INTRODUCTION TO QUARTUS

August 30th, 2007
CSC343
Fall 2007
Prepared by: Steven Medina

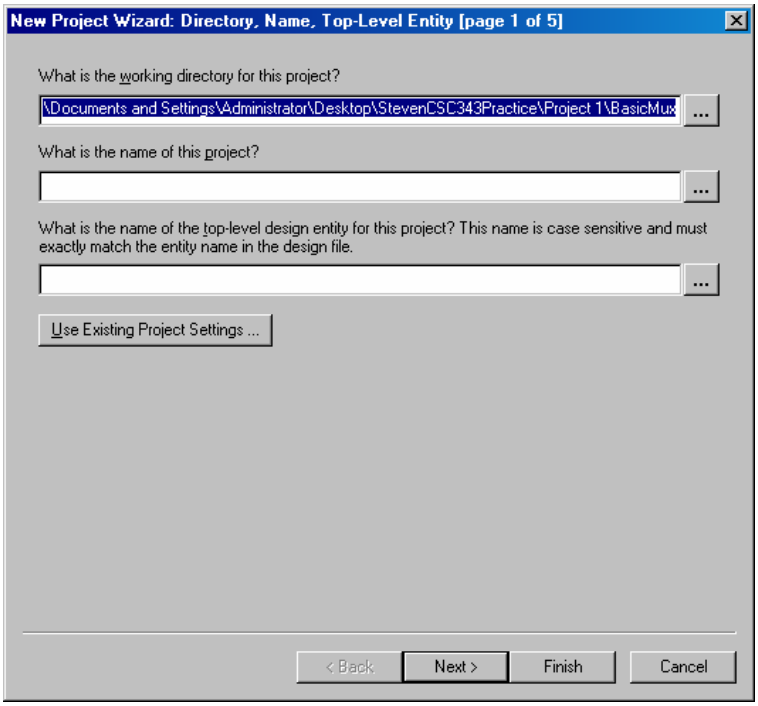
PURPOSE

The purpose of this lab is to learn how to use Quartus and Modelsim software. Most, if not all of you have never used these software packages. This lab is a step-by-step guide on how to create a Quartus project file (qpf) and simulate your design in Modelsim. You will be creating a simple 2-to-1 multiplexer to do this. Below, I briefly explain what a multiplexer is. However, you can google multiplexers if you need further explanation. To those of you who are computer engineering majors, your textbook from EE210 (switching systems) will be very helpful to your labs here.

Quartus is an important tool in developing circuits. Before an engineer builds a building, he has an architect design it. The design then may be simulated in special software to see if it can me made structurally sound. This is what Quartus does for us. We use this software to design different kinds of digital circuits BEFORE we implement them. Later on in the semester, you will be testing your designs on a programmable chip. However, it would be pointless to put your design on the chip if your simulation does not work properly on Quartus.

CREATING A NEW QUARTUS PROJECT FILE

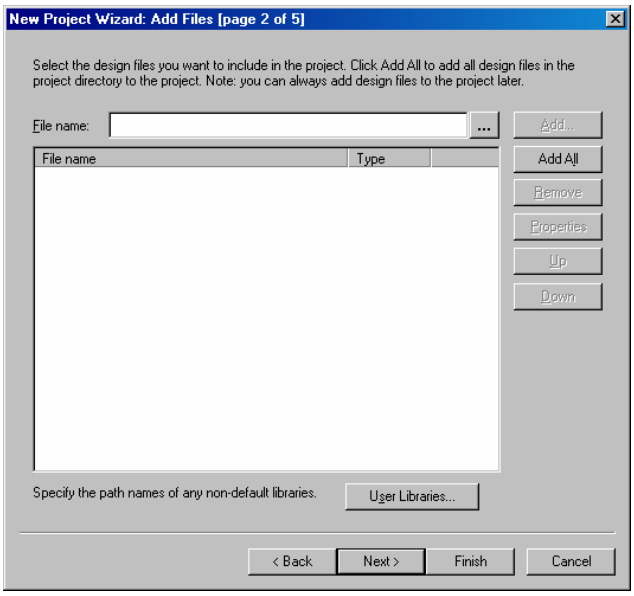
1) Right click the desktop and click “terminal. Type “Quartus” in the box that appears to open the Quartus software. Click on file=>new project wizard. The following box will appear.



