

**Ryerson University**  
**Department of Electrical Engineering**  
**COE328 – Digital Systems**

---

## Lab 1

### Introduction to CAD tools

10 Marks (2 week)

Due Date: Week 4

### 1 Objectives

- Introduction to Quartus II software and its tools
- To design, compile and simulate block diagram schematics and VHDL-based design
- Introduction to modular design and mixed-design entry

### 2 Pre-Lab Preparations

1. Locate Quartus II Tutorial 1 (available at COE328 webpage).
2. Install Quartus II version 13.0 on your personal computer. (optional)
3. Familiarize yourself with Quartus II tutorial and all the steps of this lab manual.

### 3 Laboratory Work

#### 1. Creating a Schematic Design

Schematic designs are created using Block Editor tool in Quartus II. Utilize the Quartus II Tutorial section 5 to select Cyclone- II EP2C35F672C6 FPGA board and draw the schematics illustrated in Figure 1 representing the corresponding truth table.

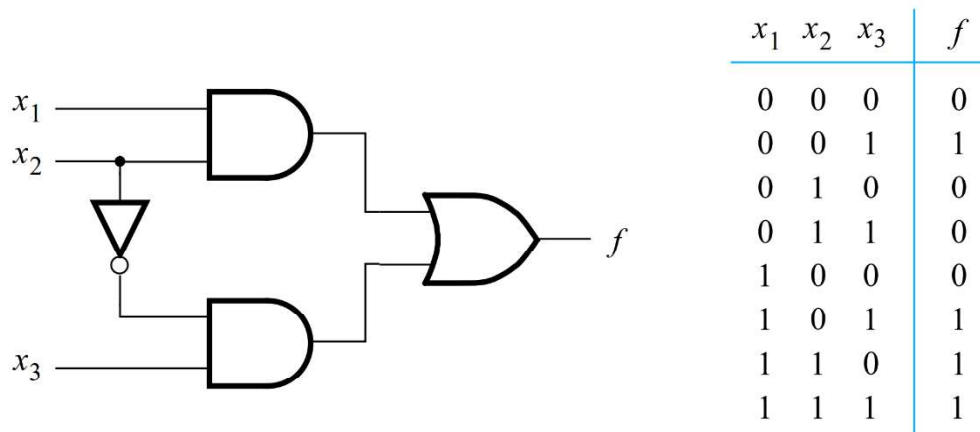


Figure 1

Save the design following instructions in the tutorial.

## 2. Compiling and Simulating a Design

A logic circuit design can be tested for faulty logic by compiling, and then tested for functionality by simulating the design. Quartus II provides powerful compilation and simulation tools which are to be learnt and utilized accordingly.

Run the compiler on your design by selecting **Processing -> Start Compilation**. By selecting the compilation tool, Quartus II will go through various stages. Successful (or unsuccessful) compilation is indicated in a pop-up box. Please refer to Quartus II Tutorial section 6 for more information on design compilation.

When the design is compiled, the verification process can move on to the simulation phase, where input values are assigned to the design, and relative output is produced by the simulation process. The Simulation Waveform Editor is a powerful tool which will enable us to perform all mentioned tasks and verify the functionality of a design.

The main goal in every simulation process is to assign the desired inputs, indicated by the truth table, and then see if the results of the simulation agree with the desired outputs indicated by the truth table. Please refer to Quartus II Tutorial 1 section 8 for step-by-step instructions on how to simulate the design. Results shown in Figure 2 are the final desired outcome.

