Lab 2: MSI Circuits

OBJECTIVES

- Learn about how to use small scale integrated (SSI) devices to create medium-scale circuits (which, when part of a ICs, are called medium scale integrated [MSI] devices).
- Convert a truth table into logical equations that describe your circuit's operation.
- Design and implement two multiplexers and a decoder.

INTRODUCTION

While small-scale integrated circuits such as AND, NAND, OR, etc. are the foundations of all digital circuits, it can be helpful to use a higher level of abstraction when using them in more complex circuits. The first level of abstraction above these gates are medium scale integrated (MSI) circuits such as multiplexers, decoders, and encoders. These MSI components are governed by characteristic equations that behave similarly across different sizes of these components.

Multiplexers (MUXs) can be thought of as multiple-position switches. A MUX has several data inputs, and some select inputs, but only one output signal. The output of the MUX is determined by the value of the data input that corresponds with the number input on the select lines. For example, when the select lines (S_1 and S_0) of a 4-input MUX (with data inputs D_{00}=D_0 through D_{11}=D_3) are equal to "00", then data input 0 (D_{00}=D_0) is logically connected to the output. When the select lines are equal to "01", then data input 1 (D_{01}=D_1) is logically connected to the output. This rule holds for any size MUX, no matter the number of data or select inputs. (Note that a 4-input MUX has six inputs!)

A traditional decoder circuit has n-inputs and 2^n outputs; the n inputs determine which single output is true. For example, with a 2-to-4 decoder (with inputs X_1 and X_0 and outputs Y_3 to Y_0), when "X_1X_0 = 01" then output Y_1 = 1 and Y_3=Y_2=Y_0 = 0. However, the word decoder also refers to the more general class of combinational circuits that convert from one representation of binary data to another. In this lab, you will design a hex-to-7-segment decoder that converts a 4-bit binary number into a graphical representation of that number's value on a 7-segment display.

LAB STRUCTURE

In this lab, you will learn about the design and use cases of MSI integrated circuits. In § 1, you will implement a 2-input MUX using SSI chips. In § 2, you will extend the concepts learned from § 1 to design a 4-input MUX. In § 3, you will design a hex-to-seven-segment decoder that you will use in all future labs. Finally, you will use a 4-input MUX to implement an arbitrary combinational equation in § 4.

REQUIRED MATERIALS

- DE10-Lite Pins
- *PLD on Breadboard Programming WARNING!*
- *Quartus Tutorial* Section IV: Programming)
- File for Part B: **`hex_to_7seg.bdf`**

SUPPLEMENTAL MATERIALS

- Dual 4-input MUX specification sheets: *National Instruments*; *Philips*
- Dual 4-input MUX in Quartus is called **`74153`**

PRE-LAB PROCEDURE

1. You will need to program your DE10-lite in this lab. In Homework 1, you skipped *Section IV: Programming* in the *Quartus Tutorial*. You will need to read this section and utilize this information in this lab (and in the remaining labs).
2. Only program pins that are listed in the DE10-Lite Pins document. These pins are specifically connected to output pins or other peripherals (such as seven segment displays) that you can use in your designs. Using other pins risks damaging the DE10-lite.
3. A single pdf document (for this lab, called `lab2.pdf`) of all design files and simulation files is required for this and all pre-labs. I suggest that you capture screen shots of each design and simulation (as they are generated) into an MS-Word (or equivalent) document. This file should also include any other required items including truth/voltage tables, or anything else specifically requested in the lab document, the *Lab Submission Template*, or required in the *Lab Rules & Policies*. After completing the document, save it as a pdf file.
4. As usual, submit your Pre-Lab Report pdf file (`lab2.pdf`) and the three Quartus archive files that you generate in the pre-lab sections under the Lab 2 assignment in Canvas.

