



DALHOUSIE UNIVERSITY

Introduction to Intel Quartus Prime and DE1-SoC Board

ECED 2200 DIGITAL CIRCUITS

Contents

1) Introduction.....	1
a) Objectives	1
b) Required Material	1
2) Quartus Setup.....	1
a) Interface.....	2
b) Design	3
3) Simulating a Design	4
4) Implementation	6
5) Laboratory Exercises.....	7
a) Logic Gates.....	13
6) APENDIX: Common Issues	15
a) Issues with Running the Simulation.....	15
b) Issues with Implementation.....	16

1) Introduction

In this laboratory, an introduction to Quartus will be given followed by a description on how to design and implement basic logic circuits. The final section of this laboratory will focus on creating a 4-bit Binary Coded Decimal (BCD) 7-segment display.

a) Objectives

The objectives of this guide are to familiarize yourself with Intel's DE1-SoC board and be comfortable with its use and operation. Students should be able to:

- Become familiar with the Quartus software
- Learn about "Schematic Entry" design
- Run a simple simulation with pre-defined stimulus
- Implement and generate a 7-segment BCD driver programming file
- Program and test your chip

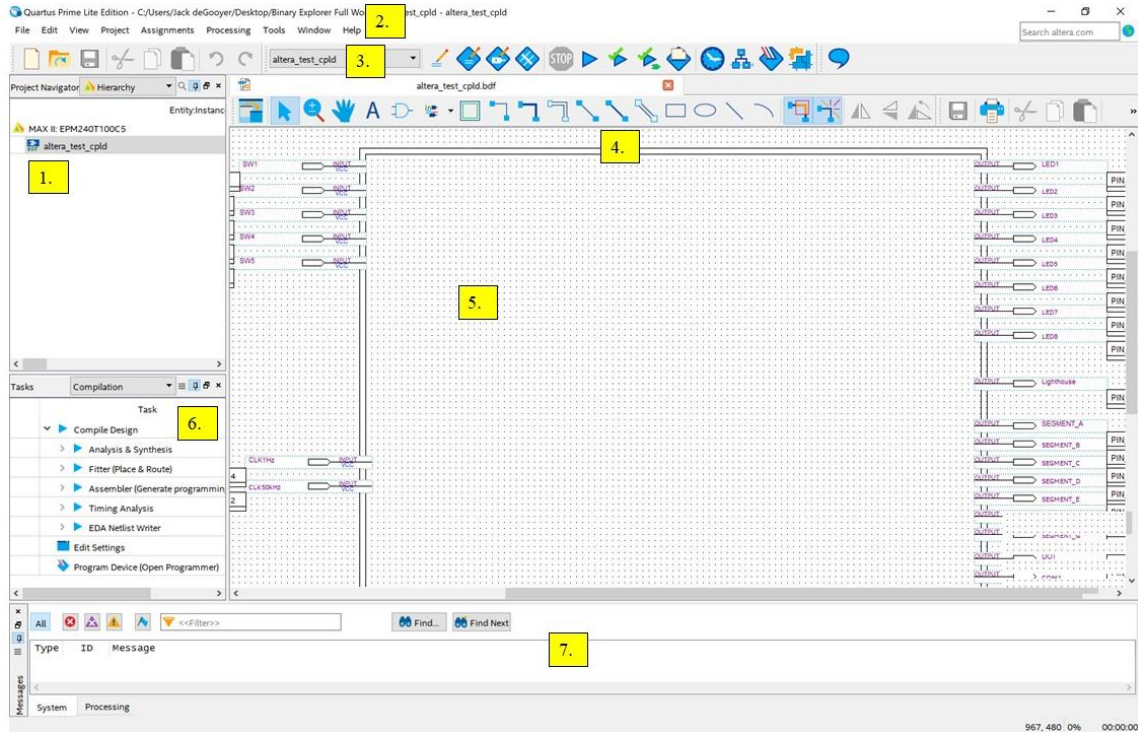
b) Required Material

To conduct the tutorial, students need the following materials.

- Intel DE1-SoC board
- Computer with Intel Quartus Prime Lite (20.1 preferred) installed:
 - All stations in C234 have Quartus installed
 - The software is free and can easily be downloaded at home for use at <http://fpgasoftware.intel.com/?edition=lite>
 - It is recommended to use the download manager when downloading at home. For individual file downloading, choose Quartus Prime, ModelSim and Max II device support file.
- The Quartus project and setting files: **Lab_Template.rar** or **Lab_Template.zip**
 - The design project templates can be downloaded from the course website.
Extract ALL files in the folder to your working directory.
Save a copy of the project templates somewhere for your future labs.
- Start **Quartus**, then **open project** -> select the **Lab_Template.qpf** file in your working directory.

2) Quartus Setup

Intel's Quartus is a powerful tool that is used in industry to program FPGAs. However, due to the introductory nature of this course and tutorial, much of the complex tools and functions of Quartus will not be explored. In this section a brief overview of the interface will be given, and a description on how to design basic logic gates will be shown.



a) Interface

Below is a screen capture of the Quartus software. Please note the numbers on the capture as their description will be below. The screen captures in the tutorial were obtained using the version 18.0. Your version of the software may be slightly different.

1. Project Navigator

In this area, the different files that make up a design will be listed. However, for the purposes of this course, there should only be one file in this folder (altera_test_cpld). Click on this file to open up the schematic for the design. After clicking the file, it should look mostly like the capture. Use the scroll bars on the side of the workspace to center the empty design.

2. Command Bar

At the top of the screen is the standard FILE, EDIT, VIEW bar.

3. Icon Bar

Most common tools can be seen in this bar. The standard tools, such as cut, copy, and paste are on the right. The dropdown bar (which says altera_test_cpld) switches between different schematics. The next icons have to do with pin assignment. This has already been taken care of and do not need to be touched. After the pin assignment buttons are the compilation and synthetization buttons. They are duplicate buttons to those found in section 6. Finally, the last buttons open the programmer and the timing analyser. Once again, these buttons are also found in section 6.

4. Workspace Toolbar

In this bar, all the tools needed to manipulate the workspace can be found. The select tool gives your cursor the ability to select different components. The zoom tool and the hand tool allow for movement on

workspace. The symbol tool allows the user to place parts. For most of the laboratory experiment the “primitive” symbols will be used. The orthogonal node tool allows the user to place a virtual wire between different inputs and outputs. The rest of the tools have various functions but are not needed for the course.

