EECS 3201 :: Lab 2
Seven-Segment Decoder

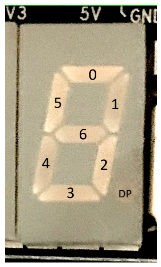
Overview

In this lab you will:

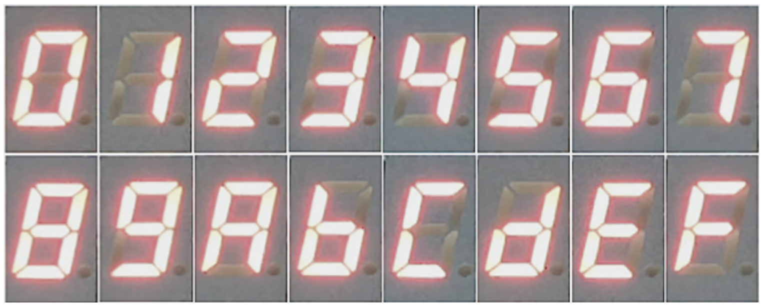
- Implement a seven-segment decoder and connect it to a seven-segment display on the DE10-lite board, and
- Create a video demonstrating your work.

Value: This lab is worth 4/20 of your total lab score.

Seven-segment displays and decoders



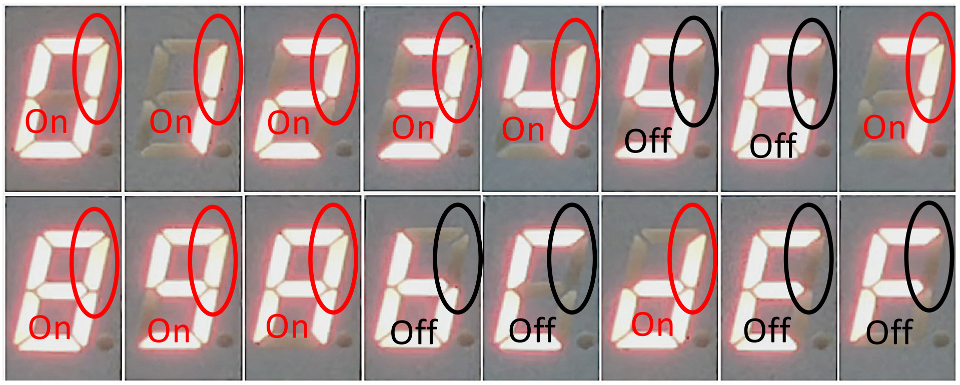
The above figure depicts a seven-segment display on the DE10-Lite, with seven LEDs (numbered 0, 1, …, 6), as well as a decimal point (DP). The LEDs can be illuminated to depict any hexadecimal number in 0,1,…,9,A,…,F. These numbers are shown below.



Each LED in the display has an associated binary function:

- The **input** to the function is a four-bit binary number, representing the hexadecimal digit 0 through F. For example, 0000 = 0, 0011 = 3, 0101 = 5, 1101 = D, etc.

- The **output** of the function represents the illumination state of the LED. For example, consider the upper right LED in the display (the LED numbered 1 in the top figure): this LED is on for digits 0, 1, 2, 3, 4, 7, 8, 9, A, and D, and off for digits 5, 6, B, C, E, and F. As an illustration of this example, see the figure below.



*An important fact about the **seven-segment displays** DE10-Lite board:*

*Logic 0 = LED **on**, and logic 1 = LED **off***

***It is the opposite for the plain LEDs** on the board, which are on for logic 1 and off for logic 0. Confusing!*

So, the function for the upper-right LED can be described on a truth table, with output 0 for the digits where it is on, and output 1 for the digits where it is off. You will need to do this for each of the 7 LEDs, designing a binary function so that the LED is on or off for each digit.

Do the following:

- Write a truth table for each of the seven LEDs in the seven-segment display.
- Generate a Boolean function for each LED using either the minterm or maxterm method, whichever you prefer.
- Write a Verilog module implementing this set of functions. You can use the Verilog module from Lab 1 as a template. The difference is that we now have four inputs and seven outputs; for each output, you can use an **assign** statement to assign the output to its function. In Verilog, the logical operators are ~ for NOT, | for OR, and & for AND. Correctly implementing the module up to this point is worth 4 out of 7 technical marks.
- For one additional technical mark (*), have your module take one input and one output, where both input and output are defined as multi-bit buses.
- For two additional technical marks (**), algebraically reduce at least one of your functions. **Fully and clearly explain your reduction in comments in the Verilog file.** For example:

```
// In minterm form, the function was originally f = ab~c + abc
// Reduction:
// f = ab~c + abc
//   = ab(~c + c)
//   = ab(1)
//   = ab
assign f = a & b;
```

- Assign the inputs to four of the switches (your choice which ones), and the outputs to one of the seven-segment displays (again, your choice). Pin labels are given in the DE10-Lite user manual.
- Upload your program to the board and demonstrate it on video.

Deliverables

- A video of yourself, stating your name and student number, followed by (without breaks/cuts) a demonstration of your seven segment decoder. (You don't have to demonstrate every digit, a few digits are fine.) **Your video should be no longer than 90 seconds in length.**
- Your Verilog module.

Submit both files via eClass.

Scoring

- **Video:** 0/3 if not submitted, 1/3 if submitted but does not conform to specification, 3/3 if submitted and conforms to specification
- **Module:** 0/7 if not submitted, 1/7 if submitted but incorrect, 4/7 if submitted and correct but (*) and (**) are missing or incorrect; an additional 1/7 if (*) is done correctly; and an additional 2/7 if (**) is done correctly.