King Fahd University of Petroleum and Minerals
College of Computer Science and Engineering
Computer Engineering Department

COE 202: Digital Logic Design (3-0-3)

Term 131 (Fall 2013)

Major Exam II

Saturday November 30, 2013

Time: 120 minutes, Total Pages: 12

Name:_________________________________ **ID:**______________ **Section:** _______

Notes:

- Do not open the exam book until instructed
- **Calculators are not allowed** (*basic, advanced, cell phones, etc.*)
- Answer all questions
- All steps must be shown
- Any assumptions made must be clearly stated

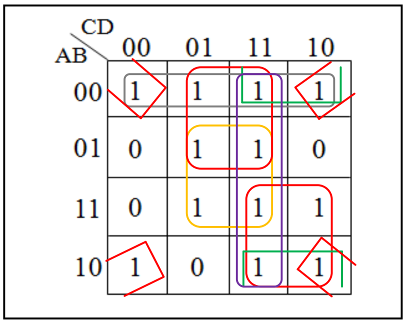
Question	Maximum Points	Your Points
1	14	
2	8	
3	20	
4	12	
5	15	
6	16	
Total	85	

Question 1. **(14 points)**

Question 1. **(14 points)**

For the following Boolean function shown in the K-map:

F(A, B, C, D)=Σ m(0, 1, 2, 3, 5, 7, 8, 10, 11, 13, 14, 15)



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

a. Identify all possible *prime implicants* of F and indicate which of these is essential.

b. Simplify the Boolean function F into a minimal sum-of-products expression.

c. Simplify the Boolean function F into a minimal product-of-sums expression.

a. **Prime Implicants:**

A'B', CD, A'D, BD, AC, B'C, B'D'

Essential Prime Implicants:

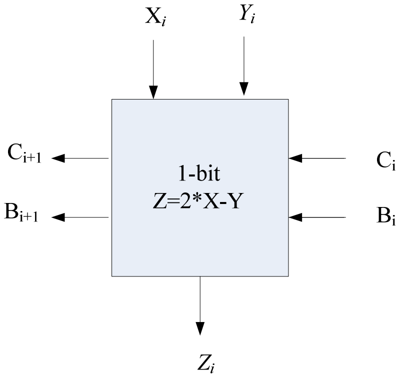
BD, AC, B'D'

b. **F =** BD + AC + B'D' + A'B' OR **F =** BD + AC + B'D' + A'D

c. **F =** (B'+C+D)(A+B'+D)(A'+B+C+D')

Question 2. **(8 Points)**

It is required to design a circuit to compute the equation Z=2*X-Y, where X and Y are two n-bit unsigned numbers. The circuit can be designed in a modular manner where it is designed for one bit and replicated *n* times. A 1-bit circuit block diagram is given below:

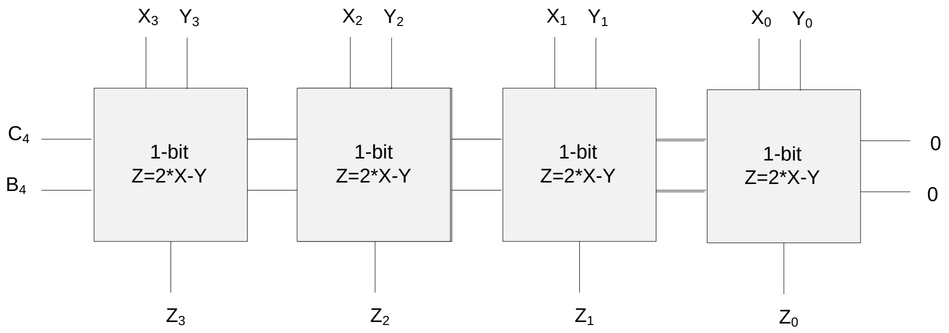


The meaning of the values of B_i and C_i is given in the table below:

B_i	C_i	Meaning
0	0	There is no carry or borrow
0	1	There is a carry of 1
1	0	There is a borrow of 1
1	1	This condition does not occur

For example, if X_i=1 and Y_i=1, then we should have Z_i=1, B_{i+1}=0 and C_{i+1}=0. If X_i=0 and Y_i=1, then we should have Z_i=1, B_{i+1}=1 and C_{i+1}=0.