5. Workspace

This area is where most of the work for the binary explorer will be done. With the tools mentioned above, the workspace can be manipulated to create complex circuits. If the area in the picture is not visible, use the scroll bars to center the workspace to the inputs and outputs. On the left of the rectangle is the inputs to the binary explorer. There are five switches which act as inputs. Below the switch inputs are two clock inputs. On the right of the device are the outputs. There are 9 different LED's, 8 for the front LED's and one for the back lighthouse. The other outputs are for the seven-segment display. At the bottom are bi-directional pins that can be used for inputs or outputs.

6. Taskbar

This tab turns the workspace design to a programmable document. By clicking on the “play” button the design will go through the stages to create the programming document. An error would indicate there is an issue with your design. When finished, make sure that there is an EDA Netlist Writer has been completed. If it has not, then press the “play” button adjacent to it. If the design has several warnings, it is alright. This is expected on the designs.

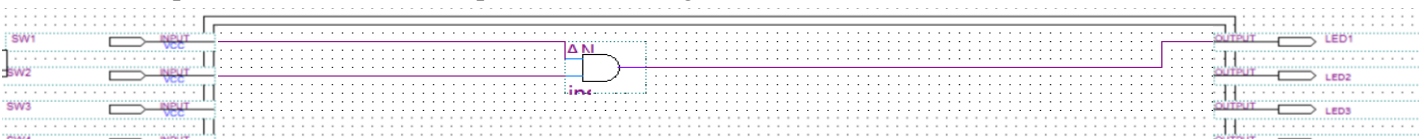
7. Console

The console allows for the user to see the results and notifications of the compilation. If there is any error, the details of the error are provided there.

b) Design

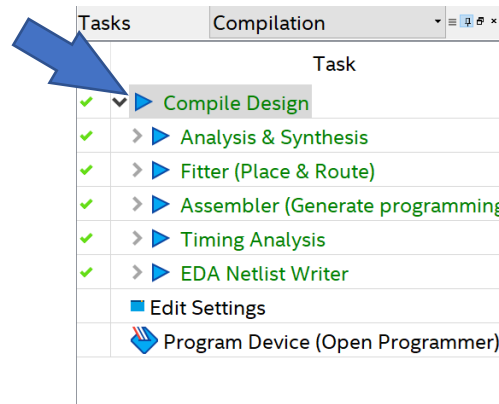
In this section the steps to design a simple logic circuit are explained.

1. To begin, open the **Lab_Template** file and **extract all** the documents into a new folder on your desktop. From this folder open the **Lab_Template** Quartus Project File (.qpf).
2. On the project navigator click on the **Lab_Template** to open the design on the workspace.
3. In the Workspace Toolbar locate the add symbol button. In the menu that pops up click on “**primitives**”, then “**logic**”. From this click on AND2.
4. Place the AND2 gate into the workspace.
5. Using the orthogonal line tool, attach the two inputs of the AND2 gates to separate switches on the inputs. Then, attach the output of the AND2 gate to LED1.



6. Ensure that no wires misconnected or other errors with the design though visual inspection.

- In the Taskbar, press the “play” button to compile the design. Ensure EDA netlist writer completes.

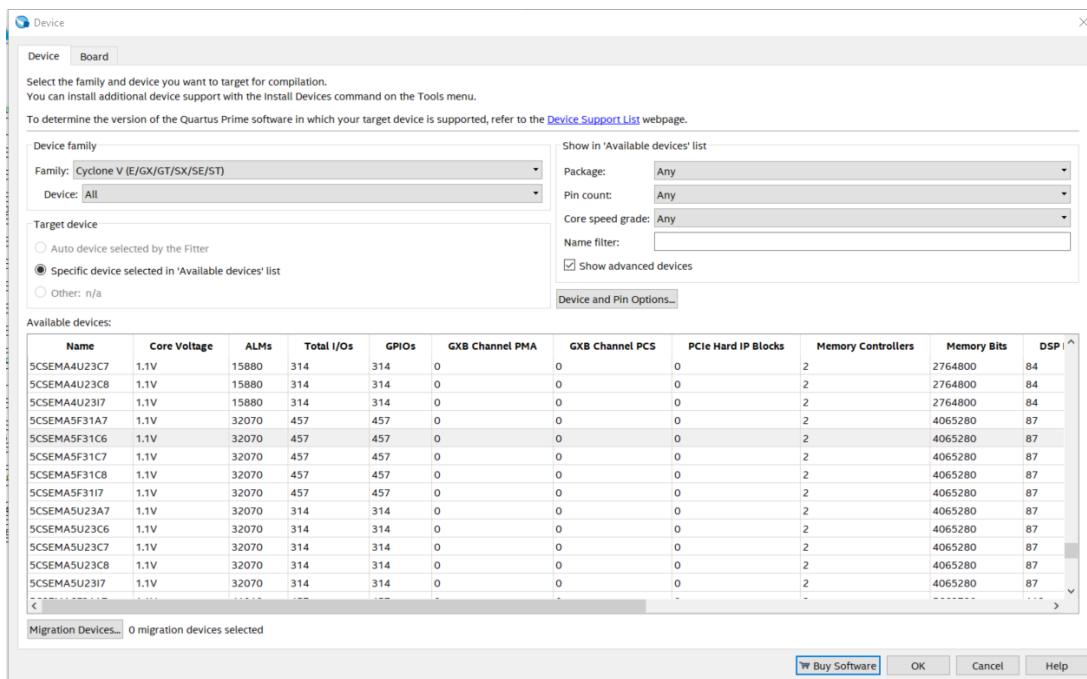


- If any errors arise, carefully check the design and ask a friend for help before the TA. The problem may be obvious to another person.
- Close out of the compilation report.

