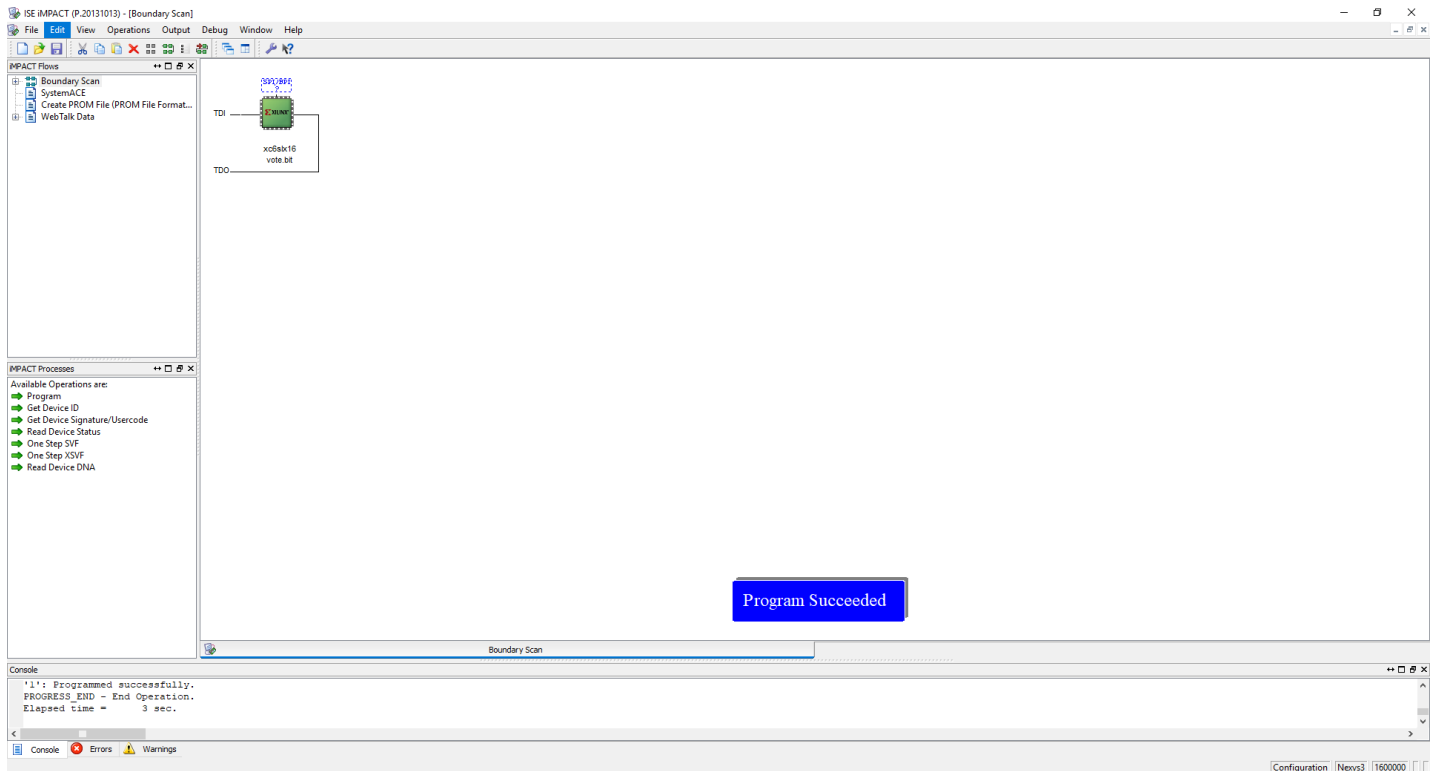


Figure 20 Initiate device programming



**Figure 21 Programming is Successful!**

## Testing the Design

Once the program is finished it is now time to demonstrate that your project works. Demonstrate your design to the demonstrators.