TAC-SIM II

ALUMAP TRUTH TABLE

Functions	o0	o1	o2	o3	msk	s0	s1	s2	s3	m
ADD	0	0	0	0	0	0	0	0	x	0
SUB	0	0	0	1	0	0	0	1	x	1
INC	0	0	1	0	0	0	1	0	0	0
DEC	0	0	1	1	0	0	1	0	1	1
AND	0	1	0	0	1	0	1	1	x	x
OR	0	1	0	1	1	1	0	0	x	x
XOR	0	1	1	0	1	1	0	1	x	x
ROR	0	1	1	1	1	1	1	0	x	x
ROL	1	0	0	0	1	1	1	1	0	x
ASL	1	0	0	1	1	1	1	1	1	x
	1	0	1	0						
	1	0	1	1						
	1	1	0	0						
	1	1	0	1						
	1	1	1	0						
	1	1	1	1						

Reasoning: s3 and m line up and are functionally the same, so I ended up combining them into just m.

I combined Inc and Dec into 1 Mux position, since the only difference between them is Subtraction being pushed in. I also combined ROL and ASL into 1 mux position and added in a 2 to 1 mux to switch between the two outputs (either the Cin or 0 depending on the function)

I used m/s3 to indicate subtraction for SUB and DEC, and kept ADD & SUB mapped to the same o to s position. The Mux positions changed between the o and s numbers for the rest, to account for INC/DEC and ROL/ASL combined to fit into the 8 to 1 Mux.

K-MAPS

msk

o2,o3 \ o0,o1	00	01	11	10
00	0	1	x	1
01	0	1	x	1
11	0	1	x	x
10	0	1	x	x

s0

o2,o3 \ o0,o1	00	01	11	10
00	0	0	x	1
01	0	1	x	1
11	0	1	x	x
10	0	1	x	x

s1

o2,o3 \ o0,o1	00	01	11	10
00	0	1	x	1
01	0	0	x	1
11	1	1	x	x
10	1	0	x	x

s2

o2,o3 \ o0,o1	00	01	11	10
00	0	1	x	1
01	1	0	x	1
11	0	0	x	x
10	0	1	x	x

S3/m

o2,o3 \ o0,o1	00	01	11	10
00	0	x	x	x
01	1	x	x	1
11	1	x	x	x
10	0	x	x	x

EQUATIONS

msk is independent of the inputs. (I provided the equation for msk below, regardless, but as you can see, the k-map shows it doesn’t rely on any of the inputs and that is how I approached its implementation.)

$msk = o0\ o1'\ o2'\ o3 + o0'\ o1'\ o2\ o3$

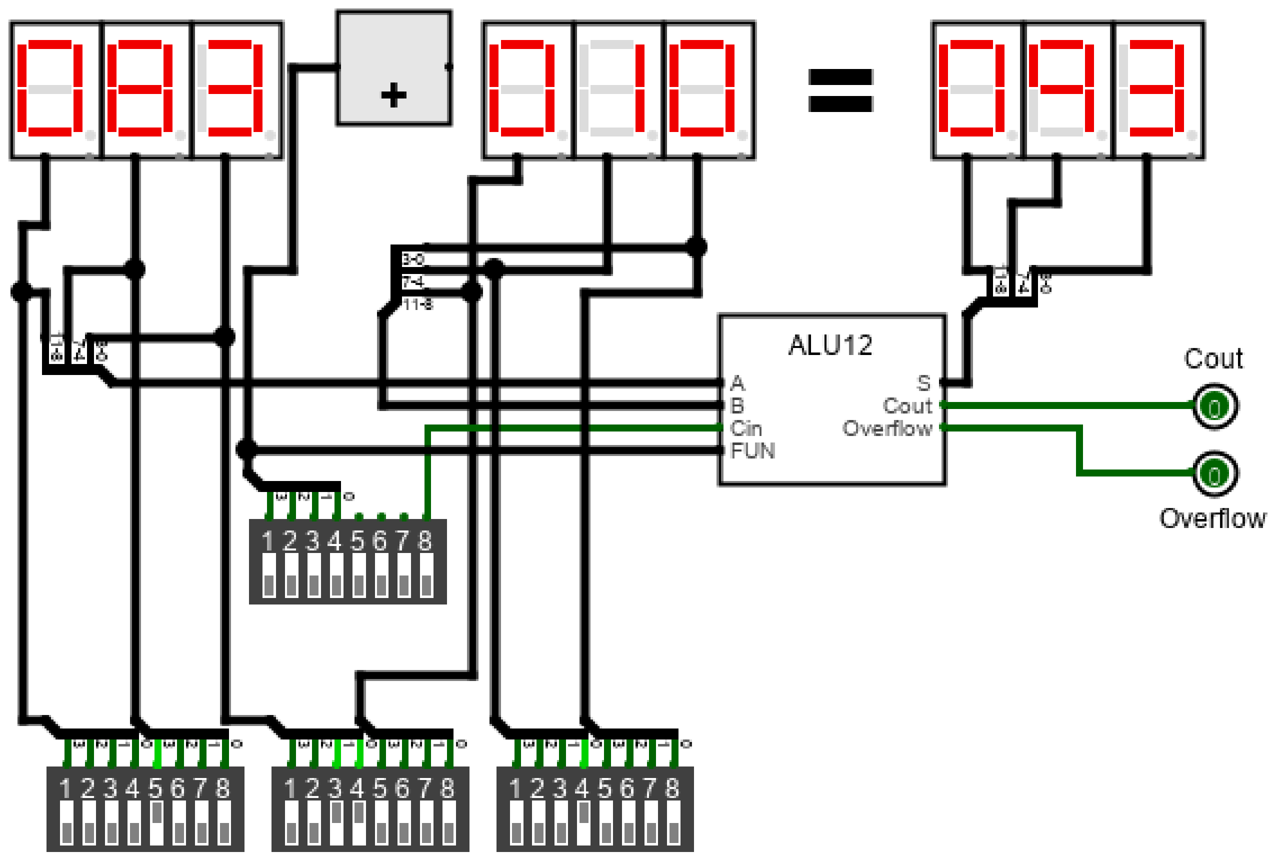
$s0 = o0\ o1'$

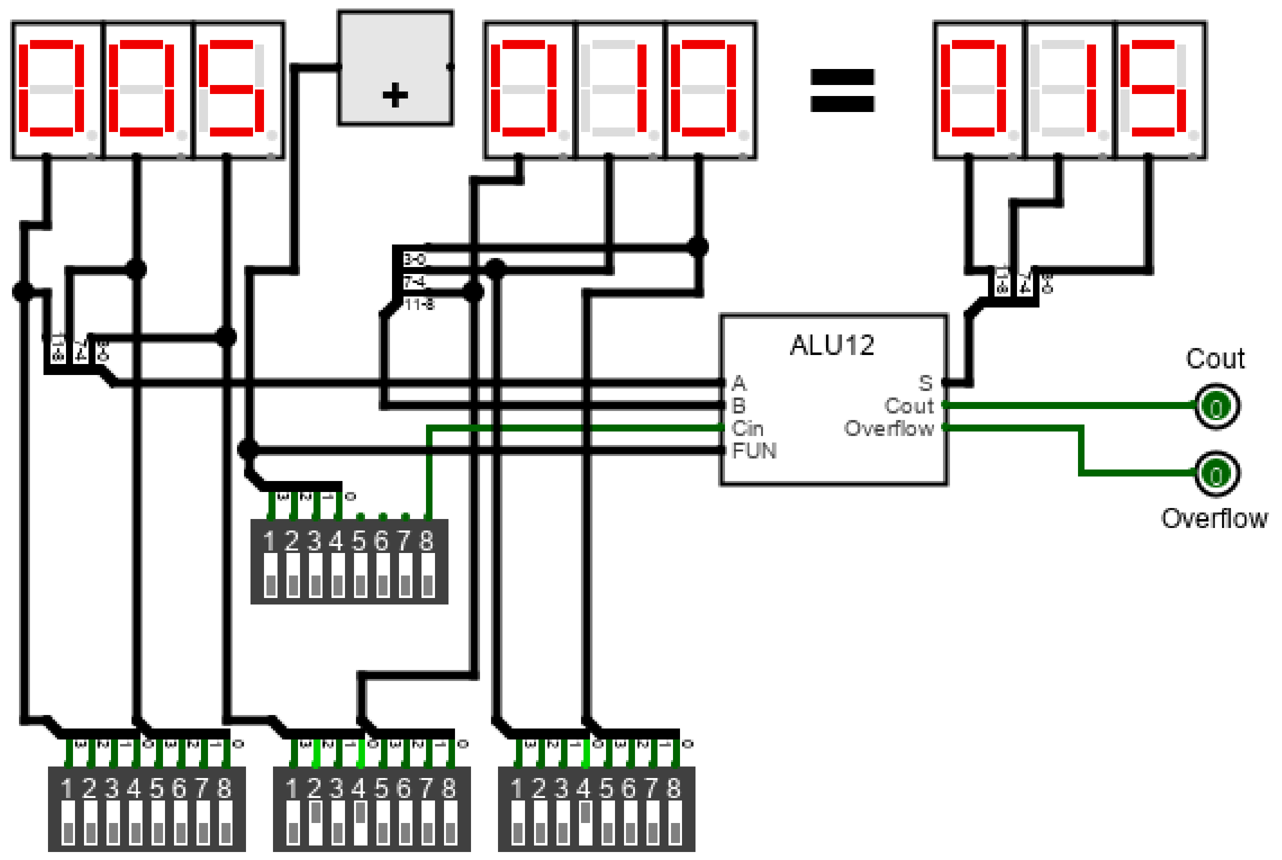
$s1 = o1\ o2'\ o3' + o0'\ o1'\ o2 + o2\ o3 + o0$