3) Simulating a Design

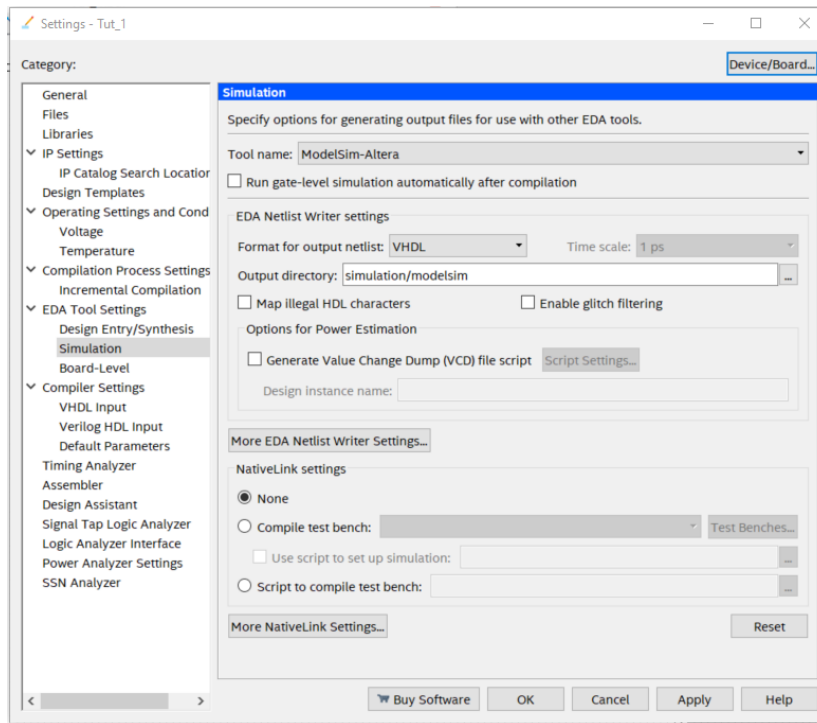
Before Simulating the design, we will first need to set up Quartus to interface with Modelsim (Questasim for versions 21 and above). The provided lab template will have this set up, however this will be useful for creating your own designs in the future.

First let us make sure we are using the right FPGA board on Quartus, go to Assignments > Device



Here you will see many options to select, however all we need to worry about is selecting the Cyclone V under family and in the available devices, find the 5CSEMA5F31C6. Alternatively, you can type this name in the Name filter in the right box.

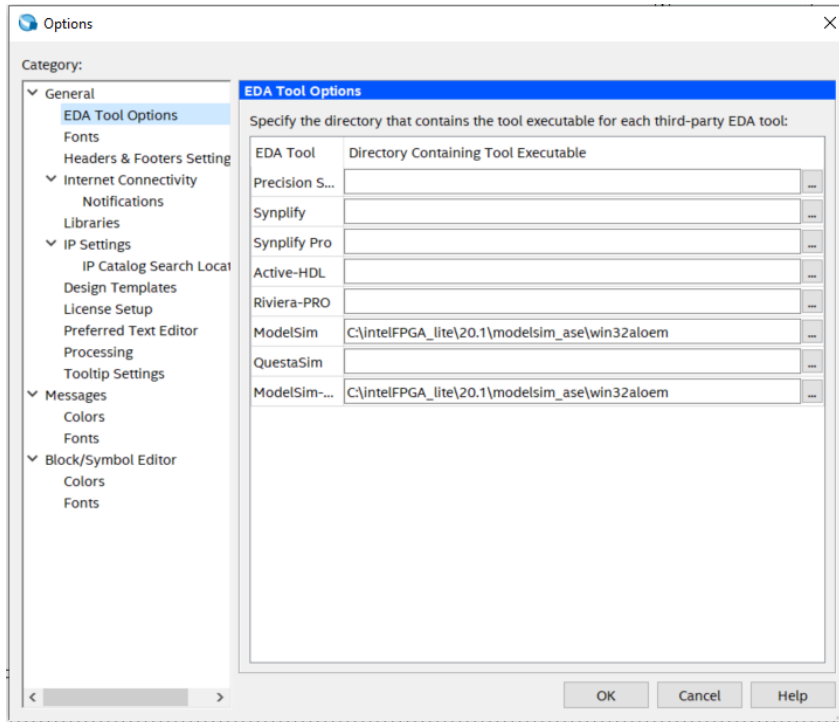
Next, we will need to set up Quartus to interface with Modelsim. To do this go to assignments > settings, in the new window select Simulation under EDA Tool Settings.



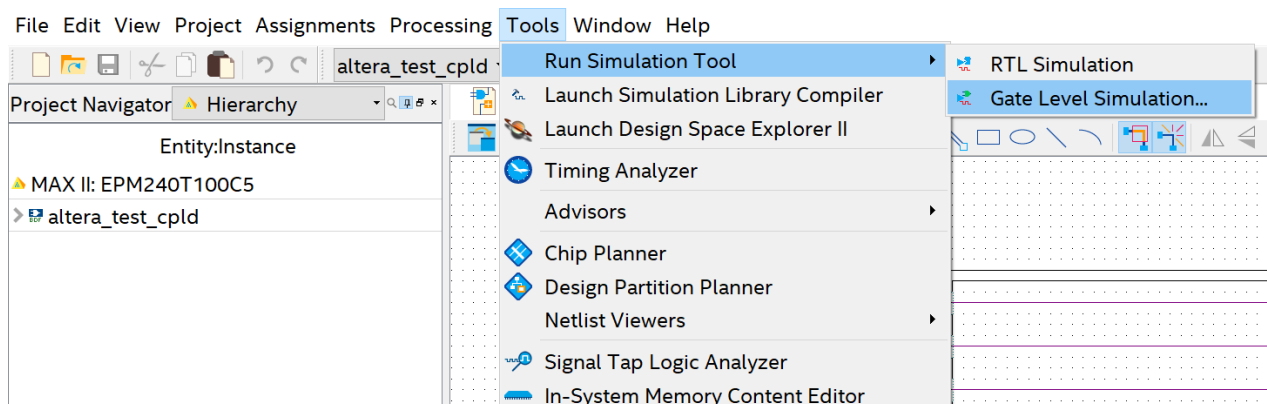
Under Tool name: we want to select Modelsim-Altera, under “Format for output netlist” we select VHDL and finally under Output directory: we type in simulation/modelsim.

Intel’s Quartus II comes with a powerful industry used simulation tool called ModelSim. The simulation process is complex and is daunting at first glance. However, by following simple steps, a proper simulation can be done.

Before you launch ModelSim, go to “Tools” and click “Options”. In this menu click at the top click “EDA Tool Options”. If the file pathway on ModelSim-Altera sections is blank, click on the “...” to the right of the pathway to locate the path. In most cases the folder can be found from the C:\ drive and down the pathway “C:\intelFPGA_lite\20.1\modelsim_ase\win32aloem”. Note: If you install Quartus to a different drive (i.e., D drive) you will need to type in “D:\intelFPGA_lite\20.1\modelsim_ase\win32aloem”.

