Homework #2 (v20180422)

(Deadline: 11:59PM PDT, May 4, 2018)

Name (Last, First): Oliveira, Christina

Student Id #: 204-803-448

INSTRUCTIONS

This homework is to be done individually. You may use any tools or refer to published papers or books, but may not seek help from any other person or consult solutions to prior exams or homeworks from this or other courses (including those outside UCLA). You're allowed to make use of tools such as Logisim, WolframAlpha (which has terrific support for boolean logic) etc.

You must submit all sheets in this file based on the procedure below. Because of the grading methodology, it is much easier if you print the document and answer your questions in the space provided in this problem set. It can be even easier if you answer in electronic form and then download the PDF. Answers written on sheets other that the provided space will not be looked at or graded. Please write clearly and neatly - if we cannot easily decipher what you have written, you will get zero credit

SUBMISSION PROCEDURE: You need to submit your solution online at Gradescope (https://gradescope.com/). Please see the following guide from Gradescope for submitting homework. You'd need to upload a PDF and mark where each question is answered.
http://gradescope-static-assets.s3-us-west-2.amazonaws.com/help/submitting_hw_guide.pdf

Problem #1

Consider the following logic block. Four input bits, *A[3:0]*, are summed together. The logic produces a one whenever the sum is either 2 or 4.

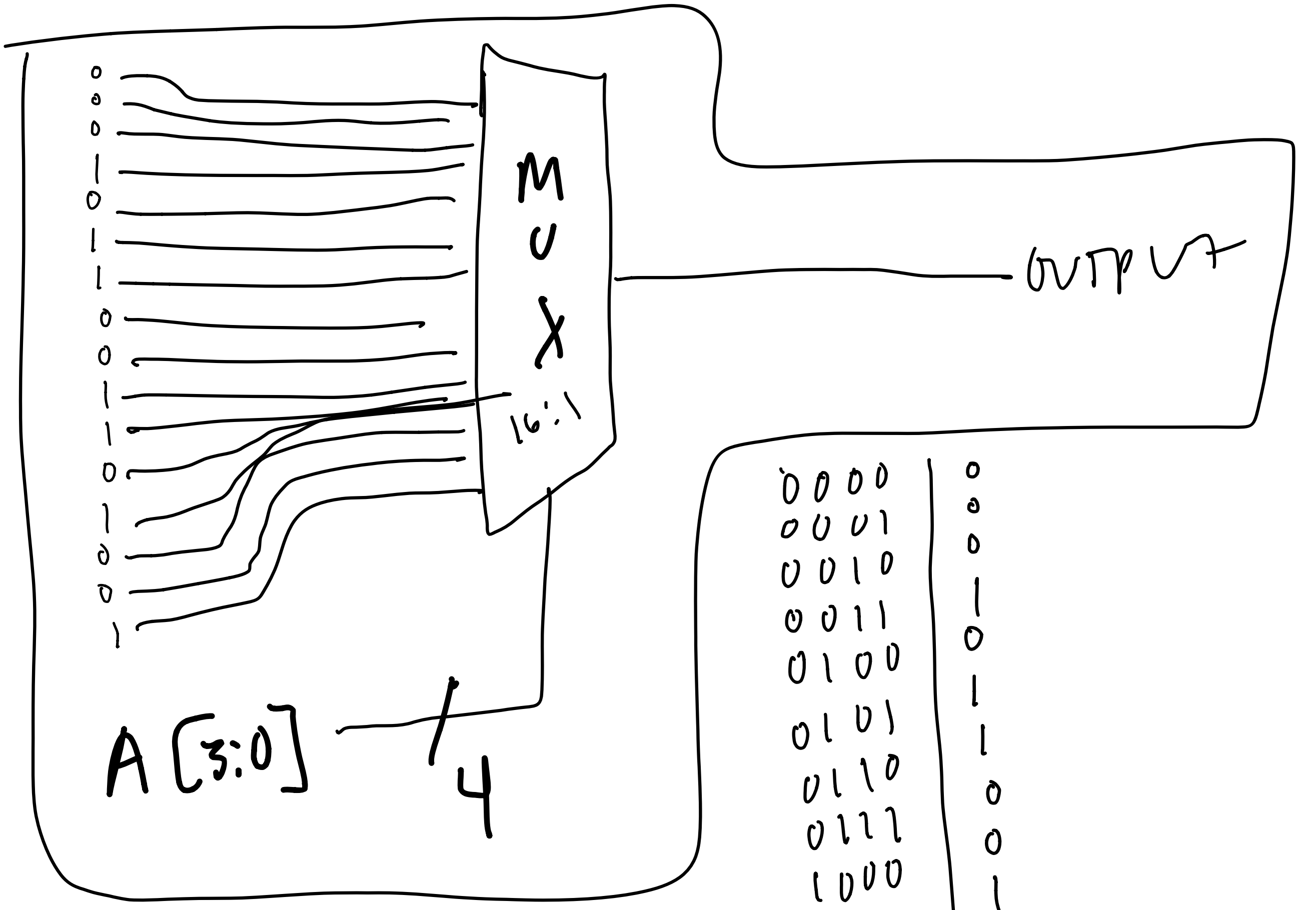
(a) Use a 16-input multiplexer to produce the output with *A[3:0]* as the select inputs.