The figure below shows how a 4-bit Z=2*X-Y circuit is implemented using 4 copies of the basic 1-bit cell.



Derive the truth table for the basic one-bit cell. You <u>do not need</u> to derive the equations for the circuit.

X_i	Y_i	B_i	C_i	B_{i+1}	C_{i+1}	Z_i
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	1	0	1
0	0	1	1	X	X	X
0	1	0	0	1	0	1
0	1	0	1	0	0	0
0	1	1	0	1	0	0
0	1	1	1	X	X	X
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	0	0	1
1	0	1	1	X	X	X
1	1	0	0	0	0	1
1	1	0	1	0	1	0
1	1	1	0	0	0	0
1	1	1	1	X	X	X

Question 3. **(20 Points)**

a. Fill in all blank cells in the two tables below.

Binary	Equivalent decimal value with the binary interpreted as:				
	Unsigned number	Signed-magnitude number	Signed-1's complement number	Signed-2's complement number	BCD number
10000000	128	- 0	- 127	-128	80

Decimal	Binary representation in 8 bits:		
	Signed-magnitude notation	Signed-1's complement notation	Signed-2's complement notation
- 75	11001011	10110100	10110101

b. Using 2's-complement signed arithmetic in 5 bits, do the following operations **in binary**. Show all your work, and:

- Verify that you get the expected decimal results.
- Check for overflow and mark clearly any overflow occurrences.

(i) 00111 00111 → +7 - 10101 - 01011 - -11 +18 > 15 00111 + 01011 0 ← 1 10010 overflow occured	(ii) 10110 -01010 → -10 - 10011 - -01101 - -13 +3 10110 + 01101 1 ← 0 00011 = +3 ✓
(iii) (+6) 00110 → 00110 + (-11) + -01011 +10101 -5 00110 + 10101 0 ← 1 11011 = - 00101 = -5 ✓	(iv) (-9) -01001 → 10111 - (+7) - +00111 -00111 -16 10111 → + 11001 1 ← 1 10000 = -16 ✓

c. Consider the signed 2's complement arithmetic operation A - B in 6 bits. With B = 101100, the largest value allowed for A in order to avoid the occurrence of overflow is $(001011)_2$.

