CSE 232 SPRING 2020

HOMEWORK 2

Abdullah ÇELİK

171044002

1. Assume for a particular year that a particular size chip using state-of-the-art technology can contain 1 billion transistors. Assuming Moore's Law (doubling each 18 months) holds, how many transistors will the same size chip be able to contain in ten years?

1 billion = 10^9
$10^9 * 2^{10*12/18} = 101{,}596 * 10^9 = 101{,}596$ billion transistors

2. Evaluate the Boolean equation `F` = (`a` AND `b`) OR `c` OR `d` for the given values of variables `a`, `b`, `c`, and `d`:

a. `a=1`, `b=1`, `c=1`, `d=0`
b. `a=0`, `b=1`, `c=1`, `d=0`
c. `a=1`, `b=1`, `c=0`, `d=0`
d. `a=1`, `b=0`, `c=1`, `d=1`

Firstly, calculates inside of parentheses evaluation. Then all the logical Gates are OR Gates, we can evaluate two variables. I choose the first two variables from left.

a. a = 1, b = 1, c = 1, d = 0
F = (1 AND 1) OR 1 OR 0
F = 1 OR 1 OR 0
F = 1 OR 0
F = 1

b. a = 0, b = 1, c = 1, d = 0
F = (0 AND 1) OR 1 OR 0
F = 0 OR 1 OR 0
F = 1 OR 0
F = 1

c. a = 1, b = 1, c = 0, d = 0
F = (1 AND 1) OR 0 OR 0
F = 1 OR 0 OR 0
F = 1 OR 0
F = 1

d. a = 1, b = 0, c = 1, d = 1
F = (1 AND 0) OR 1 OR 1
F = 0 OR 1 OR 1
F = 1 OR 1
F = 1

3. For the function `F = a + a′b + acd + c′`:

a. List all the variables.
{a, b, c, d}

b. List all the literals.
{a, a′, b, a, c, d, c′}

c. List all the product terms.
{a, a′b, acd, c′}

4. Convert the function `F` shown in the truth table in the table to an equation. Don't minimize the equation.

a	b	c	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

All cases where F is true are written as logical equations. Then these literals are combined by OR gate

F = a′b′c + a′bc′ + a′bc + ab′c + abc′ + abc

5. Use algebraic manipulation to minimize the equation obtained in Exercise 4.

F = a′b′c + a′bc′ + a′bc + ab′c + abc′ + abc

Since all Gates are OR Gates, I change the order of product terms.
F = a′b′c + ab′c + a′bc′ + a′bc + abc′ + abc

I put the first two product terms in b′c brackets. Then i put the next two product terms in a′b parenthesis. I put the last two product terms in ab parenthesis.
F = b′c * (a′ + a) + a′b * (c′ + c) + ab * (c′ + c)

I am editting the equations using E + E′ = 1 equation.
F = b′c * 1 + a′b * 1 + ab * 1

I am editting the equations using E * 1 = E
F = b′c + a′b + ab

I put the last two product terms in b parenthesis.
F = b′c + b * (a′ + a)

Using E + E′ = 1, equation is
F = b′c + b * 1

Using E * 1 = E, equation is
F = b′c + b

Distributive rule and equation is
F = (b′ + b) * (b + c)

Using E + E′ = 1, equation is
F = 1 * (b + c)

Using E * 1 = E, equation is
F = b + c

6. Determine whether the Boolean functions F = (a + b)′*a and G = a + b′ are equivalent, using:
(a) algebraic manipulation, and (b) truth tables.

(a) algebraic manipulation

First, apply the De Morgan rule in F function. De Morgan (a + b)′ = a′b′
F = a′b′ * a

All Gates are AND gate. I change the order of literals.
F = a′a b′

Using E * E′ = 0, equation is
F = 0 * b′

Using 0 * E = 0, equation is
F = 0

G = a + b′ = a * (b + b′) + b′ * (a + a′) = ab + ab′ + a′b′

F and G are not equivalent.

(b) truth tables.

F = (a + b)' * a				
a	b	a + b	(a + b)'	(a + b)' * a
1	1	1	0	0
1	0	1	0	0
0	1	1	0	0
0	0	0	1	0

G = a + b'			
a	b	b'	a + b'
1	1	0	1
1	0	1	1
0	1	0	0
0	0	1	1

F and G are not equivalent.