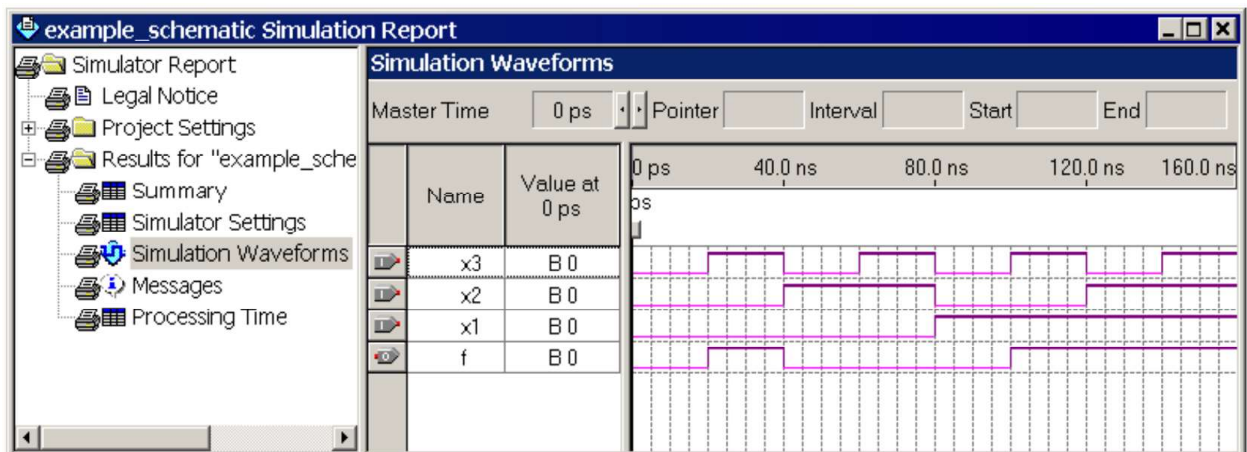


Figure 2

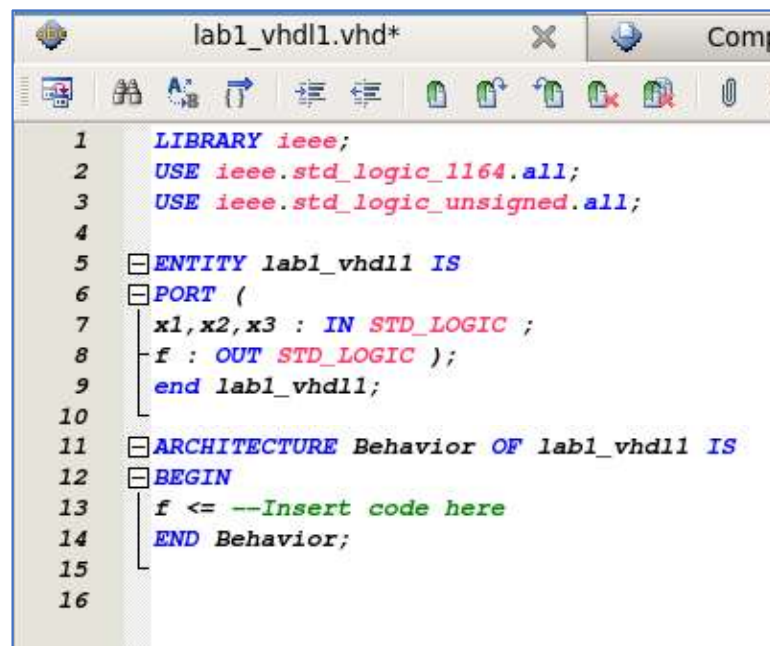
## 3. Design Entry using Truth Tables

Another method of creating a logical circuit design is through writing VHDL code. VHDL is a specific hardware description language (HDL) which is usually used to program FPGA boards and is utilized in industries such as automation.

One of the main advantages of VHDL is the fact that every logical function described in block diagrams, can be implemented by writing VHDL syntax and undergoing the same compilation and simulation processes to verify the design's functionality and performance. In order to build the circuit through VHDL, you need to create a new project, using the same initial parameters as instructed in the Quartus II Tutorial using Cyclone- II EP2C35F672C6 FPGA board. Note that we are creating this project in a new directory.

Quartus II provides a text editor that can be used for entering VHDL code. Select **File -> New**, choose **VHDL File** and click OK. This opens the Text Editor window. The first step is to specify a name for the file that will be created. Select **File -> Save As** to open the pop-up box. In the box labeled **Save as type** choose VHDL File and name it *lab1\_vhdl1*. (Quartus II will add the filename extension .vhd, which must be used for all files that contain VHDL code.)

The objective is to implement the same function illustrated in Figure 1, but this time using VHDL syntax. Figure 3 shows the skeleton code for *lab1\_vhdl1.vhd* code. You need to write the functional code (indicated by the phrase in green - "insert code here"). Pay extra attention at small details in the syntax, for instance the *ENTITY* name should be the same as ".vhd" file name and the same name also repeated in the *ARCHITECTURE* structure.



```

1  LIBRARY ieee;
2  USE ieee.std_logic_1164.all;
3  USE ieee.std_logic_unsigned.all;
4
5  ENTITY lab1_vhdl1 IS
6  PORT (
7      x1,x2,x3 : IN STD_LOGIC ;
8      f : OUT STD_LOGIC );
9  end lab1_vhdl1;
10
11  ARCHITECTURE Behavior OF lab1_vhdl1 IS
12  BEGIN
13      f <= --Insert code here
14  END Behavior;
15
16

```

Figure 3: Code Skeleton for lab1\_vhdl1.vhd

Compile and simulate your design when the code is complete. You can follow the same steps specified in the tutorial to compile and simulate any given design to verify its functionality. Create the waveform vector and try to get the same results shown in Figure 2, since they are essentially the same function being implemented in different ways.

