Question #1

Consider the Boolean function defined by the truth table below where *A, B, C, and D* are inputs, and *Y* is the sole output.

#	A	B	C	D	Y
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	X
14	1	1	1	0	1
15	1	1	1	1	X

(a) Complete the following statements

$$Y = \sum m(0, 4, 5, 6, 7, 8, 9, 11, 14 \qquad)$$

(b) Complete the Karnaugh Map shown below, **circle** the prime implicants.

AB B A

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

D C

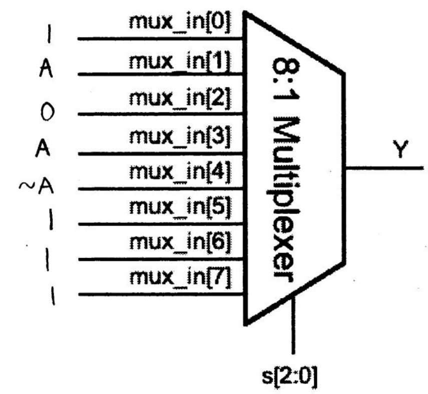
How many prime implicants are there? 7

(c) Write the Boolean (sum-of-product) expression for the essential prime implicants (if any).

EssentialPrimeImplicants = $(A \wedge D) \vee (B \wedge C)$

1xx1 x11x

(d) Implement the function *Y* using an 8-input multiplexer. The select signal is *s[2:0]={B,C,D}* where *s=3'b100* is *B*=1 and *C=D*=0 selecting the input *mux_in*[4]. *A* or ~*A* are permissible as inputs, *mux_in*[7:0]. Write the desired inputs on the figure below.

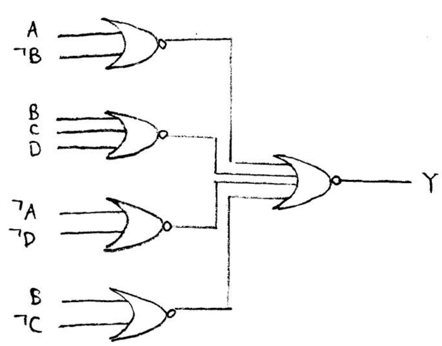


(e) Implement $\neg Y$ using the minimum # of NOR gates with fewest # of inputs (minimize literals and terms). * use product - of - sums

- Cover: 01xx, x000, 1xx1, x11x
- $Y = (\neg A \wedge B) \vee (\neg B \wedge \neg C \wedge \neg D) \vee (A \wedge D) \vee (B \wedge C)$
- $\neg Y = \neg(\neg A \wedge B) \wedge \neg(\neg B \wedge \neg C \wedge \neg D) \wedge \neg(A \wedge D) \wedge \neg(B \wedge C)$

  $= (A \vee \neg B) \wedge (B \vee C \vee D) \wedge (\neg A \vee \neg D) \wedge (\neg B \vee \neg C)$

- Implementation:



Question #2

$$Y = \neg\left(\neg(a \wedge \neg b) \vee (c \wedge \neg(d \vee e))\right)$$

(a) For the above Boolean function, if you were to convert the above expression into a sum-of-product representation, how many times did you have to apply DeMorgan's theorem?

2

(b) For part (a), what is the resulting function?

$Y = (a \wedge \neg b) \wedge \neg(c \wedge \neg(d \vee e))$

$Y = (a \wedge \neg b) \wedge (\neg c \vee (d \vee e))$

$Y =$ $(a \wedge \neg b \wedge \neg c) \vee (a \wedge \neg b \wedge d) \vee (a \wedge \neg b \wedge e)$

(c) The following expression can be written as a 6-term sum-of-product,

$$Y = (a \vee b) \wedge (a \vee \neg b \vee \neg c)$$

What Boolean property do you need to apply to do this?

distributive

Without reducing, what are the 6 product terms?

$\underbrace{(a \wedge a)}_{\text{idempotence}} \vee \underbrace{(a \wedge \neg b) \vee (a \wedge \neg c) \vee (b \wedge a)}_{\text{absorbed by first term}} \vee \underbrace{(b \wedge \neg b)}_{\text{never true}} \vee (b \wedge \neg c)$

(d) The 6-term sum-of-product of part (c) can obviously be reduced.
What is the reduced expression?