7. Using the combinational design process, create a 4-bit prime number detector. The circuit has four inputs, `N3`, `N2`, `N1`, and `N0` that correspond to a 4-bit number (`N3` is the most significant bit) and one output `P` that is `1` when the input is a prime number and that is `0` otherwise.

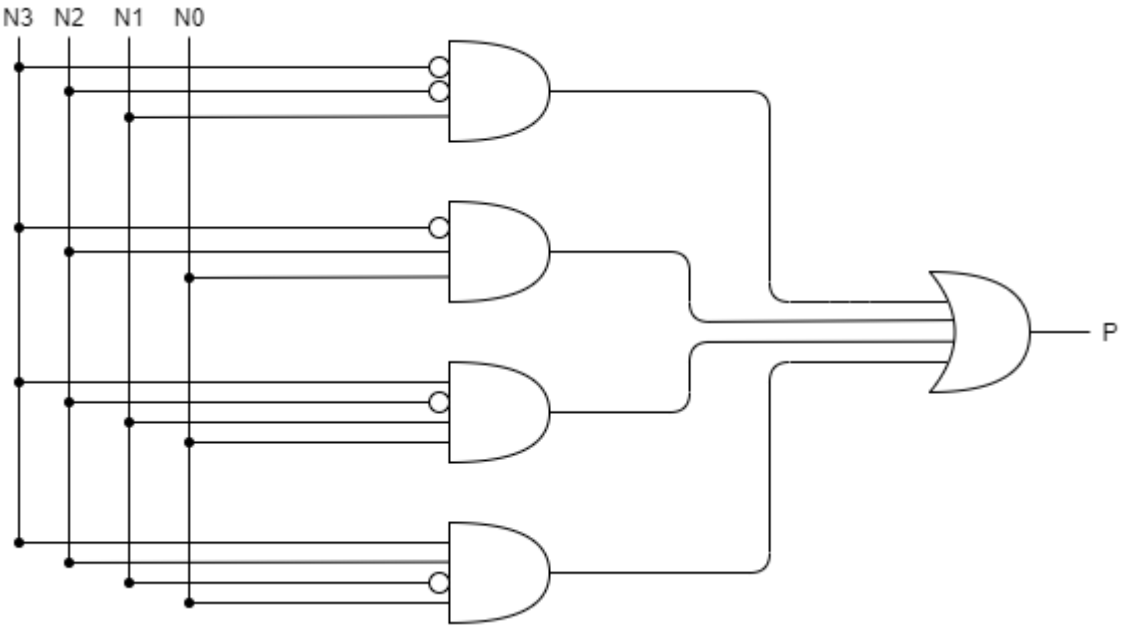
N3	N2	N1	N0	P
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

P = N3′N2′N1N0′ + N3′N2′N1N0 + N3′N2N1′N0 + N3′N2N1N0 + N3N2′N1N0 + N3N2N1′N0

= N3′N2′N1 * (N0′ + N0) + N3′N2N0 * (N1′ + N1) + N3N2′N1N0 + N3N2N1′N0

= N3′N2′N1 * 1 + N3′N2N0 * 1 + N3N2′N1N0 + N3N2N1′N0

= N3′N2′N1 + N3′N2N0 + N3N2′N1N0 + N3N2N1′N0



8. A network router connects multiple computers together and allows them to send messages to each other. If two or more computers send messages simultaneously, the messages "collide" and the messages must be resent. Using the combinational design process of Table 2.5, create a collision detection circuit for a router that connects 4 computers. The circuit has 4 inputs labeled `M0` through `M3` that are `1` when the corresponding computer is sending a message and `0` otherwise. The circuit has one output labeled `C` that is `1` when a collision is detected and `0` otherwise.

M0	M1	M2	M3	C
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

C′ = M0′M1′M2′M3′+M0′M1′M2′M3+M0′M1′M2M3′+M0′M1M2′M3′+M0M1′M2′M3′

C′ = M0′M1′M2′ * (M3′ + M3) + M0′M1′M2M3′ + M0′M1M2′M3′ + M0M1′M2′M3′

C′ = M0′M1′M2′ * 1 + M0′M1′M2M3′ + M0′M1M2′M3′ + M0M1′M2′M3′

C′ = M0′M1′M2′ + M0′M1′M2M3′ + M0′M1M2′M3′ + M0M1′M2′M3′