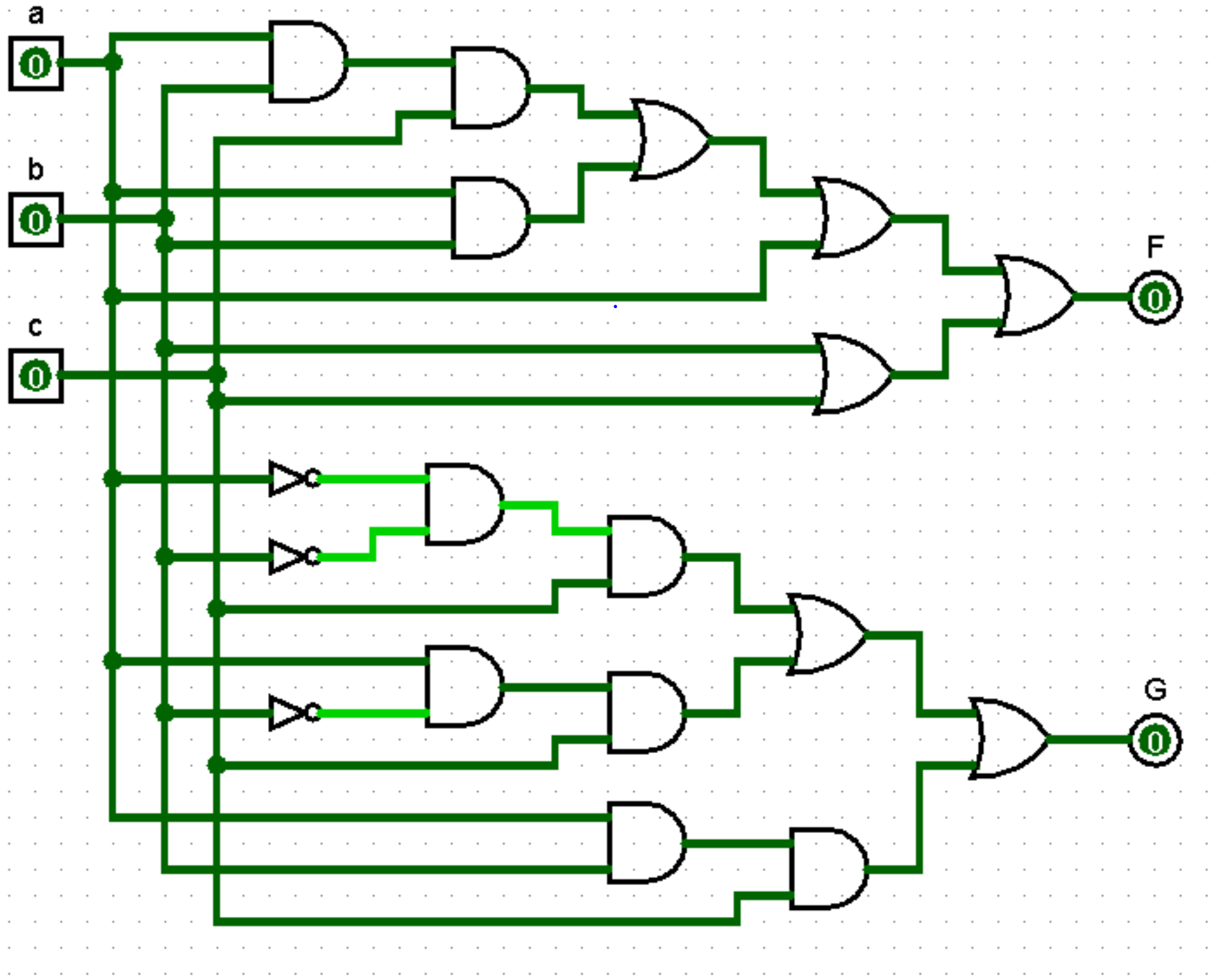
LAB 2

Problem 1:

1. 



