UF Herbert Wertheim College of Engineering
Department of Electrical & Computer Engineering
UNIVERSITY of FLORIDA

Dr. Christophe Bobda
Newman Waters
EEL3701C - Fall 2025

Lab 2: Hierarchical Component Design

Overview

- Learn the basics of hierarchical component design
- Understand more complex techniques of simplifying logic functions
- Grasp the concept of a 'top level' design

Introduction

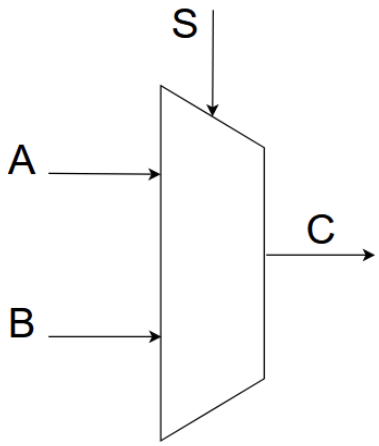
When designing systems in the real world, we tend to want to save money where possible. The easiest way to do this is to create small, reusable components rather than starting from scratch every time. Otherwise, engineers would have to reinvent the wheel every single time they start a new design, which is a huge waste of effort (do you really want to do all of this logic simplification every time?).

Thus, we will spend time designing a few basic components from scratch once that will be reused throughout the labs (and other future projects).

Pre-Lab Procedure

Part 1. You will design a two-input multiplexer. This multiplexer will have 3 inputs: **A** (input 1), **B** (input 2), and **S** (select line). There is one output **C**. You should design your multiplexer with these rules: If S = 0 then C = A or if S = 1 then C = B

You must draw out the truth table to generate the logic function $C(A, B, S)$.
Truth table must be in counting order in form: **S A B | C**
Simplify the equation as much as possible, then design it in **mux_2x1** and include a screenshot of your design in the report.



Herbert Wertheim College of Engineering
Department of Electrical & Computer Engineering
UNIVERSITY of FLORIDA

Dr. Christophe Bobda
Newman Waters
EEL3701C - Fall 2025

Part 2. You will design a four-input multiplexer. However, this one will be designed slightly differently. Sometimes you do not always have the components you want, and have to use your resources wisely to create new components.

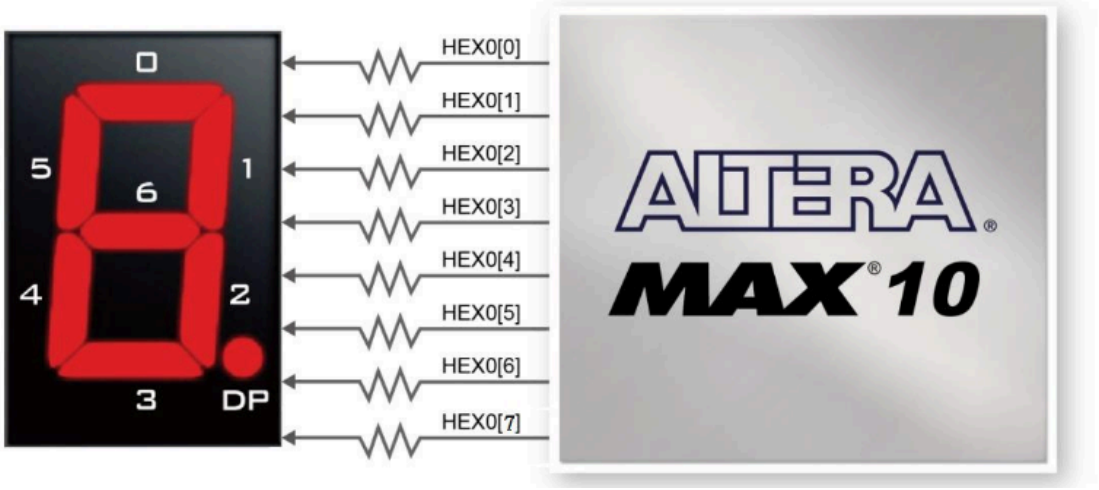
For this part, you only have 3 of the 2x1 multiplexers that you designed in part 1. You now need to devise a method to combine these multiplexers in a way that models the same behavior of a standard 4x1 multiplexer in the **mux_4x1** circuit, with two select signals **S1:S0**, which can be read as a binary number:

S1:S0	E (Output)
00	A
01	B
10	C
11	D

For information on instantiating components in Logisim, see Appendix A.