You will see three boxes that you can type in. The top box will ask for the directory that you wish to save your project in. You can click on the small box to the right and save it to a directory you feel comfortable with.

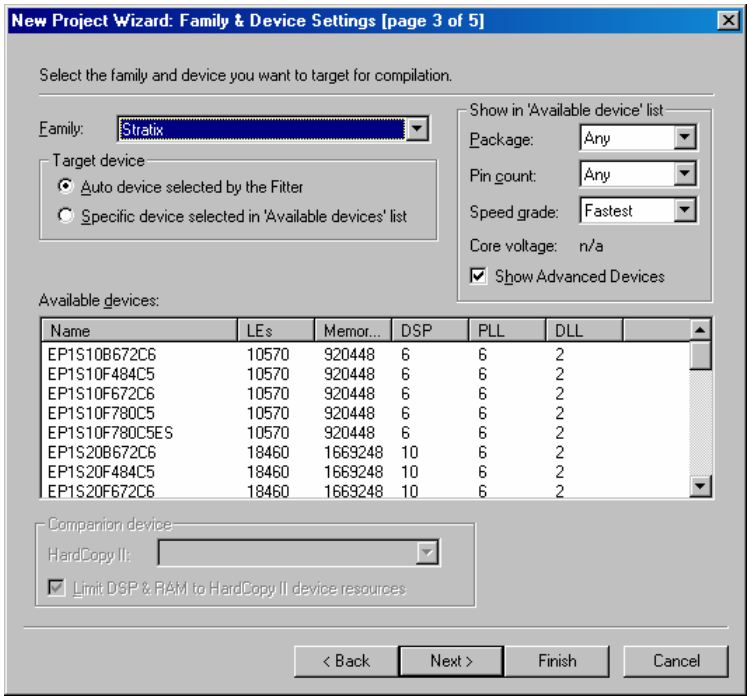
2) After doing this, click on the middle text box. This is where you name your project. I recommend calling it “Basic_Mux”. You will notice while typing, the name of the project will also be typed in the third box as well as the second. Leave it like this for simplicity for now. Click “Next” on the lower right corner.

3) You will now see the following menu.



This menu screen will eventually become important. For now, we will ignore it.

4) Click on “Next” to see the menu shown below.

