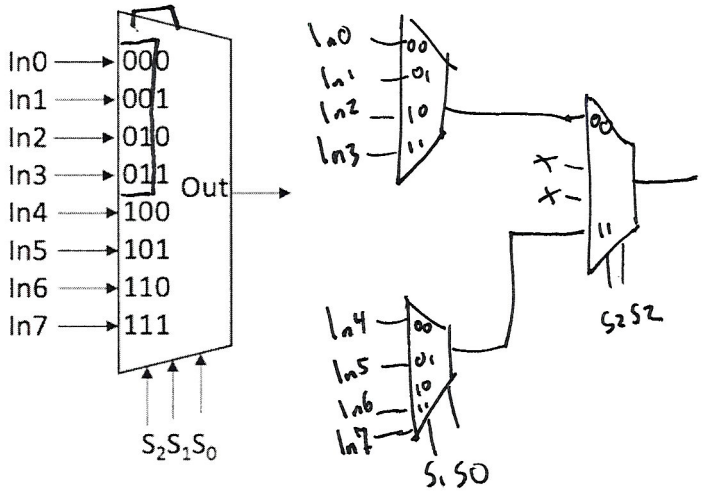
Taylor Comley
EE220 HW 4

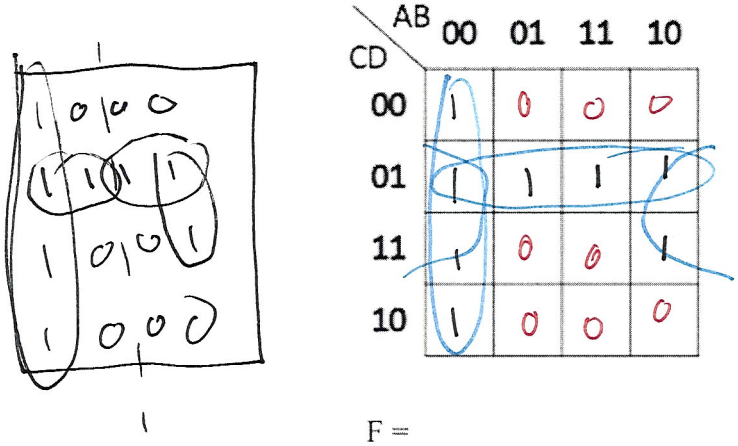
ECEn 220
Chapter 8 Homework

1. Implement an 8:1 MUX out of 4:1 MUX blocks. The symbol of an 8:1 MUX is shown on the left. Two block diagrams for two different designs are shown on the right. Complete the wiring of the control signals in each block diagram to complete these designs. You can use external logic gates if necessary.

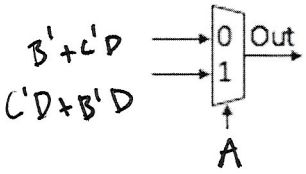


2. For the following few problems consider the following function:
   F(A,B,C,D) = Σm(0, 1, 2, 3, 5, 9, 11, 13)

Use a 2:1 MUX (A as the MUX select signal) to implement this function. Complete a 4-variable K-map of this function and circle the prime implicants used for each case. Write the Boolean equation of these cases. Draw the 2:1 MUX and its control and input signals.



A'B' + C'D + B'D

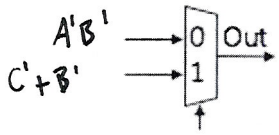


F =

When A = 0 F = B' + C'D
When A = 1 F = C'D + B'D

3. Use a 2:1 MUX (D as the MUX select signal) to implement the function from above. Complete a 4-variable K-map of this function and circle the prime implicants used for each case. Write the Boolean equation of these cases. Draw the 2:1 MUX and its control and input signals.

