Midterm Solution

Q1

1. In Boolean algebra, if $xy = xz$, then $y = z$, where x, y, z are three Boolean variables.
2. In Boolean algebra, if $x + y = xy$, then $x = y$, where x, y are two Boolean variables.
3. In Boolean algebra, if $x + y = 1$, $xy = 0$ and $x + z = 1$, $xz = 0$, then $y = z$, where x, y, z are three Boolean variables.

1. False. Counter-example: x = 0, y = 0, z = 1. xy = 0*0 = 0, xz = 0*1 = 0,
   so xy = xz, but y ≠ z.
2. True. If x + y = xy, then we can show i. x'(x+y)=x'xy -> x'y= 0 and ii. x'+x+y=x'+xy -> x'+y= 1, by the uniqueness of the complement y=(x')'=x.

3. True. By the uniqueness of the complement, we have y=z.

Q1

4. Karnaugh map: In a Karnaugh map, it is possible that none of the prime implicant is essential.
5. Universal set: $\{AND, OR\}$ is a universal set.
6. Universal set: $\{f(x,y,z) = xyz'\}$ is a universal set.
7. Universal set: $\{f(x,y,z) = xyz + x'y'\}$ is a universal set.

4. True, consider the following K-map:

A\B	0	1
0	x	0
1	1	x

5. False, cannot get NOT gate.

6. True, contains {AND, NOT}. AND: f(x,y,0) NOT: f(1,1,z)

7. True, contains {AND, NOT, OR}. NOT: f(0,y,0) AND: f(1,y,z)

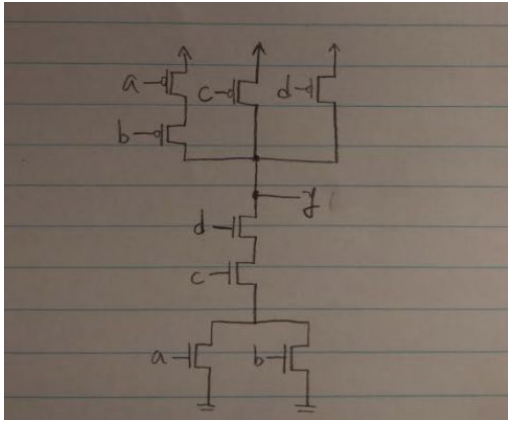
Q2

Design a CMOS gate with function $y(a,b,c,d) = [(a+b)cd]'$. Draw the schematic diagram using NMOS and PMOS transistors as the basic elements. (16 pts)

y(a,b,c,d)=[(a+b)cd]'

(1) When y = 0
(a+b)cd = 1
(a+b)=1 AND cd = 1

(2) When y = 1
(a+b)cd=0
(a+b)=0 OR c=0 OR d=0



Q3

Simplify the following two Boolean functions and state the laws/theorems in your derivation.

1. $f(a,b,c) = ab' + a'b'c + b'c$. (10 pts)
2. $f(a,b,c) = (a+bc)(a'+b'c')(bc+b'c')$. (10 pts)

1.

f(a,b,c)=ab'+a'b'c

=ab'+b'c

(consensus: b'c is consensus term)

2.

f(a,b,c)=(a+bc)(a'+b'c')(bc+b'c')

=(a+bc)(a'+b'c') (consensus)

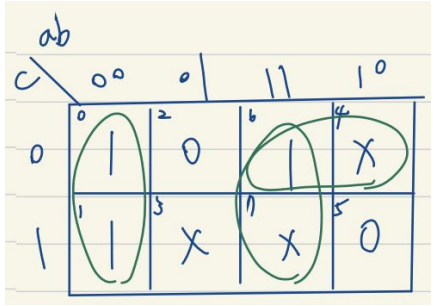
=aa'+ab'c'+a'bc+bb'cc' (distributive)

=ab'c'+a'bc (complements)

Q4

4.1 Sum of Products Expressions

Use Karnaugh map to simplify function $f(a,b,c) = \sum m(0,1,6) + \sum d(3,4,7)$.
Write the K-map and list all possible minimal sum of products expressions. No need for the logic diagram.