(b) Now assuming that *A[3]* is allowed to be a data input to the multiplexer (as opposed to being one of the select signals). Implement the same logic using an 8-input multiplexer.

(c) Implement the same logic as (b) assuming that you can only use a tree of 2-input multiplexers. Similar to (b), *A[2]* are the select inputs of the 2-input multiplexers closest to the inputs, *A[1]* controls the next set of multiplexers, and *A[0]* controls the multiplexer closest to the output.

Answer the question for all parts in the space below.

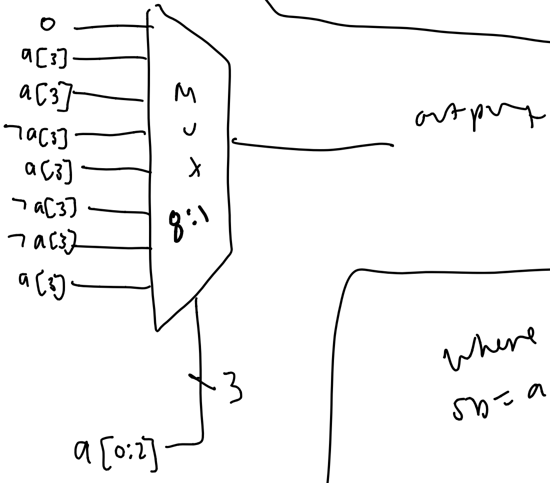
a)



truth table =

0000	0
0001	0
0010	0
0011	1
0100	0
0101	1
0110	1
0111	0
1000	0
1001	1
1010	1
1011	0
1100	1
1101	0
1110	0
1111	1

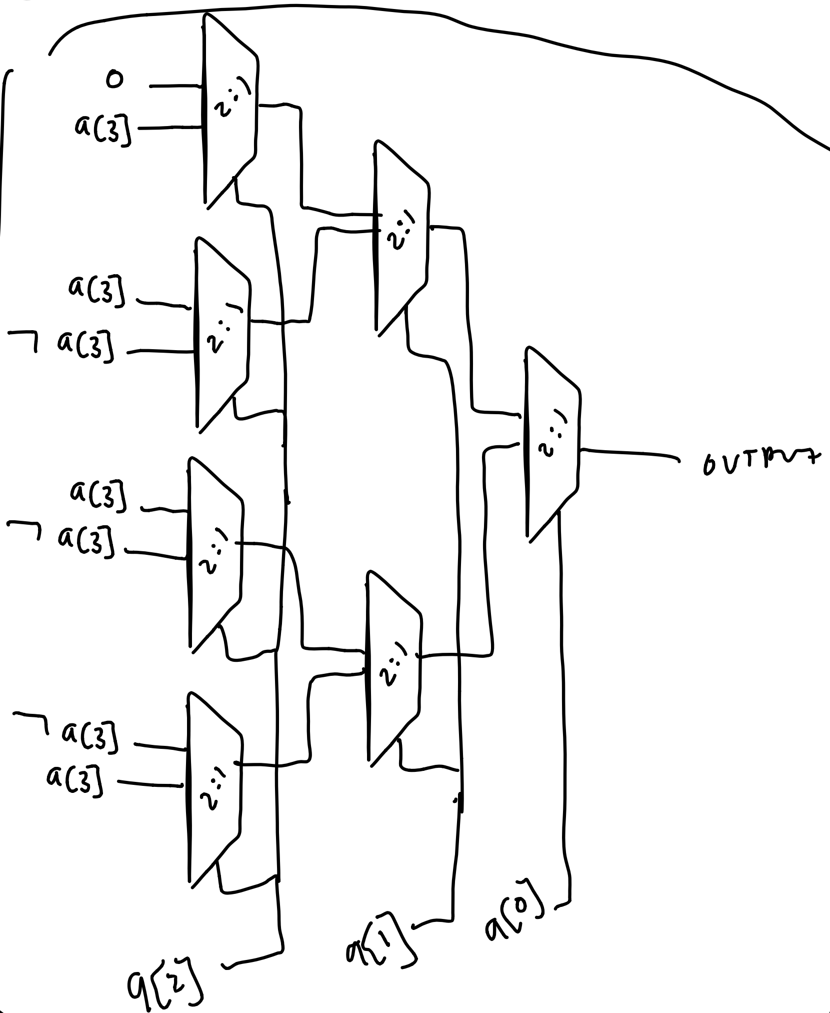
b)



