CprE 381, Computer Organization and Assembly-Level Programming, Spring 2019

Lab 2 Report

Student Name	Sean Gordon		
Section/Lab Time	C/10:00 a.m		

Refer to the highlighted language in the lab 2 instruction for the context of the following questions.

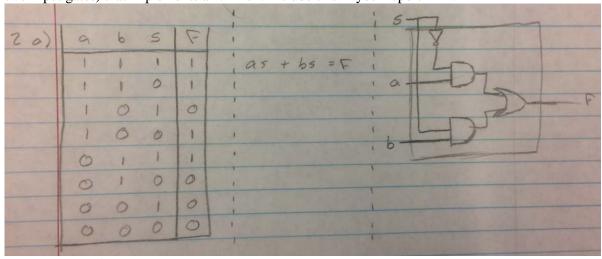
a. [Part 0] At the end of Chapter 3, answer questions 4b), 5a), and 6.

Prelab Z	1/15/10A Sen Cord
46) entiry syst is	
gorr (
inputew: in 8td-logic;	
a - data : in Std. logic - vector (7 downlo 0);	
5- data: in std-lyc- vector (7 downto 0);	
clk in std.logic;	
date 4 : ON Std_logic_vector (7 Janua 0);	
dat-5; out std-legic_vector(2 downso 0));	
end 596 2;	
50) 8	
bun_4 /8 8	
bun-10 to to to a	
burne + ckt-c 3 reg-b	
170	
100	
6) a) semicolon on line 5 misplaced, before closing pe	rendeses
rather than after	
	THE STATE OF
b) Missing and paranthese at the end of line 3 to	for the
seniculan	

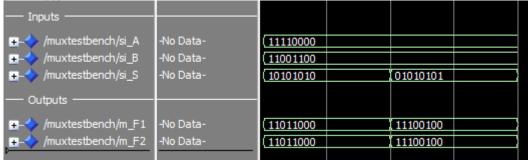
b. [Part 1 (c)] Waveform.

Inputs									
 → /octestbench/s_A	32'hFFFFFFFF	32'hFFFFF	FFF	32'h00000	000	32'h0000F	FFF	32'hFFFF0	000
Outputs									
+> /octestbench/s_F1	32'h00000000	32'h00000	000	32'hFFFFF	FFF	32'hFFFF0	000	32'h0000F	FFF
II - ◇ /octestbench/s_F2	32'h00000000	32'h00000	000	32'hFFFFF	FFF	32'hFFFF0	000	32'h0000F	FFF

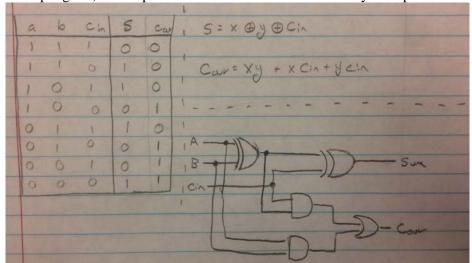
c. [Part 2 (a)] Draw the truth table, Boolean equation, and Boolean circuit equivalent (using only two-input gates) that implements a 2:1 mux. Include this in your report PDF.



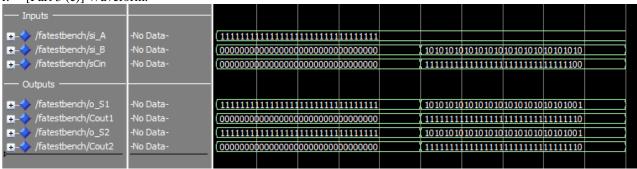
d. [Part 2 (e)] Waveform.



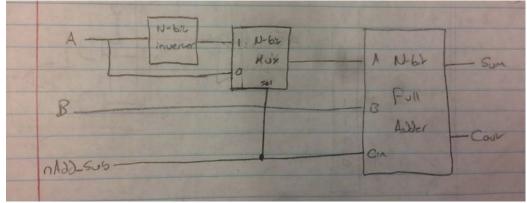
e. [Part 3 (a)] Draw the truth table, Boolean equation, and Boolean circuit equivalent (using only two-input gates) that implements a full adder. Include this in your report PDF.



f. [Part 3 (e)] Waveform.



g. [Part 4 (a)] Draw a schematic (don't use a schematic capture tool) showing how an N-bit adder/subtractor with control can be implemented using <u>only</u> the three main components designed in problems 1), 2), and 3) (the N-bit inverter, N-bit 2:1 mux, and N-bit adder). How is the 'nAdd Sub' bit used?



