# COE608 Computer Organization and Architectures
# Winter 2017

## Lab3: PART-2
## 8-bit ALU Implementation and Testing
## Due Date: Week 6 (Start of Lab Session)
**Programming the Cyclone IV FPGA to realize 8-bit ALU**

## 1 Objectives

In the second part of Lab 3, students will learn to implement and test an 8-bit version of the ALU designed in part-1. The 8-bit ALU will be implemented on a Cyclone IV FPGA at the DE2 board and tested by employing the I/O facilities available at the DE2 board.

## 2 Introduction

Students will be implementing a full 32-bit version of the ALU as part of the CPU design during the coming lab sessions of this course. Until then, the ALU implementation and testing is meant to verify the proper operation of an 8-bit ALU and to familiarize students with more experimentation of programming and testing VHDL designs on the DE2 board.

The DE2 board has eighteen user-controllable switches, four de-bounced push-buttons, eight seven-segment displays and twenty-seven LEDs (18 red and 9 green). Using some of these components students will test and demonstrate the operation of an 8-bit version of the ALU design. The ALU will be tested by using the switches as inputs, and 7-segment to display the two 8-bit operands as well as the 8-bit result. Moreover, two LEDs will be driven by the Carry and Zero signals from the ALU.

## 3 ALU Circuit Design

You should take the original design you prepared for Lab3 Part-1 and convert it to the 8-bit version. It can be easily achieved by changing the 32-bit data widths of various inputs and outputs to 8-bit, as well as the internal architecture of the ALU so that the circuit operates on 8-bit operands.

## 4 ALU and Test Circuit Implementation

The test circuit to be used for 8-bit ALU verification is shown in Figure 1. The ALU receives all the inputs from the 18 switches present on the DE2 board. Three switches are used for the ALU op-code, which leaves 15 switches available for the two operands. Since the operands are both 8-bits wide, where one input of SW[10] will be shared among the two operands. The ALU inputs and outputs are displayed using the 7-segment displays, while Carry and Zero signals are displayed on green LEDs 0 and 1 available at the DE2-board.

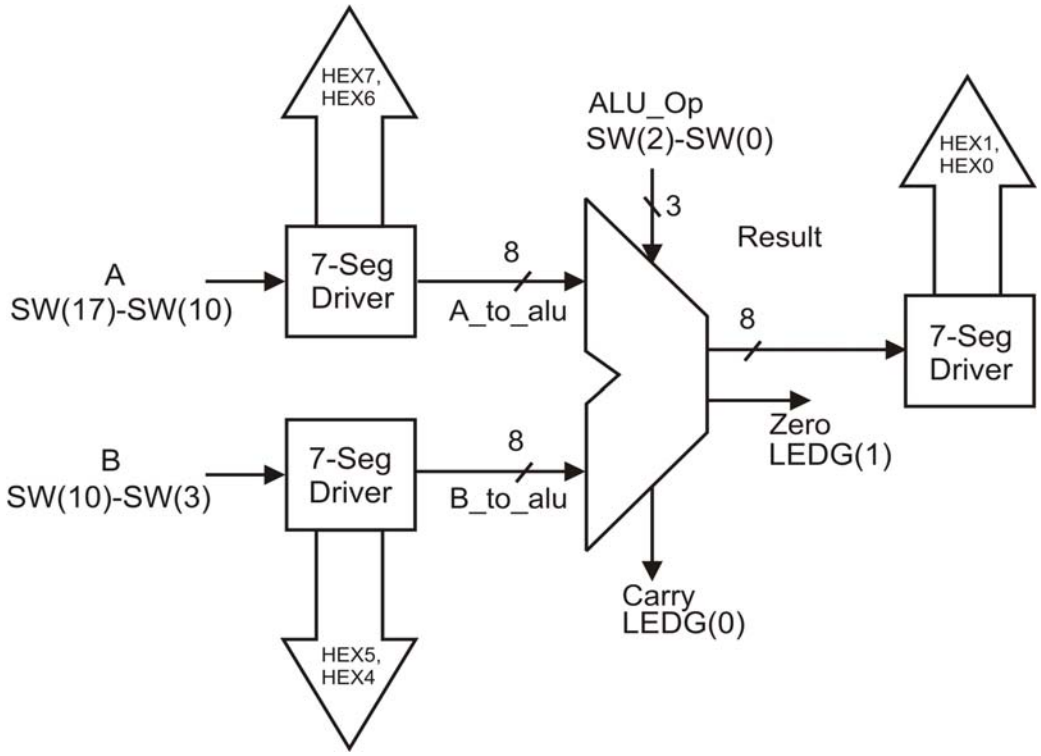


*Figure1: ALU Test Setup*

## 4.1 Top-Level VHDL Entity and 7-Segement Driver Design Files

To test the ALU, students are provided with two VHDL files which together make up a test-fixture design. One file is the seven-segment display decoder, which will simply be imported and used in the circuit as a "black-box". The 2nd file is a top-level VHDL entity file for this project, named "lab3_b.vhd". Students may need to edit this file and instantiate the 8-bit ALU inside it. All input signals of the ALU as well as port map statements for the seven-segment drivers are provided inside the file. Moreover, the input and output pin names are provided in this file. For correct operations, the board pin assignment file, "DE2_115_pin_assignment.qsf", should be imported into the project. All the files are available in the course directory /home/courses/coe608/labs/lab3b/, and should be copied to your project folder. Don't forget to start a new project and include these and your ALU file in the project.