#### 4. Mixed Design

Here we will create a schematic design that includes two different modules. The objective is to assume the following function

$$f = x_1x_2 + \overline{x_2}x_3$$

where

$$x_1 = w_1 w_2 + w_3 w_4$$

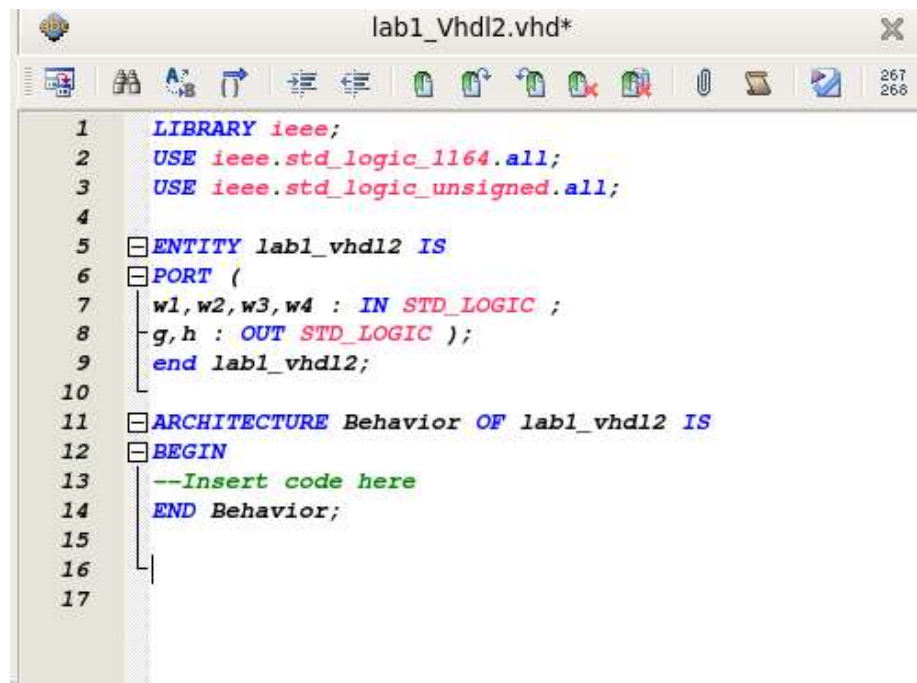
$$x_3 = w_1 w_3 + w_2 w_4$$

Hence the circuit will have 5 inputs of  $x_2, w_1, w_2, w_3$  and  $w_4$ , and one output  $f$ . We have already designed a circuit for

$$f = x_1 x_2 + \overline{x_2} x_3$$

in the previous part. To show how different modules can be connected, whether schematic design or VHDL-coded design, we will create a new VHDL file for  $x_1$  and  $x_3$  expressions and then make a top-level schematic that connects the two designs together.

Using the approach explained in part 3, create a new VHDL file under the same project, save it as lab1\_vhdl2.vhd and then use the code skeleton in Figure 4 to write the VHDL code for  $x_1$  and  $x_3$  expressions. Again, you need to pay extra attention to small details involved in writing VHDL syntax. When VHDL code is completed, compile your VHDL design to verify its basic logical functionalities.



```
1  LIBRARY ieee;
2  USE ieee.std_logic_1164.all;
3  USE ieee.std_logic_unsigned.all;
4
5  ENTITY lab1_vhdl2 IS
6  PORT (
7    w1,w2,w3,w4 : IN STD_LOGIC ;
8    g,h : OUT STD_LOGIC );
9  end lab1_vhdl2;
10
11 ARCHITECTURE Behavior OF lab1_vhdl2 IS
12 BEGIN
13   --Insert code here
14 END Behavior;
15
16
17
```

Figure 4: Skeleton code for lab1\_vhdl2.vhd

In order to couple the two designs, we need to create what is known as symbols for each design. Quartus II is able to make the symbol only for the *last compiled design*. So when you have one of the designs compiled, select **File -> Create/Update -> Create Symbol Files for Current File**. In response, Quartus II will generate a Block Symbol File (with the same name as the design file and with *.bsf* extension). Create design symbols for both - VHDL design and schematic design.

Next step is to create a Block Diagram in which the two designs are to be imported. In the same project, create a **Block Diagram Schematics** file following the steps in part 1. Now you can add the symbols that you have created by double-clicking on the Block Editor screen and select the created symbol to import it into the design. The two symbols *lab1\_vhdl1.bsf* and *lab1\_vhdl2.bsf* are available in the *projects* directory as portrayed in Figure 5 and Figure 6. Select one of them at a time and click OK to import them to your design.