The nAdd_Sub bit is used to decide between adding and subtracting, using twos complement to achieve the latter. It is fed into the multiplexor as a select bit and into the full adder as a carry in.

h. [Part 4 (c)] Provide multiple waveform screenshots in your write-up to confirm that this component is working correctly. What test-cases did you include and why?

Inputs	ipolient is working correct	iy. What test-ease	s ara you merade and wily.
Compute	— Inputs ———		
Compute	+-4 /addsubnbit/i_A	-No Data-	(11111111111111111000000000000000000000
Internals	→ /addsubnbit/i_B	-No Data-	
Internals		-No Data-	
Addsubnbit/s_mux	- "		
Outputs Outputs Inputs Inputs Addsubnbit/is a	Internals		
Outputs	± - ♦ /addsubnbit/s_inv	-No Data-	(00000000000000011111111111111111111111
/addsubnbit/o_S //addsubnbit/Cout Inputs Inputs Inputs Inputs Inputs Inputs Inputs Inputs Internals Internals Inputs Inputs Internals Inputs Inputs Internals Internals Inputs Internals Inte	+> /addsubnbit/s_mux	-No Data-	(000011110000111111111000011110000
Inputs	Outputs		
Inputs	- /addsubobit/o S	-No Data-	(00000000111111111111111111111100000000
Addsubnbit/i_A	- Jadasabribit/Cout	140 Data	\(\frac{11111111000000001111000011110000}{111111111111111111111111111111111111
Addsubnbit/i_A			
Addsubnbit/i_A	— Inpute —		
Addsubnbit/nadd			
Internals -No Data- (000000000000000000000000000000000000	— 		
Internals Internals (000000000000000000000000000000000000			
→	+- / /addsubnbit/nAdd	-No Data-	(00000000000000000000000000000000000000
+ → /addsubnbit/s_mux -No Data- (111111111111111111111111111111111111	Internals		
+ → /addsubnbit/s_mux -No Data- (111111111111111111111111111111111111	Addenthabit/s inv	-No Data-	(00000000000000001111111111111111111111
- Outputs -			
		110 2010	\(111111111111111111111111111111111111
Inputs	Outputs		
— Inputs	+-4 /addsubnbit/o_S	-No Data-	(0000000011111111111111111100000000
Addsubnbit/i_A	💶 🔷 /addsubnbit/Cout	-No Data-	(11111111000000000000000000000000000000
Addsubnbit/i_A	v		
Addsubnbit/i_A	Inputs		
Addsubnbit/i_B		No Data	(11111111111111111111111111111111111111
— Internals			
	- /audsubribit/HAdd	140 Data*	
	Internals		
	+	-No Data-	(00000000000000001111111111111111111111
— Outputs — — — — — — — — — — — — — — — — — — —			
/addsubnbit/o_S -No Data	_ •		
	Outputs —		
		-No Data-	
		-No Data-	1111111110000000011111111111111111111

I included a case alternating the nAdd_Sub bit, one where the bit is all 0, and one where it is all 1. I felt this was an apt enough test of the performance.

- i. [Feedback] You must complete this section for your lab to be graded. Write down the first response you think of; I expect it to take roughly 5 minutes (do not take more than 10 minutes).
 - a. How many hours did you spend on this lab?

Task	During lab time	Outside of lab time
Reading lab	30 min	10 min
Pencil/paper design	0 min	10 min
VHDL design	60 min	30 min
Assembly coding	0 min	0 min
Simulation	10 min	20 min
Debugging	20 min	70 min
Report writing	0 min	20 min
Other:		
Total	120 min	160 min

- b. If you could change one thing about the lab experience, what would it be? Why? That you have to write about your experience at the end. I have much more important things to do, and unless this class is one in a million, the metrics I list here will be disregarded in two or three years' time.
- c. What was the most interesting part of the lab? Going home