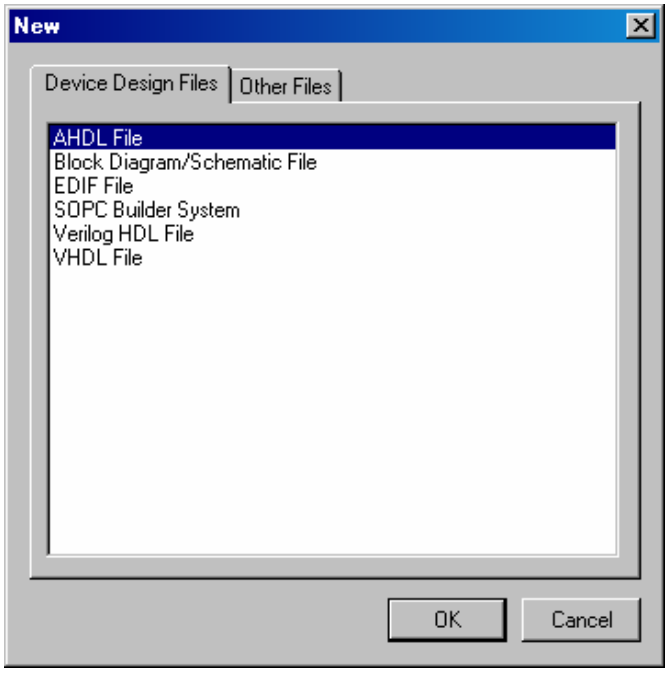


This is the menu where you select the programmable board you will use to test your designs. In the “Family” drop-down menu, chose “Cyclone II”. Under “Available Devices”, chose EP2C35F672C6. That is the name of the board we will be using for this lab. This will also be known as the DE2 board (it is much easier to say than the EP2C35F672C6 board). After selecting the proper board, click “Finish”. You have just created a new Quartus Project File.

FIRST BLOCK DIAGRAM - BASIC MULTIPLEXER

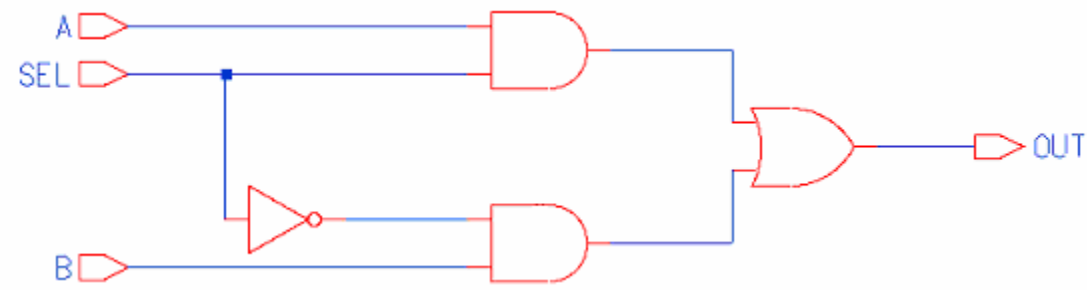
A multiplexer is a circuit design that takes 2 or more inputs, and has only one output. Its purpose is to select which one of those inputs will be driven through to the output. There is one more input called a selector. It *selects* which input will be driven to the output. The selector is at least N bits long for 2^N inputs. This means that if there are 2 inputs, the selector is 1-bit long, since there are 2^1 inputs, with N=1. If there are 8 inputs (2^3 inputs), the selector is 3-bits long. For now, we will be dealing with a multiplexer that has 2 inputs, making the selector 1-bit long. As you read on, you will learn how to create a simple one.

1) To open a blank block diagram file, click on file=>new. The following menu will appear.

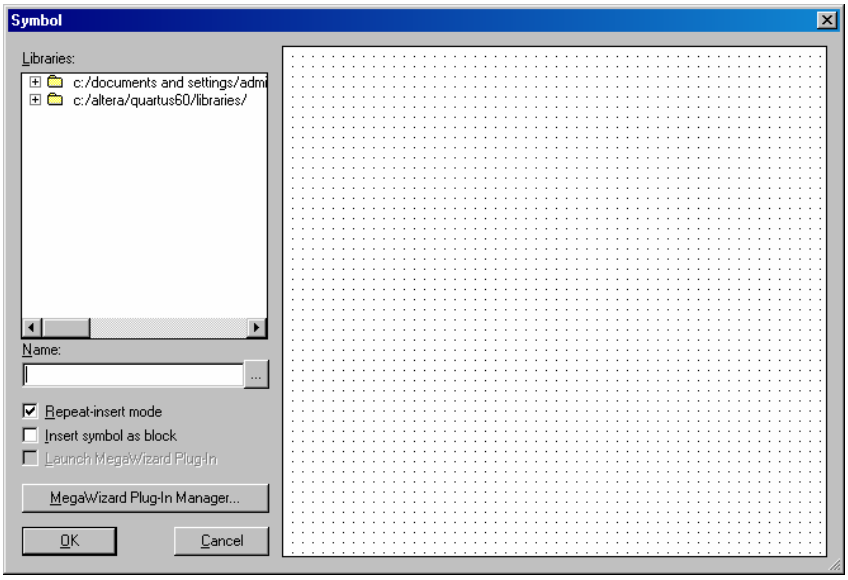


Click on "Block Diagram/Schematic File" under the "Device Design Files" tab. A large white grid will appear. This will serve as a canvas for your work. Any block diagrams you create will be on this grid.

