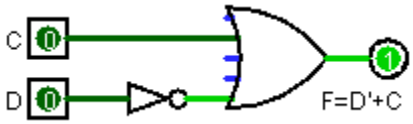
Andy Lee
500163559
Assignment 3

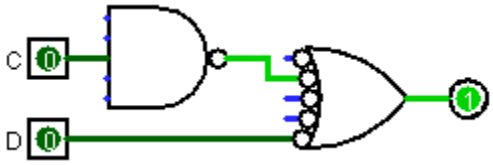
Question 1a

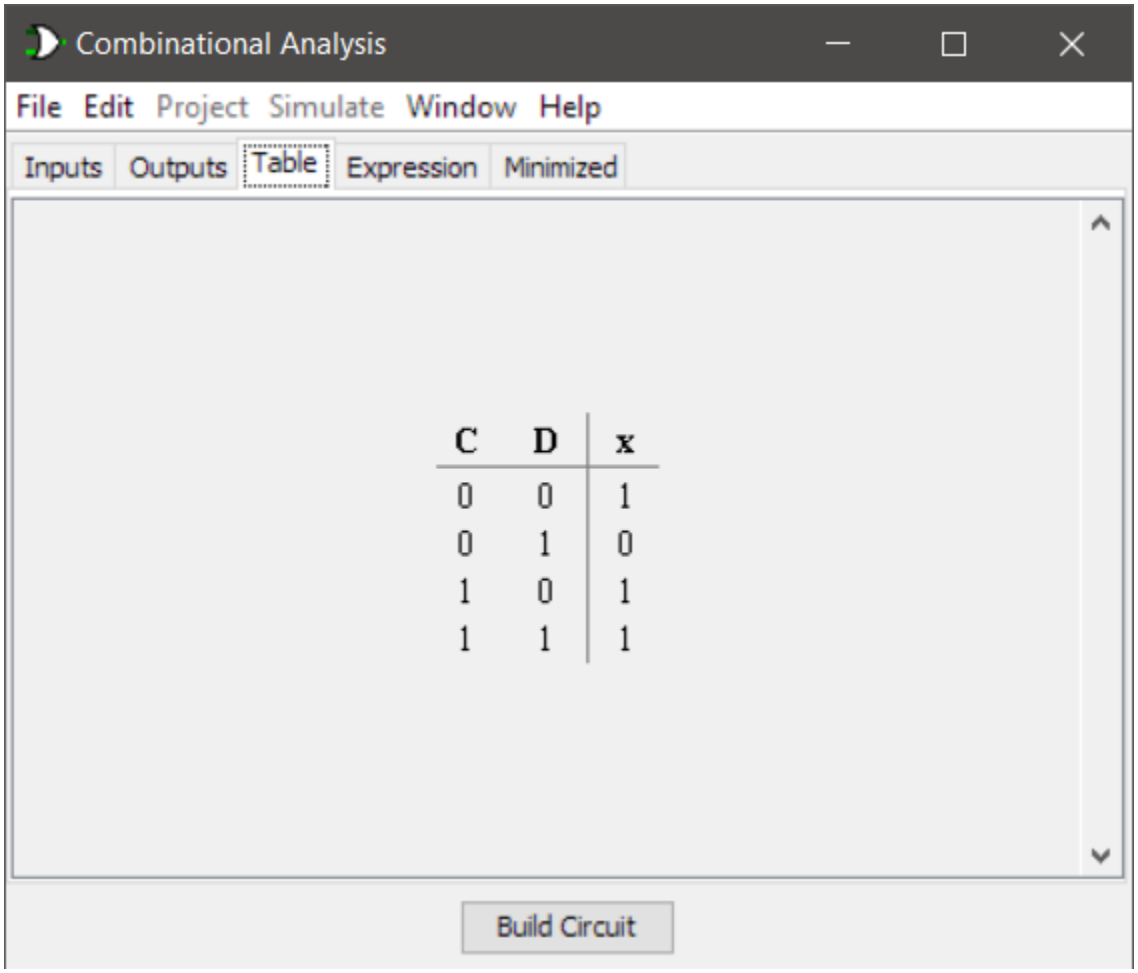
$$F(ABCD) = AC'D + A'C + ABC + AB'C + A'C'D'$$