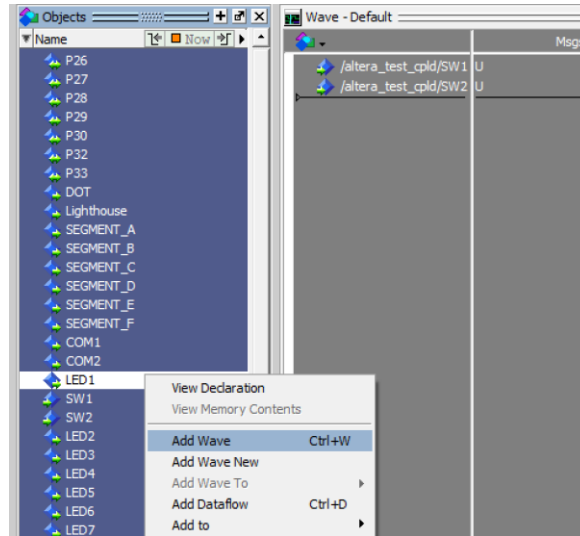


1. Launch ModelSim by pressing the Gate-Level Simulation button in the “**Run Simulation Tool**” menu of the Tool’s Section. After pressing the Gate Level Simulation, a box will pop-up asking for the EDA Gate Level Simulation Timing Model. Select “**Slow Model**” from the drop-down menu and press Run. If a menu appears asking for a simulation language, chose VHDL and continue. Wait for ModelSim to launch. If an error appears, see the Appendix of this document.

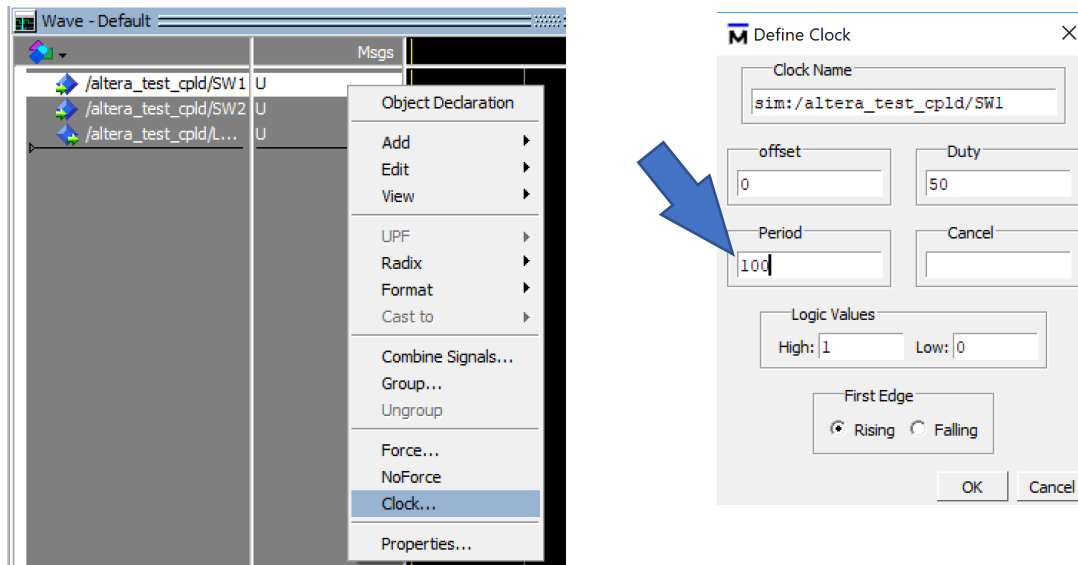


2. ModelSim should now be launched. Locate the “**Library**” window. Locate the “**work**” file (it should be the top file). Double click on it to see its contents and double click “**Lab_Template**”.

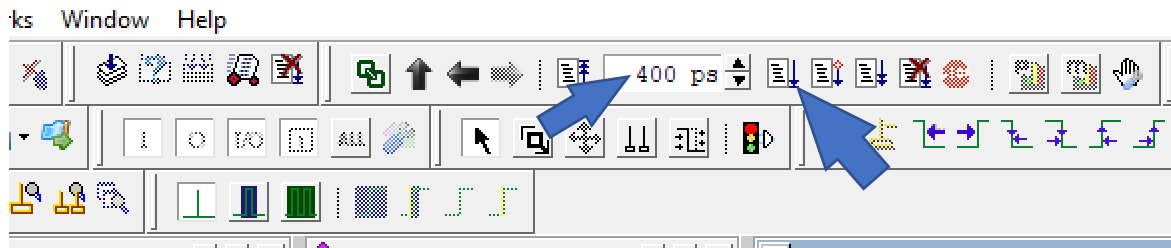
3. In the center column should be an “**Objects**” window. These are all of the signals to which the on-board chip is physically connected. Right click individually on the input and output signals that are being used (SW1, SW2, LED1) and click “**Add Wave**”. In the “**Wave – Default**” window, the selection should appear. You can rename the signals by right clicking on the signals in the “**Wave – Default**”, clicking properties, and changing the display name.



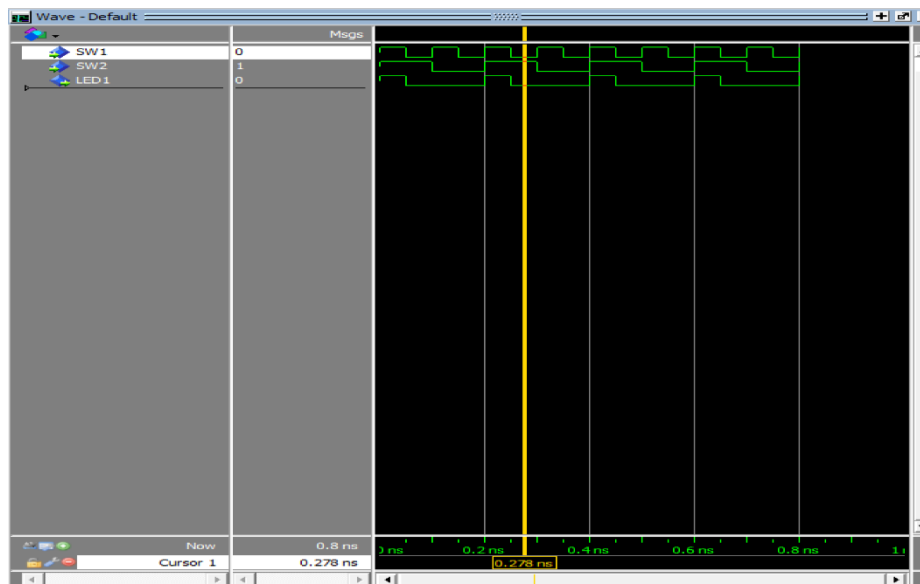
4. The set of waves should now be in the simulation window. Now values must be assigned to them so that a simulation takes places. To do this we make each input a clock cycle with different periods. Right click on the SW1 input and click “**Clock...**”. A box should pop-up. Change the value of the clock period to 100. Repeat for SW2, however change the period to 200.



