Lab 4
(Full Score: 100 points)

(**Due by 4/14/Thursday at at Moodle**)

Your name:	Score:

Design and Implementation of an 4-bit ALU (Arithmetic and Logic Unit)

Goal

- To design and implement an 4-bit ALU.
- To design and implement the logic circuits for generating 4 status bits, Z, N, V, C.
- To learn how to work as a team to work on a project

Objectives

- To design a 4-bit ALU module based on a set of given requirements
- To implement the 4-bit ALU design using Altera's Quartus II CAD software, including design entry, compilation, simulation and download to DE2 board

Detailed Requirements

I. Building a 4-bit ALU module

Design the 4-bit ALU module to implement the following function table.

f_2	f_1	f_0	**Function**	**Meaning**
0	0	0	F = OPA	transfer
0	0	1	F = - OPA	negate
0	1	0	F = OPA $\wedge$ OPB	logic AND
0	1	1	F = OPA $\oplus$ OPB	logic XOR
1	0	0	F = OPA + 1	increment
1	0	1	F = OPA - 1	decrement
1	1	0	F = OPA + OPB	add
1	1	1	F = OPA - OPB	subtract

- $f_2f_1f_0$ represents a 3-bit function select code with code values defined in the table
- OPA = $OPA_3OPA_2OPA_1OPA_0$ and OPB = $OPB_3OPB_2OPB_1OPB_0$ are two 4-bit signed numbers (for arithmetic operations) or unsigned numbers (for logic operations) to the inputs of the ALU
- The outputs of the ALU should be F = $F_3F_2F_1F_0$
- The ALU should 4 status outputs, N, V, Z, C with their values defined as:
  N = 0/1 – Non-Negative Result/Negative Result
  V = 0/1 – No Overflow/Overflow
  Z = 0/1 – Non-Zero/Zero
  C = 0/1 – Carry of 0 /Carry of 1
- You **must use the following PIN assignment** table for the inputs and outputs of your circuits. I'll use this table to test your circuit. If you ***don't use*** this table in the PIN assignments in your circuit, your implementation will ***have NO chance*** to pass the testing even though you have correctly designed/implemented the project, which means a ***huge deduction*** of your credits!

INPUTS	X_3	**SW[11],PIN_P1**
	X_2	**SW[10],PIN_N1**
	X_1	**SW[9],PIN_A13**
	X_0	**SW[8],PIN_B13**
	Y_3	**SW[7],PIN_C13**
	Y_2	**SW[6],PIN_AC13**
	Y_1	**SW[5],PIN_AD13**
	Y_0	**SW[4],PIN_AF14**
	S_3	**SW[3],PIN_AE14**
	S_2	**SW[2],PIN_P25**
	S_1	**SW[1],PIN_N26**
	S_0	**SW[0],PIN_N25**
OUTPUTS	G_3	**LEDG[7],PIN_Y18**
	G_2	**LEDG[6],PIN_AA20**
	G_1	**LEDG[5],PIN_U17**
	G_0	**LEDG[4],PIN_U18**
	Z	**LEDG[3],PIN_V18**
	N	**LEDG[2],PIN_W19**
	V	**LEDG[1],PIN_AF22**
	C	**LEDG[0],PIN_AE22**

We learned the two options in the design of a 4-bit ALU (1) by using a 4-bit adder and (2) by customizing the 74181 module. You have may choose either one for your design and implementation.

II. Test the 4-bit ALU Module

Perform a complete functional simulation of your 4-bit ALU module. To do so, you must develop a simulation plan that should cover as many typical cases and as many special cases as possible. Using the following table to test your implementation (you may need to design more test cases).

ALU Control	**OPA input**	**OPB input**	**ALU output**	**Status Bits**
f2 f1 f0	**OPA_3, OPA_2, OPA_1, OPA_0**	**OPB_3, OPB_2, OPB_1, OPB_0**	**F_3, F_2, F_1, F_0**	**N V Z C**
0 0 0	0 0 1 1	1 0 0 0	0 0 1 1	0 0 0 0
0 0 1	0 0 1 1	1 0 0 0	1 1 0 1	1 0 0 0
0 1 0	0 0 1 1	1 0 0 0	0 0 0 0	0 0 1 0
0 1 1	0 0 1 1	1 0 0 0	1 0 1 1	1 0 0 0
1 0 0	0 0 1 1	1 0 0 0	0 1 0 0	0 0 0 0
1 0 1	0 0 1 1	1 0 0 0	0 0 1 0	0 0 0 1
1 1 0	0 0 1 1	1 0 0 0	1 0 1 1	1 0 0 0
1 1 1	0 0 1 1	1 0 0 0	1 0 1 1	1 1 0 0
0 0 0	1 1 0 1	0 1 1 1	1 1 0 1	1 0 0 0
0 0 1	1 1 0 1	0 1 1 1	0 0 1 1	0 0 0 0
0 1 0	1 0 0 0	0 1 1 1	0 0 0 0	0 0 1 0
0 1 1	0 1 1 1	0 1 1 1	0 0 0 0	0 0 1 0
1 0 0	1 1 1 1	0 1 1 1	0 0 0 0	0 0 1 1
1 0 1	0 0 0 1	0 1 1 1	0 0 0 0	0 0 1 1
1 1 0	0 1 0 1	0 1 1 1	1 1 0 0	1 1 0 0
1 1 1	1 1 0 1	0 1 1 1	0 1 1 0	0 1 0 1

This is a **group project**. A team of **2 members** work together and only one submission is required!

- **But** if you have completed the lab **individually**, you will receive ***up to 15%*** extra points.

Submissions:

- Your COMPLETE QuartusII project (compresses in a zip file).
- (Submitted to Moodle by the due time).