EE M16: Logic Design of Digital Systems	**Fall 2015**
Midterm Solutions	
Instructor: Prof. Danijela Cabric	*Scribes: Ghaith Hattab, Mihir Laghate*

1 (*10 points*). Reduce the following expression using Boolean algebra postulates and theorems. Show intermediate steps and mention the rules used at each step.

$$f(w,x,y,z) = \overline{\bar{x}\bar{y} + z} + z + xy + wz$$

The solution is as follows:

$$\begin{aligned}
f(w,x,y,z) &= \overline{\bar{x} \cdot \bar{y} + z} + z + xy + wz \\
&= \left(\overline{\bar{x} \cdot \bar{y}} \cdot \bar{z}\right) + z(1+w) + xy \quad \because \text{ De Morgan's Law} \\
&= (x+y)\bar{z} + z + xy \qquad \because \text{ De Morgan's Law} \\
&= x + y + z + xy \qquad \because a\bar{z} + z = a + z \\
&= x(1+y) + y + z \\
&= x + y + z
\end{aligned}$$

2. (*15 points*). Consider the following function

$$f(a,b,c,d) = \Sigma m(1,3,4,9,11,13).$$

(a) Use K-maps to minimize **both** the sum of products and products of sums form. Write the simplified Boolean expressions.

(b) Implement the function using minimal number of gates. You can use either NOR gates or NAND gates with maximum number of four inputs. Assume that inputs are available in both uncomplemented and complemented form.

(a)

The K-map for sum of products is shown to the right. The minimal sum of products is

$$f(a,b,c,d) = \bar{b}d + a\bar{c}d + \bar{a}b\bar{c}\bar{d}$$

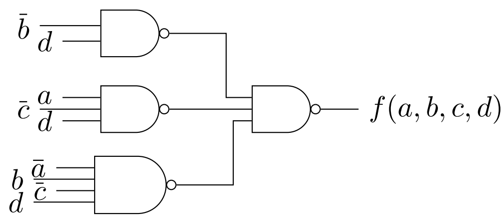
ab \ cd	00	01	11	10
00	0	1	1	0
01	1	0	0	0
11	0	1	0	0
10	0	1	1	0

The K-map for product of sums is shown to the right. The minimal product of sums is

$$f(a,b,c,d) = (\bar{b}+\bar{c}) \cdot (\bar{a}+d) \cdot (b+d) \cdot (a+\bar{b}+\bar{d})$$

ab \ cd	00	01	11	10
00	0	1	1	0
01	1	0	0	0
11	0	1	0	0
10	0	1	1	0

(b) Since the minimal sum of products form has fewer terms, it will result in a circuit with fewer gates as shown below.



3. (*20 points*). Consider the following function

$$f(a,b,c,d) = \overline{(\bar{a}b + d)(\bar{b} + \bar{d})}.$$

(a) Find minimal sum-of-products and products-of-sums expressions for the switching expression f.

(b) Determine the prime implicants and the essential prime implicants for this expression.

(c) Does this function have a unique minimal sum-of-products? If not, list any other minimal sum-of-products expressions. Does this function have a unique minimal product-of-sums? If not, list any other minimal product-of-sums expressions.

(a) First, simplify the given expression:

$$\begin{aligned} f(a,b,c,d) &= \overline{\bar{a}b} \cdot \bar{d} + bd \\ &= (a + \bar{b})\bar{d} + bd \\ &= a\bar{d} + \bar{b} \cdot \bar{d} + bd. \end{aligned}$$

Using this expression, we construct the K-map on the right. Note that this expression is the minimal sum of products expression for f.
Also from the K-map, we can see that the minimal product of sums expression is:

$$f(a, b, c, d) = (b + \bar{d}) \cdot (a + \bar{b} + d)$$

ab \ cd	00	01	11	10
00	1	0	0	1
01	0	1	1	0
11	1	1	1	1
10	1	0	0	1

Note that since the function f does not depend on the input c, we can also construct a K-map using just a, b, and d. The resulting algebraic expressions do not change. HOWEVER, the prime implicants and essential prime implicants still need to use all 4 variables. Due to this reason, we avoid using the K-map of just a, b, and d.

a \ bd	00	01	11	10
0	1	0	1	0
1	1	0	1	1

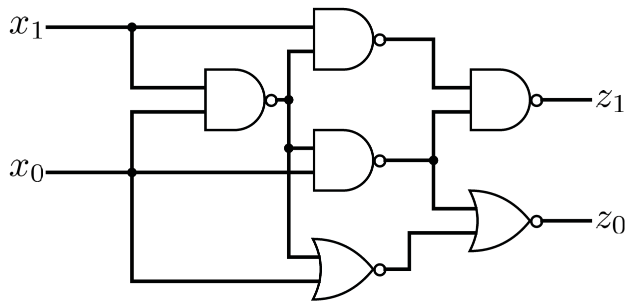
(b) The prime implicants are x1x1, 11xx x0x0, and 1xx0.
The essential prime implicants are x1x1 and x0x0.

