ETE4451 FPGA Lab Dr. Boskovich

**LAB 2 / 3**

**Simple ALU with**

**Seven Segment Display Outputs**

#### Objective

* Become more familiar with using Xilinx Vivado
* Continue to Learn to use Test-Fixtures
* Develop design techniques for Decoders
* Be able to implement Multiplexor designs

#### Work to be demonstrated

* Implementation of an ALU
* Implementation of Seven Segment Hex Displays

#### ***For each part of the lab be sure to simulate each module created. Demonstrate the simulation waveform outputs that validate the design.***

#### **PART 1 - *ALU Design (Lab 2)***

1. Create a simple 4-bit ALU.

Using Continuous Assignment Behavior, design an ALU with the following features where A and B are 4-bit numbers.

1. **Adder / Subtractor.** Using 8 slide switches, 4 for A and 4 for B, a button for carry-in, and a button for mode select. When the mode select button is not asserted, the Adder/Subtractor is in the Add mode. When the button is pressed, the Adder Subtractor is in the Subtract mode. The results of the adder/subtractor are to be displayed on 5 (sum + carry) LEDs.
2. **Comparator.** Concurrent to the add and subtract operations, the ALU must also indicate if A<B, A>B, or A=B. For each condition, a separate output is used to indicate the result of the comparison. The comparator uses individual bits to determine the logic condition. For example if A is less than B then the *lt* output would be true. Similarly, if A is greater than B, then the *gt* output would be true.

**PART 2 - *Seven Segment Decoder (Lab 3)***

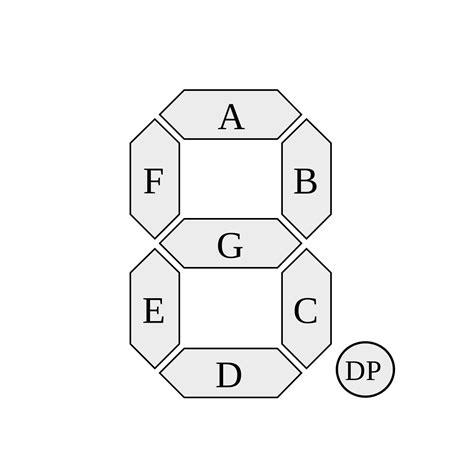
Modify the output section to be able to decode the five bits into four bits 0-F plus 1 bit where the additional 1 bit is the carry out. The four bits will be applied to a seven-segment decoder that you design as a hex value. To indicate a carry out, use one of the discrete LEDs. Continue to use the LEDs to indicate the comparison function results.

#### **PART 3 - *Multiplexor Design (Lab 3)***

#### Modify the design from Part 1s and 2 to operate upon two 8-bit numbers such that the inputs are 0-FF. To display the results as hex numbers 00-FF, choose a button to use which will display the least significant number when not pressed, and the most significant number when pressed.

***Demonstrate Part 3.***

For each part consider the inputs and outputs to be positive logic. Also use the following format to define the 7-segment display to the following bits.



Bit Segment

1. A (least significant)
2. B
3. C
4. D
5. E
6. F
7. G
8. DP (most significant)