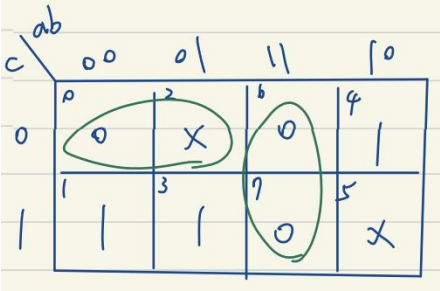


f(a,b,c)=a’b’+ab, or
=a’b’+ac’

Q4

4.2 Product of Sums Expressions

Use Karnaugh map to simplify function $f(a,b,c) = \sum m(1,3,4) + \sum d(2,5)$.
Write the K-map and list all possible minimal product of sums expressions. No need for the logic diagram.



f(a,b,c)=(a+c)(a'+b')

Q5

Consider the function $f(x,y)$ where $\oplus$ is an Exclusive OR operator and the priority of the operators is AND (first), Exclusive OR and OR (last):

$$f(x,y) = (xy) \oplus (x' + y') \oplus y \oplus x \oplus (xy) \oplus (x'y').$$

Simplify the function into a minimal sum of products expression using switching function techniques e.g. Shannon's expansion. Show your derivation. Write the solution in a Boolean expression. (20 pts)

$$\begin{aligned}
& f(x,y) = (xy) \oplus (x' + y') \oplus y \oplus x \oplus (xy) \oplus (x'y') \\
&= xf(1,y) + x'f(0,y) = x(y \oplus y' \oplus y \oplus 1 \oplus y \oplus 0) + x'(0 \oplus 1 \oplus y \oplus 0 \oplus y') \\
&= x(0 \oplus 1 \oplus 1) + x'(0 \oplus 1 \oplus 1) \\
&= x(1 \oplus 1) + x'(1 \oplus 1) \\
&= 0
\end{aligned}$$

A	B	A **XOR** B
0	0	0
0	1	1
1	0	1
1	1	0

B1

B.1. Boolean Algebra: In the framework of Boolean algebra, we would like to construct a "$-$" operator such that $(x+y)-y=x$, where x, y are Boolean variables. In other words, "$-$" is an inverse operator of the "$+$" operator. Is it possible to construct using Boolean algebra? State your choice of Yes or No. Explain your solution with no more than four sentences. (5 pts)

It is not possible to recover the original value of an operand after performing the OR operation. Let z = x + y denote the operation. Suppose y = 1. Then regardless of input x, z = 1 and consequently x = 0 and x = 1 are indistinguishable from each other

B2

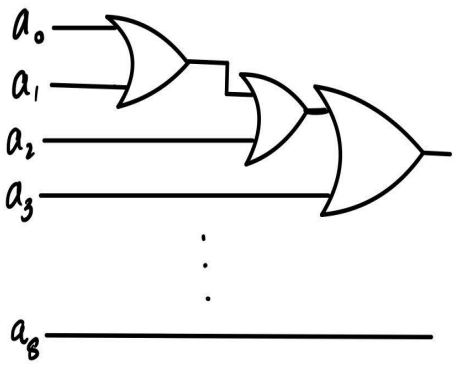
B.2. Voting machine: Given a nine input function $f(a_8, a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$, construct the function according to the following specification using a minimal network of AND, OR, NOT gates. Assuming that NOT gate is free. The network can have more than two levels of gates. Show your designs with logic diagrams. (5 pts)