$s2 = o1'\ o2'\ o3 + o1\ o2'\ o3' + o1\ o2\ o3' + o0$

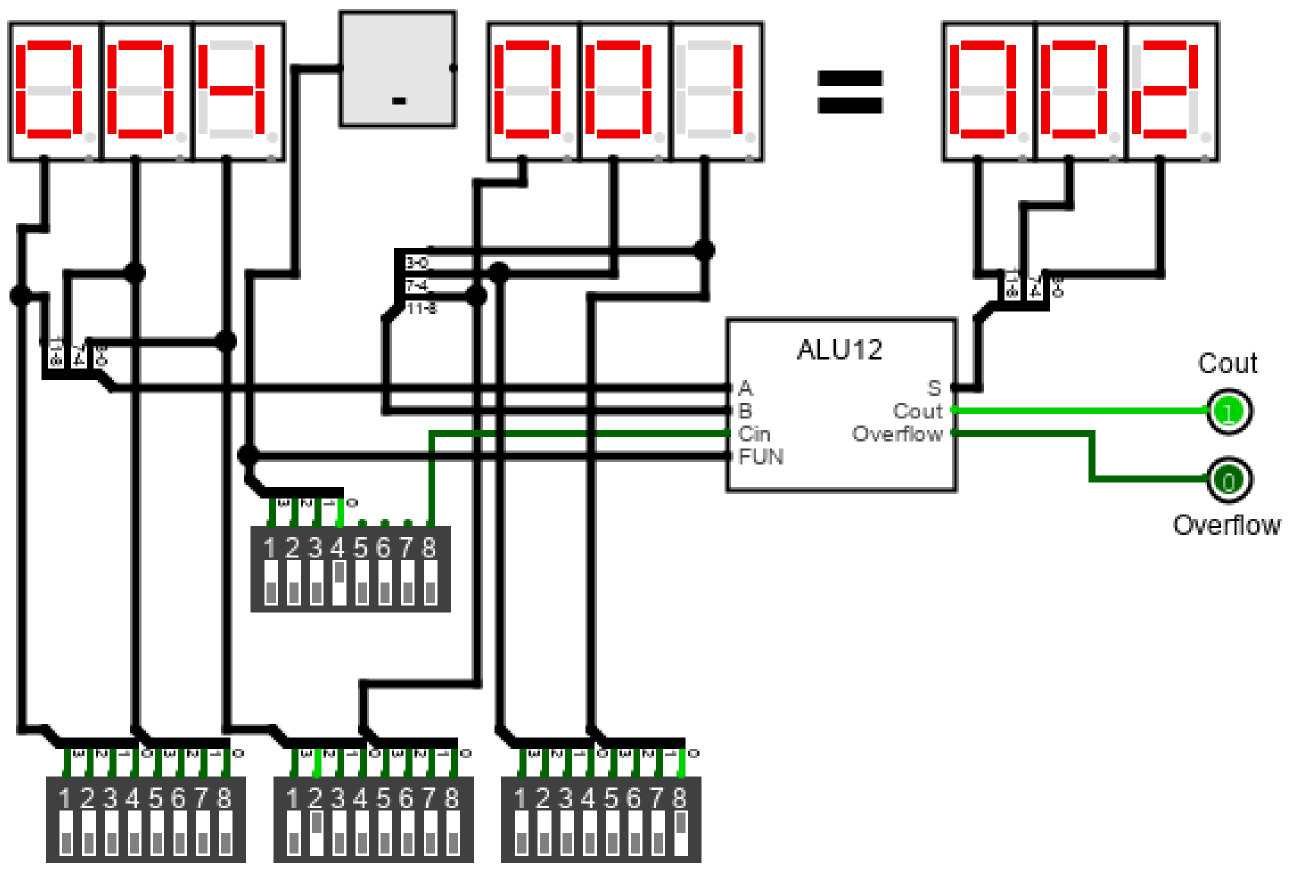
$s3/m = o3$

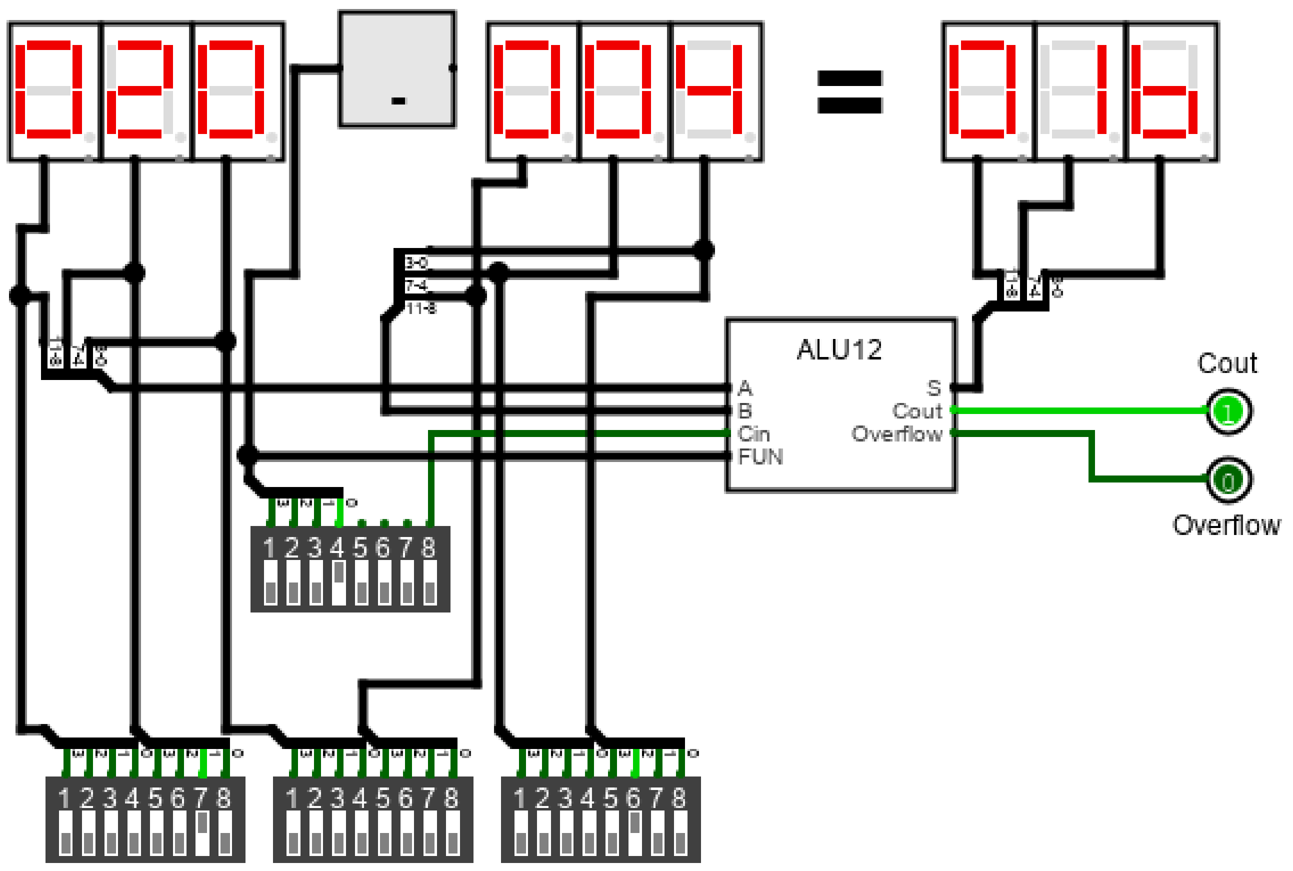
TAC-DEMO



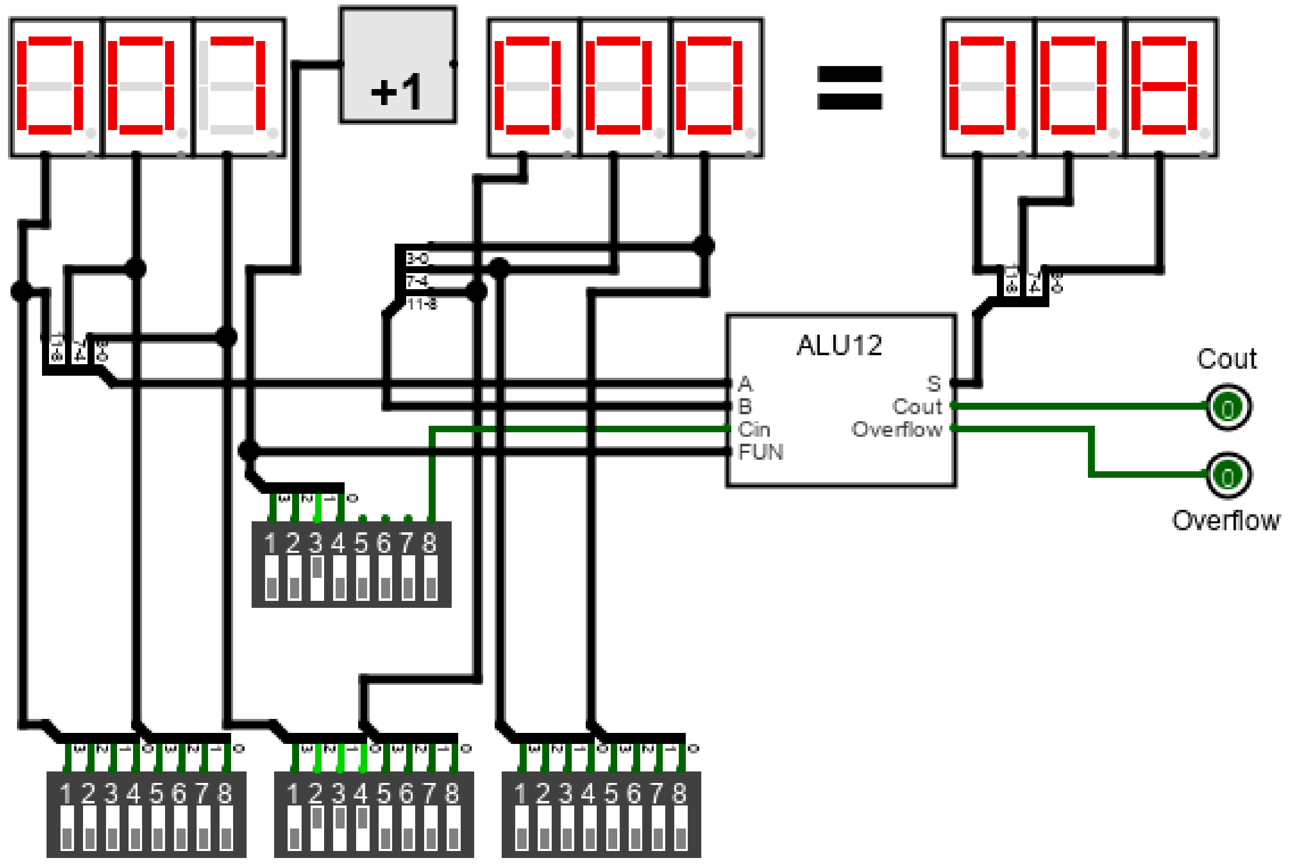


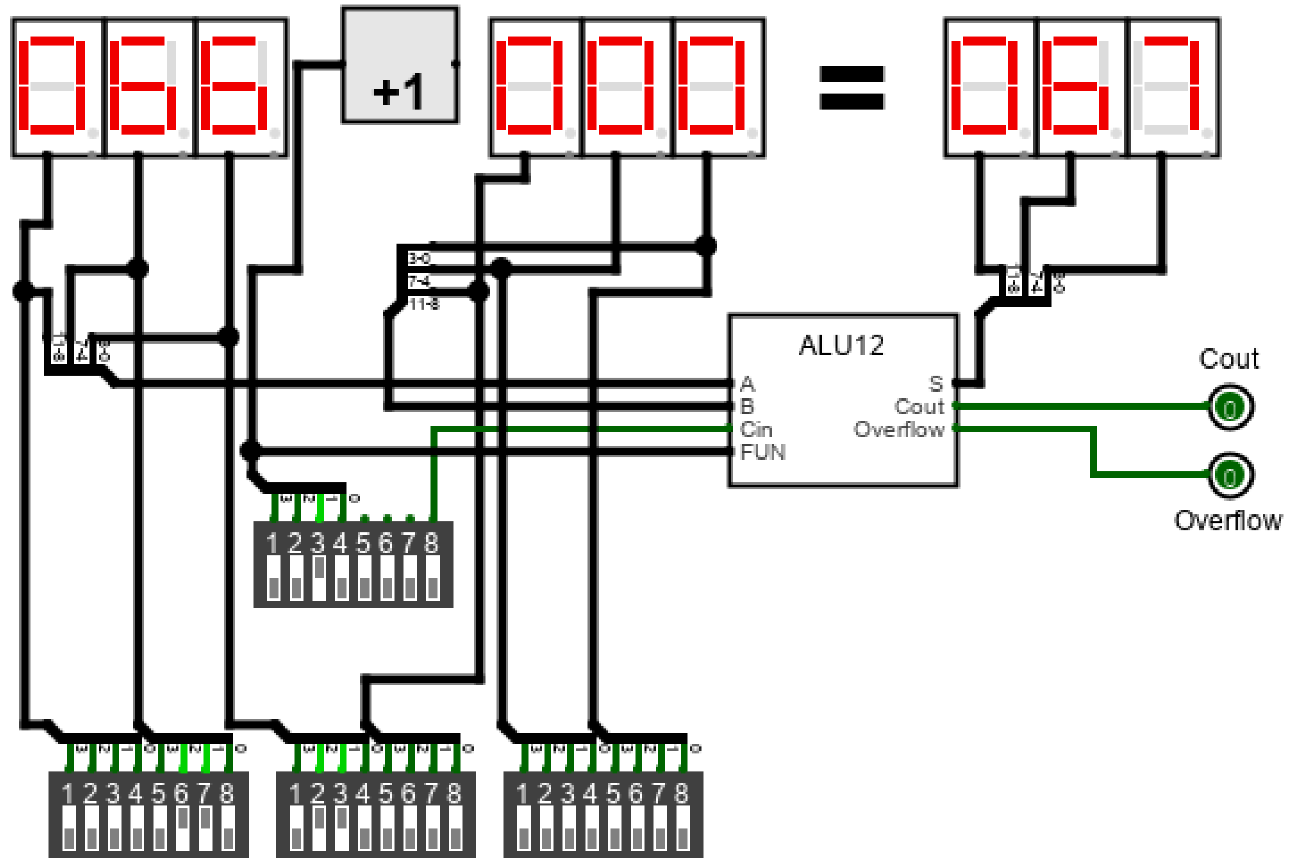
Reasoning: Showing addition, adding A to B, with 83 + 10 = 93 and 5 + 10 = 15.