sb a[0-2]	sb a[3]=1	sb a[3]=0	
0 0 0	0	0	0
0 0 1	1	0	sb
0 1 0	1	0	sb
0 1 1	0	1	¬sb
1 0 0	1	0	sb
1 0 1	0	1	¬sb
1 1 0	0	1	¬sb
1 1 1	1	0	sb

where
sb = a[3]

c)



Problem #2

The idea for this problem is to build building blocks that are then reused in bigger blocks.

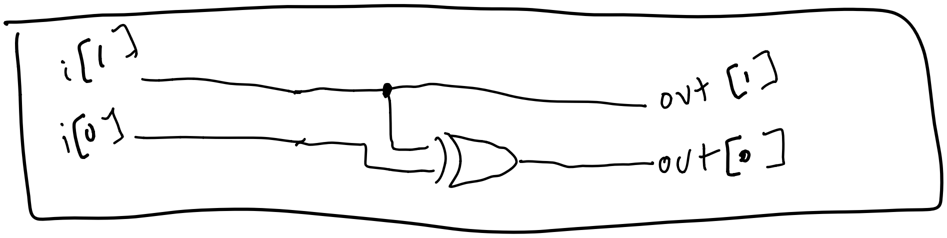
(a) Implement 2-bit Binary-to-Gray Encoder. There are several possible implementations. You should need only a single XOR gate.

(b) Implement a 3-bit Binary-to-Gray Encoder using the 2-bit encoder of (a) as a building block.

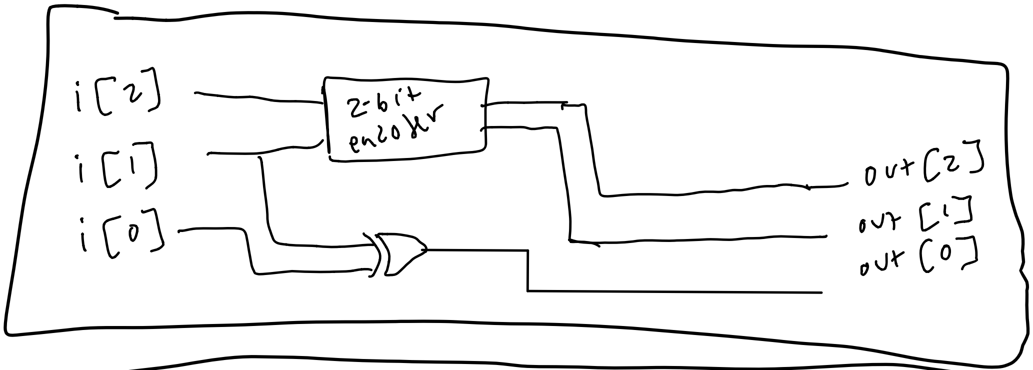
(c) Implement a 4 bit Binary-to-Gray using the 3-bit encoder of (b) as a building block.

Answer the question for all parts in the space below.