2) Now that you have a new block diagram, we will create a basic 2-to-1 multiplexer. This multiplexer will consist of two AND gates, an OR gate, and a few inverters. A schematic of a 2-to-1 multiplexer is shown below.



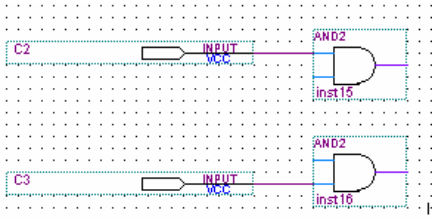
3) To the left of our blank grid, you will see a long vertical toolbar with many symbols to click on. Click on the third one from the top. It looks like an AND gate, and if you leave your cursor on it without clicking, it will say "Symbol Tool". After clicking on this, you will see a box come up like the one shown below.



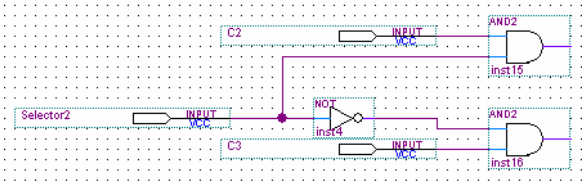
The folders on the left contain circuits and gates we will use for this design, as well as every other design for all other projects. Although these will not be the *only* design blocks we will be using, they will be used very frequently.

4) From the two folders to the left, click on the plus sign to the left of the bottom folder. It will drop down 3 more folders. Of those three folders, click on "primitives". This will drop down drop down 5 more folders. Next, click on "logic". This will drop down many simple circuits and gates for us to choose from. We will start by clicking on "and2". This is a two-input, one-output and gate. After clicking on this gate, click "ok". You will see that you can now place this gate wherever you want on the grid. Click anywhere on the grid to place a gate down. You can place as many of this gate you want on the grid until you right click and hit cancel. Try to place the gates in such a way that it will look like the schematic diagram of the multiplexer shown on the beginning of page 4.

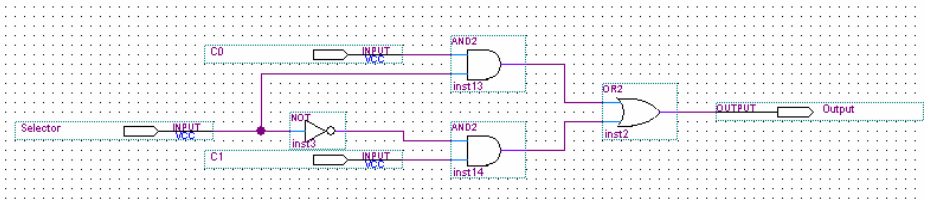
5) Next, click on the "symbol tool" box again. Under the "primitives" folder, click on "pin". You will see three choices of pins. Click on "input" and press ok. Place one input pin to the left of each AND gate. To connect the input pins, first click on the box that says "Orthogonal Node Tool", which is usually two boxes under "Symbol Tool". Place the cursor over the right side of the pin. You will know the cursor is in the right place when it turns into a + symbol. Click and drag the cursor to either input of the AND gate and release the mouse button. You should now have something that looks like the picture below.



6) Now, in the “logic” folder in the “symbol tool” menu, click on the NOT gate. Place the NOT gate somewhere between the two input pins. Then, place an input pin going into the not gate. Connect two more wires to look like the image below.



7) Next, click on the “logic” folder again in the symbol tool menu. Click on the gate labeled “or2” and click ok. Place this gate to the right of the two AND gates. Connect the output of the AND gates to the inputs of the OR gate. Your design should look similar to the one shown below. Place an input pin to the right of your OR gate. Your design is now complete! It should look similar to the image below.