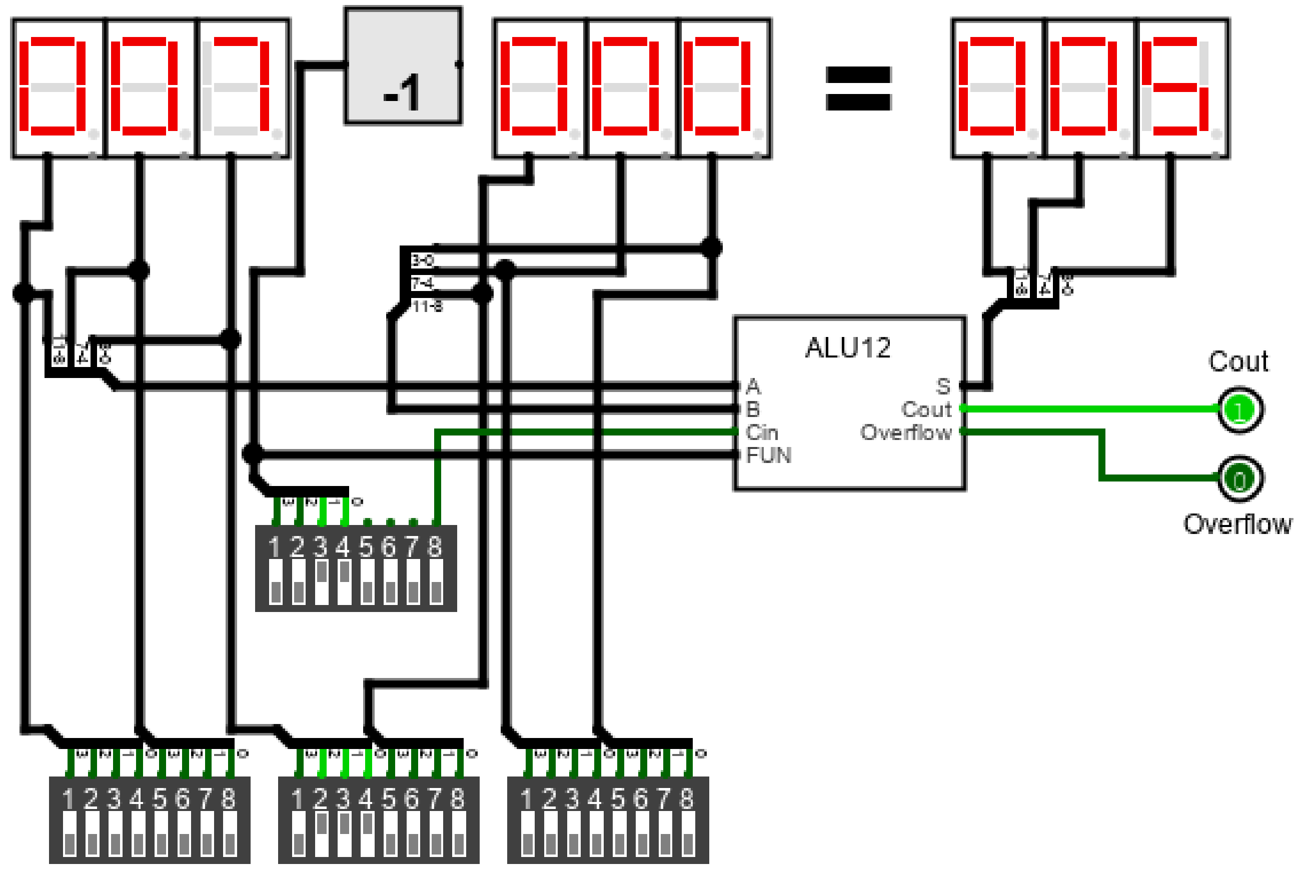


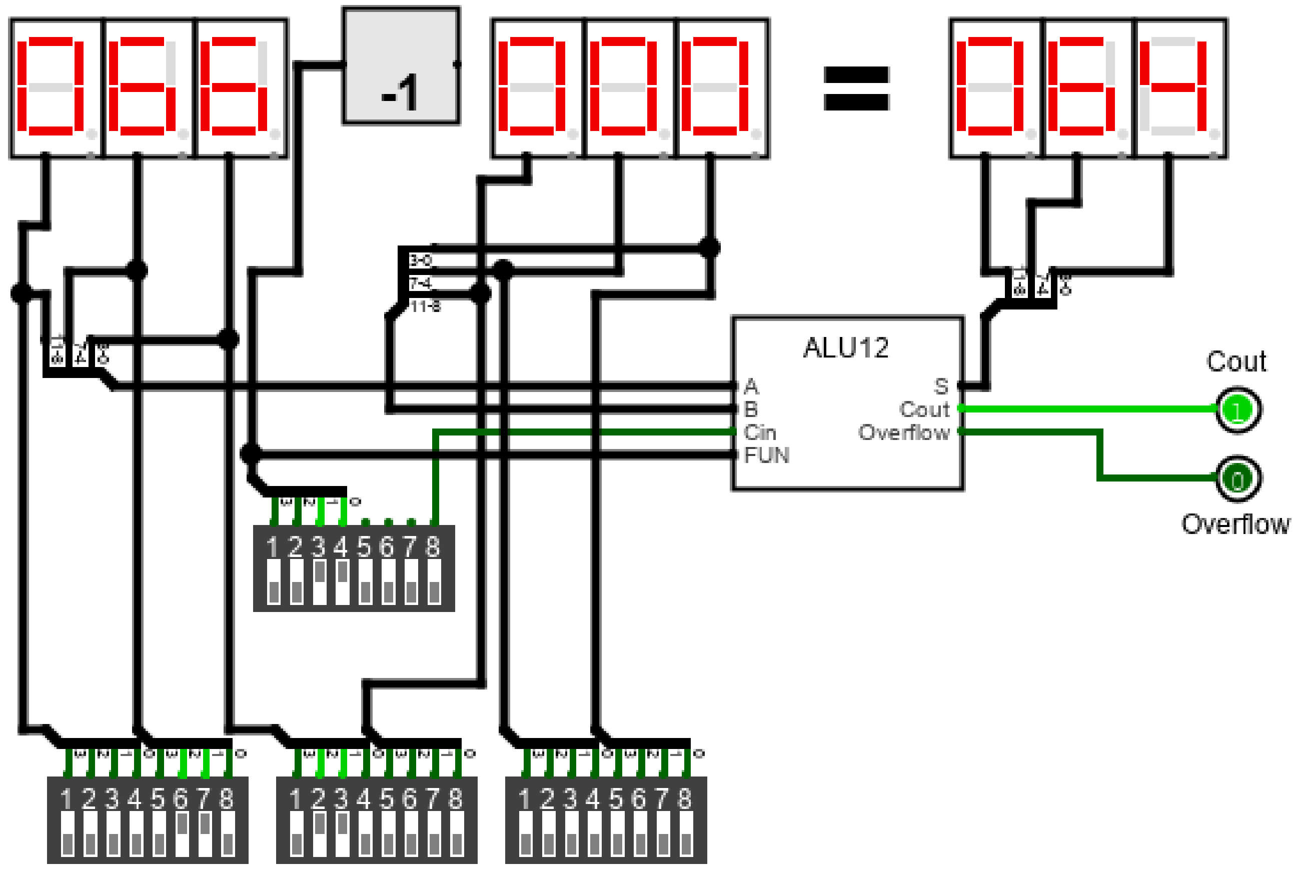
Reasoning: Showing Subtraction, subtracting B from A, with 4 - 1 = 2 with a Cout and 20 - 4 = 16 with a Cout.