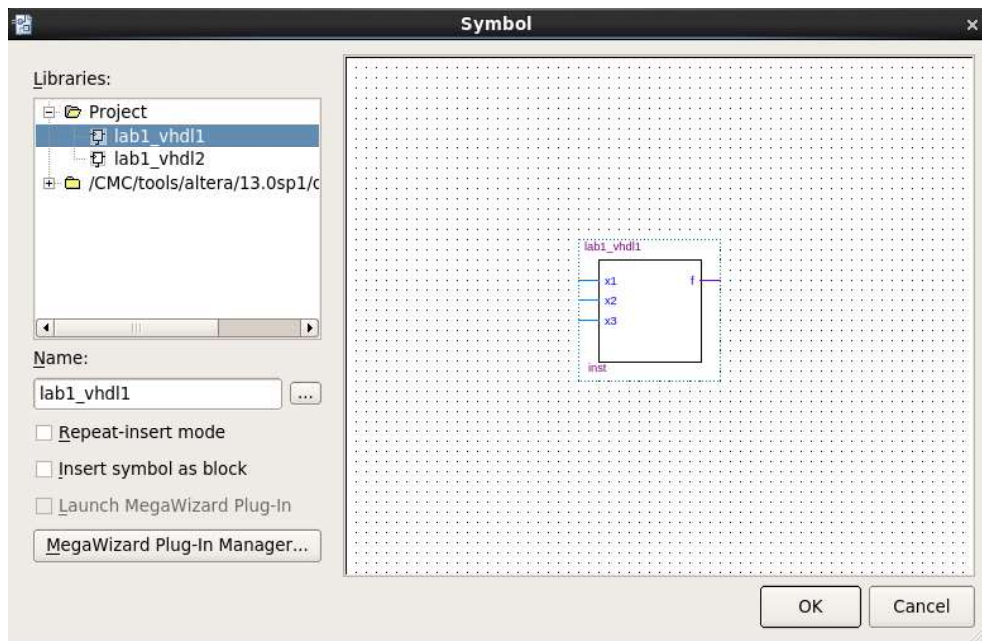


Figure 5: lab1\_vhdl1.bsf symbol import

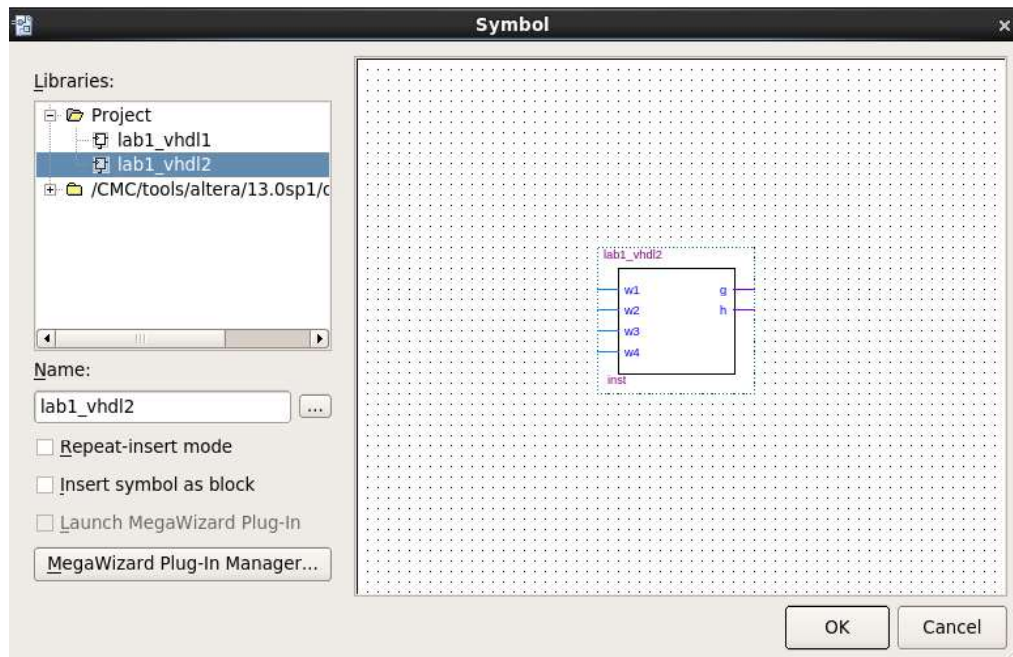


Figure 6: lab1\_vhdl2.bsf symbol import

Now that both symbols are imported to the block diagram schematics file, you can finalize the design. Figure 7 illustrates a completed block diagram schematic design.