Part 3
P3A) You will complete the design of a 2-bit hex to seven-segment display decoder.
The design of this component requires understanding a few things about the hex-segment display that you are working with.

From the manual, page 29:



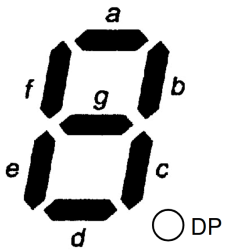
Herbert Wertheim College of Engineering
Department of Electrical & Computer Engineering
UNIVERSITY of FLORIDA

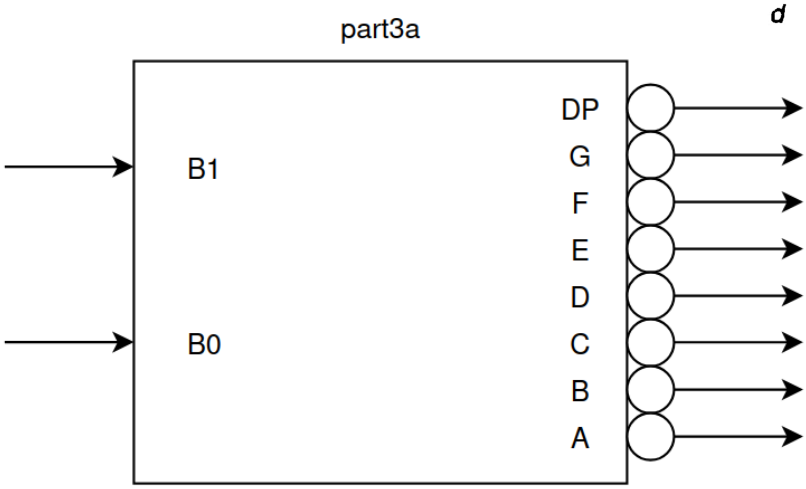
Dr. Christophe Bobda
Newman Waters
EEL3701C - Fall 2025

"The segment can be turned on or off by applying a low logic level or high logic level from the FPGA, respectively." This means:

- Segment turns **on** when pin is at 'low' logic level
- Segment turns **off** when pin is at 'high' logic level

This means that your design will need to send out 'active-low' outputs. When generating a logic equation for a specific segment, this means you should place a '0' where it should be on, and a '1' when it should be off. You will create your display to process the first 4 numbers (0, 1, 2, 3):





Note: The "DP" signal (decimal point) should always be set to 1 to ensure it is OFF.
Draw out the truth table in counting order for each digit in this form and include it in your report:
B1 B0 | DP G F E D C B A
0 0 | 1 1 0 0 0 0 0 0 # Example: 0xC0: all segments are on except middle (G) and decimal point
…Continue

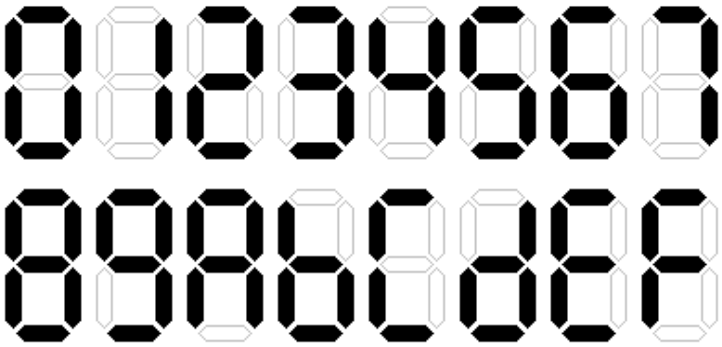
Include a screenshot of part3a in your pre-lab submission, along with the logic function for each segment (A, B, C, …) under the corresponding K-Map. Show your work in your pre-lab report.