B = - 010100 = - 20

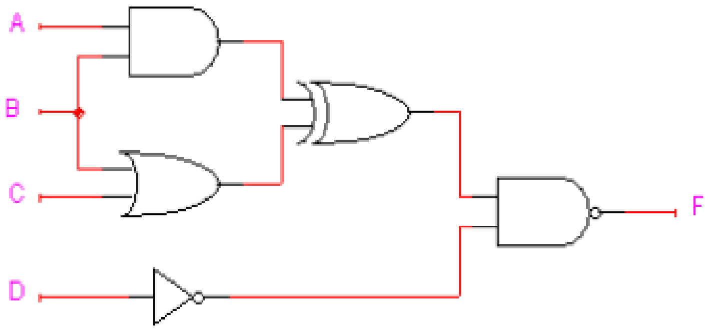
A - B ⩽ 31 → A ⩽ B + 31

A ⩽ -20 + 31

A ⩽ +11

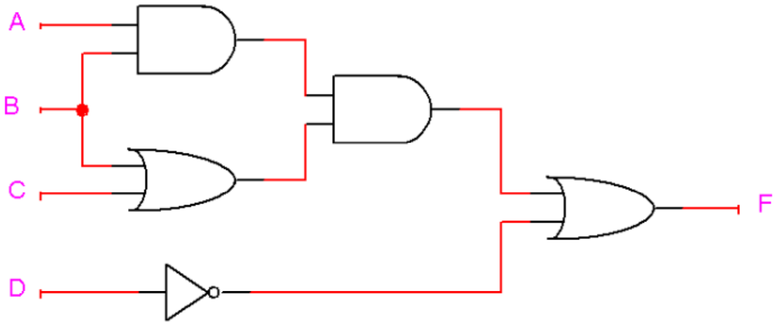
Question 4. **(12 Points)**

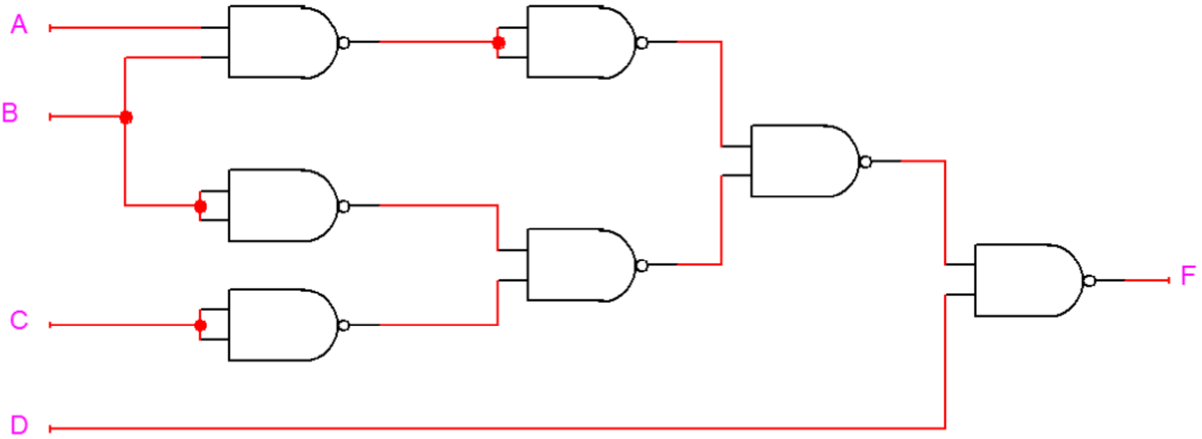
1. **(4 points)** Considering the following circuit, provide a <u>minimized **SOP**</u> expression of *F*(*A*, *B*, *C*, *D*).



$$F = \overline{\left[AB \oplus (B+C)\right]\ \overline{D}} = \overline{\left[AB \oplus (B+C)\right]} + D = \left[AB\square\ (B+C)\right] + D$$
$$= AB(B+C) + \overline{AB}\ \overline{(B+C)} + D$$
$$= AB + ABC + (\overline{A} + \overline{B})\ \overline{B}\ \overline{C} + D$$
$$= A\,B + \overline{B}\ \overline{C} + D$$

2. **(4 points)** **<u>Using only NAND gates</u>**, redraw the following circuit to show a multi-level **NAND** circuit. Only the **true** form of each input variable is available.

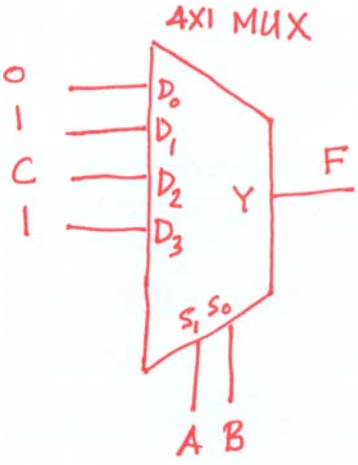




3. **(4 points)** Implement $F(A,B,C) = \prod M(0,1,4)$ using a 4-to-1 MUX. Show how you obtained your solution, and properly label all input and output lines.

Connecting *A* and *B* to the MUX select lines S_1 and S_0, respectively, yield the following:

A	*B*	*C*	*F*	MUX Inputs
0	0	0	0	$D_0 = 0$
0	0	1	0	
0	1	0	1	$D_1 = 1$
0	1	1	1	
1	0	0	0	$D_2 = C$
1	0	1	1	
1	1	0	1	$D_3 = 1$
1	1	1	1	



Question 5. **(15 Points)**

A Triple adder **circuit** adds three *n*-bit numbers a, b, and d. The triple adder circuit consists of *n*-stages of the single bit circuit slice shown in Fig. (i) (called 3-Add). The i^{th} stage receives 5 inputs 3 of which are the i^{th} bits of a, b, and d and the other two are carry-in inputs $C0_i$ and $C1_i$. It has 3 outputs; one sum bit (S_i) and two carry-out bits $C0_{i+1}$ and $C1_{i+1}$.