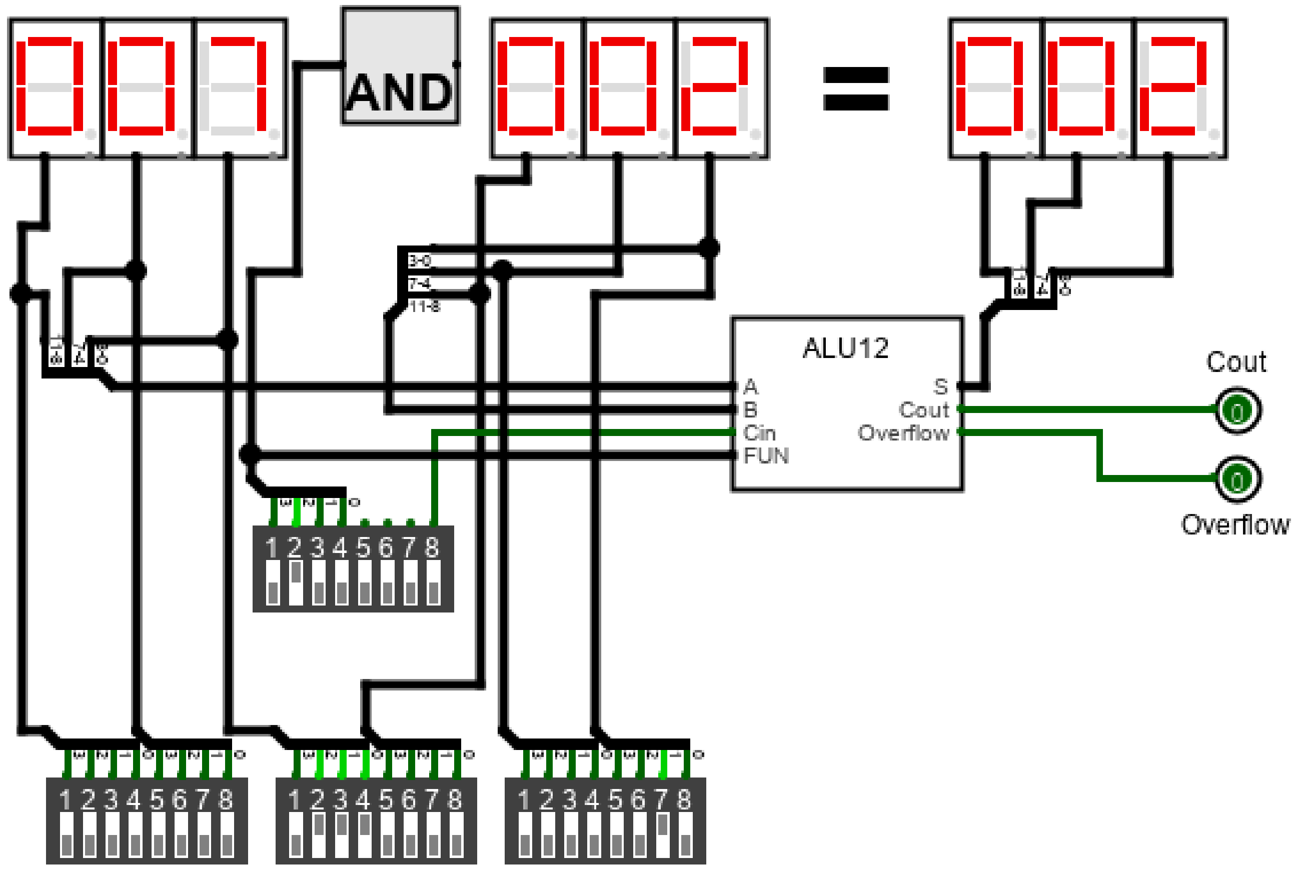


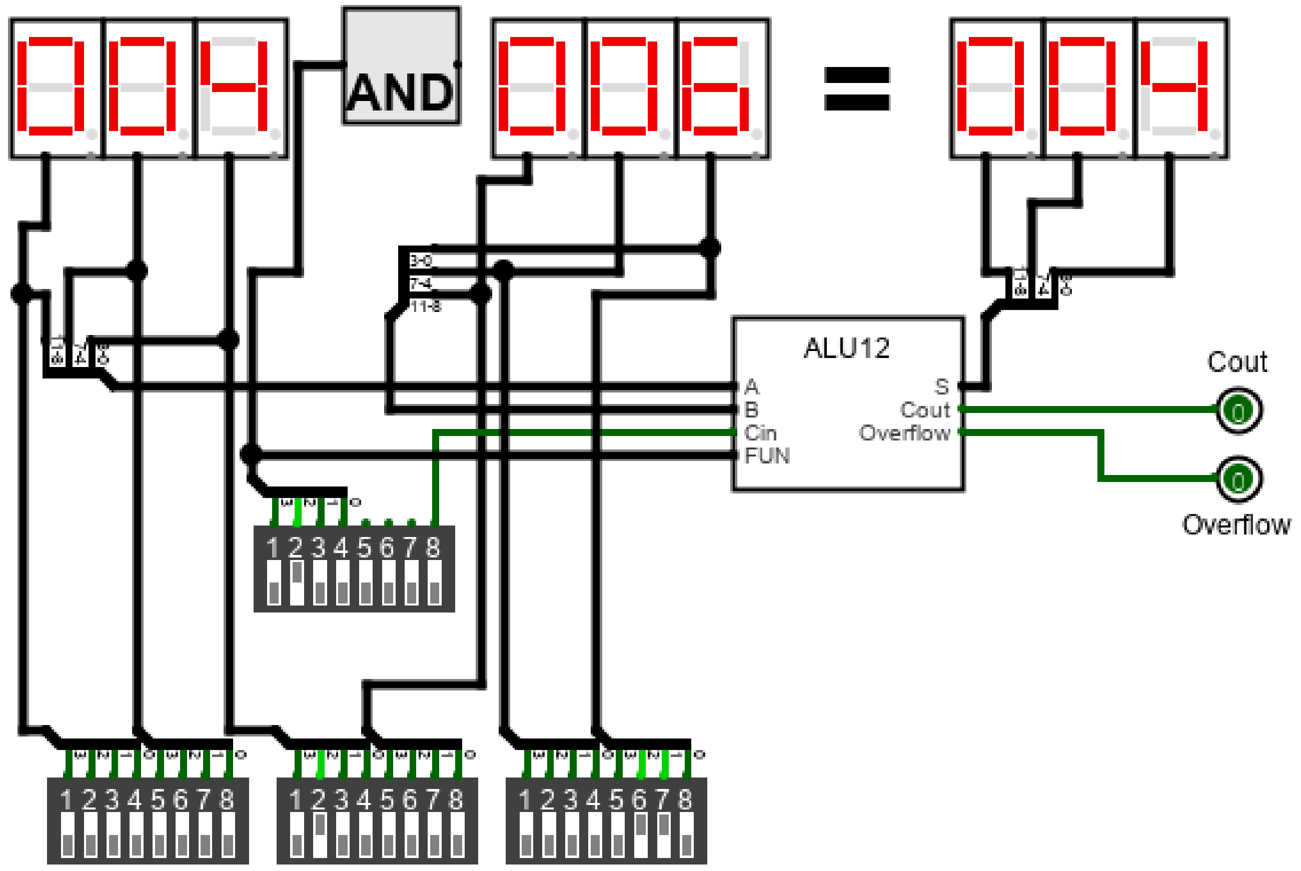
Reasoning: Showing the Increment function, which only affects A and adds +1 to it, as shown here with 7 + 1 = 8 and 66 + 1 = 67.



