Lab Project 1: 32-Bit ALU

Revision: January 23, 2019

CECS 440 – CSULB

STUDENT

On my honor, this is my own work; I understand severe penalties will be assessed if I submit work for credit that is not my own.

Print Name _____ **ID Number** _____

Signature _____ **Date** _____

Estimated Work Hours

1	2	3	4	5	6	7	8	9	10

5%will be deducted from total score for each day late

Point Scale

4: Exemplary
3: Complete
2: Incomplete
1: Minor effort
0: Not submitted

Score = Points awarded (Pts) x Weight (Wt)

GRADER

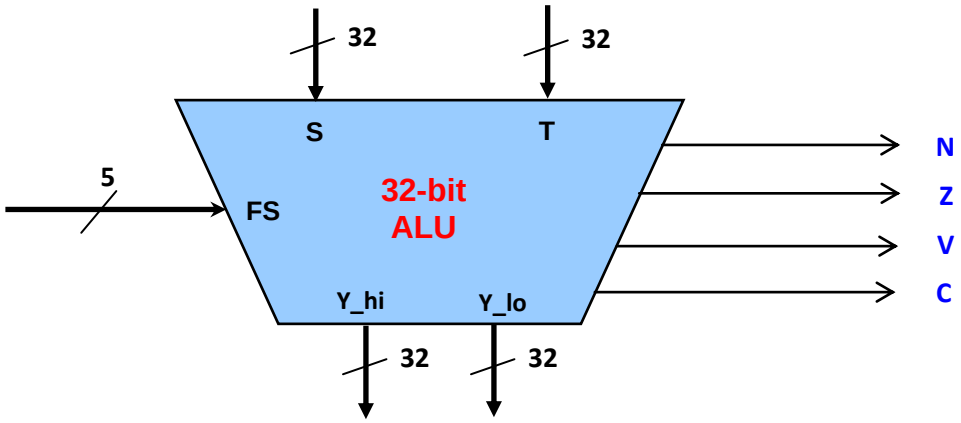
Deliverables / Test / Verification	Wt	Pts	Score	Grader Signature	Date	Days Late
Documentation / Source Code	2					
Documentation / Testbench	1					
Simulation/Verification & Results	2					

Total In-Lab Score

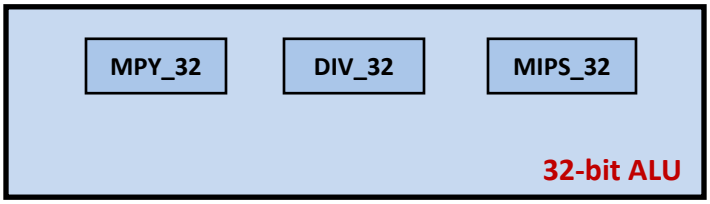
General Statement: You are to write a **"behavioral"** Verilog module for a 32-bit ALU where inputs **S** and **T** are 32-bit values. **ADDU**, **SUBU** and **SLTU** are "unsigned" operations; all other arithmetic operations are "signed." The ALU has the following operations (with respective 5-bit hex "function select," i.e. **FS**):