It would be wise to change the name of all the input and output pins. I recommend calling the input pins into the AND gates “C0” and “C1”, the input pin into the inverter “Selector”, and the output pin “Output”.

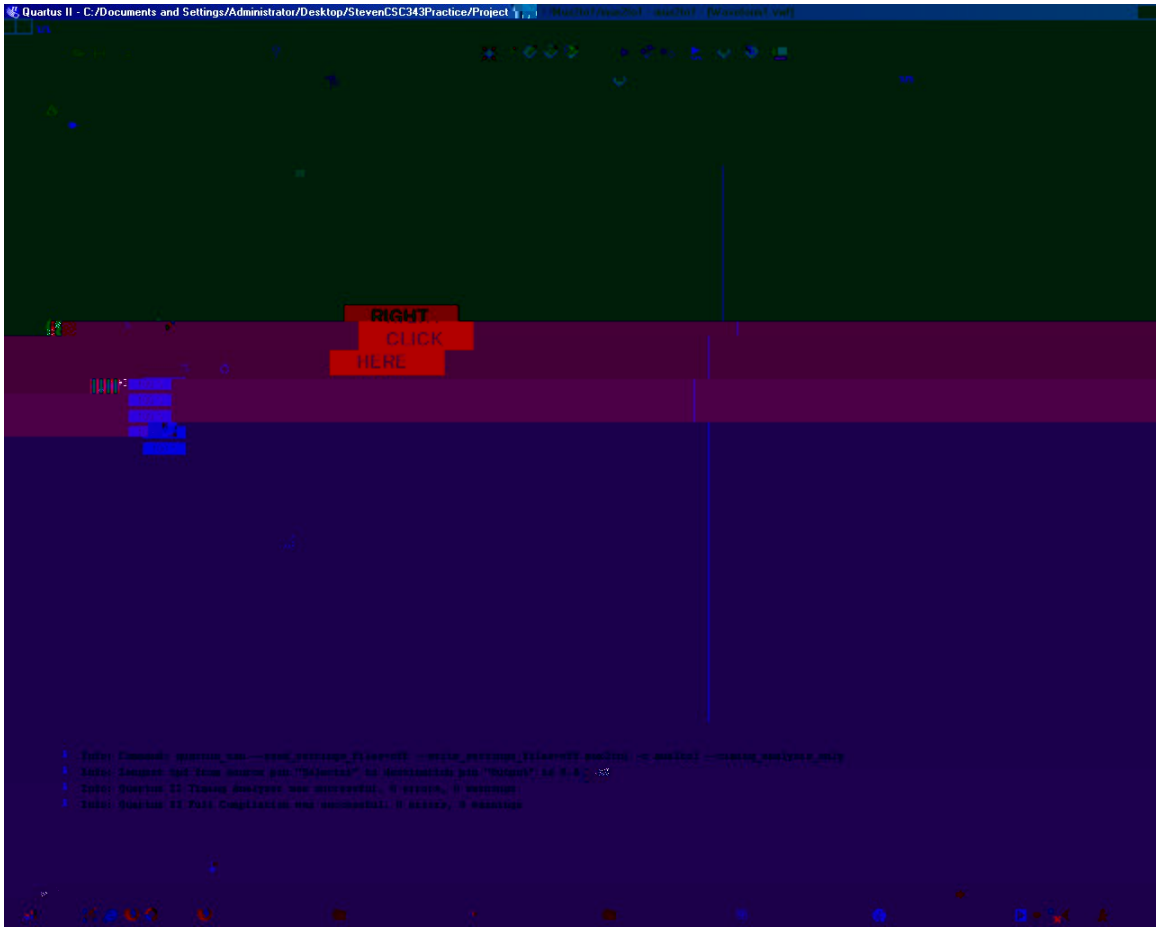
COMPILING AND SIMULATING YOUR DESIGN

COMPILING

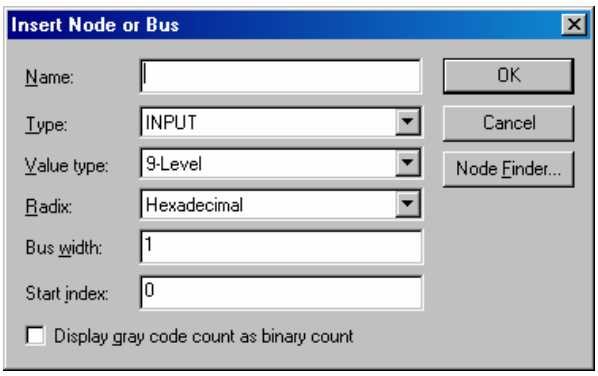
To simulate your design, simply click on processing=>start compilation. If your design has no loose wires or incorrect inputs, all the bars under the status menu on the left side will reach 100%.

SIMULATING

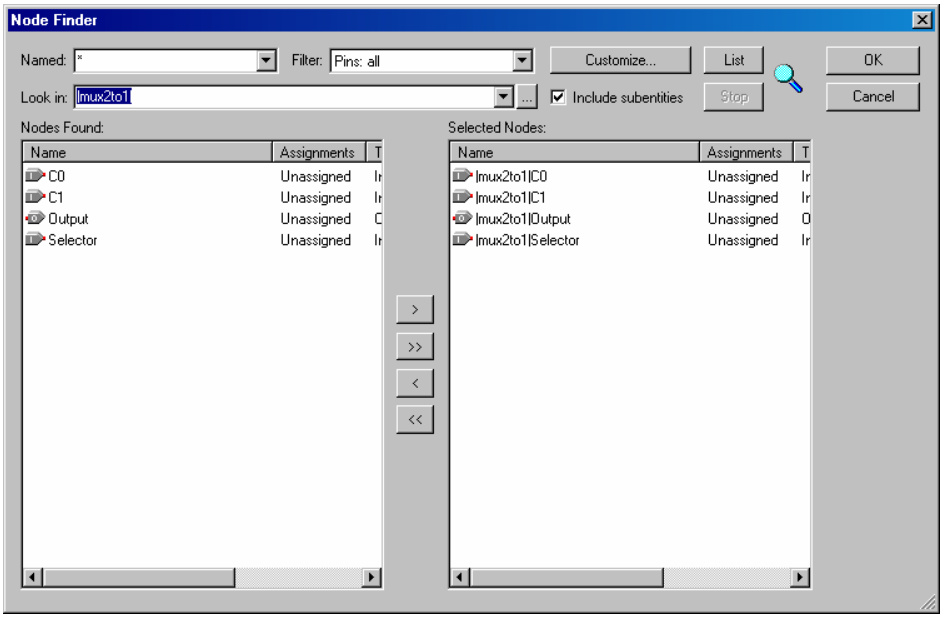
1) Now that you have compiled, you are ready to simulate. First, click on file=>new. In the box that appears, click on the “other files” tab. In the choices that follow, click on “vector waveform file”. A new screen will appear on top of your design. It will look like the image below.



2) Looking at the image above, I placed a red box that says “right click here”. As I am sure you have already guessed, you are going to right click in that area. In the menu that appears, click “insert node or bus”. You will see the following box appear.



From this box, click on “Node Finder”. The menu that will appear is shown below.