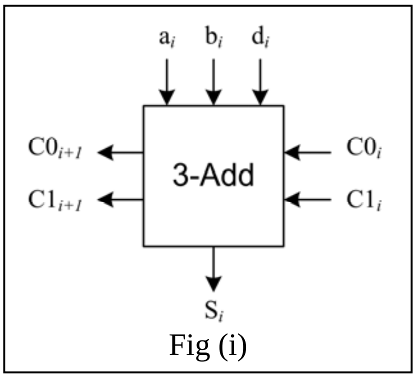


Fig (i)

Fig. (ii) shows the *n*-bit Triple adder circuit

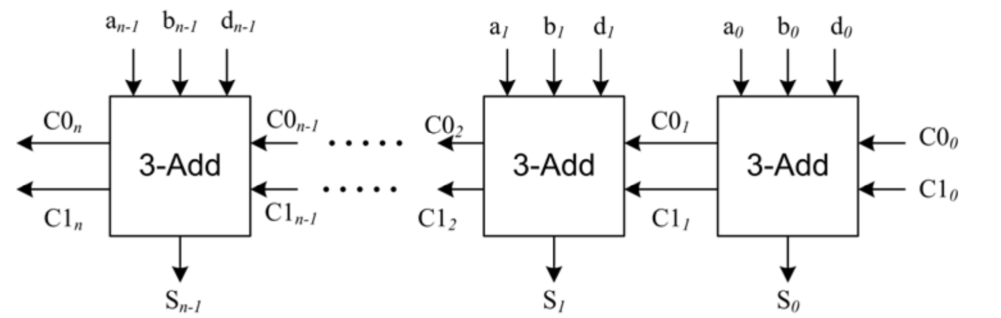


Fig (ii)

Fig. (iii) shows an example for such addition.

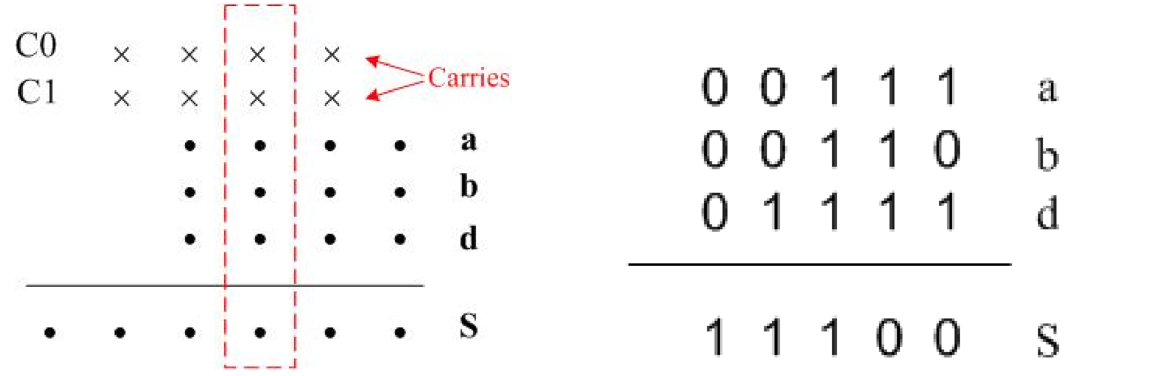


Fig. (iii)

The circuits used to generate the output carry bits are shown in Fig. (i**v**)**. a**nswer the following:

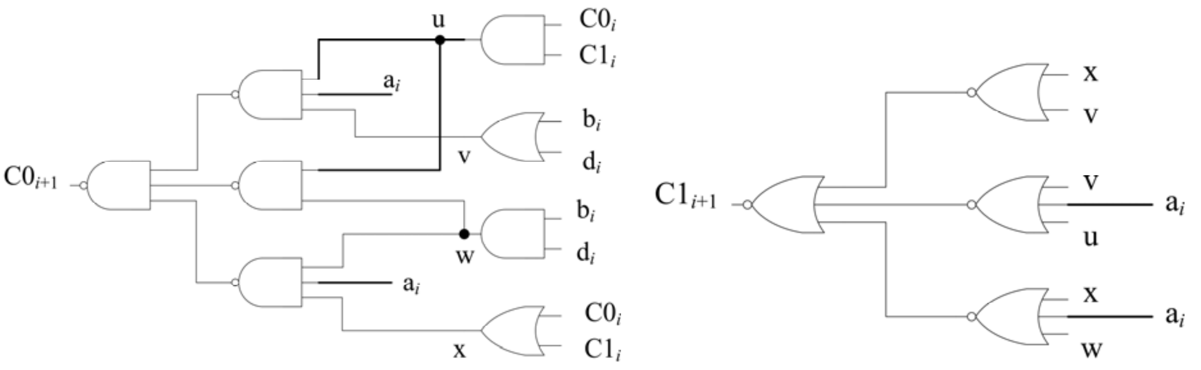


Fig. (iv)

(I) Using the gate propagation delays of Table (i)**,** what is the carry propagation delay *per single stage* **for both of the output carries (C0 and C1)?** (5 Points)

Gate	Delay
AND , NAND	**1 ns**
OR , NOR	**3 ns**
XOR	**4 ns**

Table (i)

Solution

Propagation Delay ($C0_i$ → $C0_{i+1}$) =2 T_{NAND} + Max (T_{AND} , T_{OR}) = 2 x 1 ns + 3 ns = 5 ns.

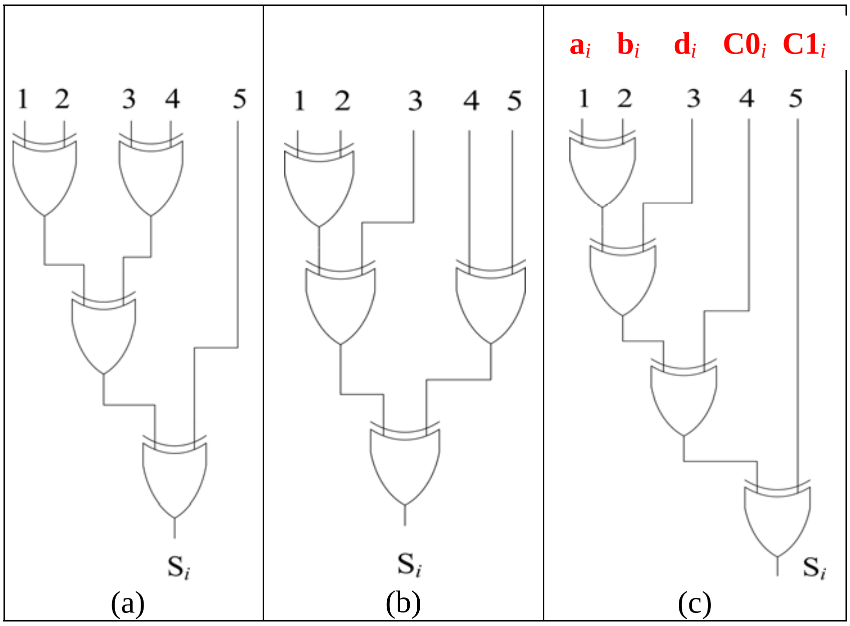
Propagation Delay ($C1_i$ → $C1_{i+1}$) =2 T_{NOR} + Max (T_{AND} , T_{OR}) = 2 x 3 ns + 3 ns = 9 ns.