(c) No, this function does not have a unique minimal sum-of-products expression. Apart from the expression given above, another minimal sum of products expression of f is

$$f(a, b, c, d) = ab + \bar{b}\bar{d} + bd$$

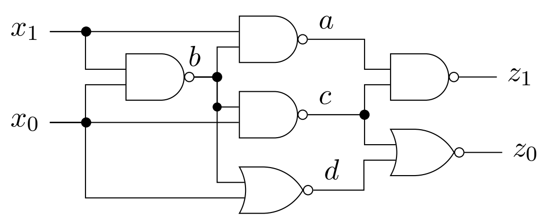
Yes, this function has a unique minimal product of sums expression.

4. (*10 points*). Analyze the NAND-NOR network shown in the figure below. Obtain switching expressions for the outputs.



We provide two solutions for this problem.

First, label the outputs of each gate as shown below.



Algebraically, we can analyze this circuit as follows.

$$a = \overline{x_1 b}, b = \overline{x_0 x_1}, c = \overline{b x_0}, d = \overline{b + x_0}$$
$$z_1 = \overline{ac}, z_0 = \overline{c + d}$$

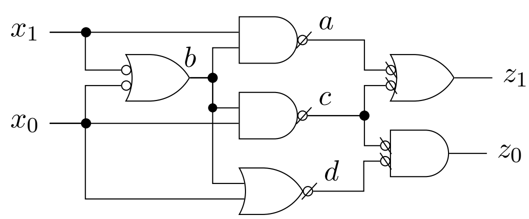
By substituting the appropriate variables, we get

$$\begin{aligned} z_1 &= \overline{\overline{x_1 b} \cdot \overline{b x_0}} \\ &= x_1 b + b x_0 \qquad \because \text{ De Morgan's Law} \\ &= (x_1 + x_0) b \\ &= (x_1 + x_0)(\overline{x_1 x_0}) \\ &= (x_1 + x_0)(\overline{x_1} + \overline{x_0}) \qquad \because \text{ De Morgan's Law} \\ &= x_1 \oplus x_0 \end{aligned}$$

and

$$
\begin{aligned}
z_0 &= \overline{\overline{bx_0} + \overline{b + x_0}} \\
&= bx_0(b + x_0) \qquad \because \text{ De Morgan's Law} \\
&= bx_0b + bx_0x_0 \\
&= bx_0 + bx_0 \qquad \because w \cdot w = w \\
&= bx_0 \qquad \because w + w = w \\
&= \overline{x_1x_0}x_0 \\
&= (\overline{x_1} + \overline{x_0})x_0 \qquad \because \text{ De Morgan's Law} \\
&= \overline{x_1}x_0 + \overline{x_0}x_0 \\
&= \overline{x_1}x_0 \qquad \because \overline{w}w = 0.
\end{aligned}
$$

Alternately, by propagating the inversion circles backwards, we get the following circuit. The NOT operations that cancel out have been denoted by crossed out circles.



We can now express the gate outputs as:

$$
\begin{aligned}
a &= x_1b, b = \overline{x_1} + \overline{x_0}, c = bx_0, d = b + x_0 \\
z_1 &= a + c, z_0 = cd.
\end{aligned}
$$