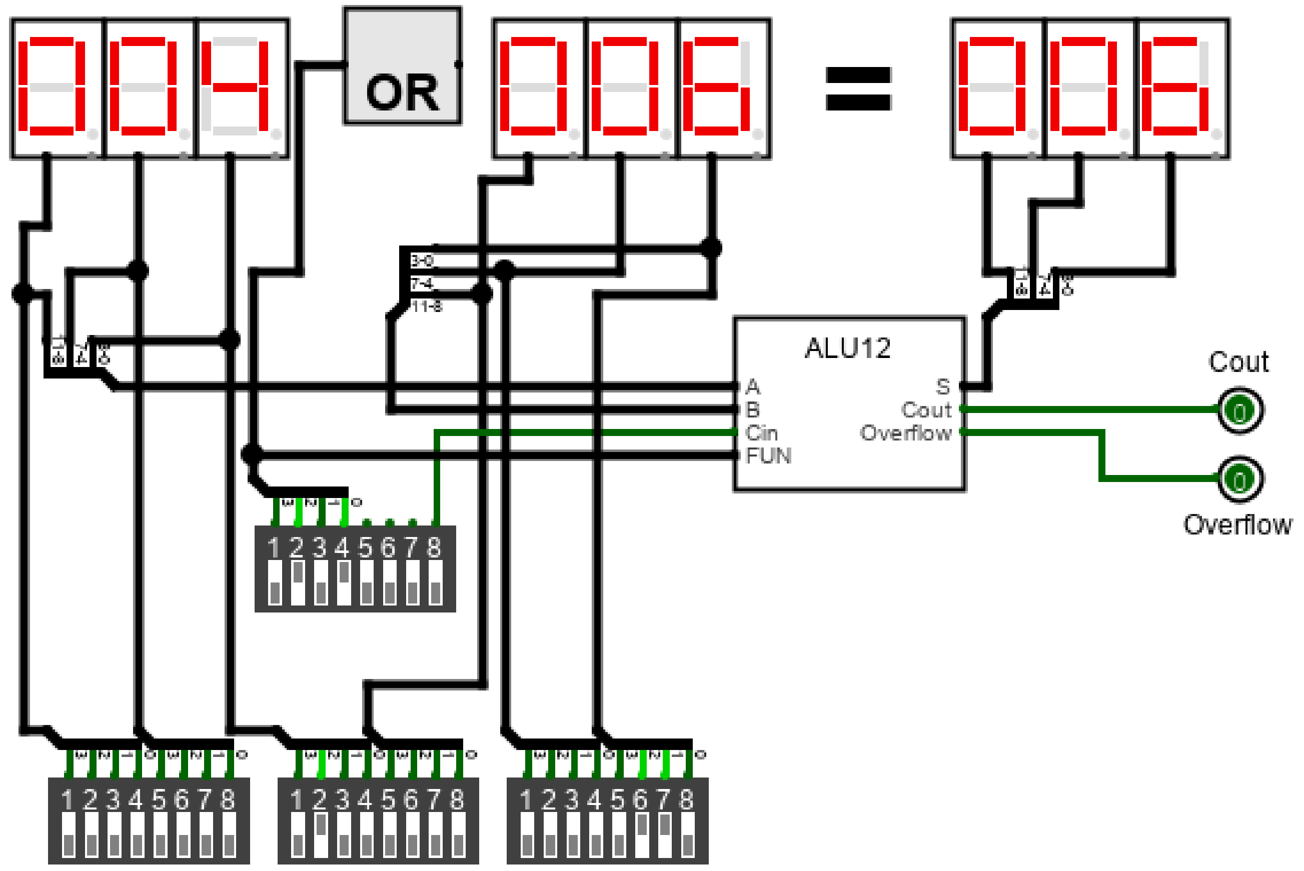


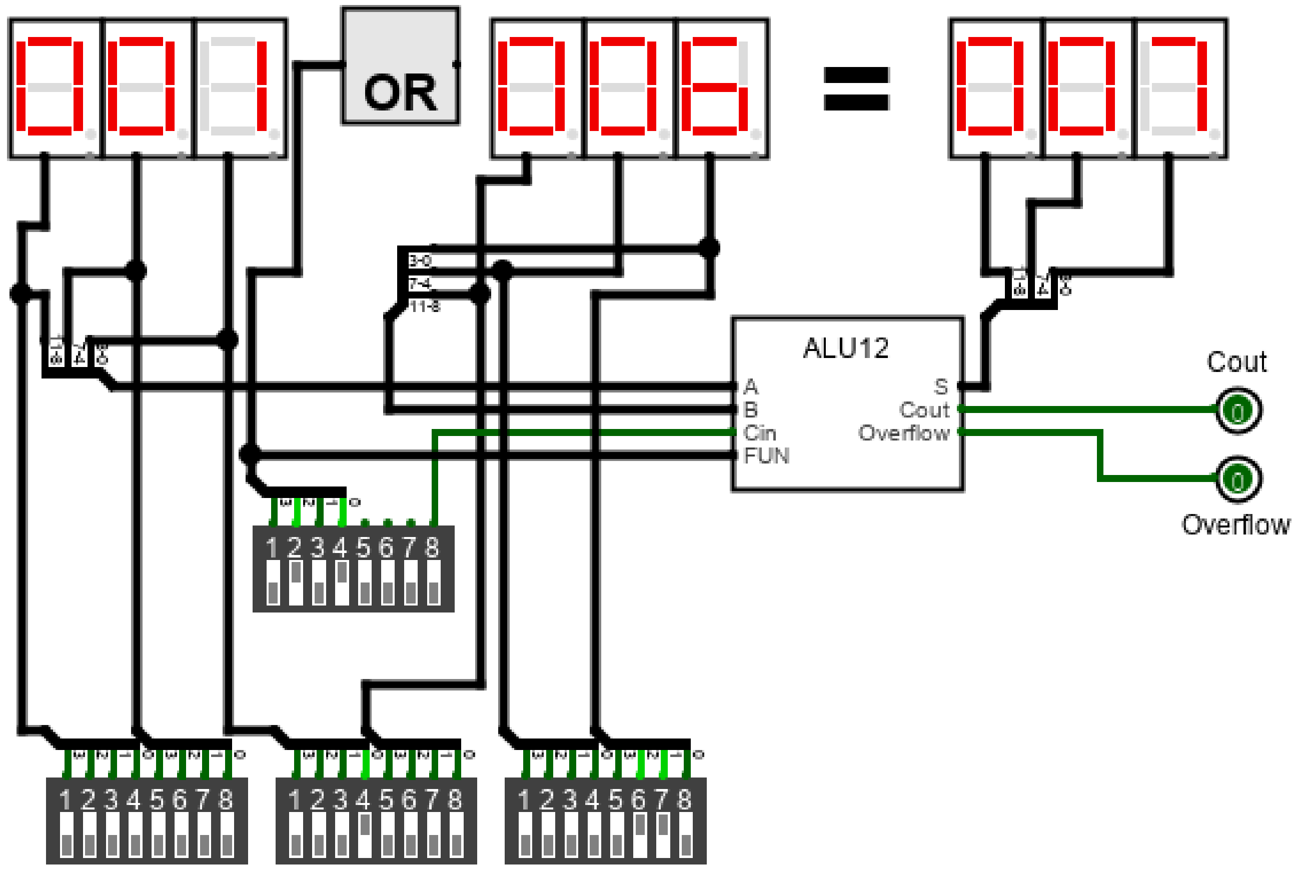
Reasoning: Showing the Decrement function, which only affects A and can be seen with 7 - 1 = 5 with a Cout and 66 - 1 = 64 with a Cout.