AB\CD	C'D'	C'D	CD	CD'
A'B'	1		1	1
A'B	1		1	1
AB	1		1	1
AB'	1		1	1

$$F = D' + C$$





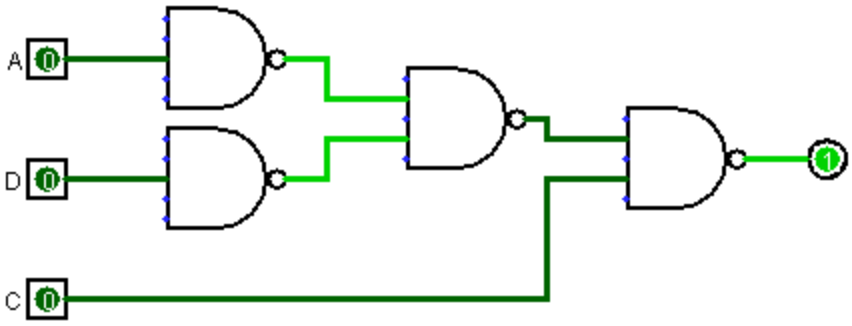


Question 1b

$$F(ABC) = (A' + C' + D')(A' + C')(C' + D')$$
$$F' = [(A' + C' + D')(A' + C')(C' + D')]'$$
$$F' = (A' + C' + D')' + (A' + C')' + (C' + D')'$$
$$F' = ACD + AC + CD$$

A\CD	C'D'	C'D	CD	CD'
A'	1	1	0	1
A	1	1	0	0

$$F = C' + A'D'$$



Question 2. Build half adders (HA) using NOR gate; construct 4-bit adders using built HA.

Step 1. Build carry and sum circuits with NOR gates base on the following truth table.

X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$C(x,y) = xy$$

x\y	y	y'
x'	0	0
x	0	1

$Collect\ 0s = C' = x' + y'$

$(C')' = (x' + y')' = xy = (xy)'' = (x' + y')'$

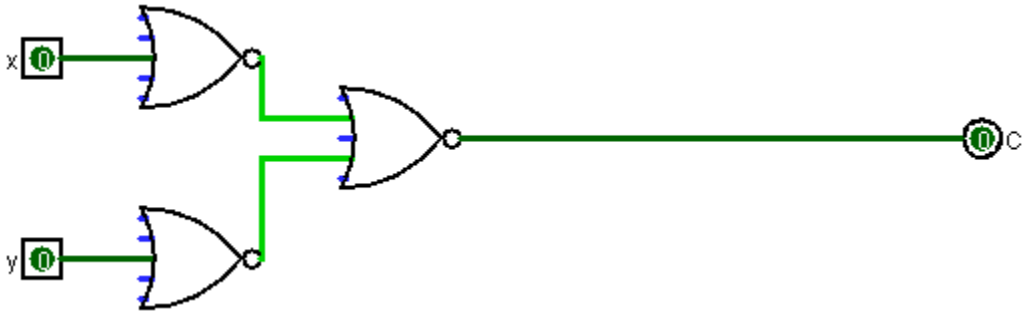


*Figure 1 Diagram of **carry** circuit*

$$S(x,y) = x'y + xy'$$

x\y	y	y'
x'	0	0
x	0	1

$Collect\ 0s = S' = x'y' + xy$

$$
\begin{aligned}
(S')' &= (x+y)(x'+y') \\
&= \big((x+y)(x'+y')\big)'' \text{ apply double negation} \\
&= ((x+y)' + (x'+y')')'
\end{aligned}
$$

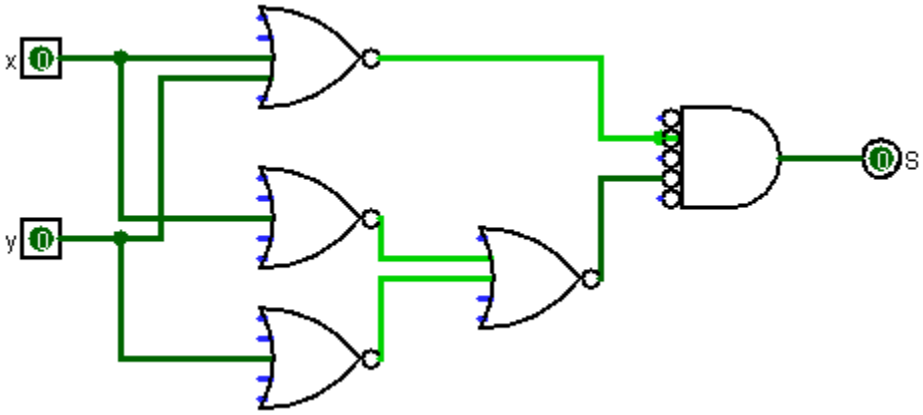


*Figure 2 Diagram for **sum** circuit*

Step 2. Next construct half adder using C and S.