5. Before we simulate the design, we must ensure that the simulation length is adequate. Change the simulation length to double the longest period. In the current case, change it to 400 ps. After changing the length, press “**Run**”.



6. The yellow line is your cursor. By clicking on different time points in the simulation, the values of the inputs and outputs can be read off. In this simulation, the names of the inputs and outputs were changes to simplify viewing. It is recommended to do so. You can click on the waveforms to see their value in the switches and the corresponding outputs. Play around with the simulation to get an understanding on its use.

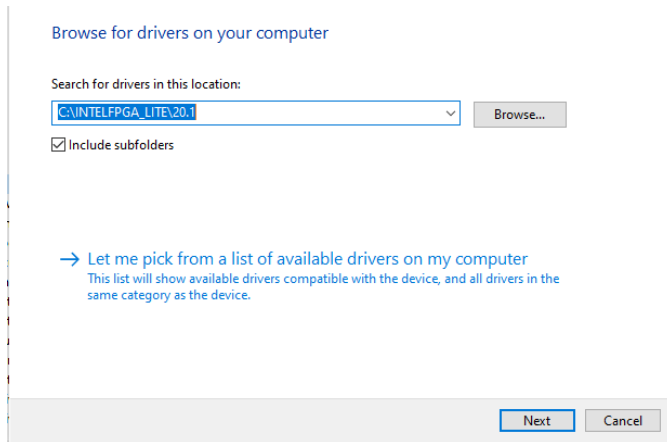


7. When finished, quit the simulation.

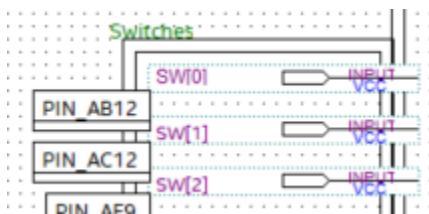
4) Implementation

For any students who install Quartus on their personal computers, we will need to install the JTAG drivers, before we can do that however we will need to disable driver signature enforcement. To do this you will need to follow the instructions in this link: <https://avalonsciences.com/wp-content/uploads/2018/06/Disabling-Driver-Signature-Verification-on-Windows-8-or-10.pdf> Note that you will need to restart your computer to do this. Enforcement will be re-enabled once you restart your computer again, so it is best to install the drivers as soon as it is disabled.

Once disabled, plug in your DE1-SoC board and go to device manager, you will see a device called “unidentified device”, right click and select update drivers > Browse my computer, then enter the path to your Quartus_lite folder. Check “include subfolders” and the drivers will be ready to go



Now we are ready to upload to the board, assuming we use the provided template. If you wanted to create your own project from scratch, you will need to assign the pins in your design file to the physical pins on the board. Using the Lab template as an example we will go through this process:



We see here that switch SW[0] is associated with PIN_AB12, but where did this come from? The pinouts for all inputs and outputs for the DE1-S0C board used can be found here: <http://mems.ece.dal.ca/eced4260/DE1.pdf>

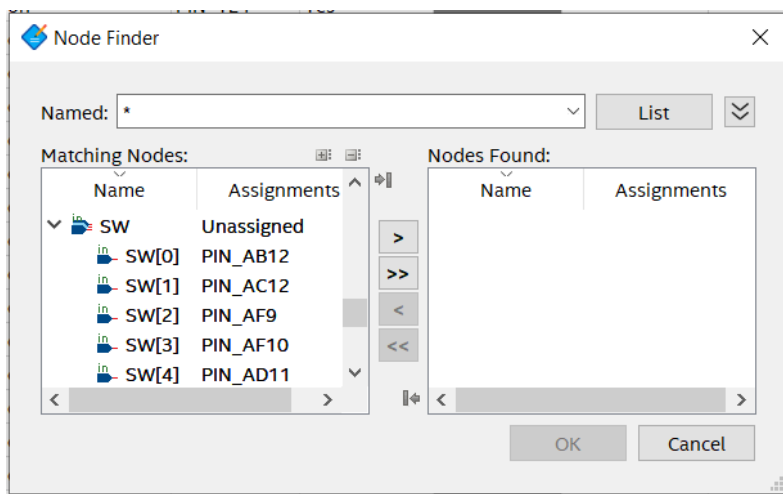
Table 3-6 Pin Assignment of Slide Switches

Signal Name	FPGA Pin No.	Description	I/O Standard
SW[0]	PIN_AB12	Slide Switch[0]	3.3V
SW[1]	PIN_AC12	Slide Switch[1]	3.3V
SW[2]	PIN_AF9	Slide Switch[2]	3.3V
SW[3]	PIN_AF10	Slide Switch[3]	3.3V
SW[4]	PIN_AD11	Slide Switch[4]	3.3V
SW[5]	PIN_AD12	Slide Switch[5]	3.3V
SW[6]	PIN_AE11	Slide Switch[6]	3.3V
SW[7]	PIN_AC9	Slide Switch[7]	3.3V
SW[8]	PIN_AD10	Slide Switch[8]	3.3V
SW[9]	PIN_AE12	Slide Switch[9]	3.3V