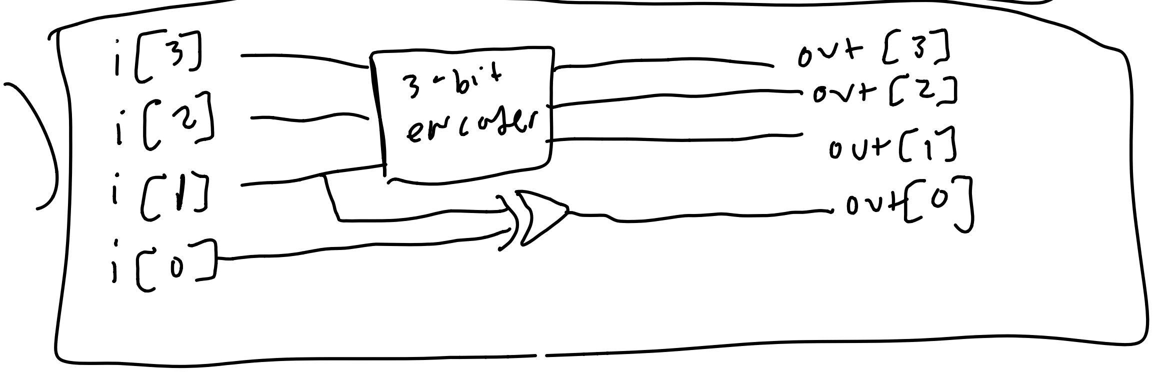
a)



b)



c)



(IMPORTANT: Keep this page in submission even if left unused)

Problem #3

Universal Time Coordinated (UTC+*nnn*) has replaced Greenwich Mean Time (GMT) as the means to determine time zones, where *nnn* specifies the time zone. In the first homework we determined the number of bits *nnn* must be (7 bits) to represent the time zone without being specific about the representation except that we set the binary number to count in intervals of 15 minutes.

(a) If we represent *nnn* as sign magnitude, how would one represent French Polynesia at -9:30?
(b) Repeat (a) if *nnn* is instead 2's complement.
(c) What is the relative accuracy of this representation?

A wacky designer decided that to represent time of day, instead of representing the hours as 0 to 12, she learned about 2's complement and chose to use 24's complement for the hours with 0 corresponding to 12 noon. She simply uses one base-24 digit to represent the hour.

(d) How would the clock dial look? Note that two numbers would map to each of the 12 positions.
(e) Before noon, effectively a negative number, what would 10 AM appear in 24's complement. Note that 10 AM is effectively -2 in the counting.
(f) What would be the time of day (in a normal clock) if 12 is the hour in 24's complement?
(g) Adding time becomes quite easy. Suppose a rocket ship leaves at 10 AM (your answer in (e)) and takes 135 hours to get to its destination, Mars. Show the 24's complement arithmetic and the resulting hour in 24's complement.

The wacky designer gets to Mars and finds out that the Martian day is 25 hours long. So, she decides to use 5's complement to represent the hour in Mars. She needs 2 "quinits" (base-5 digits) to represent the hour

(h) What does she write corresponding to the 12th hour?
(i) What does she write corresponding to the 24th hour?

Answer the question for all parts in the space below.

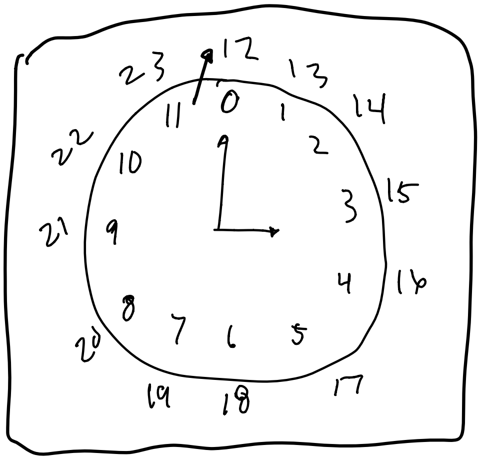
a) -9:30 = - 38 in 8 bit signed mag $= \boxed{1100110}$

b) -38 in 2's comp. $= \boxed{1011010}$

c) $\frac{\text{ABSOLUTE acuracy}}{-9{:}30} = \frac{7.5}{} = \boxed{\frac{1}{76}}$

(IMPORTANT: Keep this page in submission even if left unused)

d)



e) 10 AM: $\boxed{22}$

f) MIDNight

g) $22 + 135$ in 24s $= \overset{(22)}{(5)(15)}$

comp.