(II) For the *n*-bit triple adder circuit of Fig. (ii), assuming a 12ns delay from the *i*th input carries to the *i*th sum signal (S_i), calculate the worst case delay to generate the *n*-bit sum (S_{n-1} S_{n-2}….. S_1 S_0) of the three *n*-bit operands. (5 Points)

Solution

W. Case Carry propagation delay for the 1st (*n*-1) stages = (*n*-1) x 9ns= 9(*n*-1) ns
Delay to get the S_{n-1} = 9(*n*-1) ns + 12 ns = (9*n*+3) ns = 3(3*n*+1) ns

(III) The *i*th output sum bit is given by $\boldsymbol{S_i = a_i \oplus b_i \oplus d_i \oplus C0_i \oplus C1_i}$, select one of the following logic implementations of S_i to yield the fastest *n*-bit triple adder. *You must Label the 5-inputs of this circuit* (as a_i, b_i, d_i, $C0_i$, $C1_i$) and **justify** your answer. (5 Points)


1 2 3 4 5
S_i
(a)
1 2 3 4 5
S_i
(b)
$\mathbf{a}_i$ $\mathbf{b}_i$ $\mathbf{d}_i$ $\mathbf{C0}_i$ $\mathbf{C1}_i$
1 2 3 4 5
S_i
(c)

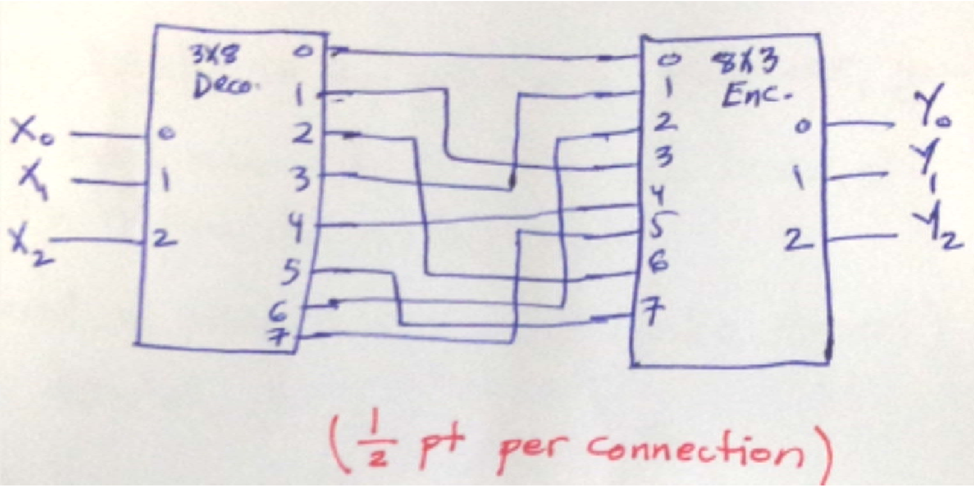
Configuration (c) yields the fastest output because of the following:

1. $\mathbf{a}_i$, $\mathbf{b}_i$, and $\mathbf{d}_i$ are all (∀i) available at time **0,** thus at the n^{th} stage these signals should have propagated through gates and S_{n-1} would only be waiting for the incoming carry signals ($\mathbf{C0}_{n-1}$, $\mathbf{C1}_{n-1}$) to be computed.
2. $\mathbf{C0}_{n-1}$ arrives **4ns <u>earlier</u>** than $\mathbf{C1}_{n-1}$. Thus, inputs of the last XOR gate are available at the same time. Thus Delay from $\mathbf{C1}_{n-1}$ to $\mathbf{S}_{n-1}$ is only 4 ns.