$a \vee (b \wedge \neg c)$

What Boolean axioms or properties are needed for the reduction?

absorption, idempotence

Question #3

(a) The following 8 bits can be used to represent different numbers depending on the encoding

8b'10010111

If this was unsigned, what is the corresponding integer? 151

(b) If the 8 bits in (a) was sign magnitude, what is the corresponding integer? -23

(c) If the 8 bits in (a) was 2's complement, what is the corresponding integer? -105

(d) If the 8 bits in (a) was hexadecimal, what is the corresponding hexadecimal? 97

(e) If the 8 bits in (a) was binary coded decimal, what is the corresponding integer? 97

(f) If the 8 bits is fixed point 1001.0111, what is the corresponding number? $9\frac{7}{16}$

(g) If the 8 bits in (a) was a 4E3 floating point number (IEEE format S+EEE+MMMM),

* minimum magnitude = 0.25°C

What is the bias? $2^{3-1}-1=3$

What is the corresponding real number? $-1\frac{7}{16} \times 2^{1-3} = -\frac{23}{64}$

(h) Military temperature range is -55°C to +125°C with 1% accuracy.

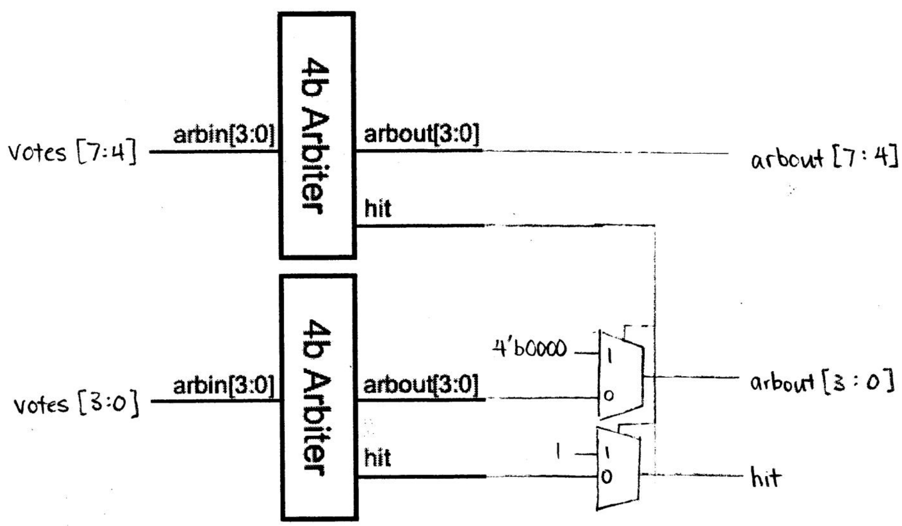
Would you choose floating point or fixed point? fixed point

If you are to represent this in floating point,
what is the minimum # of bits for mantissa? $\lfloor -\log_2(0.01) \rfloor = 6$ b (assuming rounding, not truncation)

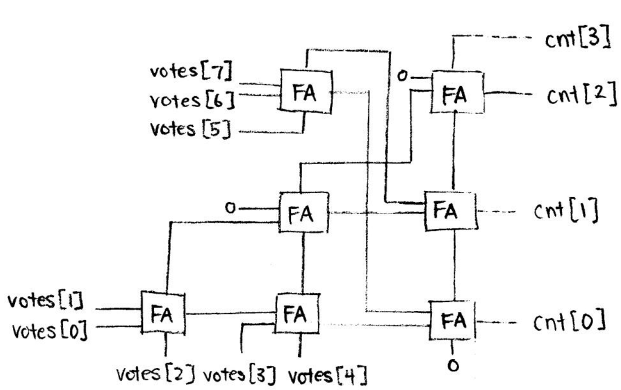
And, what is the minimum # of bits for exponent? $\lceil \log_2(125) \rceil = 7$ b

Question #5

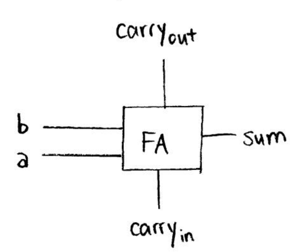
(a) Given 8-bit input, *votes*[7:0], in which any number of the inputs can be a 1'b1. Build an **arbiter** that provides an 8-bit output, *arbout*[7:0], that is 1-hot. The hot signal corresponds to the position with the highest priority. Note that votes[7] has higher priority than votes[6] etc. You have available to you a module ARB that is a 4-bit arbiter already built that you **must** use. ARB accepts as inputs *arbin*[3:0] and outputs *arbout*[3:0] and a *hit* signal to indicate that one or more of the signals is a 1'b1. You also have available to you INV (inverters), and 2-input MUX (multiplexers). Recall that you can implement considerable arbitrary logic with 2-input MUXs.



(b) Now, the *votes*[7:0] need to be counted. You have available Full Adders (FA) as building blocks for implementing a design. If the delay of the logic is determined by the number of hops where each hop is the traversal of a Full-Adder from any input (*a,b,and c*) to any output (*sum, carry*). Design your block to minimize this delay. Note that your design should output 4 bits to indicate the binary count, *cnt*[3:0].



Full Adders are oriented in this drawing as follows:



How many Full Adders to you need? 7

How many hops is your design? maximum of 5

Blank page for scratch work

Blank page for scratch work