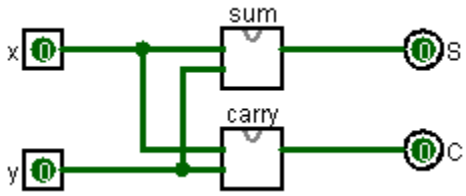
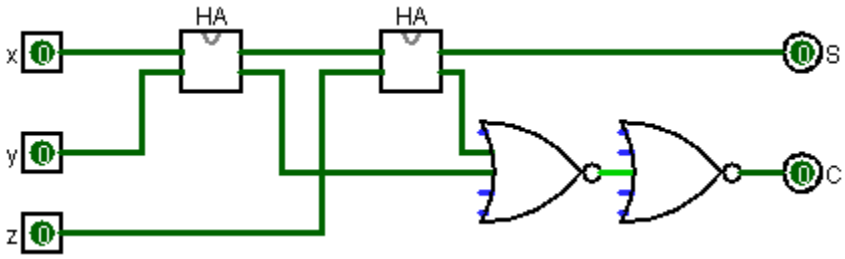


Figure 3 Diagram of half adder

Step 3. Construct full adder (FA) using on C and S and NOR gates were applicable. Design taken from lecture notes and modified to NOR gate only.



Step 4. Use FA as component and repeat 4 times to build a 4-bit adder.

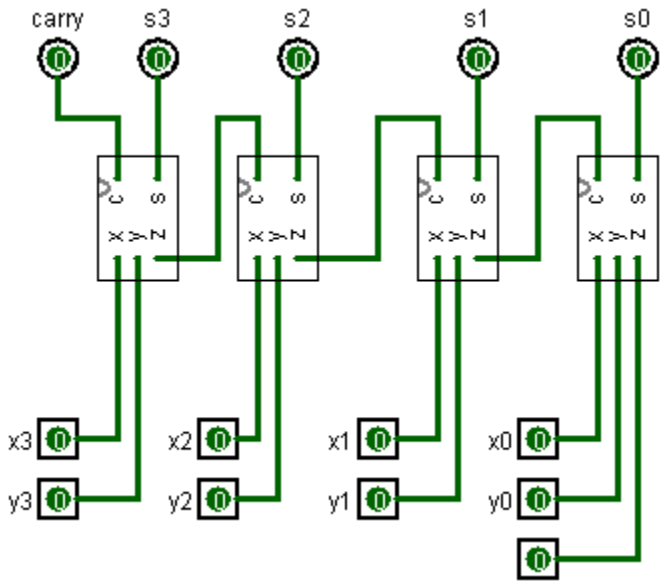


Figure 4 diagram of 4bit adder

Note: logicism randomly refuse to give an output and rebuilding the exact same circuitry or restarting the program fixes the program. So I've included samples below.