1. INTRODUCTION TO MULTIPLEXERS

The multiplexer (MUX) is a device that acts as a multi-position switch (see Figure 1). A MUX has multiple DATA inputs (labeled D_i) and at any given time, one of these inputs is "logically" connected to the output Y. The SELECT lines (labeled S_i) control which input is passed to the output. You will begin this lab by implementing a 2-input MUX, which has two data lines and one select line. For a 2-input MUX, when S0 = 0, D0 will be "logically" connected to the output. When S0 = 1, D1 will be "logically" connected to the output. In general, reading the value of the select lines as a binary number (with S_0 being the least significant bit) will connect the data line with that same number to the MUX output.

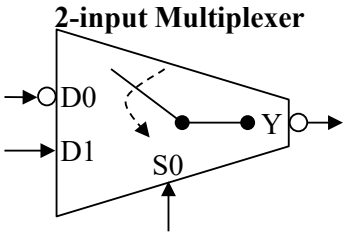

Figure 1: 2-input MUX.

1. Draw a truth table for the 2-input multiplexer (3 inputs and 1 output) using "wild cards" as appropriate. Wild cards, usually designated with an "*" or a "–", indicate that any value can replace it. If one * exists in a row of a truth table, the one row represents two rows; one with the * replaced by a 0 and the other with the * replaced by a 1. If there are two asterisks in a single row, that row is an abbreviation for four rows.
2. Derive the logic equation for this MUX from the truth table (both an SOP and a POS).
3. Draw a voltage table for this MUX. The figure shows that one of the D inputs and the select line are active-high; one of the D inputs and the output are active-low.
4. Draw a functional block diagram of the MUX; draw signal definitions (i.e., activation levels and signal names **outside** the MUX).
5. Design a circuit for this MUX using logic gates available in your lab kit. Use project name `lab2_2inMUX`. (It is usually easier to design first on paper.) There is no requirement, in this lab, to minimize the number of gates or chips; but of course, a simpler design will be easier to build.
6. If you designed the circuit in part 4 on paper, now design it using Quartus (with filename `lab2_2inMUX.bdf`) When using discrete logic chips, i.e., 74'xx ICs, add pin numbers and chip labels to the logic circuit diagram to make this a wiring diagram. You will use this circuit design to build the MUX circuit on your breadboard in item 7. When you create designs exclusively with the DE10-lite (i.e., not using any logic ICs), it is NOT necessary to use 74'xx pin and chip labels. Just make sure that the BDF screenshot that you include in your lab document includes the DE10-lite pin assignments generated by the Pin Planner.
7. Simulate the complete mixed-logic circuit in Quartus. Annotate the simulation and include it in your lab report.
8. Submit the archive file `lab2_2inMUX.qar` through Canvas.

9. Build your MUX circuit on your breadboard using the parts in your lab kit. You will need switch circuits for the inputs and an LED circuit for the output. Use the GPIO header (JP1) on the DE10-Lite to provide Vcc (labeled 3.3V) and GND, just as you did in Lab 1. You will demonstrate this circuit in your prelab demo.
10. Use the truth table from item 1 to demonstrate to yourself that your MUX circuit operates correctly for all possible input combinations. Create switch and LED legends for this circuit and include them in your lab report. Have them ready when your PI asks you to demo your MUX circuit. Note that LEDs should always be lit when the related signal is true.

2. FOUR-INPUT MULTIPLEXER DESIGN

In this part of the lab, you will implement the 4-input multiplexer described by Figure 2. The steps are largely identical to the steps in Part 1, except you are not required to implement this MUX using the gates in your lab kit; you will instead design it with SSI gates available in Quartus.

1. Create a truth table describing the behavior of the 4-input MUX, using wildcards where appropriate.
2. Derive the logic equation for this MUX from the truth table in both SOP and POS form.
3. Draw a voltage table for this MUX, paying close attention to the activation levels of the inputs and outputs shown in Figure 2.
4. Draw a functional block diagram of the MUX; draw signal definitions (i.e., activation levels and signal names **outside** the MUX).
5. Design a circuit that implements the MUX equation on your scratch paper and in Quartus. You may use SSI gates available in Quartus. Use project name `lab2_4inMUX`.
6. Simulate your MUX design to verify that it functions correctly. Do not just cycle through all possible inputs in counting order, instead use your knowledge of MUXs to design a more concise simulation.
7. Program the 4-input MUX circuit to your DE10-Lite and create the necessary switch and LED circuits on your breadboard to view the MUX's operation. To connect switch and LED circuits to the DE10-Lite, connect the necessary wires on your breadboard to selected GPIO header pins. Do **not** use the same switch circuits that you created for the 2-input MUX.