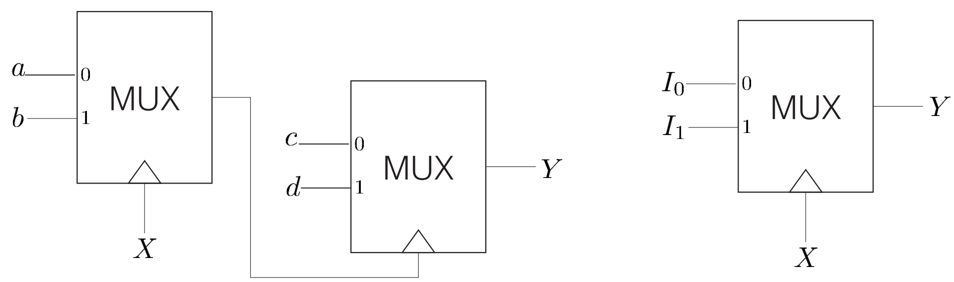
Repeated substitution gives us the required expressions:

$$
\begin{aligned}
z_1 &= a + c \\
&= x_1b + bx_0 \\
&= b(x_0 + x_1) \\
&= (\overline{x_0} + \overline{x_1})(x_0 + x_1) \\
&= x_0 \oplus x_1
\end{aligned}
$$

and

$$\begin{aligned} z_0 &= cd \\ &= bx_0(b + x_0) \\ &= bx_0b + bx_0x_0 \\ &= bx_0 + bx_0 \qquad \because w \cdot w = w \\ &= bx_0 \qquad \because w + w = w \\ &= (\overline{x_0} + \overline{x_1})x_0 \\ &= \overline{x_0}x_0 + \overline{x_1}x_0 \\ &= \overline{x_1}x_0 \qquad \because \overline{w}w = 0. \end{aligned}$$

5. (*15 points*). Consider the two circuits



(a) Find I_0, I_1, so that both circuits are equivalent.

(b) Write the Boolean expression of Y in terms of X, a, b, c, and d.

(a) When $X = 0$, then $g = a$ and $Y = I_0$. Hence, $I_0 = \bar{a}c + ad$.
When $X = 1$, then $g = b$ and $Y = I_1$. Hence, $I_1 = \bar{b}c + bd$.
(b) The switching expression for the circuit on the right is

$$Y = \overline{X}I_0 + XI_1$$
$$\therefore Y = \overline{X}(\bar{a}c + ad) + X(\bar{b}c + bd)$$

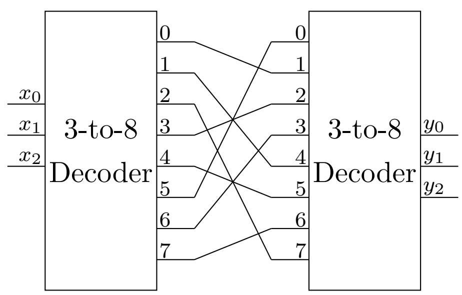
6. (*15 points*).Design a combinational network that has a three-bit input x representing the digits 0 to 7, and a three-bit output y representing the same set of integers. The function of the system is

$$y = (3x + 1) \mod 8.$$

Use a binary decoder, a binary encoder, and (one or more) OR gates.

First, create an input-output table as shown to the right. Note that each input has a unique output.
So, we are going to use a binary 3-to-8 encoder to obtain a 1-hot representation of the input and then connect the output of the encoder to the input of a 8-to-3 decoder as per this table. The circuit is shown below.

Input (x)	Output (y)
0	1
1	4
2	7
3	2
4	5
5	0
6	3
7	6



7. (*15 points*). Compute $z = a + 2b - c$ in 2's complement for $a = -7$, $b = 12$, and $c = -97$. Perform calculations on bit-vectors representing a, b, and c and show every step of your work. How many bits should z have to represent the correct result? Check your work by showing, for each step, the corresponding values in decimal number system.

Of the given input numbers, c has the largest magnitude. 7-bits are required to represent $|c|$. Hence, including the sign bit, we use a 8-bit representation. At the end, we will check to see if 8-bits are sufficient to store z.

First, lets compute the 8-bit 2's complement representation of the inputs:

$$a = -7 \Rightarrow \underline{a} = 1111\ 1001$$
$$b = 12 \Rightarrow \underline{b} = 0000\ 1100$$
$$c = -97 \Rightarrow \underline{c} = 1001\ 1111.$$

Next, we compute z as $z = 2b + a + (-c)$. We do this rearrangement because multiplication should occur first.

Algebraic Operation	2's Complement Representation	Decimal Representation
b	0000 1100	12
$2b$	0001 1000	24
a	1111 1001	-7
$2b + a$	0001 0001	17
c	1001 1111	-97
$-c$	0110 0001	97
$2b + a$	0001 0001	17
$-c$	0110 0001	97
$2b + a - c$	0111 0010	114

Since $z = 114 < 128 = 2^7$, it is correct to use 8-bits for representing z.