(5)(15)

$$\begin{array}{r} \overset{(22)}{(5)(15)} \\ +\ 37 \\ \hline 15 \\ 6(13) \end{array}$$

$\boxed{13}$ = 7am

(IMPORTANT: Keep this page in submission even if left unused)

if clock from

15 hour DAY

h) $\boxed{44}$ 12th hour

$-12 \rightarrow 12$

0 is 13th hour

i) $\boxed{21}$ 24th hour

$$\begin{array}{r} 44 \\ -44 \\ \hline 00 \\ +01 \\ \hline 01 \end{array} = -1$$

44

24 (→21)

Problem #4

Signal processors use a number of common building blocks where one such block is a Subtract-Compare-Select (SCS). An SCS logic takes the difference between two input words, R_i and D and generates two outputs, a single bit Q_i and an output word R_{i-1}. If $R_i - D < 0$, the output word $R_{i-1} = D$ else $R_{i-1} = R_i - D$. If $R_i - D < 0$, $Q_i = 0$ else $Q_i = 1$.

(a) The SCS can be built using building blocks such as a Single-Bit Subtractor (SBS) where each bit position takes two input bits, x_i and y_i, and a borrow input, b_{i-1}, from the lesser significant bit position. It outputs a difference, d_i and a borrow output, b_i. Show a truth table for this SBS logic and the Boolean functions for d_i and b_i.

(b) Implement the functions in (a) using INV, XOR, AND, OR, or 2:1 Multiplexers as available logic gates.

(c) Use the SBS of (a) as a building block to implement an 8-bit Subtractor. Inputs of *x[7:0]*, *y[7:0]*, and outputs of *b[7]*, and *d[7:0]*, Assume that the borrow input for the LSB is 1'b0. You can use the block you designed in (b). Any additional logic gates you need, you may use the same logic gates as provided in (b).

(d) Build an 8-bit SCS block using the 8-bit Subtractor of part (c) and any additional logic gates as provided in (b).

Answer the question for all parts in the space below.

a)

Truth Table

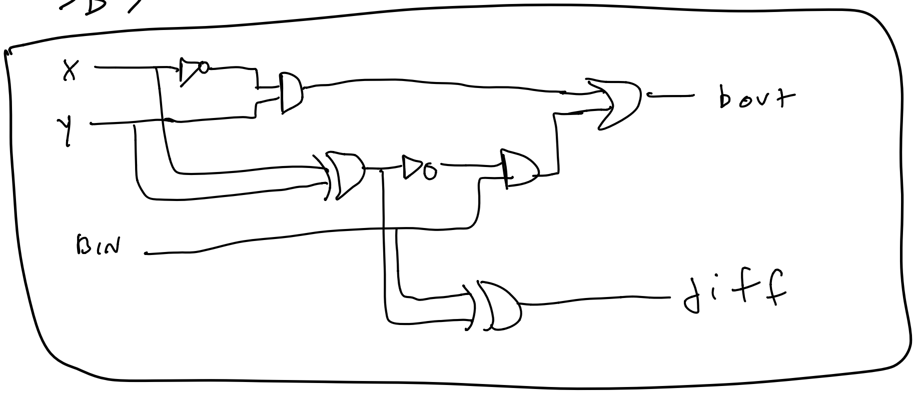
bin	X	Y		diff	bout
0	0	0		0	0
0	0	1		1	0
0	1	0		1	1
0	1	1		0	0
1	0	0		1	1
1	0	1		0	0
1	1	0		0	1
1	1	1		1	1

$$diff = (\overline{X} \wedge \overline{Y} \wedge BIN) \vee (\overline{X} \wedge Y \wedge \overline{BIN}) \vee (X \wedge \overline{Y} \wedge \overline{BIN}) \vee (X \wedge Y \wedge BIN)$$

$$bout = (\overline{X} \wedge \overline{Y} \wedge BIN) \vee (\overline{X} \wedge Y \wedge \overline{BIN}) \vee (\overline{X} \wedge Y \wedge BIN) \vee (X \wedge Y \wedge BIN)$$

b)

