LAB Assignment #5 for ECE 443

Assigned: Wed., Sept 30, 2009
Due: Mon., Oct 7, 2009

Description: Implement the remaining components of the ALU for the micro-controllers.

You built the adder portion of the ALU last lab. This lab will focus on building the remaining components of the ALU, including a Boolean function unit and shifter. The input/output signals to the entire ALU are given as follows:

Input:

- A, B: input buses (12 bits wide)
- inv_ins: Invert **B** bits
- and_ins: AND A and B bits
- or_ins: OR A and B bits
- xor_ins: XOR A and B bits
- add_ins: Add A and B
- sub_ins: Subtract B from A (A-B)
- shltf_ins: Shift **B** left by 1 bit
- shrtf_ins: Shift **B** right by 1 bit
- skwlf_ins: set *skw_success* to 1 if register W is less than register F
- skwgf_ins: set *skw_success* to 1 if register W is greater than register F
- skwnf_ins: set *skw_success* to 1 if register W is NOT equal to register F
- skwef_ins: set *ske_success* to 1 if register W is equal to register F
- movf_ins: pass the B operand to the output unchanged
- movlw_ins: pass the B operand to the output unchanged

Output:

- DEST_OUT: output bus (12 bits wide)
- skw_success: Boolean (carry-out bit of adder)

Last week, you wrote the AddUnit which took parameters (parameter *skw1* is explained below):

ADD_OUT, A, B, skw1, skwlf_ins, skwgf_ins, sub_ins, add_ins

For this lab, you will need to build the Boolean function unit and Shifter unit and a parent (ALU) that combines all three modules. The Boolean function unit uses the following parameters (first two are output parameters, others are input):

BOOL_OUT, skw2, A, B, and_ins, or_ins, inv_ins, xor_ins, skwnf_ins, skwef_ins

The operation of the Boolean function unit is straightforward for the *and_ins*, *or_ins*, *inv_ins*, and *xor_ins* instructions -- simply carry out the logic operation on the A and B operands in a bitwise fashion. Note that operands A and B are used for the binary logical instructions while ONLY operand B is used for the *inv_ins* instruction. Use a tri-state to select the logical operation that drives the output bus BOOL_OUT. For the *skwnf_ins* (not equal) and *skwef_ins* (equal), you should XOR the A and B operands and use a reduction OR or NOR (or both) to decide if any of the 12 bits are zero or if they are all zero (dependent on the operation). You can assume that only one of the *xxx_ins* will be 1 and the others will be 0.

The parameters of the Shifter Unit are (only the first parameter is an output -- others parameters are inputs):

SHIFT_OUT, B, shltf_ins, shrtf_ins, movf_ins, movlw_ins

Only operand B is needed as input. SHIFT_OUT is the output bus. If *shlft_ins* is 1, then left shift operand B by 1 bit (shifting a '0' in as the new rightmost bit). If *shrtf_ins* is 1, then right shift operand B preserving the sign bit. If either *movf_ins* or *movlw_ins* are 1, then pass the B operand through with NO shifting. Use a tri-state to select the appropriate operation that drives the output bus SHIFT_OUT. You can assume that only one of the *xxx_ins* will be 1 and the others will be 0.

In the main module (ALU), you should tri-state the appropriate bus, either ADD_OUT, BOOL_OUT or SHIFT_OUT, to the output bus DEST_OUT based on the *xxx_ins* that is set to 1. You can assume that only one of the *xxx_ins* will be 1 and the others will be 0. You should OR *skw1* and *skw2* to create *skw_success*.

Laboratory Report Requirements:

1) Turn in a commented copy of your VHDL code along with a schematic.
2) Write a test bench and run simulation(s) that shows the inputs and output behavior of the circuit (be sure to include a set of 'sample' waveforms in your report).
3) Be prepared to give a demonstration in class on Wed. I will provide the UART driver code to enable you to type values into hyperterminal that will be delivered to the FGPA and converted into numbers. You will enter a decimal number for operand A, hit return, and enter a second number of operand B, hit return, and then you will press one of the pushbutton (or combinations of puttons) to carry out an operation using your ALU. The result of the operation will be automatically displayed in the hyperterminal screen. I will include comments in the code on how to use the driver..

Grading:

Your lab grade will consist of two parts. The first part is associated with the in-class demo, 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 of the lab grade is derived from your lab report. Correct implementation counts for 15 pts (of the 50 pts). Well documented simulation results are also worth 15 pts. The remaining 20 pts will be given according to how well the VHDL code is written and documented (comments). Bonus points will be given to any implementation feature that goes above and beyond the requirements.