EEL 3701C: Digital Logic & Computer System

LAB 01: Quartus Installation and First Circuit Design
(25 Points)

Instructional Objectives

At the end of this lab you should be able to:

1. Use reduction techniques to obtain minimal functional representations
2. Design minimal three-input logical functions.
3. Build and debug three-input logical functions.
4. Gain an introductory knowledge of Quartus and VHDL

Introduction:

Boolean Algebra is used to analyze and simplify the digital (logic) circuits where there are a number of inputs. Boolean logics are composed of a number of laws which are covered in class. Boolean expressions can be optimize using De Morgan's law and Karnaugh Map. We will apply that knowledge in this lab. Moreover, NAND or NOR gates are told as universal gates and we can generate any gate using them. In this lab, we will explore that concept to obtain a simplified circuit using NAND or NOR gate.

Background

Background information give you about the idea of using NAND and NOR gates for the universal function. You can use the NAND and NOR gate to make any gate you want, for example you can use the NAND gate to make an AND gate or even a XOR gate. This is what you call a building block. In a design if you need and OR gate and an inverter, then you need two ICs. But you can use one IC for with NAND gate to reduce the number of ICs in your design. The following figures shows how you apply NAND and NOR gates for universal function.

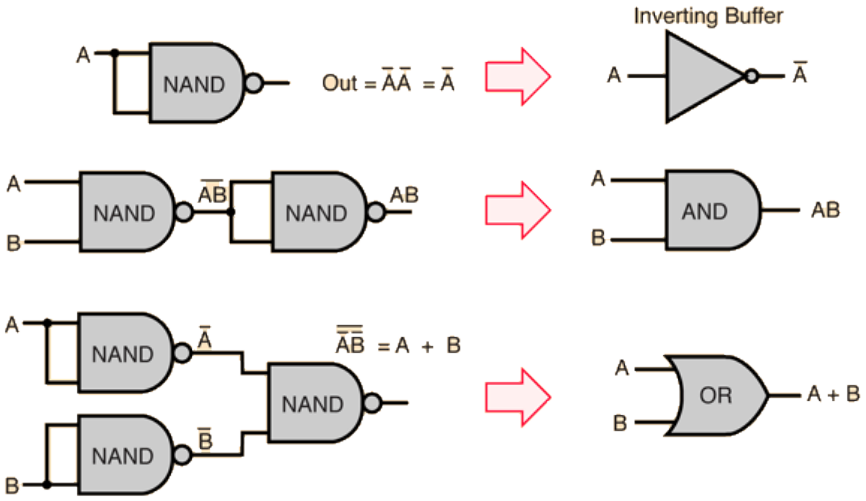

Figure 1: Basic gate design with NAND gates

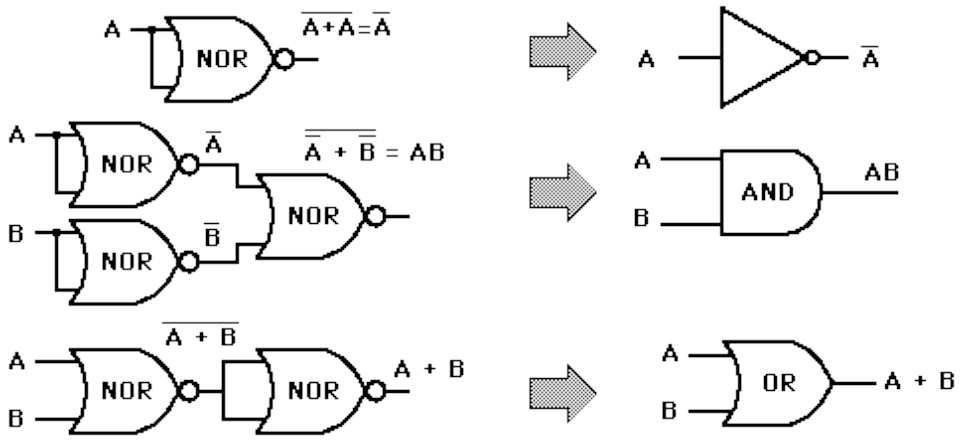

Figure 2: Basic gate design with NOR gates

Experiment:

The work for this lab is divided into pre-lab and in-lab homework. The in-lab home work has some Boolean equations. You have to express the expression using minimum gates and then implement with NAND or NOR gates. To show the output, you are advised to use LEDs.