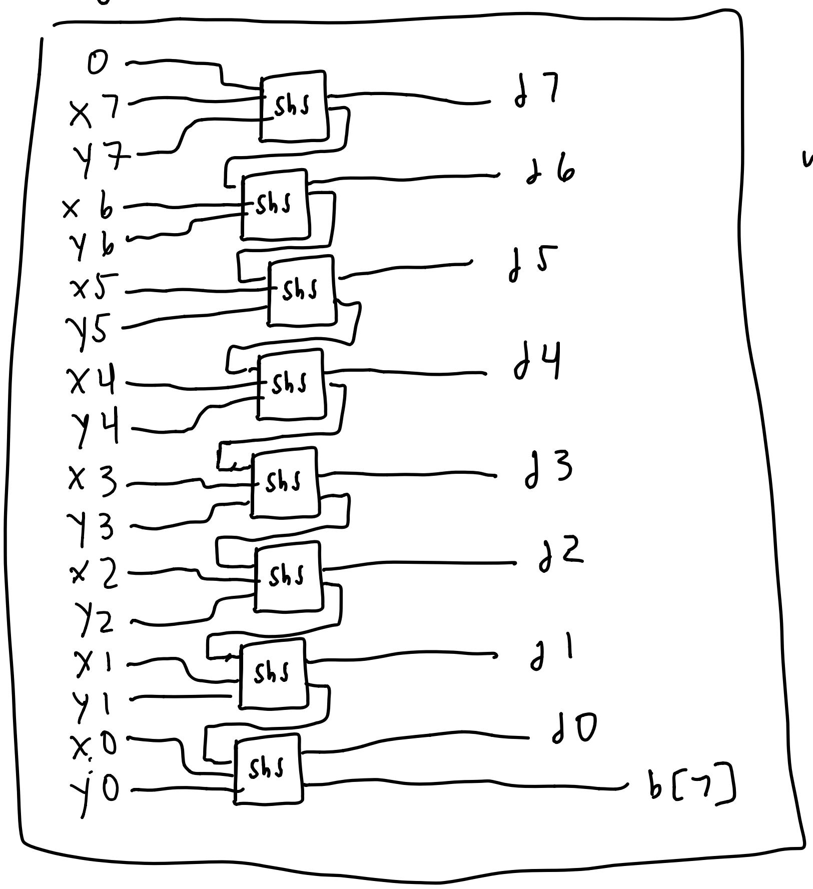
SBS

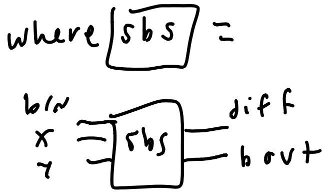


(IMPORTANT: Keep this page in submission even if left unused)

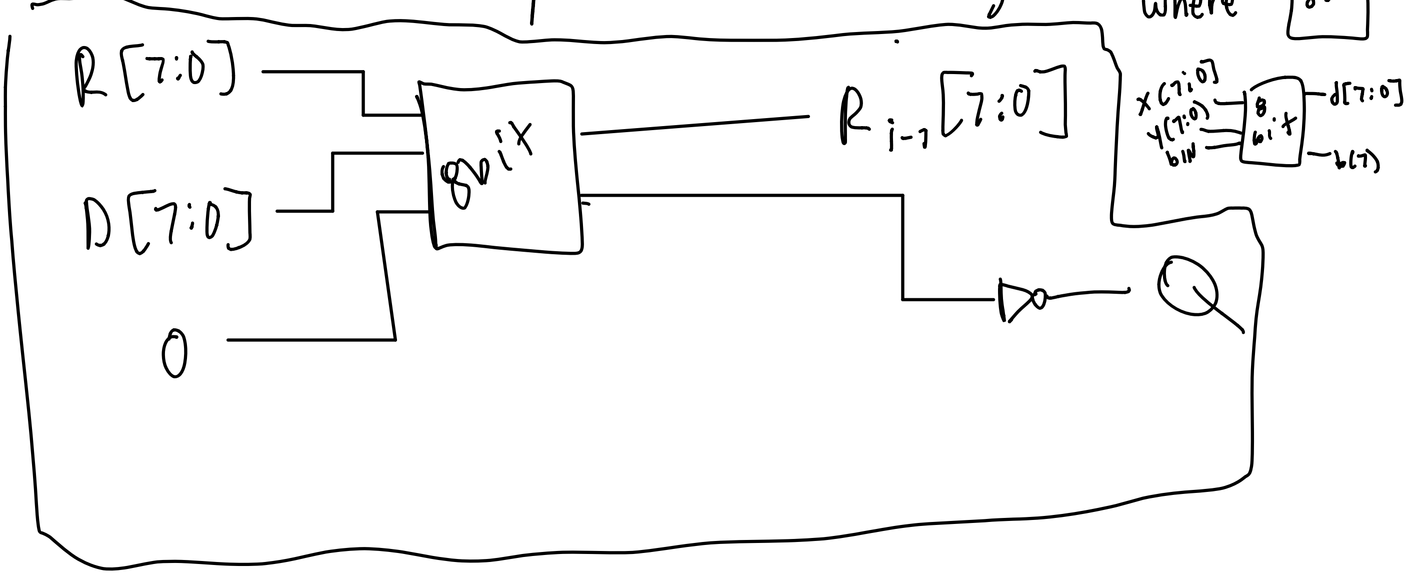
8-bit sub.