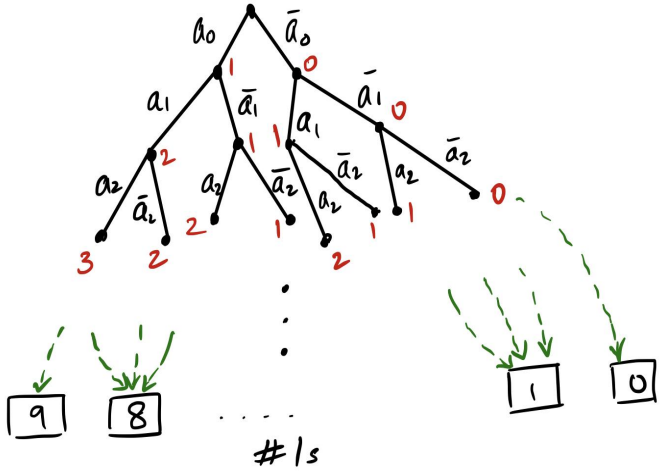
i. Function $f = 1$, when one or more inputs are true, otherwise, $f = 0$.

ii. Function $f = 1$, when two or more inputs are true, otherwise, $f = 0$.

iii. Function $f = 1$, when five or more inputs are true, otherwise, $f = 0$ (a majority function).

i. The below simple combination of OR gates suffices





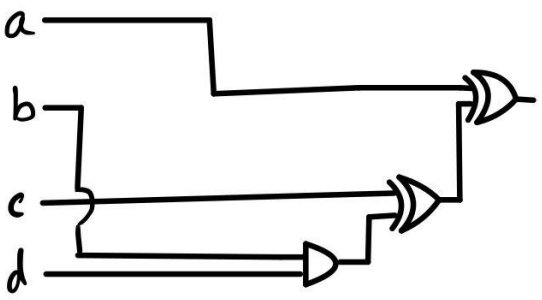
ii, iii. The minimal number of gates is obtained using a binary decision diagram that tracks the number of 1s. The diagram leaves terminate when the derived number of 1s have been obtained. The diagram may be implemented via AND & NOT gates.

Or, we can use a “bit counter” to count the number of true bits (ref. multiplication). We provide the output based upon the counted values. The diagram is shown for ii is shown to the left.

B3

B.3. Karnaugh Map: Given a four input combination function $f(a,b,c,d)$, we have On-Set $F = \sum m(2,3,5,6,8,9,12,15)$ and an empty Don't-Care set. Implement the function with {AND, XOR, NOT} gates only (no other gates). Assuming that we can have multiple (more than 2) input gates and NOT gates are free. Design with a minimal networks in terms of the total number of AND, XOR gates. Draw a logic diagram to illustrate the design. (5pts)

cd \ ab	$\bar{a}\bar{b}$	$\bar{a}b$	ab	$a\bar{b}$
$\bar{c}\bar{d}$	0 (0)	0	1 (3)	1 (2)
$\bar{c}d$	0 (4)	1 (5)	0 (7)	1 (6)
cd	1 (12)	0 (13)	1 (15)	0 (14)
$c\bar{d}$	1 (8)	1 (9)	0 (11)	0 (10)



From the above KMap we obtain

$f(a,b,c,d) = a \oplus c \oplus bd$

B4

B.4.1. Given a three-input, three-output quantum gate $f(a, b, c) = (a, b, (ab) \oplus c)$, can $\{f(a, b, c)\}$ be a universal set? State your choice of Yes or No. Explain your choice.

Yes

- AND: $f(a, b, 0) = (a, b, ab)$
- NOT: $f(1, 1, c) = (1, 1, 1 \oplus c) = (1, 1, c')$
- OR: De Morgan's Laws.

B.4.2. Given the output of the quantum gate in B.5.1, can we recover the input? State your choice of Yes or No. If Yes, provide your formula to recover the input. If No, explain your choice.

Yes. $f(a, b, c) = (x, y, z) \implies g(x, y, z) = (x, y, (xy) \oplus z)$.

Note here, x=a, y=b, z=ab xor c, thus xy xor z = c.