CS 303 LAB2 PRE-LAB REPORT

Elif Cemre Durgut – 26493

Pre-Note: To run my circuit, lab2.dig and decoder.dig should be located in the same directory since lab2.dig uses decoder.dig as a component.

First of all, to add two 3-bit signed 2's complement binary numbers, we need 2 full-adders and 1 half adder. For this purpose, I used the adder component from Digital but there is no half adder, so I used the full adder with a constant carry value 0.

The first adder should take the least significant bits and sends the carry to the next. The second adder takes the middle bits and sends the carry to the final adder. And the last adder takes the most significant two bits.

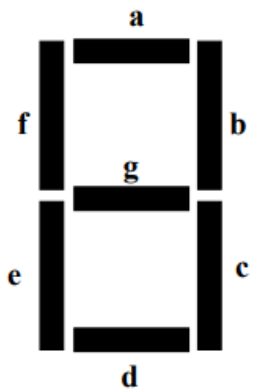
I searched for 7-segment decoders in Digital. There exists 7448 however it considers binary numbers as unsigned, but I need them to be signed. That's why I wrote my own 7-segment decoder for 2's complement.

The truth table is as follows:

For 3-bit binary numbers, the range for possible number is [-4, 3]. Therefore, I did **not** consider the numbers out of this range.

CBA is the result of the addition operation, it is 3-bit signed binary number. a, b, c, d, e, f, g corresponds to edges of the 7-segment display component.

	C	B	A	a	b	c	d	e	f	g	sign
0	0	0	0	1	1	1	1	1	1	0	0
1	0	0	1	0	1	1	0	0	0	0	0
2	0	1	0	1	1	0	1	1	0	1	0
3	0	1	1	1	1	1	1	0	0	1	0
-4	1	0	0	0	1	1	0	0	1	1	1
-3	1	0	1	1	1	1	1	0	0	1	1
-2	1	1	0	1	1	0	1	1	0	1	1
-1	1	1	1	0	1	1	0	0	0	0	1



Then, I converted this truth table into K-maps for each edge.

for a :

AB \ C	0	1
00	1	0
01	1	1
11	1	0
10	0	1

for b:

AB \ C	0	1
00	1	1
01	1	1
11	1	1
10	1	1

$a = A'C' + A'B + BC' + AB'C$

$b = 1$

$c = A + B'$

$d = a$

$e = A'C' + A'B$

$f = A'B'$

$g = BC' + A'C + B'C$

$sign = C$

for c:

AB \ C	0	1
00	1	1
01		
11	1	1
10	1	1

for d:

AB \ C	0	1
00	1	
01	1	1
11	1	
10		1

for e:

AB \ C	0	1
00	1	
01	1	1
11		
10		

for f:

AB \ C	0	1
00	1	1
01		
11		
10		

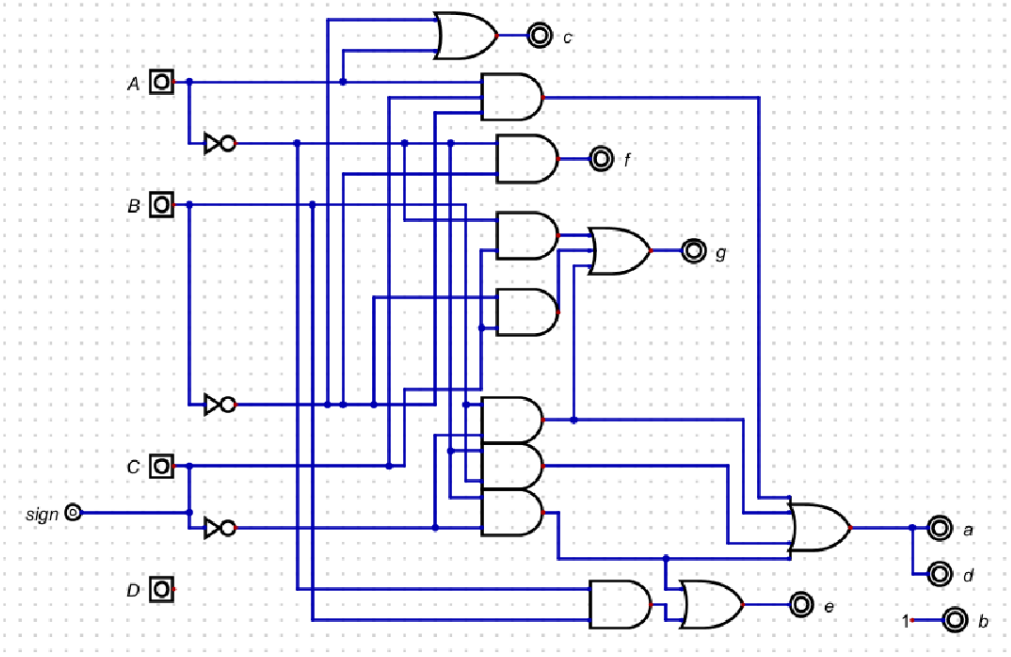
for g:

AB \ C	0	1
00		1
01	1	1
11	1	
10		1

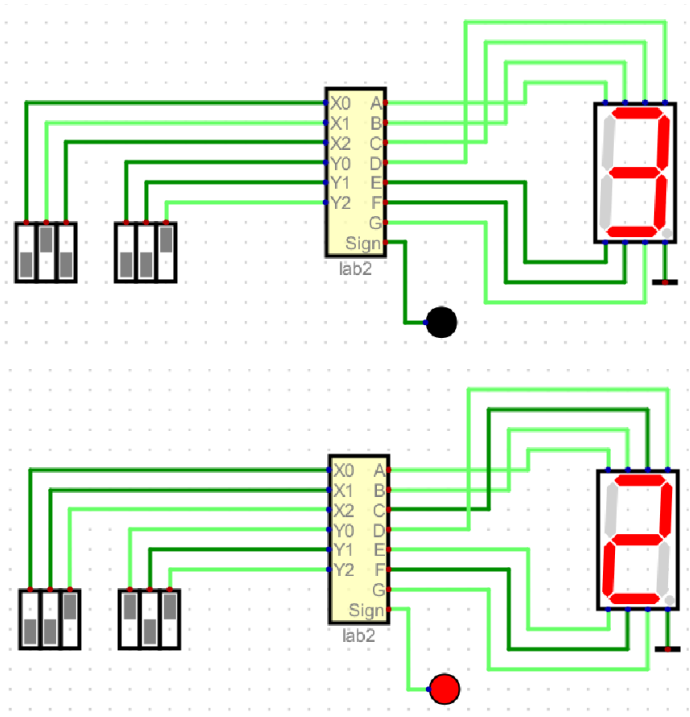
for sign:

AB \ C	0	1
00	0	1
01	0	1
11	0	1
10	0	1

After I converted each edge K-map to Boolean algebra, I constructed the logic circuit of my encoder as you can see below:



The test cases in the pdf document works:



Note: I am aware that it shows 1 for 101 (-3) and 100 (-4). The expected result in decimals is -7 however, the pdf document says that it is 3-bit adder, that's why the result is in 3-bit too.