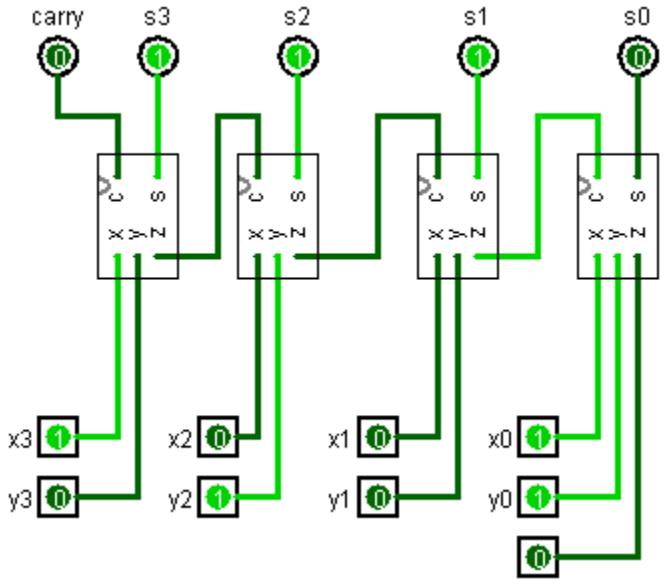


Figure 5 demo of 1001+101

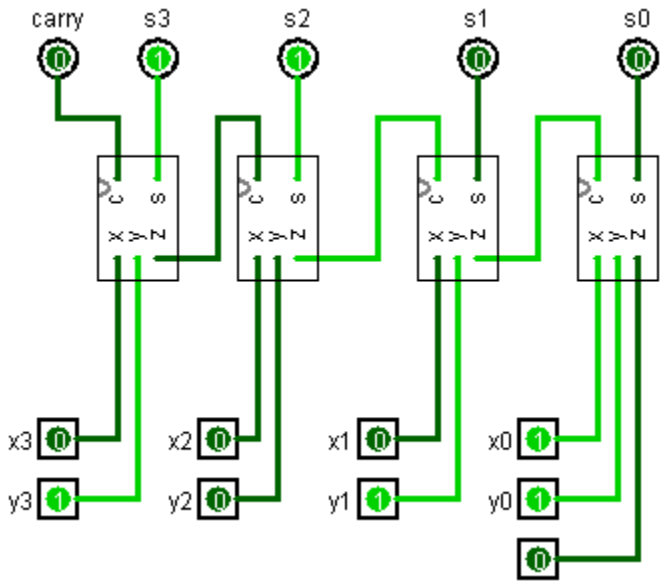


Figure 6 demo of 1+1011

Q3 Build 4-bit binary to gray code converter from 0 to 9 using NAND gates only.

	$Binary(i_3, i_2, i_1, i_0)$	O_3	O_2	O_1	O_0
0	0000	0	0	0	0
1	0001	0	0	0	1
2	0010	0	0	1	1
3	0011	0	0	1	0
4	0100	0	1	1	0
5	0101	0	1	1	1
6	0110	0	1	0	1
7	0111	0	1	0	0
8	1000	1	1	0	0
9	1001	1	1	0	1
10	1010	X	X	X	X
11	1011	X	X	X	X
12	1100	X	X	X	X
13	1101	X	X	X	X
14	1110	X	X	X	X
15	1111	X	X	X	X

$$O_3(i_3, i_2, i_1, i_0) = wx'y'z' + wx'y'z$$
$$O_2(i_3, i_2, i_1, i_0) = w'xy'z' + w'xy'z + w'xyz' + w'xyz + wx'y'z' + wx'y'z$$
$$O_1(i_3, i_2, i_1, i_0) = w'x'yz' + w'x'yz + w'xy'z' + w'xy'z$$
$$O_0(i_3, i_2, i_1, i_0) = w'x'y'z + w'x'yz' + w'xy'z + w'xyz' + wx'y'z$$

wx\yz			y		
	0	0	0	0	
	0	0	0	0	x

w	x	x	x	X	
	1	1	x	x	
		z			

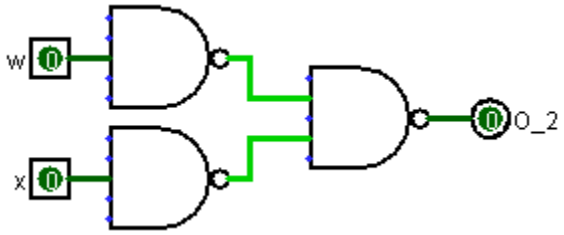
Figure 7 k-map for i_3

$$O_3 = w$$

wx\yz			y		
	0	0	0	0	
	1	1	1	1	x
w	x	x	x	X	
	1	1	x	x	
		z			

Figure 8 k-map for i_2

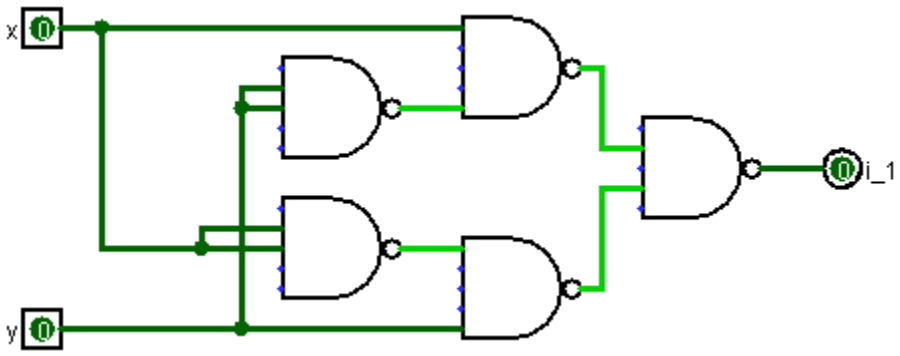
$$O_2 = w + x$$



wx\yz			y		
	0	0	1	1	
	1	1	0	0	x
w	x	x	x	X	
	0	0	x	x	
		z			

Figure 9 k-map for i_1

$$O_1 = xy' + x'y$$



wx\yz			y		
	0	1	0	1	
	0	1	0	1	x
w	x	x	x	X	
	0	1	x	x	
		z			

Figure 10 k-map for i_0

$O_0 = y'z + yz'$ ← Boolean logic same as O_1, therefore will be reusing circuitry.