In order to assign an input/output to a pin on the board, we will go to assignments > Assignment Editor. You will see an empty table (table is full in the screenshot) with 3 cells that say <<new>>

tatu	From	To	Assignment Name	Value	Enabled	Entity	Comment	Tag
20	✓	out LED[4]	Location	PIN_W1 /	Yes			
21	✓	out LED[5]	Location	PIN_W19	Yes			
22	✓	out LED[6]	Location	PIN_Y19	Yes			
23	✓	out LED[7]	Location	PIN_W20	Yes			
24	✓	out LED[8]	Location	PIN_W21	Yes			
25	✓	out LED[9]	Location	PIN_Y21	Yes			
26	✓	in CLK2_50	Location	PIN_AA16	Yes			
27	✓	out 0_HEX_0	Location	PIN_AE26	Yes			
28	✓	out 1_HEX_0	Location	PIN_AE27	Yes			
29	✓	out 2_HEX_0	Location	PIN_AE28	Yes			
30	✓	out 3_HEX_0	Location	PIN_AG27	Yes			
31	✓	out 4_HEX_0	Location	PIN_AF28	Yes			
32	✓	out 5_HEX_0	Location	PIN_AG28	Yes			
33	✓	out 6_HEX_0	Location	PIN_AH28	Yes			
34	✓	out 0_HEX_1	Location	PIN_AJ29	Yes			
35	✓	out 1_HEX_1	Location	PIN_AH29	Yes			
36	✓	out 2_HEX_1	Location	PIN_AH30	Yes			
37	✓	out 3_HEX_1	Location	PIN_AG30	Yes			
38	✓	out 4_HEX_1	Location	PIN_AF29	Yes			
39	✓	out 5_HEX_1	Location	PIN_AF30	Yes			
40	✓	out 6_HEX_1	Location	PIN_AD27	Yes			
41		<<new>>	<<new>>	<<new>>				

Click the leftmost cell and select the binoculars, in the new window select “list” and then pick the desired input/output as you have named it, in this case SW[0] and hit ok.



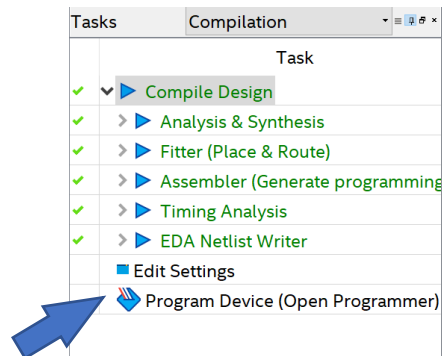
Then, under Assignment name select “Location (accept wildcards and groups)”.

Finally, under value, enter the pin name as defined in the DE1-S0C manual and recompile your project.

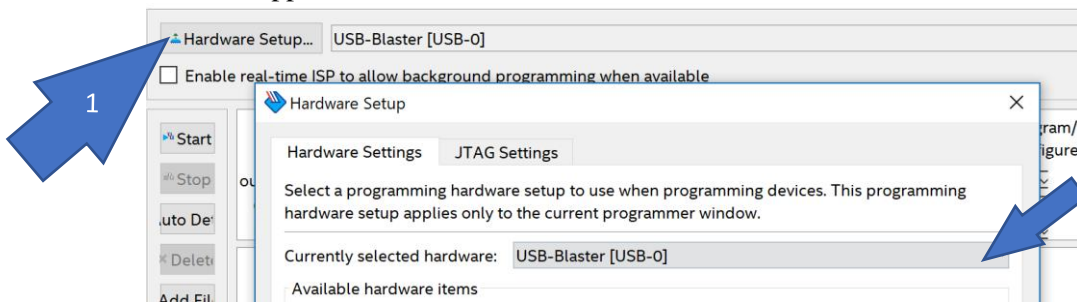
After a design has been made and it has been simulated, it must be transferred to the CPLD for implementation. When the compilation report was done during design, a programming file (.pof) was created for programming the device.

1. Connect the USB from the Binary Explorer to the computer.

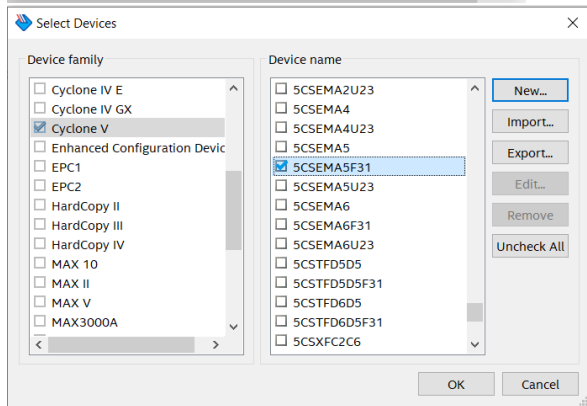
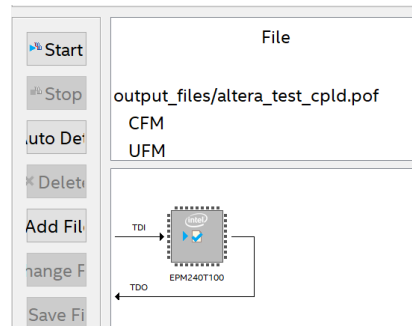
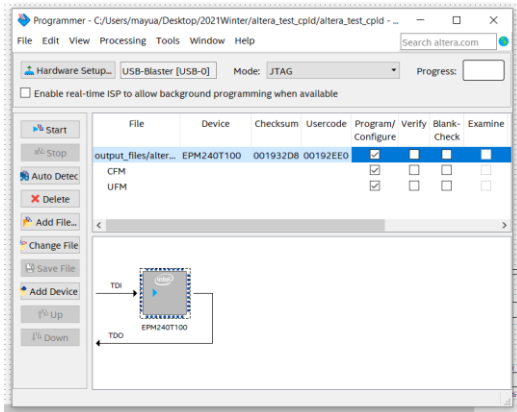
2. In the “Tasks” bar select “**Program Device (Open Programmer)**”



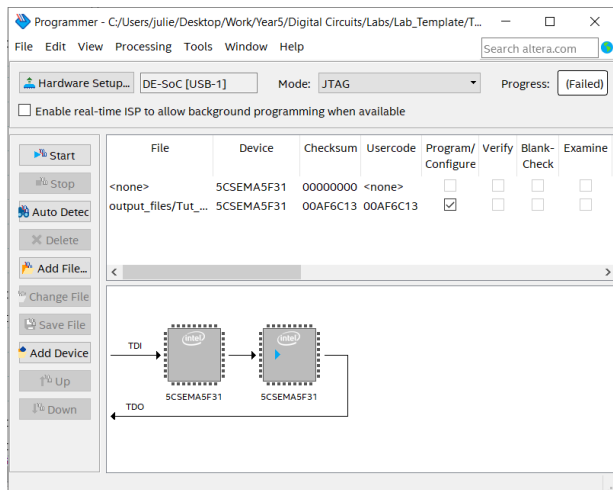
3. Click Hardware Setup and ensure that the selected hardware device is USB-Blaster [USB- a number]. If none exist, see the Appendix for a solution.