Arithmetic		Logic		Other	
PASS_S (Y_lo = S)	5'h00	AND (Y_lo = S & T)	5'h08	INC (Y_lo = S + 1)	5'h0F
PASS_T (Y_lo = T)	5'h01	OR (Y_lo = S | T)	5'h09	INC4 (Y_lo = S + 4)	5'h10
ADD (Y_lo = S + T)	5'h02	XOR (Y_lo = S xor T)	5'h0A	DEC (Y_lo = S - 1)	5'h11
ADDU (Y_lo = S - T)	5'h03	NOR (Y_lo = ~[S | T])	5'h0B	DEC4 (Y_lo = S - 4)	5'h12
SUB (Y_lo = S + T)	5'h04	SRL (Y_lo = logical T > 1)	5'h0C	ZEROS (Y_lo = 32'h0)	5'h13
SUBU (Y_lo = S - T)	5'h05	SRA (Y_lo = arithmetic T > 1)	5'h0D	ONES (Y_lo = 32'hFFFFFFFF)	5'h14
SLT if (S < T) Y_lo = 1 else Y_lo=0	5'h06	SLL (Y_lo = logical T < 1)	5'h0E	SP_INIT (Y_lo = 32'h3FC)	5'h15
SLTU if (S < T) Y_lo = 1 else Y_lo=0	5'h07	ANDI (Y_lo = S & { 16'h0, T[15:0] })	5'h16		
MUL (Y_hi, Y_lo = S * T)	5'h1E	ORI (Y_lo = S | { 16'h0, T[15:0] })	5'h17		
DIV (Y_hi, Y_lo = S / T)	5'h1F	LUI (Y_lo = { T[15:0], 16'h0 })	5'h18		
		XORI (Y_lo = S ^ { 16'h0, T[15:0] })	5'h19		

Note the individual status outputs to indicate a result that was negative (**N**), zero (**Z**), overflow (**V**) or carry (**C**).



The 5-bit "Function Select" will select which ALU operation is to be performed. Also, note that we will be enhancing the capabilities of this ALU in the very near future, thereby using more combinations of the opcode. Both multiplication and division are to be "signed," and are to be implemented by **separate** verilog modules of a 32-bit multiplier and a 32-bit divider. The multiplication algorithm yields a 64-bit product {**Y_hi,Y_lo**}. The division algorithm yields a 32-bit quotient (**Y_lo**) and a 32-bit remainder (**Y_hi**). The remainder of the operations are to be implemented in a third verilog modules that yields a 32-bit result (**Y_lo**) where **ALU_hi** will be set to all 0's. Thus, the actual **32-bit ALU** module is a "wrapper" that instantiates the three modules and chooses the correct outputs for **Y_hi**, **Y_lo** and {**C**, **V**, **N**, **Z**} based upon the value of "**FS**."



Status Flag Considerations:

Since this is an "integer ALU," the status outputs are dependent upon the ALU Opcode. The following summarizes your responsibility to update the particular flags based upon the ALU operations.

Alu Operations	Status Flags Affected
add, sub, addu, subu, inc, inc4, dec, dec4	C, N, V, Z
srl, sra, sll	C, N, Z
pass, slt, sltu, mul, div, lui, all "logic," zeros, ones, sp_init	N, Z

When designing your ALU, which is combinational logic, all of the outputs (**Y, C, V, N, and Z**) are a function of all the inputs (**S, T, and FS**). Thus, for any change of any of the inputs, all of the outputs are to be changed appropriately! If a status flag output is not affected by the ALU operation, you are to set its value to "**X.**"

Deliverables: You must turn in **(1)** the first two pages of this document, filled in appropriately, followed by **(2)** a printout of your verilog module for the 32-bit ALU (e.g. **alu_32.v**), followed by **(3)** printouts of the 3 modules that make up the 32-bit ALU (e.g. **MIPS_32.v**, **DIV_32.v** and **MPY_32.v**), followed by **(4)** a printout of the verilog file for the testbench (i.e. **MIPS_ALU32_TB.v**), followed by **(5)** a professionally formatted printout of the Xilinx "log" file of the simulation that verifies all the operations of the ALU (i.e. **isim.log**).

Grading: At this point in your educational experience, we are very serious about your development of habits and practices involved with submission of "Deliverables" to our customer—documentation which has your name on it, and for which you are accountable. With this heart, and in this spirit, will your lab assignments be graded. You will **lose points** for each violation related to inputs, outputs, and operation specifications. You will **lose points** for not adhering to documentation specifications of the "deliverables." You will **lose points** for "substandard" documentation practices (i.e. uncommented logic, meaningless comments, "wrap-around" in print outs, illegible documentation, etc.). You will **lose points** for failing to verify the correctness of your testbench results, especially "delivering to the customer" results in which there are verification errors.

Due Date: Monday, February 4, 2019 **{Monday of Week 3}**