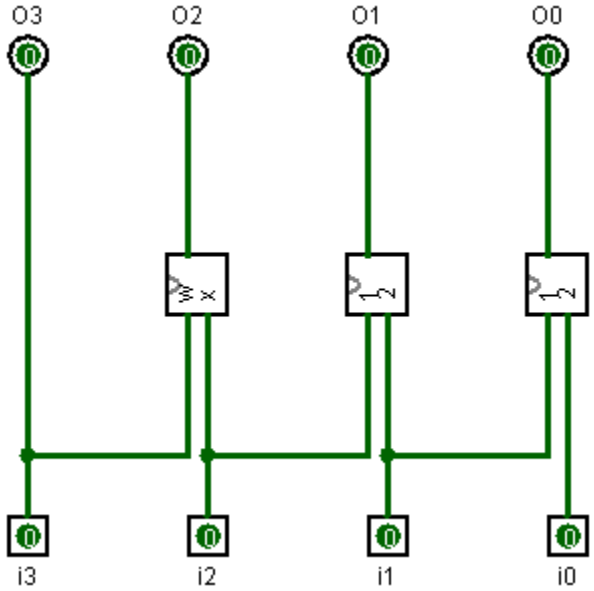


Figure 11 4bit to gray converter

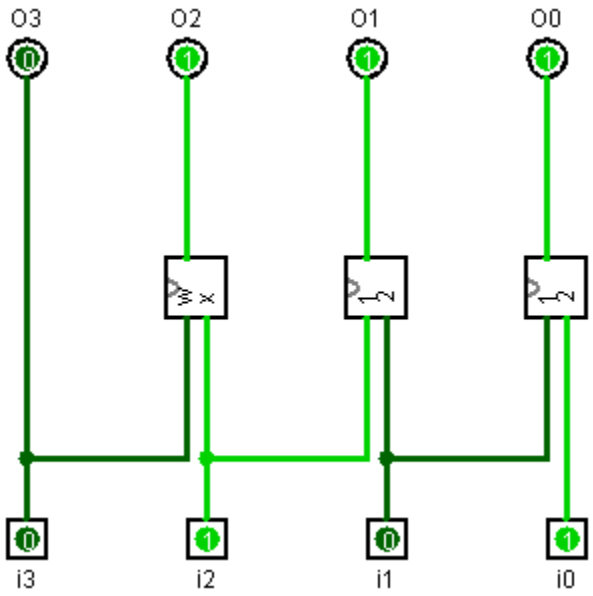


Figure 12 gray converter demo 1

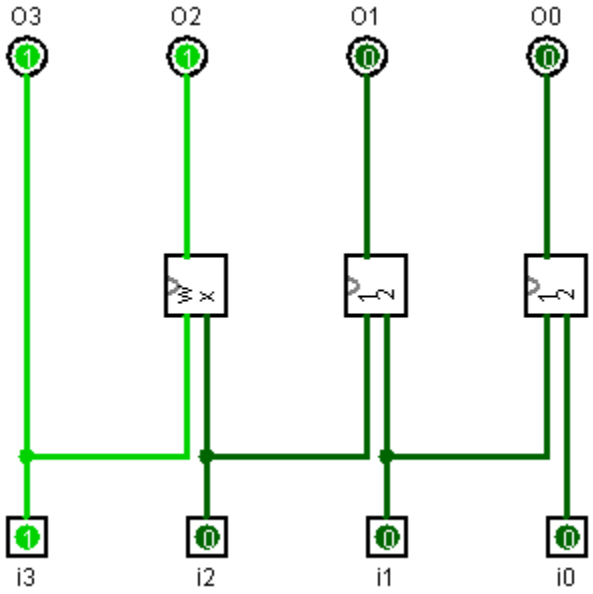


Figure 13 gray converter demo 2