c)





d) subtract compare select?



Problem #5

Let's take a 24-bit word, *W[23:0]*.

(a) If *W* is 2's complement, what's the most positive and most negative possible values in decimal?

(b) How do you represent 160160 and -51051 in 2's complement (binary)?

(c) If *W* is unsigned, what's the largest magnitude value in decimal?

(d) If *W* is hex, repeat (b).

(e) If *W* is 1's complement, repeat (b).

(f) If *W* is signed fixed point of 13.11 (2's complement), what is the largest and smallest magnitude negative number that is achievable (in decimal)? What is the fixed point binary number for these respectively.

(g) If *W* is an extension of the IEEE floating point format (incidentally, it doesn't actually exist for a 24-bit word) where mantissa (or fraction) is 15 bits and the exponent is 8 bits, what is the bias?

(h) Extending on (f), what is the largest and smallest magnitude (positive sign) achievable? Show both the integer value and the binary representation. Recall that the exponent cannot be 8'b0000_0000 or 8'b1111_1111.

(i) How do you represent the decimal, -495.0594, in binary (closest value) if it is fixed point (f)? What about floating (g)?

(j) What is 1011_0110_0101_1100_1100_1011 (binary number) in decimal if the number is fixed point (f)? What about floating (g).

Answer the question for all parts in the space below.

a) $\boxed{-8,388,608}$ Most negative $2^{24}/2$

$\boxed{8,388,607}$ Most positive

b) 160,160 → $\boxed{000\ 0001\ 00\ 111\ 00011\ 0100000}$

-51,051 → $\boxed{11111111\ 00\ 111\ 0001\ 0010101}$

c) $\boxed{16,777,215}$ $2^{24}-1$

d) W is Hex: 160160 → 0x0271A0

-51051 → 0x FF3895

e) W is 1s complement

160160 → $\boxed{000\ 0001\ 00\ 111\ 00011\ 0100000}$

-51051 → $\boxed{11111111\ 00\ 111\ 0001\ 0010100}$

(IMPORTANT: Keep this page in submission even if left unused)

f) largest mag: 1000 00 0 000 0000000 00000000 $= 2^{-12}$

smallest mag: 111 111111 111111 111111 111 $= \frac{1}{2048}$

g) $\frac{2^{8}-1}{2} = \boxed{127}$

h) largest positive

$\underbrace{0}_{s}\ \underbrace{11111110}_{exp}\ \underbrace{111111111111111}_{M} = 2^{254-127} = \boxed{2^{127} \times \frac{2^{15}-1}{2^{15}}}$

smallest positive

$0\ \underbrace{00000001}_{exp}\ \underbrace{000000000000000}_{M} = 2^{1-127} = \boxed{2^{-126}}$

i) -495.0594IN

fixed point = 1111 0000 1 0000 1111 0000111

floating point = 1100000111 11 1011111 0000111

j) 1011 0110 0101 1100 1100 1011

fixed point : -2357 . 59912

floating point: $(-1)\ 2^{108-127}\ (1.59912)$