4. Select the .pof (or .sof) file, click the Program/Configure column, then. Select add device and select “Cyclone V” in the left box and “5CSEMA5F31” in the right box and select OK.



Then Select the Device Box and select “Up” the device order should look as below:



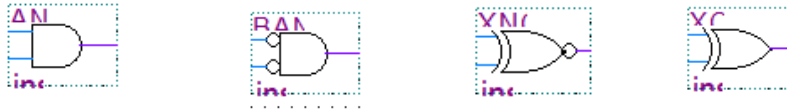
Now press “**Start**” to begin programming. Wait until the LED’s finish being dim and turn off. Keep the cord plugged in during testing. The cord provides power to the device

5) Laboratory Exercises

With an understanding of the design, simulation, and implementation of the binary several exercises can be done. There are two exercises, one requires repeating the AND2 design with other primitive gates and the other is a binary to seven segment display.

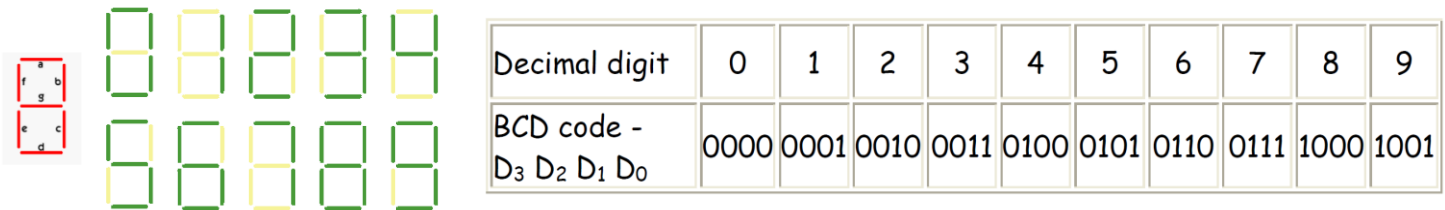
a) Logic Gates

Repeat the steps taken with the AND2 gate with the XNOR, XOR, BAND2 gates and record the results.

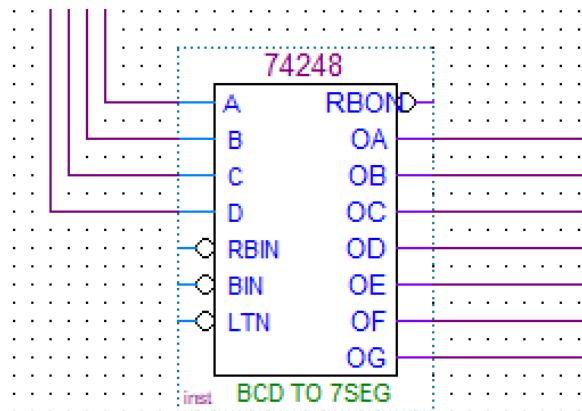


b) Seven Segment Display

The two white square boxes on the top left of the Binary Explorer is a seven-segment display. Each segment of a seven-segment display is a small light-emitting diode (LED), and - as is shown below - a decimal number is indicated by lighting a particular combination of the LED's elements. Binary-coded-decimal (BCD) is a common way of encoding decimal numbers with 4 binary bits:



In the second part of this lab, a test circuit will be constructed to drive the 7-segment display using a 4-bit BCD signal. Press “**Add Symbol**” and chose the “**others**” section instead of the primitives. Then press “**maxplus2**” and pick the **74248** chip. Connect the four switches to A, B, C, D and the outputs to their respective letters as shown. Then compile and implement (no need to simulate) the design and record the results.



Digital Circuits - ECED 2200 Tutorial 1 Observations

Student Names: _____ B00 _____
 _____ B00 _____

AND Gate Truth Table:

A	B	Y
0	0	
0	1	
1	0	
1	1	

XNOR Gate Truth Table:

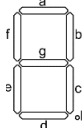
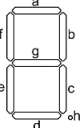
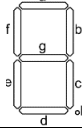
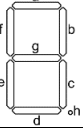
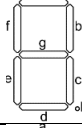
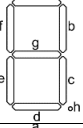
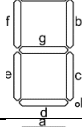
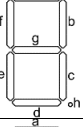
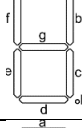
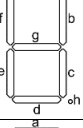
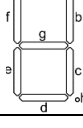
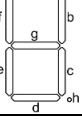
A	B	Y
0	0	
0	1	
1	0	
1	1	

XOR Gate Truth Table:

A	B	Y
0	0	
0	1	
1	0	
1	1	

BAND2 Gate Truth Table:

A	B	Y
0	0	
0	1	
1	0	
1	1	

Switches SW ₄ SW ₃ SW ₂ SW ₁	Display (colour the light-up segments)	Switches SW ₄ SW ₃ SW ₂ SW ₁	Display (colour the light-up segments)
0000		0110	
0001		0111	
0010		1000	
0011		1001	
0100		1010 (what happens @ 10?)	
0101		1011	