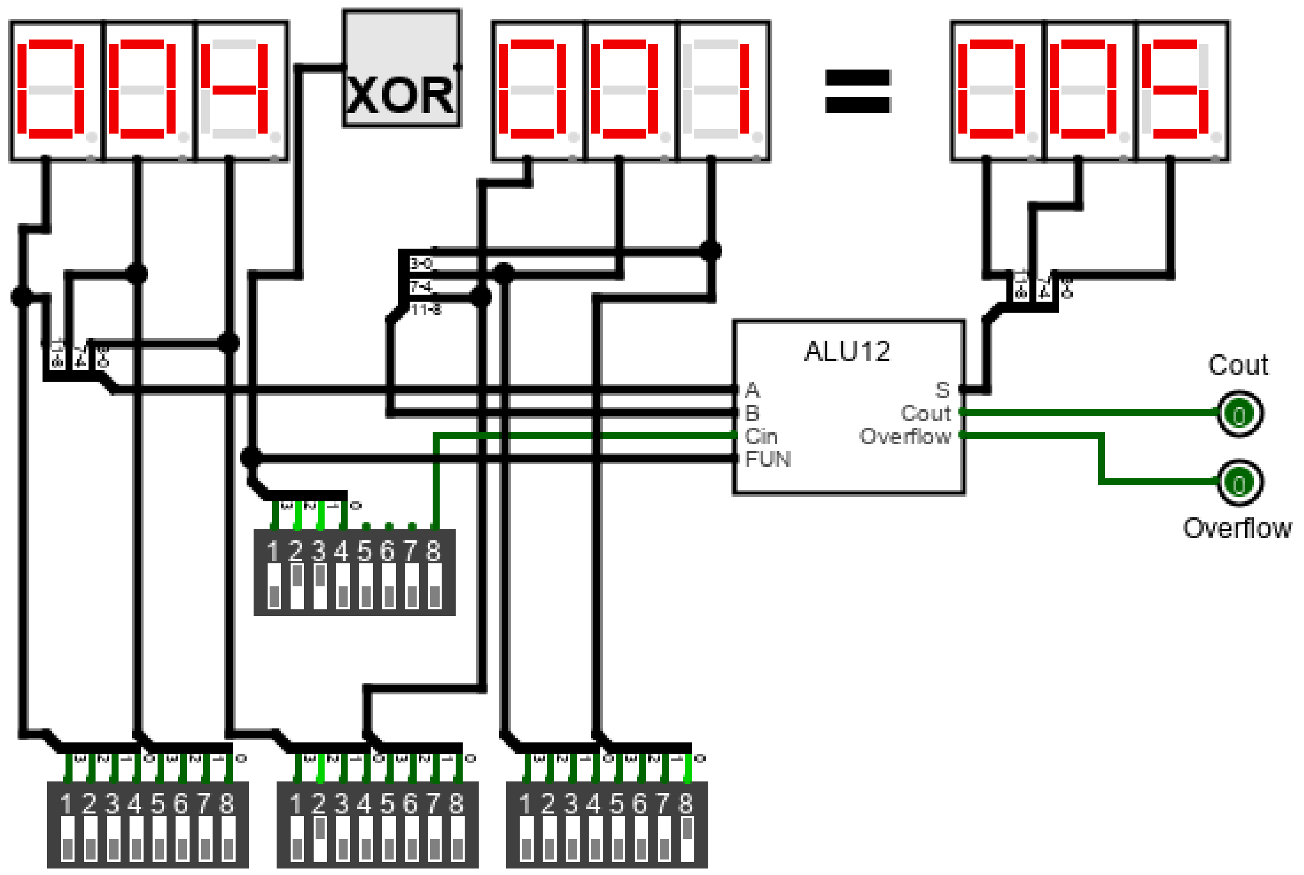


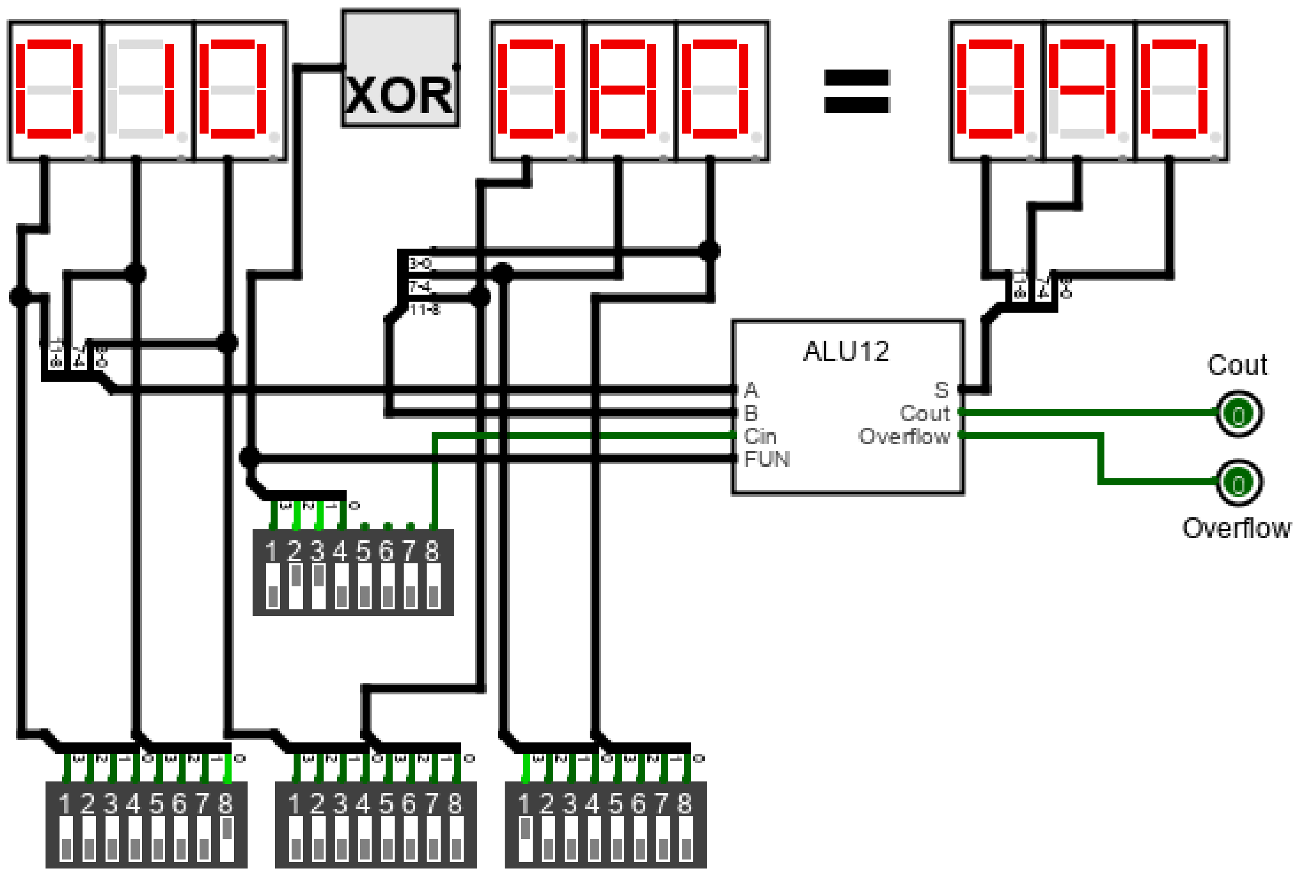
Reasoning: Showing the AND function accepting A and B, so 7 and 2 = 2, as well as 4 and 6 = 4. Also showing the higher number in the A and then then the B positions.