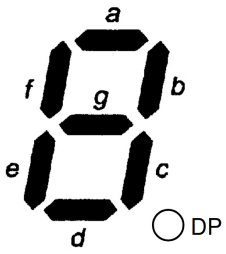
UF Herbert Wertheim College of Engineering
Department of Electrical & Computer Engineering
UNIVERSITY of FLORIDA

Dr. Christophe Bobda
Newman Waters
EEL3701C - Fall 2025

P3B) You will extend the functionality of your seven-segment display to show more digits. However, we don't want to generate logic by hand anymore, since it takes a really long time. With more digits, we need more binary inputs. In this part, you will create a 4-bit hex to seven-segment display decoder.

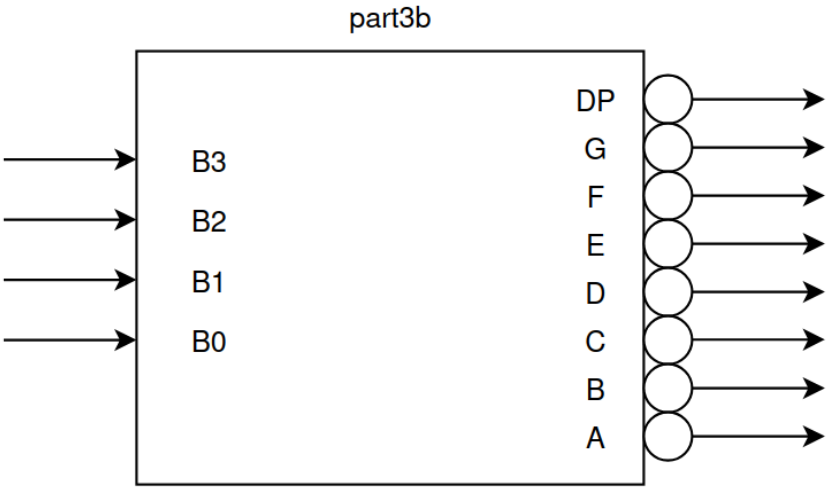
The end result is a component that will be able to display the following digits:





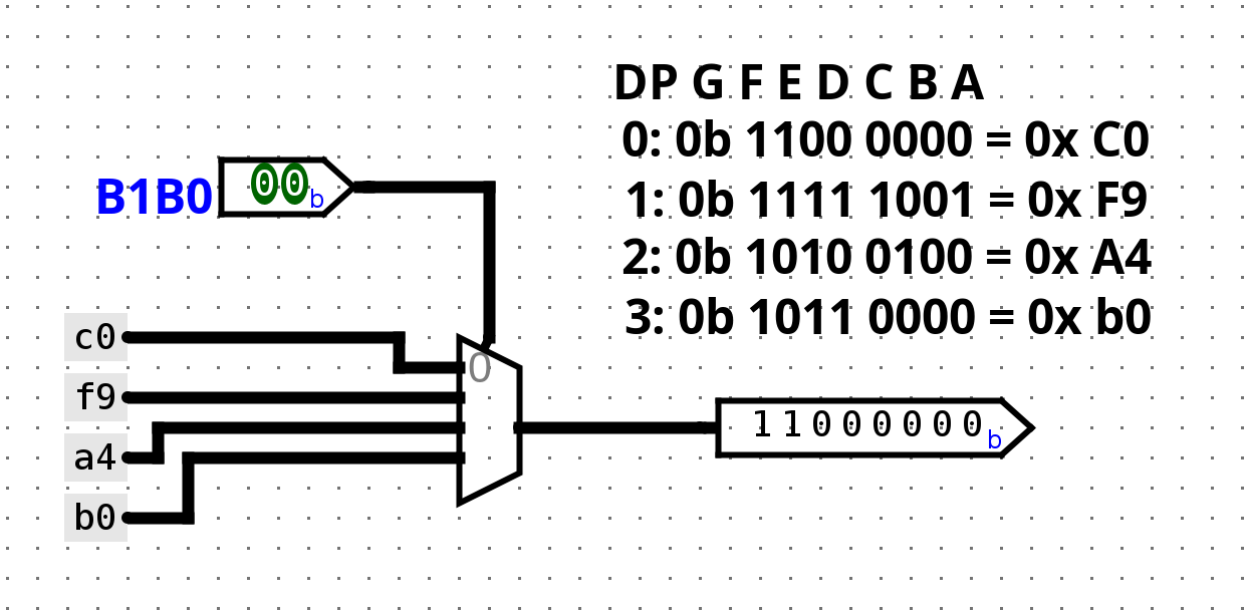
Draw out the truth table in counting order for each digit in this form and **include in your report**:
B3 B2 B1 B0 | DP G F E D C B A
0 0 0 0 | 1 1 0 0 0 0 0 0 # Example: 0xC0: all segments are on except middle (G) and decimal point
…Continue
1 1 1 1 | 1 0 0 0 1 1 1 0 # Example: 0x8E: draws out an F on the display, removes the decimal point