6) APENDIX: Common Issues

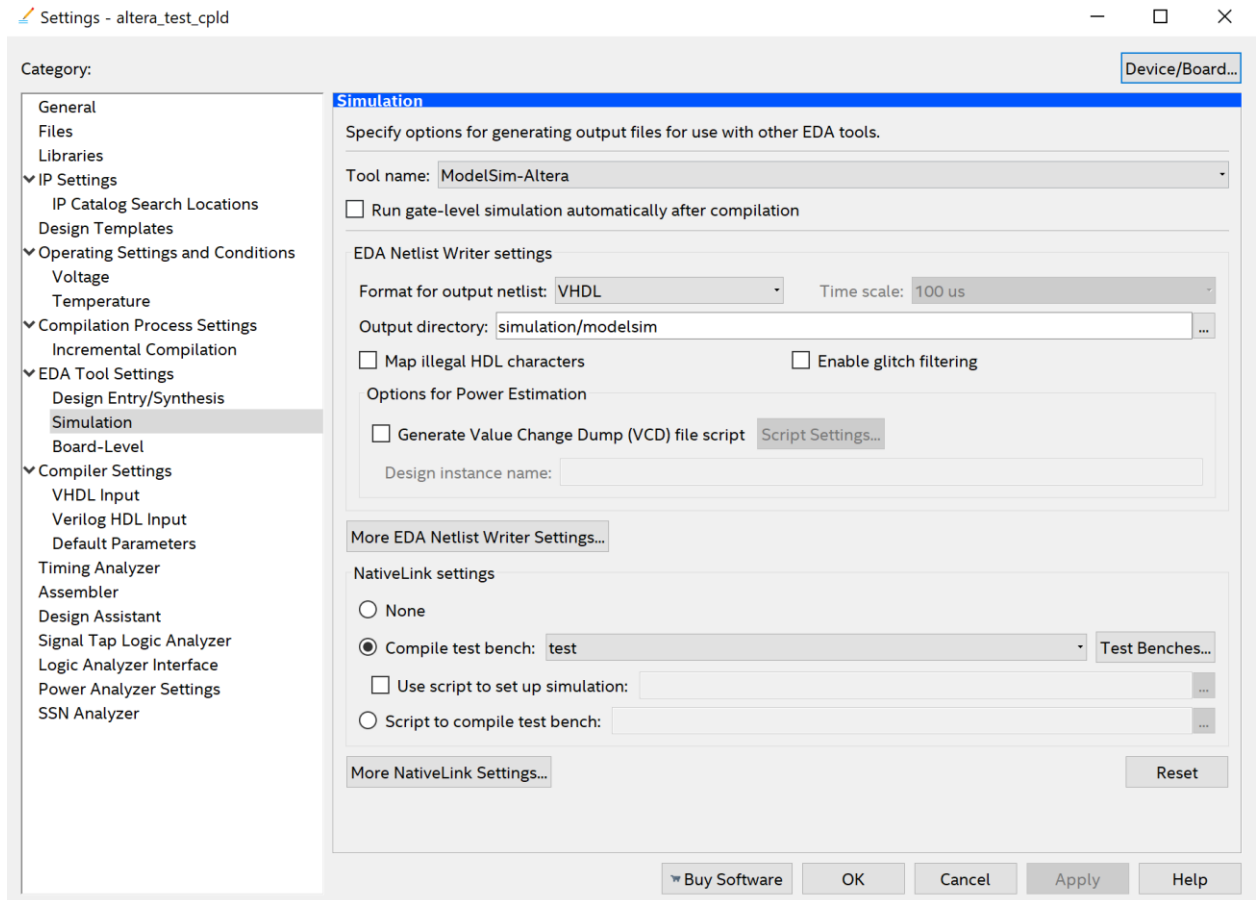
While the Binary Explorer, Quartus, and ModelSim may be a product of countless hours of many teams of people, there are sometimes issues that plague the initial start-up of the device. This section covers the two most prominent issues that occur usually when installing Quartus at home and attempting to program. It follows with a simple Q and A of common issues in the lab.

a) Issues with Running the Simulation

A common issue with the Quartus is the failure of Quartus to link with ModelSim. If this occurs an “Nativelink Error” message will be seen.

To solve this, go to “**Tools**” and click “**Options**”. In this menu click at the top click “**EDA Tool Options**”. There should be a file pathway on either the ModelSim or ModelSim-Altera sections. If both are blank, click on the “...” by ModelSim and double click on the “win32aloem” folder and press “**Select Folder**”. In most cases the folder can be found from the C:\ drive and down the pathway “C:\intelFPGA_lite\18.0\modelsim_ase\win32aloem”. (Note: 20.1 for version 20)

Now, exit out of that window and go to “**Assignments**” then “**Settings**”. In the window that pops up find the “**Simulation**” settings in the “**EDA Tool Settings**” section. Ensure it looks exactly as shown. All sub-menus (such as the “more settings”) are set to their defaults. If the “**Compile test bench:**” is not set to “**test**” then click “**Test Benches...**” and create a bench called “**test**” with default settings.



After these changes, the simulation should work. Ensure the “**Gate Level Simulation**” is pressed and not the one above it.

b) Issues with Implementation

During initial start-up of the program at home or in the laboratories, Quartus may have some trouble discovering the Binary Explorer during implementation. The most likely issue (after checking to see if the device is plugged in) is to check which drivers are installed on the device. A driver allows a computer to communicate with an external peripheral, like the binary explorer.

Below is the full solution from the Intel Website. The solution is designed for windows platforms.

1. Plug the USB-Blaster download cable into your PC. The **Found New Hardware** dialog box appears.
 - o Note from Dalhousie: If no box appears or there is trouble with this solution, search “**run**” in your windows search bar in the bottom right corner. Run the program “**devmgmt.msc**”. With the device manager opened, go to the bottom and open the “**Universal Serial Bus controllers**” section. Then right click on “**Altera USB Blaster**” and click “**Update Driver**”. Then click “**Browse my Computer for Driver Software**” and skip to step 6.
2. Select **Locate and install driver software (recommended)**.
3. Select **Don't search online**.
4. When you are prompted to **Insert the disc that came with your USB-Blaster**, select **I don't have the disc. Show me other options**.
5. Select **Browse my computer for driver software (advanced)** when you see the **Windows couldn't find driver software for your device** dialog box.
6. Click **Browse**, and browse to the **C:\intelFPGA_lite\18.0\quartus\drivers\usb-blaster** directory.
 - o Note: Do not select the x32 or x64 directories. (Note: **20.1** for version 20)
7. Click **OK**.
8. Select the **Include subfolders** option, and click **Next**.
9. If you are prompted **Windows can't verify the publisher of this driver software**, select **Install this driver software anyway** in the **Window Security** dialog box. The installation wizard guides you through the installation process.
10. When **The software for this device has been successfully installed** dialog box appears, click **Close**.
11. To complete your installation, set up programming hardware in the Quartus Prime software.

The following solution is the “Set up Programing Hardware in the Quartus Prime Software”

1. Start the Quartus® II software.
2. Choose **Programmer** from the Tools menu. The Programmer window will open.
3. Click the **Hardware Setup...** button to open the Hardware Setup window.
4. The selected programming hardware is identified as Currently Selected Hardware.
5. Programming hardware that is already set up appears in the Available hardware items window.
6. Click the **Add Hardware** button to open the Add Hardware window if the programming hardware you would like to use is not listed in the Available hardware items window.
7. Select the appropriate programming cable or programming hardware from the Hardware Type list.
8. Select the appropriate port and baud rate if necessary.
9. Click **OK**.
10. Select the programming hardware you would like to use by choosing it in the Available hardware items list.
11. Click **Close**.
12. Your programming hardware has been set up.