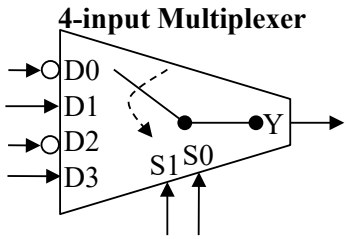

Figure 2: 4-input MUX.

8. Submit the archive file `lab2_4inMUX.qar` through Canvas.
9. Prove to yourself that your MUX functions correctly. Create switch and LED legends for this circuit and include them in your lab report. Have them ready when your PI asks you to demo your MUX circuit.

3. DECODER DESIGN

In this part of the lab, you will design a Hex to 7-segment Decoder that you will use in all future lab assignments. This decoder will take 4-inputs (a hexadecimal number) and output 7 active-low signals that will in turn drive an

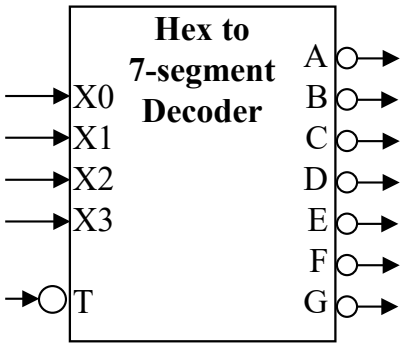

Figure 3: Hex to 7-segment decoder.

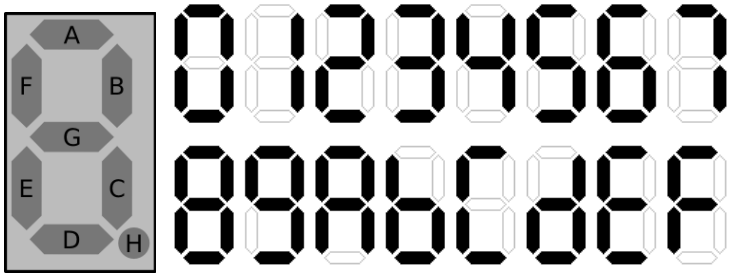

Figure 4: LED arrangement on 7-segment display (left) and proper display of each hex character (right). Note that the 7-segment displays on the DE010-lite are active-low.

active-low 7-segment LED display on your DE10-lite. Figure 3 shows a block diagram of the part that you are required to design.

The left item in Figure 4 shows the arrangement of each of the LEDs on the 7-segment LED device. Each of the LEDs are active-low. The right item in Figure 4 shows the proper display for each of the 16 hex characters. The T input in Figure 3 is used to test the outputs; when T is true, all the outputs A through G are true.

