CSE 331 Computer Organization
MIPS Processor Design
Project 2: 32-bit ALU Implementation

Due Date: 23:59 November 19 (Tuesday), 2013
(Later submissions will not be accepted)

In the next projects for CSE 331, you will design and implement a subset of the MIPS32 architecture in Xilinx ISE using Verilog. The goal of these projects is to move you to building complex, general-purpose CPUs. Read this document entirely, paying special attention to the section "Rules". The MIPS ALU (arithmetic and logic unit) performs all of the core computations dictated by the assembly language.
We do not describe all ALU functions in full but will explain the crucial ones as below:

Full Adder: C = A + B + cin
Input: A[32], B[32], cin
Output: R[32], cout
The output *R* is computed by adding *A*, *B*, and *cin*. Any remaining carry bit is output on *cout*.

Left Shifter: C = (B << sa)
Input: B[32], sa[5]
Output: R[32]
The output *R* is computed by shifting *B* to the left *sa* bits. The shift amount *sa* can be anything from 0 to 31, encoded as an unsigned integer.

ALU: R = f(A, B, sa)
Input: A[32], B[32], op[4], sa[5]
Output: R[32]
The function is selected according the value of the *op* input:

op	***R***	***Function name***
0000	R = B >> sa	shift right logical
0001	R = B >>> sa	shift right arithmetic
001x	R = B << sa	shift left logical
010x	R = A & B	and
011x	R = A | B	or
100x	R = ~(A | B)	nor
101x	R = A ^ B	xor
110x	R = A + B	add
111x	R = A - B	subtract

Table 1: Functions your ALU should be able to perform

Note the difference between logical right shift (which fills with zero bits), and arithmetic right shift (which fills the empty bits with the copies of the sign bit of B).

Rules (very crucial and all-important):

1. **Submit a single zip file to Moodle containing all of your Xilinx ALU project files such that your TA could be able to open your project and simulate in Xilinx ISE.**
2. **Name your zip file as "YourID_Name_Surname_Project2.zip" otherwise you will lose 5 points.**
3. **You not only design the desired ALU but also have to simulate it with Xilinx ISE. The simulation files should also be included in the zip file of course.**
4. **You will put a txt file named "FunctionalReport.txt" into the zip file, in which you explain which functions in Table 1 work accurately and which ones wrongly or even do not work. Your TA will check your ALU and if the txt file and the simulation of TA matches with each other, you will get extra 10 points otherwise if the information you gave does not match with the reality you get -15 points.**
5. **Comment your Verilog files clearly (You may comment in Turkish). Otherwise you will lose 30 pts.**
6. **Your design will be similar to the described in the Friday lecture given by your instructor Alp.**
7. **You have to use structural Verilog as explained in the PS hours. Data flow or behavioral Verilog are not allowed. Otherwise you will lose between 20-40 pts.**
8. **You have to use one adder for both addition and subtraction.**
9. **You will not get much partial credits. Submit fully working simulating designs for credit.**

For the Adventurous and For Bonus Points:

Note: These suggestions for an extra challenge will be examined (and commented on, if your project works well) and they will have impact on the project grades only if your project runs smoothly.

- Implement a carry-lookahead adder (15 pts)
- Use the same block for 3 different shift instructions (10 pts)

Deviation from the MIPS Standard:

You can ignore any MIPS instruction or feature not mentioned in this document.
In addition:
- Ignore overflow conditions
- Assume traps or exceptions will not occur (i.e., do not implement them).
- Do not implement floating point, multiply/divide, or any other arithmetic
used by the MIPS processor.