Pre-Lab Homework

a) Optimize the following logic equations using Boolean Algebra and find the truth table
   i) $Y1 = A\,\overline{B}\,C + \overline{A}\,B\,C + A\,B\,\overline{C} + A\,B\,C$
   ii) $Y2 = A\,\overline{B}\,\overline{C} + \overline{A}\,B + B\,\overline{C} + B\,C$

b) Implement **Y1 only** (after optimization) using NAND gates on your breadboard. **Use pull-up switches and LEDs with resistors.** Demonstrate to yourself that it works properly. Take a picture of your circuit.
   i) You may use both Triple and Double NAND gates.
   ii) Hint: Minimal form of Y1 is 3 Double-NAND's and 1 Triple-NAND

c) Implement **Y2 only** (after optimization) on your breadboard. Take a picture of your circuit. **Keep this on your breadboard, you will demo it in your lab.**
   i) Hint: Minimal form of Y2 is 4 NOR gates.

d) Install Quartus using the guide on Canvas. Follow the tutorials on how to create a project, how to make a BDF file, and how to run simulations.

e) In a new project, Implement the Y1 and Y2 in a single BDF file using NAND gates. Simulate the result and take screenshots of your BDF and simulation. **Annotate your simulation with arrows or text describing what is happening.**

f) In a new project, implement the Y1 and Y2 in a single BDF file using NOR gates. Simulate the results and take screenshots of your BDF and simulation. **Annotate your simulation with arrows or text describing what is happening.**

g) Program the circuit you made in Part F to your Out of the Box MXDB. Create a circuit using pull-up switches and LEDs (with resistors!) to demonstrate its correctness.

h) Record using your phone or a webcam a live demonstration of Part C and Part G functioning. Submit this recording to Canvas with your Pre-Lab Document
    i) If you are unable to do Part H due to not having the means to record your circuit, please contact your PI.

Pre-Lab Deliverables (What you will record and submit to Canvas along with your Prelab Report)

1. The **breadboard** implementation of Y2 using NOR gates (Part C)
2. The **MXDB BDF** implementation of Y1 and Y2 using NOR gates (Part G)

Pre-lab Report
Include all requested pictures, screenshots, and **annotated** simulations in your report. Fill in only the prelab sections of the lab template provided on canvas.

Please follow this steps in these documents very carefully to install the fpga board and simulation platform :

1. Go to ***Quartus-Installation-Configuration.pdf*** file and install necessary softwares.
2. Go ***MXDB manual*** and follow the steps to install the driver.
3. Files for installing the FTDI chip (in MXDB manual) is also shared in the folder as a zip file. Name is ***programmer driver bat file install.zip*** .

In any problem installing the FPGA board see the MXDB manual troubleshooting pages at the end.

Post-Lab Implementation (10)

1. Your In-Lab implementation will consist of implementing Y1 and Y2 using VHDL.
    a. You do not have to use only NAND or NOR gates in VHDL. The compiler will do that for you. Instead, you should enter the minimized forms of Y1 and Y2 you found in Part A.
    b. A sparse template will be provided to you for the VHDL. It will consist of only the skeleton of a VHDL program: The includes, the entity definition, and the architecture definition. ***(Use Quartus Tutorial.pdf for help) details are given***
        i. Do not use this as a crutch. Be prepared to write this on your own in the future.
2. Screenshot your simulation your VHDL. **Annotate your simulation with arrows or text describing what is happening.** *(Waveform Simulation Tutorial.pdf for help)*
3. Program your VHDL circuit to the MXDB.
    a. The circuit you built in Part G to verify the BDF file should work to verify the VHDL circuit without modifications.
4. Record your VHDL circuit from Part 3 functioning to submit to canvas along with your Postlab Report.

Post-Lab Deliverables: (due a week after your lab session)

- Summary of your work in a report. Template for lab report is provided on Canvas
    - You should include all screenshots, code, and **annotated** simulations from the **Post-Lab Implementation**.
    - Keep the prelab report intact in its section. You will be **updating** your prelab report to transform it into your post-lab report.

i) A recording of your VHDL circuit working, as described in Part 4.