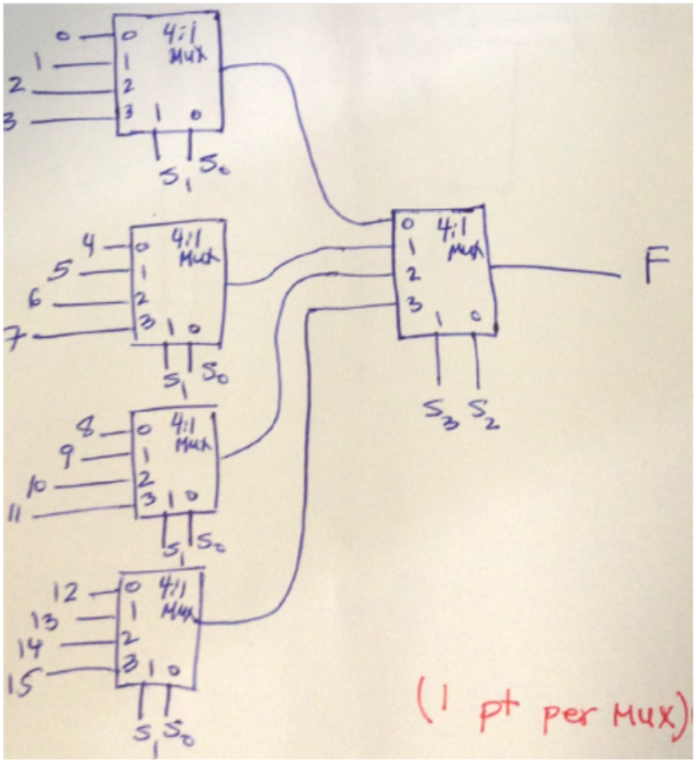
Question 6. **(16 Points)**

X	Y
0	0
1	3
2	6
3	1
4	4
5	7
6	2
7	5

a. Design a circuit that has a three-bit input X and three-bit output Y. Both X and Y represent the integers 0 to 7 (i.e., $X, Y \in \{0, 1, \dots, 7\}$). Using a *single* decoder and a *single* encoder of appropriate sizes, show how can you build a circuit that performs the function **[Y = 3X mod 8]**. Make sure you *label all signals*. The truth table for this circuit is shown in decimal notation. **[4 pts]**



b. Construct a 16-to-1 multiplexer using the minimum number of 4-to-1 multiplexers. **[5 pts]**



c. Using *only* MSI parts, design a circuit that takes two 4-bit binary numbers A = $A_3A_2A_1A_0$ and B = $B_3B_2B_1B_0$ together with a 2-bit selection input S = S_1S_0. The circuit produces a 5-bit output O = $O_4O_3O_2O_1O_0$ according to the shown table:

Clearly label all inputs and outputs of the MSI parts. **[7 pts]**

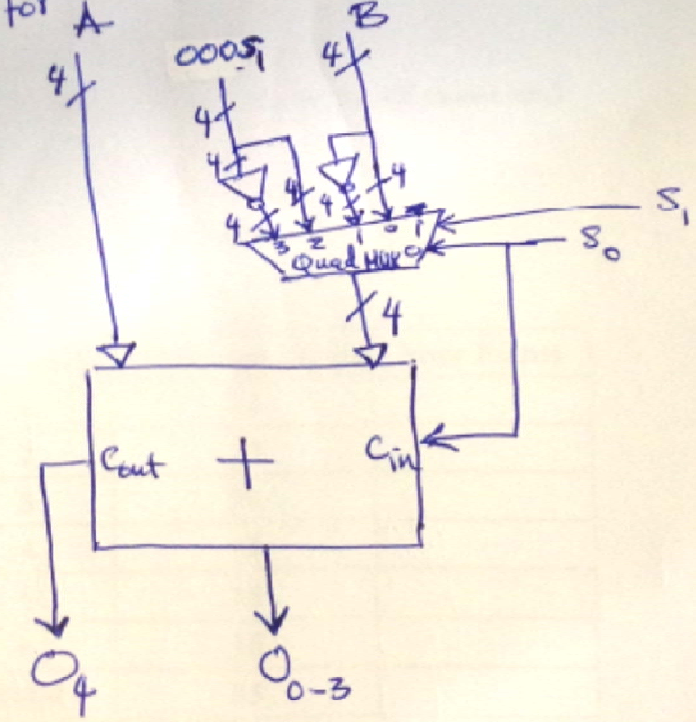
S_1S_0	O
00	A+B
01	A-B
10	A+1
11	A-1

notice A is common

Also, whenever there is a subtraction operation, $S_0 = 1$

Note:

- Many solutions are possible. However, you must use one adder. You will lose ≥ 2 points if you do not do that.
- Instead of using a constant (like 0001), use $S_0S_0S_0S_1$ for → A+1


A
B
4
000S_1
4
4
4
4
4
4
3
2
1
0
Quad MUX
S_1
S_0
4
Cout
+
Cin
O_4
O_{0-3}