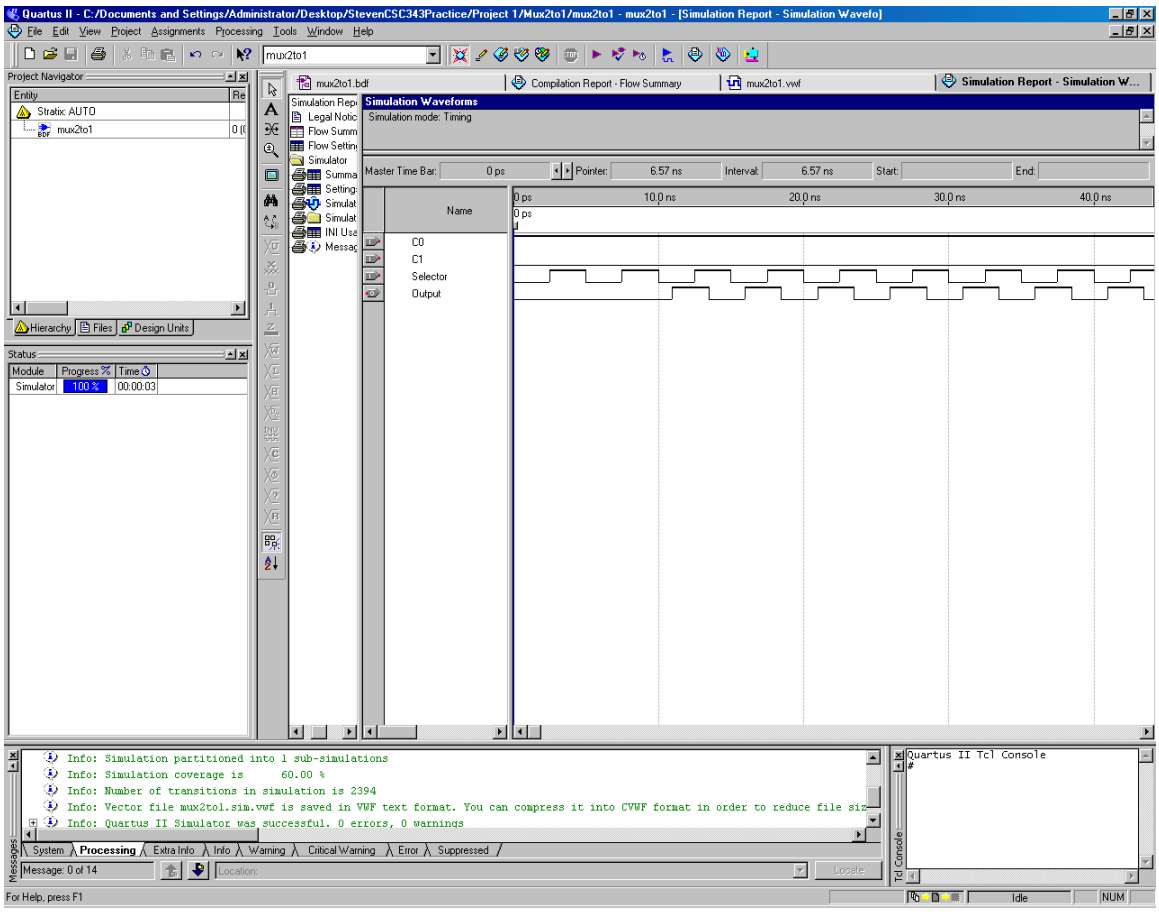


Make sure the drop down box called “filter” says “Pins: all”. Click on “List” on the upper right of this screen. A List will appear on the left menu called “nodes found”. Those are the inputs and outputs of our 2-to1 multiplexer. Click on the double arrow >> to send all those inputs and outputs to the “selected nodes” menu. Click “ok”.

3) You are now back to the original vector waveform file screen. You will now notice all of your inputs and outputs. To simulate, we need to give all the inputs initial values. Right click on the first input and select “Value”. From the menu that appears, click on “Forcing Low”. Next, do the same with the second input, but select “Forcing High” instead.

4) Right click the selector. Under the “Value” menu, select “clock. A box will appear. Change the vale of period from 10 to 5, and click ok.