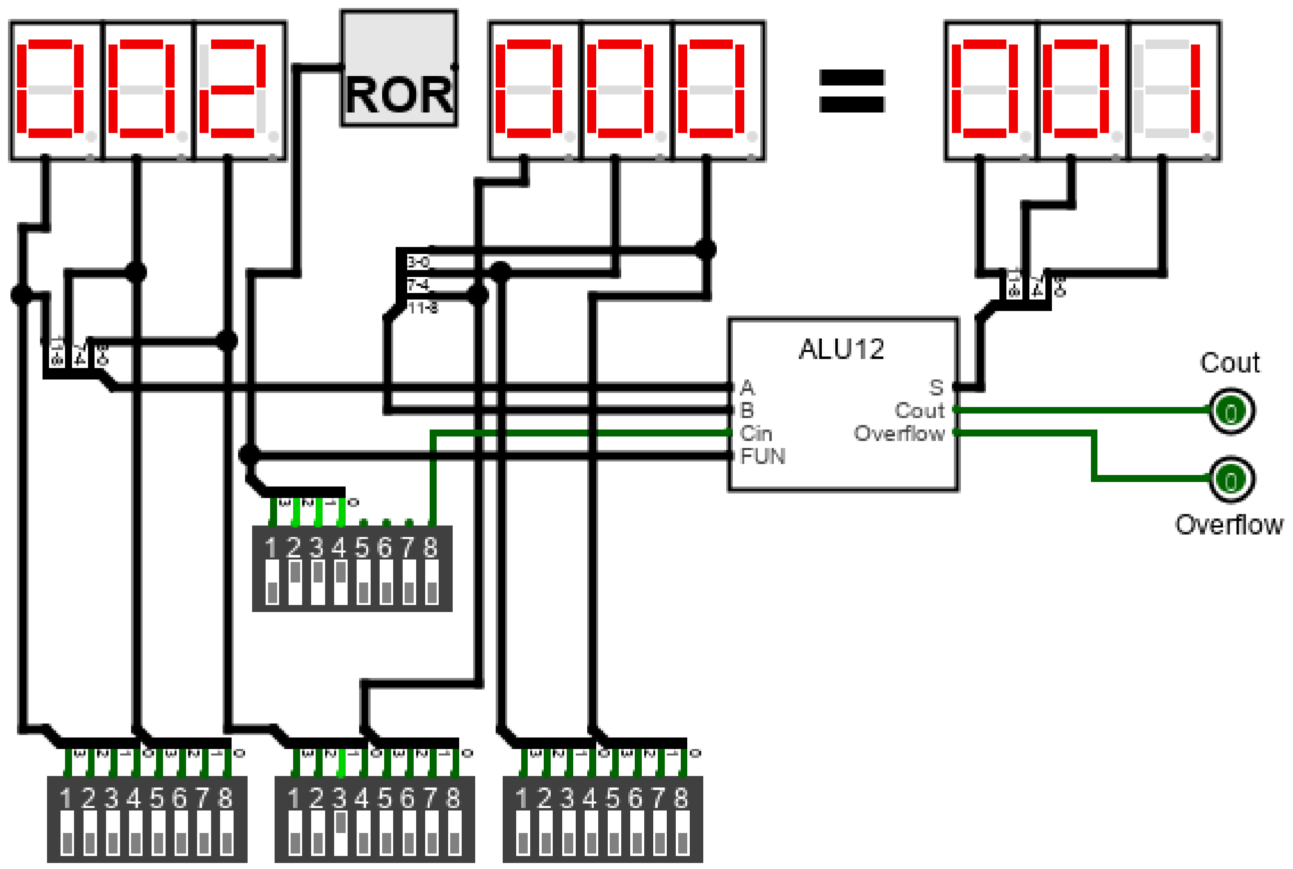


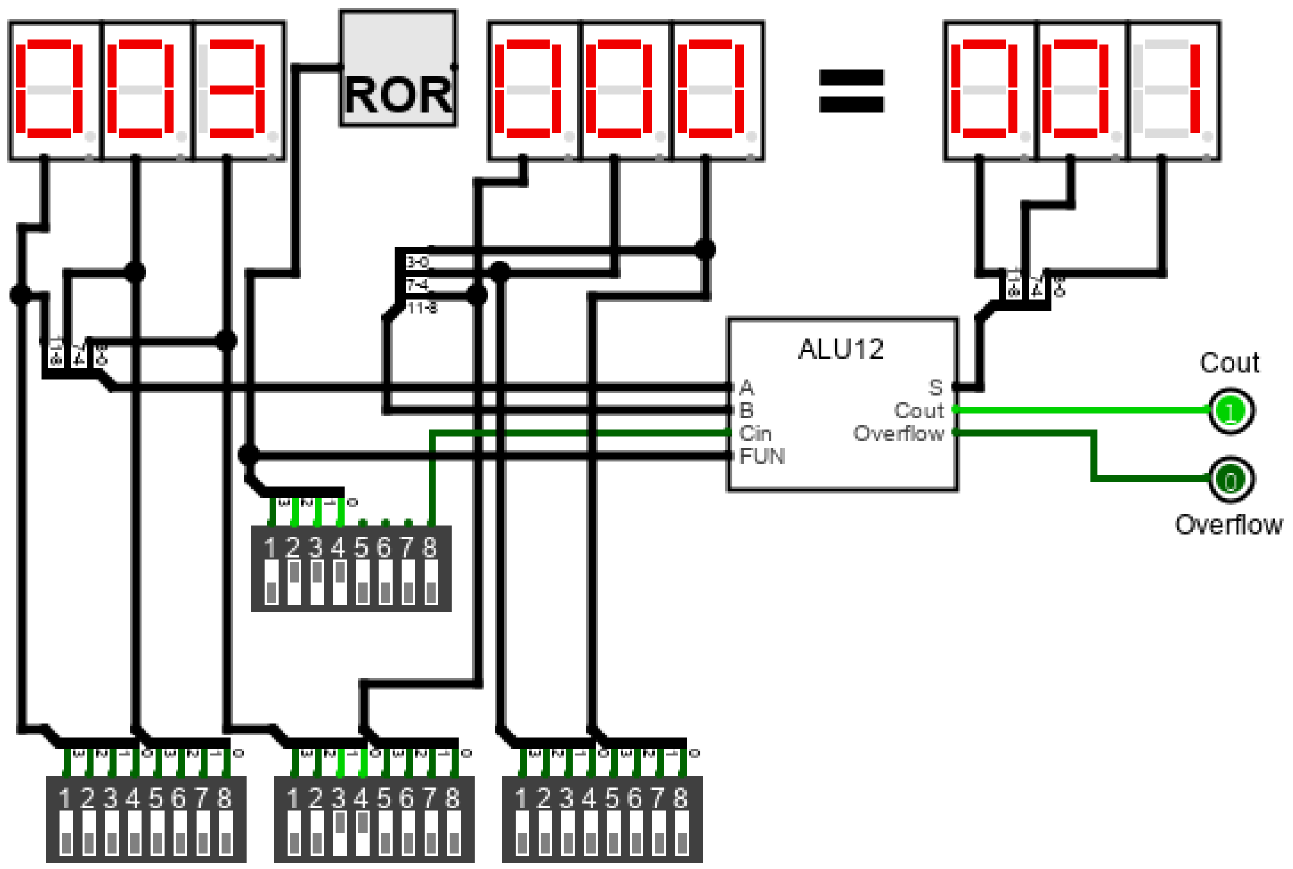
Reasoning: Showing the OR function, accepting input from A and B, so 4 or 6 = 6, and then 1 or 6 = 7.



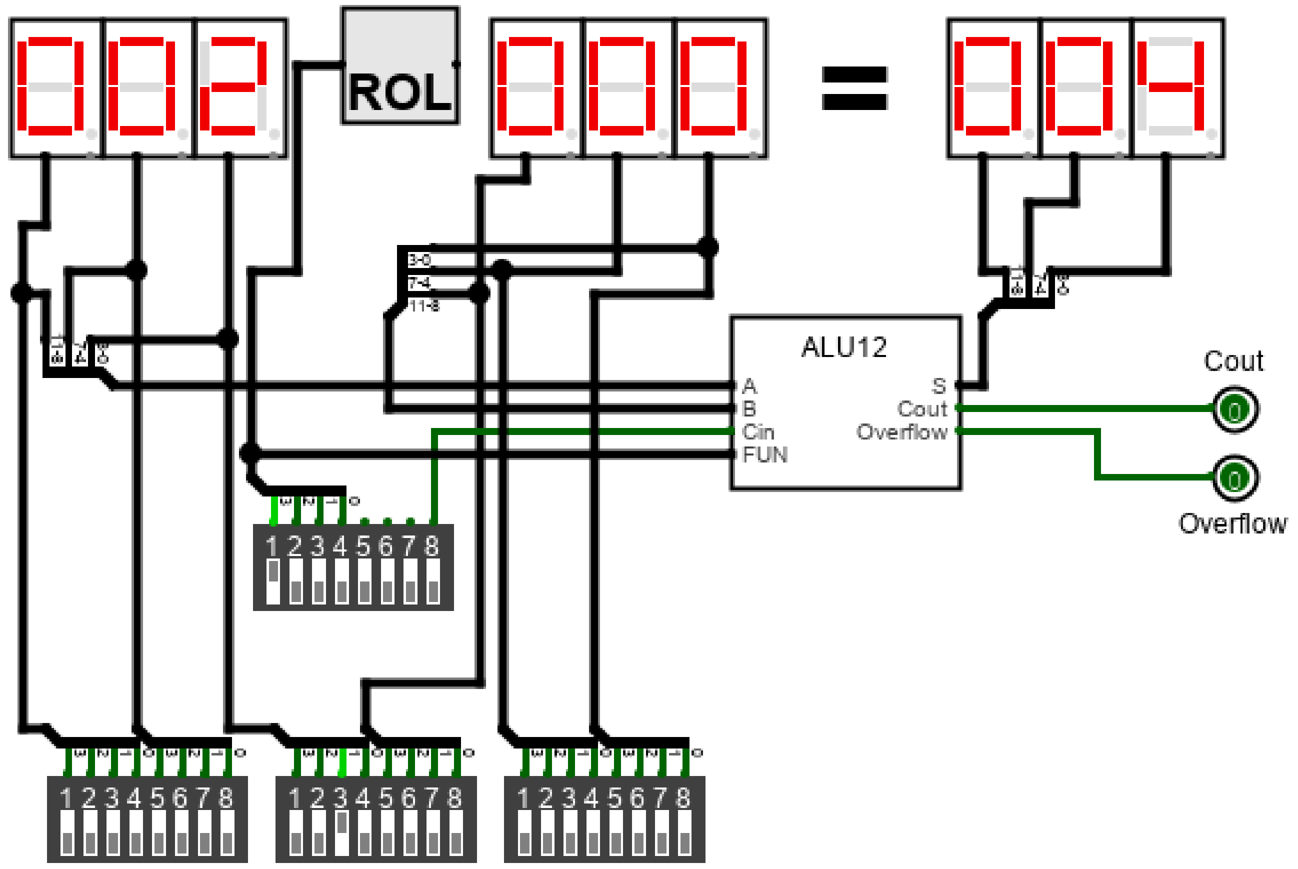


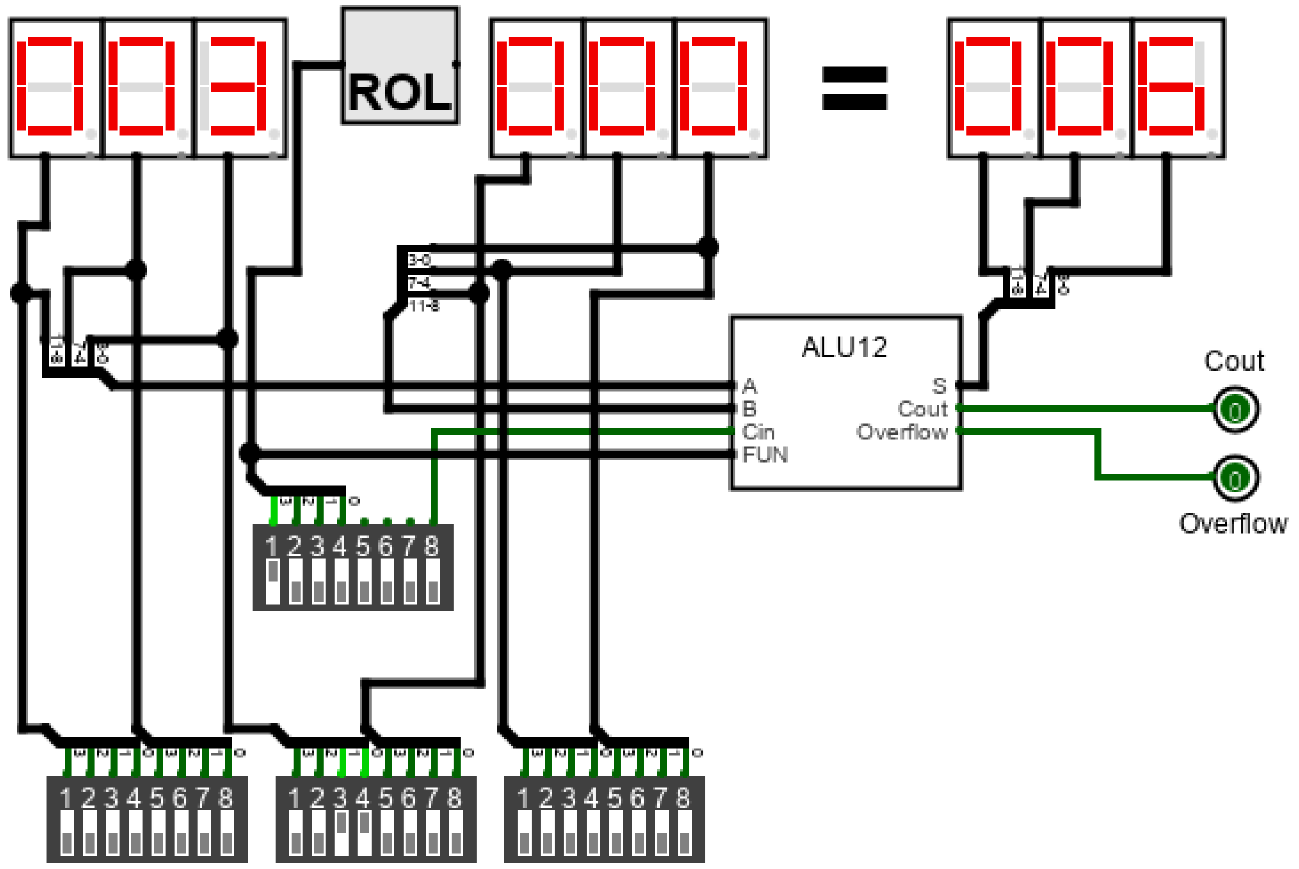
Reasoning: Showing the XOR function, with A and B as inputs, so 4 XOR 1 = 5, and then 10 XOR 80 = 90.