Now, we want to create the following component:



To do this, you will 'hardcode' the truth table values, and use your multiplexer components to select between the different values within the table.

For example, here is what your component from **part3a** would look like with this method:



Now, there will be 16 different values you need to handle, which means it is not as simple as using one multiplexer to choose between the values. You will need to do something similar to **part 2**, where you implement the logic of an 8-bit, 16x1 multiplexer using 8-bit, 4x1 multiplexers. These 8-bit 4x1 multiplexers are already created for you in the circuit called *mux_4x1_8bit* provided your 4x1 mux is designed correctly. You must use this component in your design.

Complete the design in **seven_segment**, and include a screenshot of your design in your report. You have completed your 'binary to 7 segment display' (display in hex) driver circuit.

In your lab report, complete the following question:
Generate the logic equation for segment D using a 4-variable K-map. Show all work.

Part 4. Master of Tones

You have been hired as a guitar technician for a band called MUXallica. They saw the amazing 4x1 MUX and the displays you created, and want you to use this component to help manage their guitars. They have asked you to devise a way to switch between different guitar tones while they are playing live (i.e. Distortion, Clean, Lead, Flanger).

UF Herbert Wertheim College of Engineering
Department of Electrical & Computer Engineering
UNIVERSITY of FLORIDA

Dr. Christophe Bobda
Newman Waters
EEL3701C - Fall 2025

Your task is to use 2 switches, **SW[1:0]** (representing a binary selector) to choose between the different tones.

They want you to use **4** 7-segment displays to give a better visual representation:

S1:S0	Tone	Displays [HEX3-0]
00	Distortion	“D157”
01	Clean	“C1EA”
10	Lead	“1EAD”
11	Flanger	“F1AE”

Implement this using constants, your 8-bit 4x1 multiplexers, and the seven_segment components you created in the previous parts in the **part4** circuit.

You will map your design to the DE-10 Lite board using the procedure in Appendix B. Make sure to have this ready **before** your lab, as you will be demonstrating this in your lab session.

In-Lab Procedure

1. You will demo your circuit working in **part 4b** to your PI. **(20)**
2. Complete the quiz in-lab as specified by your PI. **(40)**

Submission Files

- Lab2.circ (contains all subcircuits)
- lab2.pdf

UF Herbert Wertheim College of Engineering
Department of Electrical & Computer Engineering
UNIVERSITY of FLORIDA

Dr. Christophe Bobda
Newman Waters
EEL3701C - Fall 2025

Rubric

Note	Points
Pre-Lab	
P1 - truth table is correct	8
P1 - mux_2x1 completed and correct	6
P2 - mux_4x1 completed correctly with 3 mux_2x1 components	6
P3A - K-maps are correct and part3a matches logic equations	8
P3B - seven_segment uses 5 8-bit 4x1 multiplexers from P2 and has correct behavior	6
P4 - part4 included	6
In-Lab	
Part 4 demo	20
Quiz	40
Total	**100**

