# LAB Assignment #6 for ECE 443

Assigned: Wed., Oct 9, 2009
Due: Mon., Oct 14, 2009

## Description: Prepare for a demo of last week's lab using my code as driver

You should download the VHDL code that I've put on-line and run it first without modifying it. Make note of the WARNING messages you get during the synthesis so that when you later modify it to include your code, you will know which ones to ignore. You should observe the following behavior.

A) Set hyperterminal set to 38400, N81 and connect your USB to serial cable from your computer to the FPGA board.
B) Upload the synthesized design to the FPGA.
C) Reset the board after the upload by moving 'switch 3' (left most switch) from the up position to the down position and then back to the up position -- leave it in the up position.
D) Press the 'enter' pushbutton -- you should see "+0000" printed on the screen each time you press it. If you don't something is wrong with your serial connection.
E) To enter values for the two operands, do the following: Type up to 4 digits (value MUST be in range of -2048 to 2047) and hit 'enter' on the keyboard, type up to 4 more digits and hit enter again. If you then press the 'enter' pushbutton on the FPGA, the sum will be printed to the screen (this is the unmodified default behavior). If you enter another (third) number and press 'enter' on the keyboard, this new value over-writes the first value you entered. Each time you enter a value, it over-writes one of the values in the pair in a circular fashion.

Once you have tried this and confirmed that it works, you need to modify the main VHDL module (UARTNumberConvert.vhd). First create an instance of your ALU. Then find the line that says '-- ADD YOUR ALU HERE'. Connect your input operands A and B to the signals 'A_op_reg' and 'B_op_reg' and your output to 'result_binary_val'. Connect your xxx_ins signals as shown below. Connect your skw_success signal to one of the LEDs (you can remove any assignments that I have made to the LEDs).

add_ins: NONE buttons pressed -- just press Enter
sub_ins: up_pb_level
inv_ins: down_pb_level
and_ins: left_pb_level
or_ins: right_pb_level
xor_ins: up_pb_level AND down_pb_level
shltf_ins: up_pb_level AND left_pb_level
shrtf_ins: up_pb_level AND right_pb_level
skwlf_ins: down_pb_level AND left_pb_level
skwgf_ins: down_pb_level AND right_pb_level
skwnf_ins: left_pb_level AND right_pb_level
skwef_ins: up_pb_level AND down_pb_level AND left_pb_level
movf_ins: up_pb_level AND down_pb_level AND right_pb_level
movlw_ins: up_pb_level AND left_pb_level AND right_pb_level

Bear in mind that my code does not check for all possible error conditions. For example, if you enter a value smaller than -2048 or larger the 2047, your results will be difficult to interpret because of the overflow that occurred. Also, I do not enforce that you type only 4 characters before hitting enter on the keyboard. If you start noticing strange behavior, use the reset as described above. Report any bugs to me as soon as possible.

THERE IS NO WRITTEN LABORATORY REQUIREMENT
Grading:
As indicated last week, your lab grade will consist of two parts. The first part is associated with the in-class demo that you will do this week, and is worth 50% of the total grade (50 pts). Successful demonstration of the lab's stated requirements is worth 50 pts. Partial implementations will be given only partial credit. The second portion you have already turned in.