5) Now, we can simulate. Click on Processing=>Start Simulation. If all goes well, you will see a message that says “Simulation was Successful”. By default, Quartus sets its simulator to “timing simulation”. This means that the resulting simulation will have delays. Look carefully to see if your multiplexer functions as it should; taking into account its delay. You can right click and click on “zoom” to zoom in and out. Your resulting waveform will look similar to the image below (this is a zoomed out image).



CREATE VHDL FILE FROM BDF FILE

VHDL stands for **V**ery Large Scale Integrated **H**ardware **D**escription **L**anguage. It is a language used to describe hardware. This is different that programming in C++ or Java. Since this is your first lab, you aren't expected to know it yet. However, you should at least see what some of the code looks like. The good news is you can have Quartus create VHDL code automatically from your block diagram files. This is done very easily.

1) Open the 2-1o-1 multiplexer design you just created if it is not open already.

2) Click anywhere on the block diagram file to give it focus.

3) Click File=>Create/Update=>Create HDL design file for current file. When you do, a box will appear.

4) Where it says "file type", make sure you have "VHDL" selected. Click OK. You have now created a VHDL file!

5) Your VHDL file may not show up automatically. If that is the case, it is located in the project folder you created for your project. Simply look in the directory of your

project. It should be named the same as your block diagram file. However, the extension will be .vhd.

CREATE SYMBOL BLOCK FROM VHDL FILE

1) Click file=>new. In the box that appears, select “VHDL File”. Save this file as “mux_gate”.

2) Copy and paste the VHDL code below

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;

entity mux_gate is

port (I0      : in STD_LOGIC;
      I1            : in STD_LOGIC;
      I2            : in STD_LOGIC;
      I3            : in STD_LOGIC;
      Sel     : in STD_LOGIC_VECTOR(1 downto 0);
      O             : out STD_LOGIC
      );
end mux_gate;

architecture behav of mux_gate is
begin
   with Sel select
      O <= I0 when "00",
          I1 when "01",
                        I2 when "10",
                        I3 when "11";
end behav;
```

This VHDL code will describe a 4-to-1 multiplexer with a 2-bit selector. The selector must be 2-bits because there are four inputs. Therefore, the selector must have four possibilities. They are 00, 01, 10, and 11.

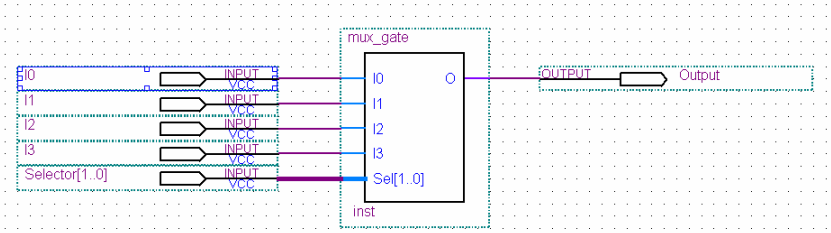
3) Compile the code.

4) Click File=>Create/Update=>Create Symbol files for current file. This will create a symbol block out of your VHDL code.

5) Open a new block diagram file. Save it as “VHDLmux”.

6) Click on “symbol tool”. In the “libraries” menu on the left side, there should be a folder on the left side called “Project”. This folder contains any symbol blocks that you, the user, have created. If you click on that folder, you should see the “mux_gate” that you have just created. Select it.

7) Place that block on the block diagram file. Add input and output pins to the five inputs and the output. Rename the pins anything you like, EXCEPT the pin that goes into the input labeled “Sel[1..0]”. You can do this by double clicking a pin and type in the name in “pin name(s)” field. Your block diagram should look something like the image below.



8) Double-click the pin that goes into the input named “Sel[1..0]”. Rename this pin “Sel[1..0]”, which is the same as the name of the input in the symbol block. The [1..0] part of the name means that the input is 2-bits long, going from bit 1 down to bit 0.

9) Compile and simulate your design. When simulating, make the selector a “counter” instead of a “clock”. This is so all inputs I0 though I3 have a chance to make it to the output. Does the multiplexer work correctly?