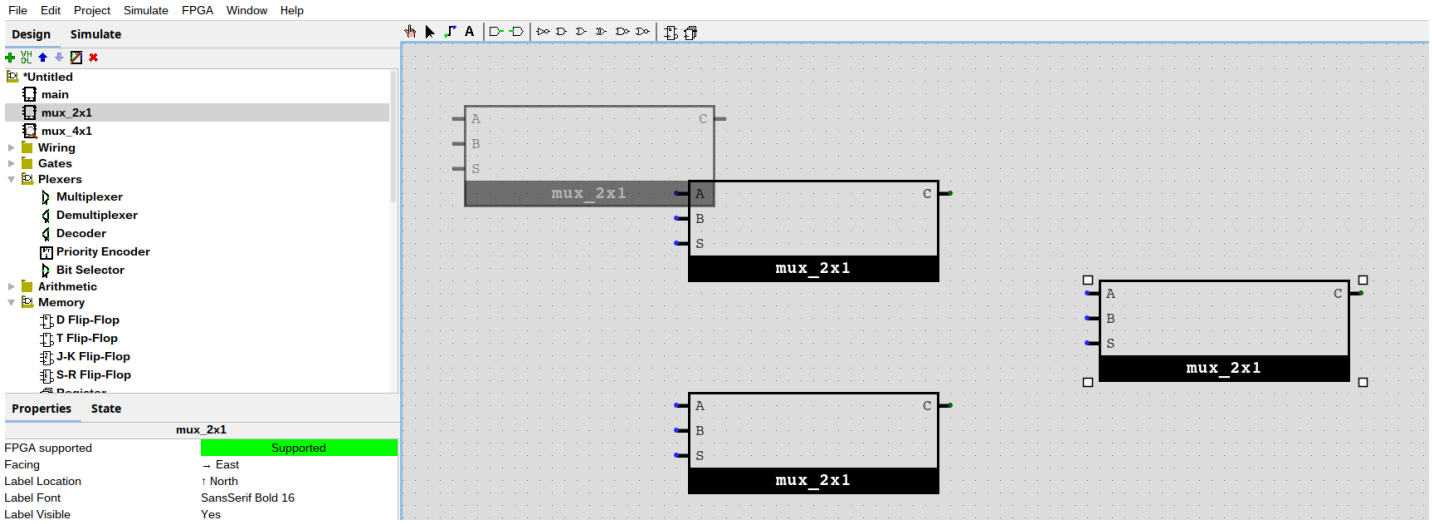
UF Herbert Wertheim College of Engineering
Department of Electrical & Computer Engineering
UNIVERSITY of FLORIDA

Dr. Christophe Bobda
Newman Waters
EEL3701C - Fall 2025

Appendix A

Placing Multiple Components in Logisim

You should click on your component in the left toolbar and you will see the component appear (slightly grayed out). You can place multiple instances of the component by repeating this process.



Appendix B

Programming DE10-Lite with Logisim

1. Before you do anything, ensure that all inputs/outputs are labeled.
2. Follow the instructions in the **Installing Logisim** section of the Software Installation Guide (integrating Logisim with Quartus/Modelsim) if you have not already.
3. Download this XML file.
4. In Logisim, go to the **FPGA** tab on the top, and select **Synthesize & Download**.
5. Under the dropdown menu for "*Target Board*", select "*Other*"
6. Find the downloaded XML file from step 2. Select it, and then use this as your target board. You only need to perform this step one time.
7. In the "Action Method" section select the desired TopLevel design, and in the drop down below that set the action to "Synthesize & Download" and then select **Execute**.
8. A pin assignment window will pop up with an Image of the DE10-Lite. Map the inputs and outputs to pins by clicking on each unmapped component to highlight it and then select the respective switch/led/pin to map to on the board. then select **Done**.
9. It will ask to confirm that your board is plugged in, and then it will program the DE10.