Computer Architecture Lab #2

Objectives

After this lab, the student should:

- Understand VHDL basics:
  - Process
  - Multiple Architectures for same entity
  - Generic Entities
  - Use of For..Generate & if…Generate
  - Understand When..Else VS If…Else Vs Case..When Vs With..Select

Requirements

Design a 16 bit ALSU that accepts two 16 bit input values A and B and provides output F, the ALSU has 4 selection inputs S3, S2, S1, S0 and Cin input. The ALSU provides a total of 20 operations specified in the following table

	S_3	S_2	S_1	S_0	Cin =0	Cin = 1
Part A	0	0	0	0	F = A	F = A+1
	0	0	0	1	F = A + B	F = A+ B+1
	0	0	1	0	F = A-B-1	F = A-B
	0	0	1	1	F= A- 1	F = 0
Part B	0	1	0	0	F = A and B	
	0	1	0	1	F = A or B	
	0	1	1	0	F = A xor B	
	0	1	1	1	F = Not A	
Part C	1	0	0	0	F=Logic shift right A	
	1	0	0	1	F=Rotate right A	
	1	0	1	0	F=Rotate right A with Carry	
	1	0	1	1	F=Arithmatic shift right A	
Part D	1	1	0	0	F=Logic shift left A	
	1	1	0	1	F=Rotate left A	
	1	1	1	0	F=Rotate left A with Carry	
	1	1	1	1	F = 00000000000	

Deliverables :

1. Write vhdl code for part A in a separate vhdl files (don't forget to output the carry out)
2. You should use the full-adder given in the explanation instead of the vhdl operators (+) and (-)
3. Add to part C & D a cout (carry out : output)
4. Integrate the 4 parts in one file.
5. Compile, your Code should be free of errors and warnings.
6. Simulate the integrated file.
7. Bonus : optimized design (hint: you can use one full adder for part A)

.

N.B. you will be graded for code neatness and understanding , Goodluck

To test part A use the following table

Operation	A	B	Cin	Cout	F
F = A	0F0F	-	0	0	0F0F
F = A + B	0F0F	0001	0	0	0F10
	FFFF	0001	0	1	0000
F = A-B-1	FFFF	0001	0	1	FFFD
F= A- 1	FFFF	-	0	1	FFFE
F = A+1	0F0E	-	1	0	0F0F
F = A+ B+1	FFFF	0001	1	1	0001
F = A-B	0F0F	0001	1	1	0F0E
F = 0	-	-	1	0	0000

To test the previous parts use the following table

Operation	A	B	F	Operation	A	Cin	F
AND	0F0F	000A	000A	F=Rotate right A with Carry	0F0F	0	0787
OR	0F0F	000A	0F0F	F=Rotate right A with Carry	0F0F	1	8787
XOR	0F0F	000A	0F05	F=Logic shift left A	0F0F	_	1E1E
NOT	0F0F	_	F0F0	F=Rotate left A	F0F0	_	E1E1
F=Logic shift right A	0F0F	_	0787	F=Rotate left A with Carry	F0F0	0	E1E0
F=Rotate right A	0F0F	_	8787	F=Rotate left A with Carry	F0F0	1	E1E1
				F=Arithmatic shift right A	F0F0	_	F878