1. Draw a truth table for the above decoder.
2. Derive both the SOP and POS logic equations for each of the outputs A and B from the truth table. Also derive the SOP equation for C.
3. Draw a voltage table for A, B, and C of the above decoder.
4. Draw a functional block diagram of this decoder; draw signal definitions (i.e., activation levels and signal names **outside** the Decoder).
5. Design a circuit to implement the decoder using SSI parts only (i.e., AND, NAND, BNAND, NOT, …). Use the project name `lab2_decoder`. (It is usually easier to design first on paper.) A partially completed and well-formatted design, hex_to_7seg.bdf, is available on our website. You may copy the designs in this file and augment them with your own designs or design the entire circuit yourself. **Note that you are required to design a POS circuit for the decoder's A and B outputs.** Note that creating all the minterms in the given `hex_to_7seg.bdf` was overkill and unnecessary for this problem but was done to facilitate a quick solution.
6. If you designed the circuit in part 5 on paper, now design it using Quartus (with filename `lab2_decoder.bdf`). There is no requirement, in this part of the lab, to minimize the number of gates or chips. Note that since you are not using discrete elements (74'xx chips) in this part of the lab (you are instead using the PLD on the DE10-lite), you should not put 74'xx chip numbers and pin numbers on your circuit diagram.
7. Simulate your decoder design and include an annotated simulation in your lab report. A comprehensive simulation in counting order is appropriate for this simulation.
8. Submit your archive file through Canvas. It should have the filename `lab2_decoder.qar`.
9. Combine your BDFs for the Hex to 7-segment Decoder and the 4-input MUX so that you can flash them both to your DE10-Lite at once. Use the project name of `lab2_combined`.
10. Submit the archive file (for this combined circuit) `lab2__combined.qar` through Canvas.
11. Use the switch circuits for part 1 and 2 again for these combined circuits. Since you only have an 8-input DIP package, use the same inputs for the 4-input MUX and the decoder. Connect the outputs of the decoder to HEX0 on your DE10-Lite.
12. Demonstrate (to yourself) that your MUX circuit and Hex to 7-segment Decoder circuit both operate correctly.
13. Create switch and LED legends for these circuits and include them in your lab report. Have them ready when your PI asks you to demo your MUX circuit.

4. MUX IMPLEMENTATION OF A LOGIC EQUATION

In this part of the lab, you will design a circuit to implement the equation,

$$F = \overline{V}\,(W + \overline{X}\,Z) + \overline{V}\,\overline{W}\,\overline{X}$$

You will only use a single 4-input MUX (with enable) for this design. Use the minimum number of additional SSI gates; zero additional SSI gates, if possible, is the best solution. You will build this circuit in **Quartus only**. Use the 74'153 dual 4-input MUX chip in Quartus.

1. When you use Quartus to design with a 74'153 MUX (search for 74153, see Figure 5), you will find that the select lines are labeled A and B. Which select inputs correspond to A and B for this MUX? Verify the functioning of a 74'153 MUX (and if A B =S1 S0 or A B = S0 S1) by creating a new Quartus circuit (in a new project and file, `lab2_MUX_TEST`) using only this chip, inputs, and outputs. Simulate this design using enough different input combinations to prove that the 74'153 works as you suspect it should. For every "new" part you use (like the 74'153), if you are not sure of a pins function, you should verify its operation before using it in a circuit. Include this design and simulation in your pre-lab document. Submit the archive file `lab2_MUX_TEST.qar` through Canvas.

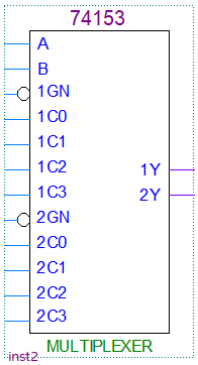

Figure 5: 74'153 in Quartus.

2. Create a truth table for the above equation.
3. Derive an MSOP or an MPOS logic expressions for the equation. (You may use K-maps, if desired.)
4. Design a circuit (`lab2_EQU.bdf` in project `lab2_EQU`) to implement this equation with a single MUX from the 74'153 Dual 4-input MUX chip (in Quartus). If useful, utilize the MUX's enable in this design. Use the MUX select lines as follows: S1 = W(H) and S0 = X(H).
5. Make a voltage table from the truth table and the activation-levels that you chose above.
6. Simulate your design using Quartus to verify that your design works (by comparing it to the voltage table).

PRE-LAB PROCEDURE SUMMARY

1. Learn about how to design and implement different sizes of multiplexers in § 1 and § 2.
2. Design a Hex to 7-segment Decoder in § 3.
3. Use MUXs to implement specific logic equations in § 4.

IN-LAB PROCEDURE

1. Complete the lab quiz.
2. Demonstrate the following circuits from the pre-lab:
   a. 2-input MUX made with SSI chips
   b. 4-input MUX made with DE10-Lite
   c. Hex to 7-segment Decoder made with DE10-Lite.