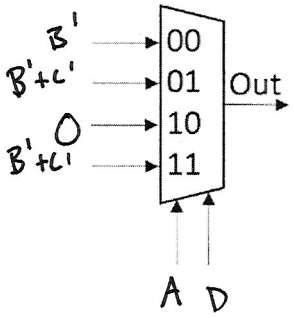
CD \ AB	00	01	11	10
00	1	0	0	0
01	1	1	1	1
11	1	0	0	1
10	1	0	0	0



F =
When D = 0 F = $A'B'$
When D = 1 F = $C'+B'$

4. Use a 4:1 MUX (AD as the MUX select signals) to implement the function from above. Complete a 4-variable K-map of this function and circle the prime implicants used for each case. Write the Boolean equation of all four cases. Draw the 4:1 MUX and its control and input signals.

CD \ AB	00	01	11	10
00	1	0	0	0
01	1	1	1	1
11	1	0	0	1
10	1	0	0	0



F =
When AD = 00 F = B'
When AD = 01 F = $B'+C'$
When AD = 10 F = 0
When AD = 11 F = $C'+B'$

ECEn 220
ALU Homework Solutions

Background

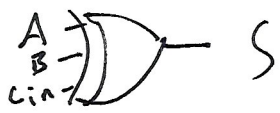
In the LC-3 there is an ALU which performs the functions PASS, ADD, AND, NOT. The last three should be self-explanatory. The PASS function simply passes the first input through to the output. Assuming inputs A and B as well as control signals C1 and C0, the behavior of the ALU is outlined in the table below.

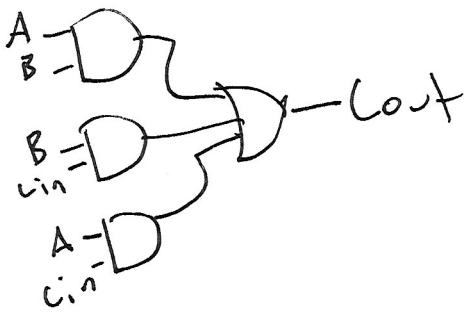
Control		Function Select	Function
C1	C0		
0	0	PASS	Out = A
0	1	ADD	Out = A + B
1	0	AND	Out = A • B
1	1	NOT	Out = A'

One way to build an ALU is to build a bit-slice. That is, you build one bit worth of ALU first. Then, you combine n of those together to make an ALU that works on n-bit values. Further, a good way to design such an ALU is to simply implement each function independently (AND, ADD, NOT, PASS) and then combine their outputs together using a 4:1 MUX.

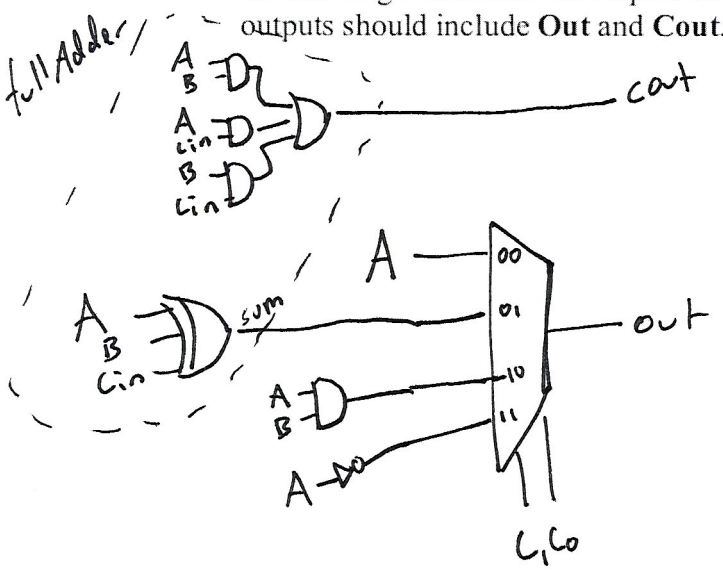
Problems

1. Begin by drawing the schematic of the full adder as described in section 9.6 of the text.





2. Now incorporate your single bit adder into a single bit ALU. This ALU contains a 4-to-1 mux with inputs of **A**, **A+B** (addition), **A AND B**, and **NOT A**. Draw the Logic Diagram for this single bit ALU. The inputs should include **A**, **B**, **Cin**, and **C[1:0]** (control). The outputs should include **Out** and **Cout**.



3. Now draw a symbol for this 1-bit ALU and show how you would combine 4 of them into a 4-bit ALU.