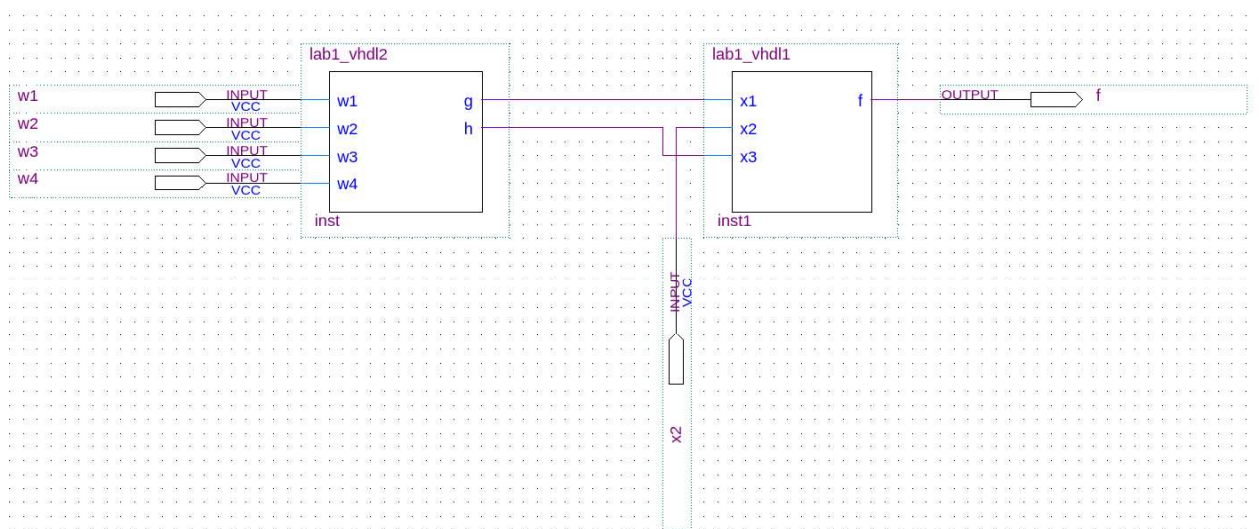


Figure 7: Block diagram schematics of mixed design



Next step would be to compile the design and perform simulations on the design. Follow the steps from part 1 to compile your design, then create a waveform vector file and introduce input values for the input ports and simulate your design. Figure 8 illustrates the waveform output for the mixed design, which is the final objective in this experiment.

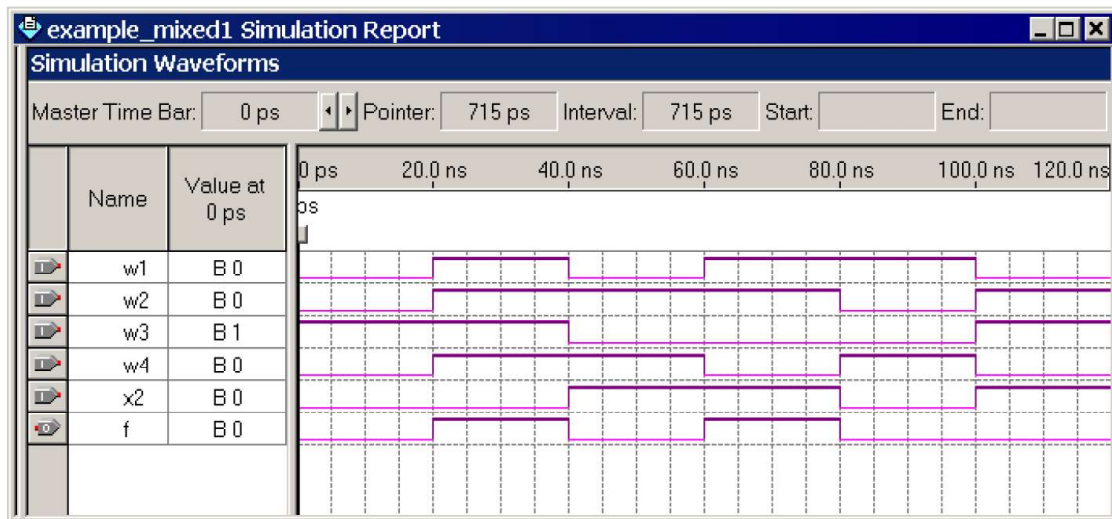


Figure 8: Waveform output for mixed design