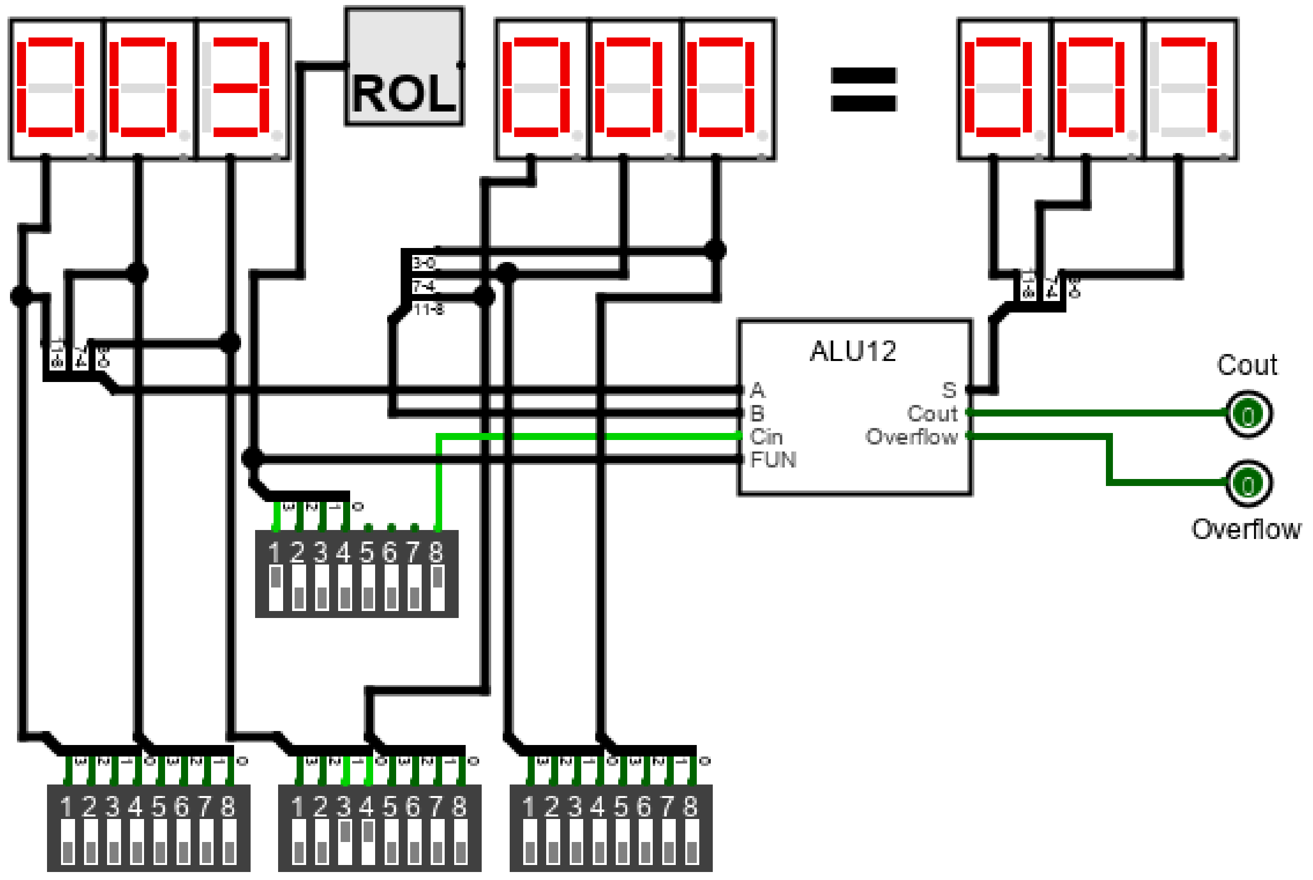


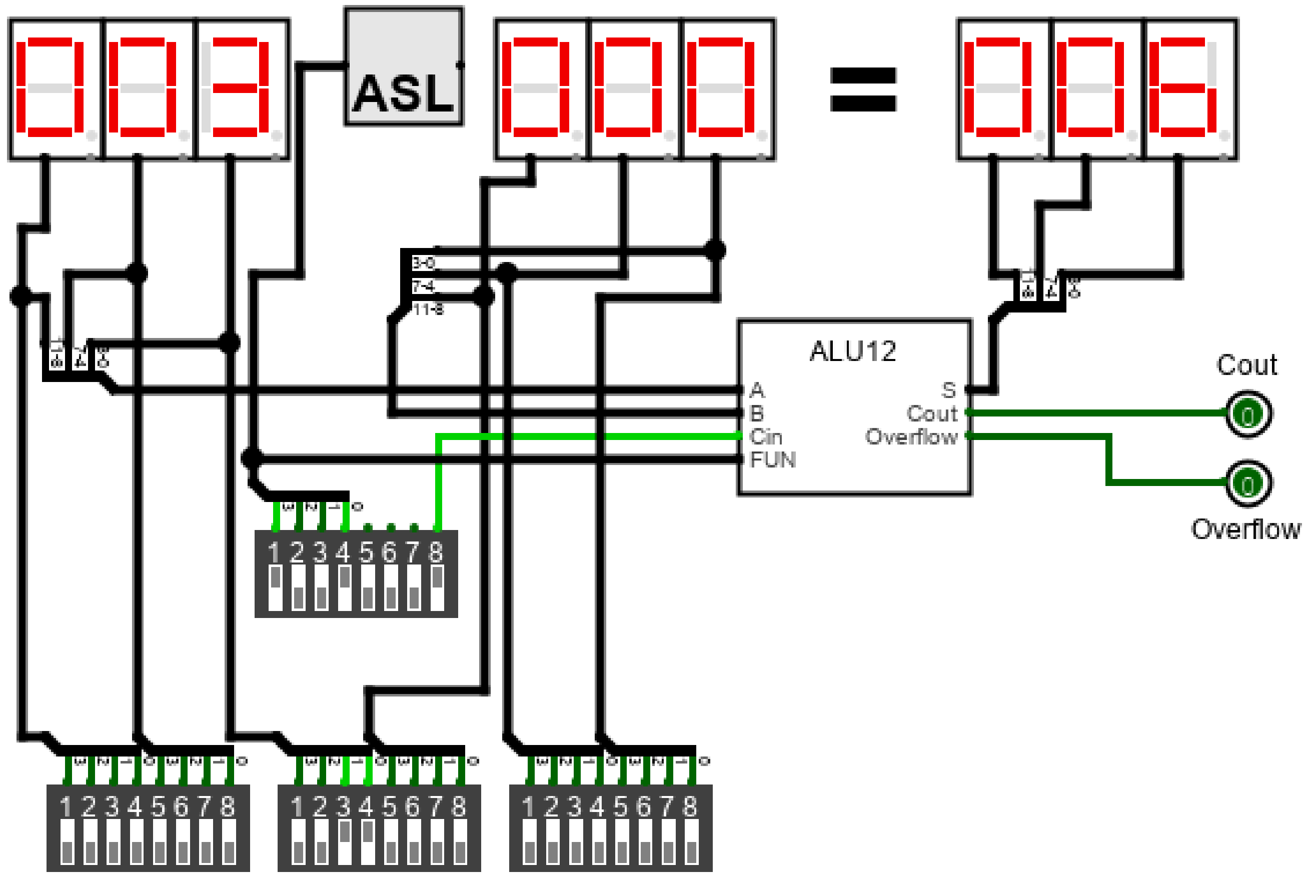
Reasoning: Showing the ROR function, with only A being a valid input, so 2 ROR = 1, and 3 ROR = 1, with a Cout. This showcases the inputs rotating right or down.





Reasoning: Showing the ROL function, with A being the only valid input, so 2 ROL = 4, and 3 ROL = 6. This showcases the values rotating left or up.





Reasoning: Showing the ASL function right next to the ROL function with the same values. This showcases that ROL takes the Cin into the function, while ASL ignores the Cin. This leads to 3 ROL = 7 while 3 ASL = 6.