## 4.2 Pin assignments

In order for the ALU test and design setup to operate properly, it must be physically connected to its respective inputs and outputs (Switches, 7-segment displays and LED's). This can be accomplished either through manual pin assignment (like Lab 1) or by a pre-made pin assignment. The "DE2_115_pin_assignment.qsf" file is provided in the course directory that you should copy to your project folder. Once you have edited the top-level entity file and added your ALU to the design you are ready to import the required pin assignments. This is done through the **Assignments** menu, by selecting the **Import Assignments** option, and then selecting the "DE2_115_pin_assignment.qsf" file in your project directory.

## 4.3 Seven Segment Display and Additional Outputs

The seven-segment driver design provided takes an 8-bit input and generate 7-segment signals for the displays. A copy each of the 7-segment driver is connected to input operands A, B, and the output result of your ALU at one end and to the HEX pins of 7-segments. The A and B operands will be displayed on HEX7–HEX6 and HEX5–HEX4, respectively. The ALU output result is displayed on HEX1–HEX0. The seven-segment displays on the board are named as HEX0-to-HEX7 to identify, which seven-segment display is assigned to which particular data. You may consult the DE2-board user manual for detailed information regarding all the components on the board. The Zero signal is routed to Green Led 0, and the Carry signal to Green LED 1. The sign bit of operand A is routed to Red LED 16 and that of operand B to Red LED 13; the sign bit of the output is displayed on the 7-segment display HEX2. All these connections have been made in the top-level VHDL entity file provided.

## 5. What to Hand in and What is required form this Lab

Following is required to obtain full credit for your lab session.

- You must provide a printed copy of your modified top-level and 8-bit ALU VHDL files.
- You must demonstrate the correct operation of your ALU on the DE2 board to the lab instructor.
- You must understand the VHDL files (top-level and 8-bit ALU) provided for this lab, your lab-instructor will quiz you on the test setup and the ALU design.

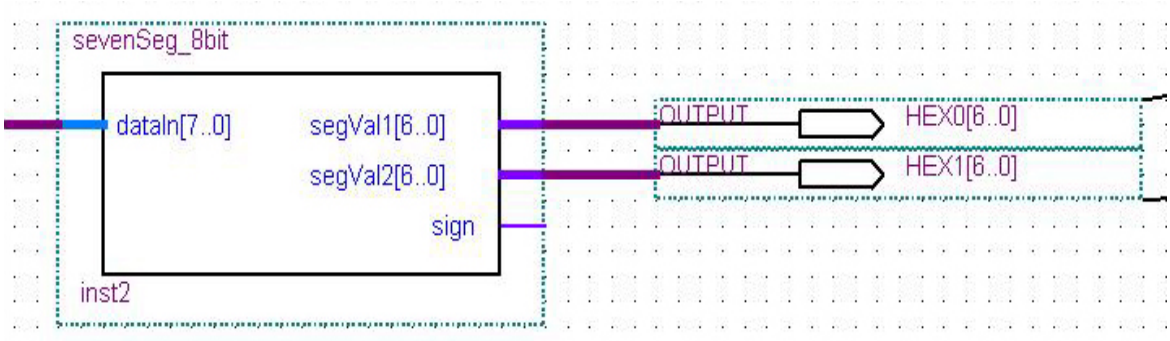


*Figure 2: 7-Segment Connection Example*

# Appendix: Seven-Segment Display Decoder

This document introduces the 7-segment decoder circuit, it's basic operation and requirements, as well as the connection methodology required to make use of the device.

## 1. Functional Description

The 7-segment decoder can be thought of as a black box which one can use to display any 8-bit 2's complement number using two seven segment displays and an additional led for the sign of operand. The number is displayed using hexadecimal notation.

The input to the circuit can be any 8-bit binary number. The circuit will treat the number as a signed number, and decode it according to the two's complement method. The output will then display the original number, along with an indicator (such as a led or portion of another 7-segment display) the sign of number.

The circuit can also be used to display unsigned numbers, but the most significant bit of such an arrangement must always be set to 0, to indicate to the circuit that no decoding should be performed.

## 2. Circuit Input and Output

The input to the circuit consists of a binary number that is to be displayed. The circuit generates two sets of 7-bit vectors, which are then used to drive two 7-segment displays. Two displays must be used to accommodate the -127 to 127 range of numbers in hexadecimal notation. Output “segVal1” displays the lower four bits of the input number, while “segVal2” displays the upper four bits eluding the sign-bit. In addition, the circuit will also generate an indicator signal for the sign of the number.

Since the circuit is built specifically for the DE2 board 7-segment displays, active-low devices are assumed to be used. If this is not the case, then all output signals must be inverted. As well, each output signal corresponds directly to the segment with the corresponding index (meaning that segVal1(0) corresponds to segment 0 in a standard display). In the case of the sign signal, the signal will be low in case of positive numbers and high in the case of negative numbers (if an active low device is used to display the sign, this output should be inverted).

## Connection Example

Figure 2 shows an example of the device displaying an unsigned number (for this reason the sign bit is left disconnected). The two 7-bit patterns are connected to the HEX0 and HEX1 outputs as specified in the “DE2_115_pin_assignment.qsf” file.