1. F
   1. 1 – quad 2-input AND gate (74x08)
   2. 1 – quad 2-input OR gate (74x32)

G

1. 1 – hex inverter gate (74x04)
2. 2 – quad 2-input AND gate (74x08)
3. 1 – quad 2-input OR gate (74x32)
4. F
   1. Quad 2- input OR gate (SN74LS32) – 22 ns
   2. Quad 2 – input AND gate (SN74LS08) – 25 ns
   3. MAX DELAY: AND AND OR OR OR = 116 ns

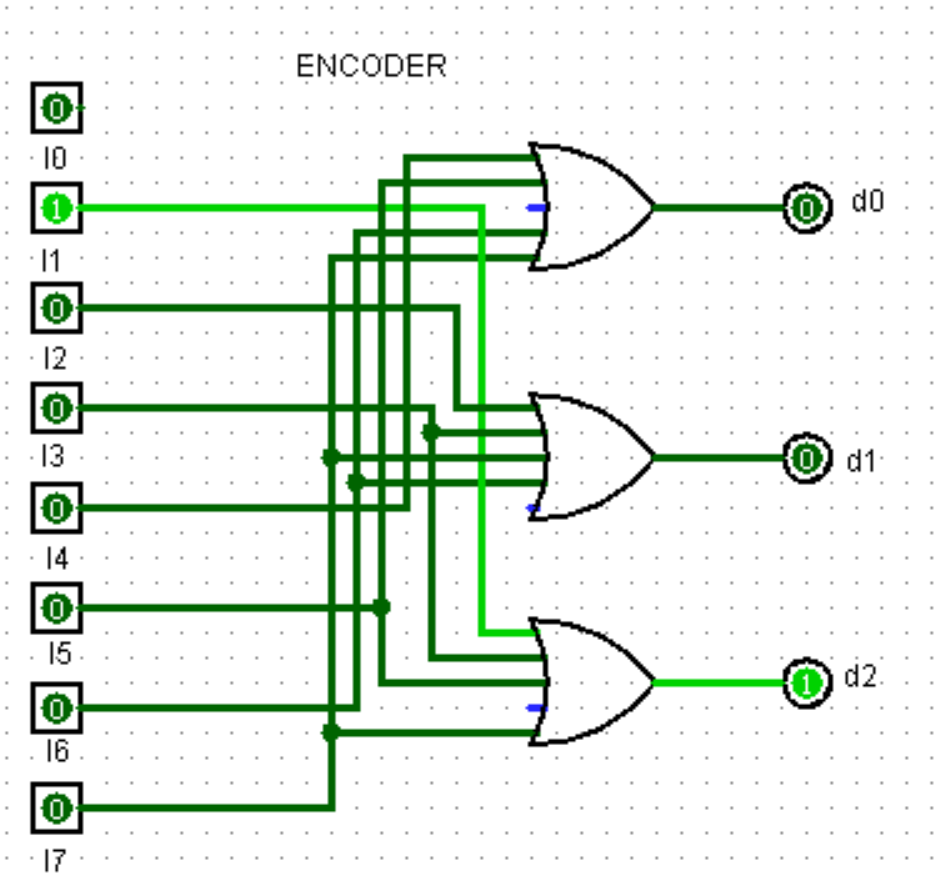
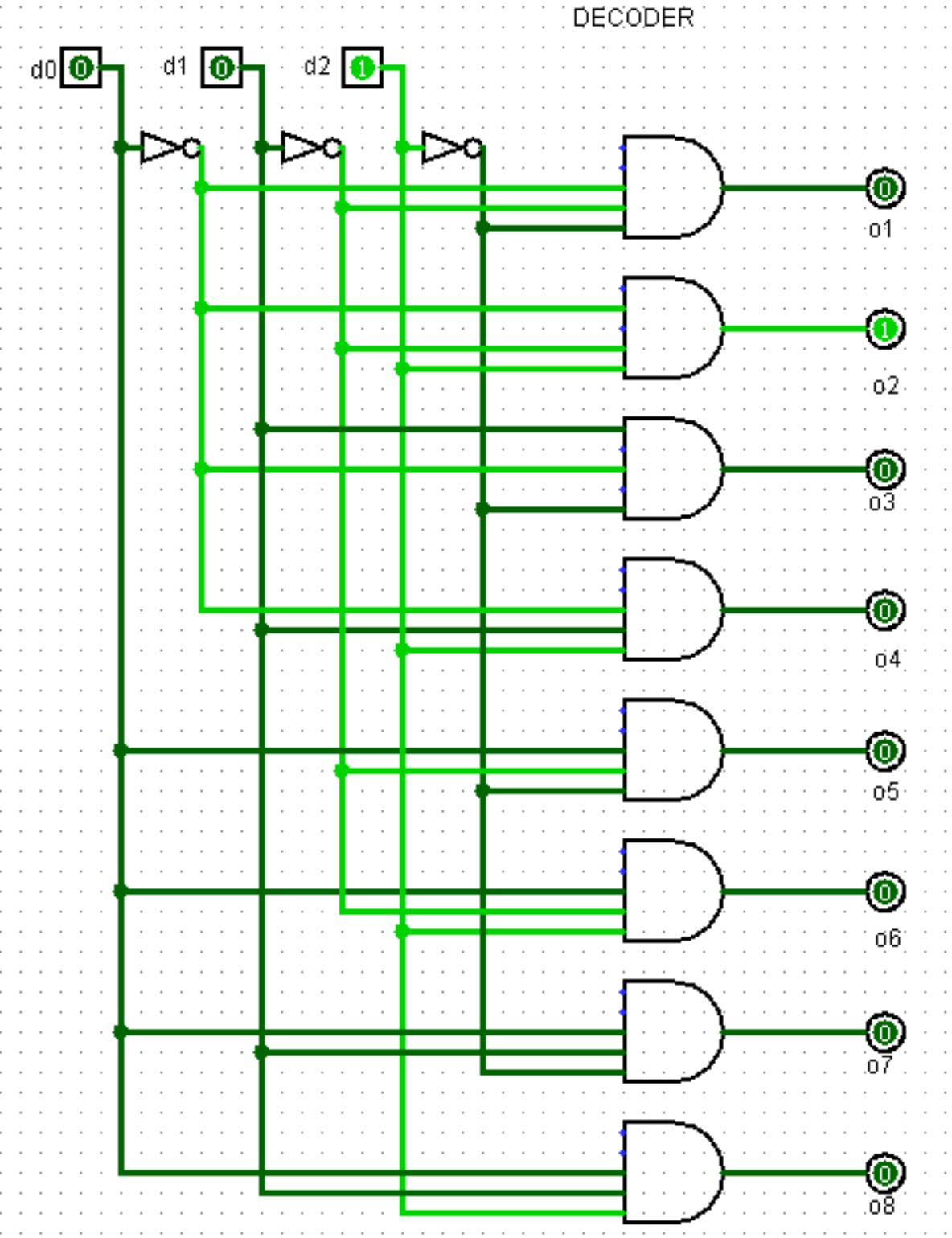
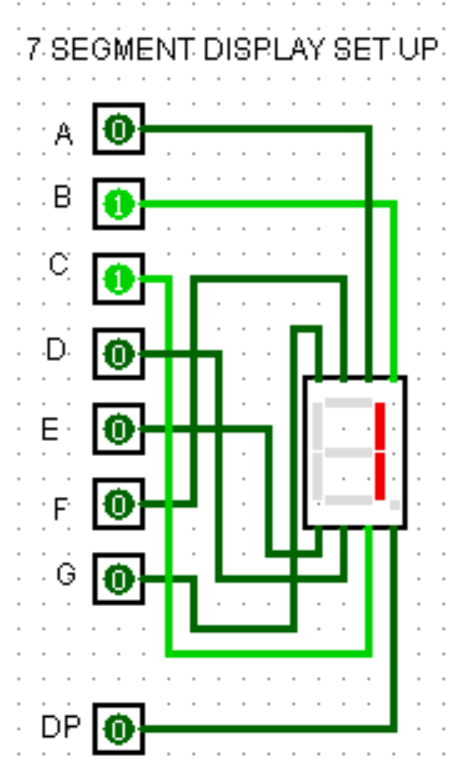
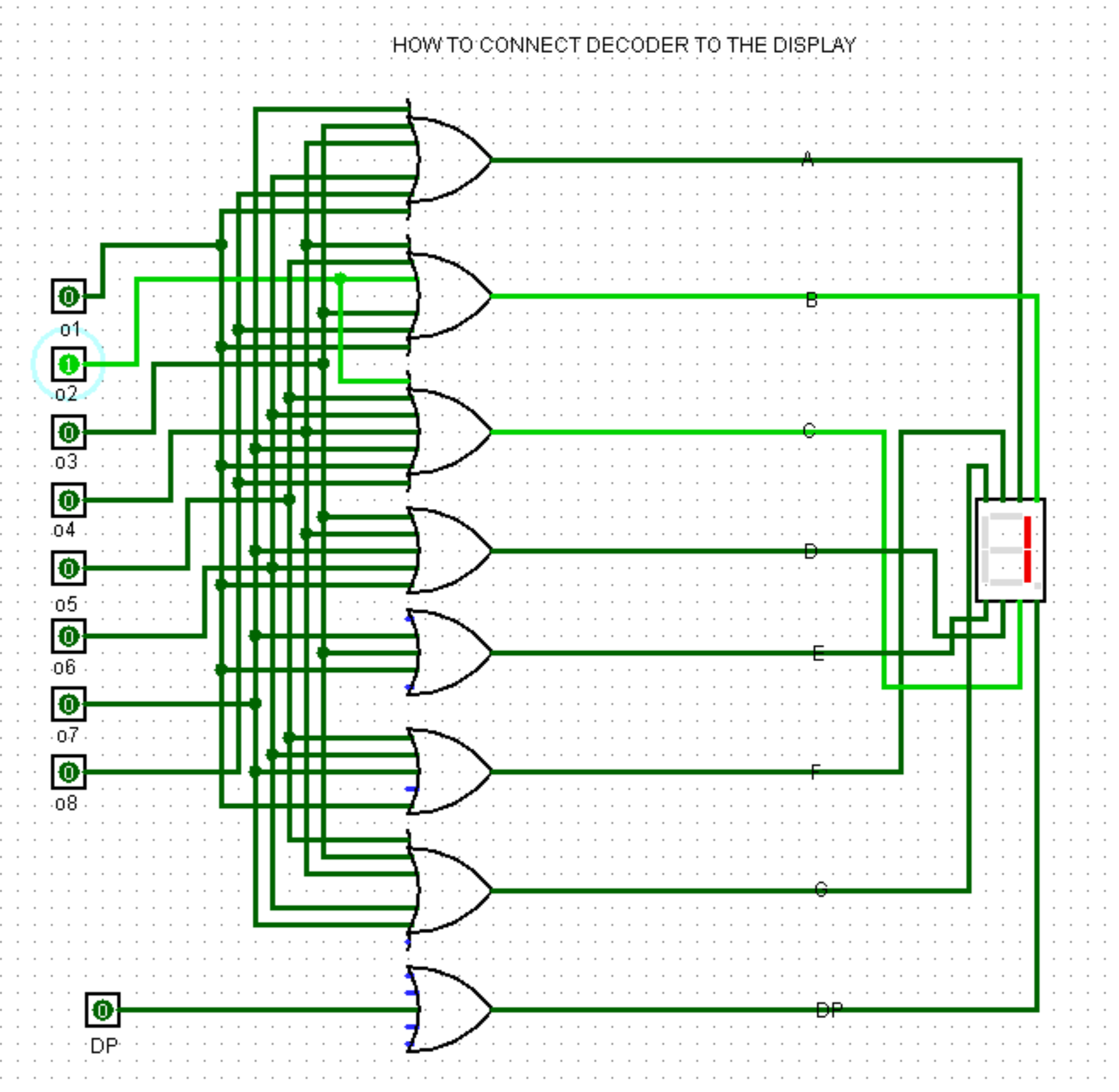
G

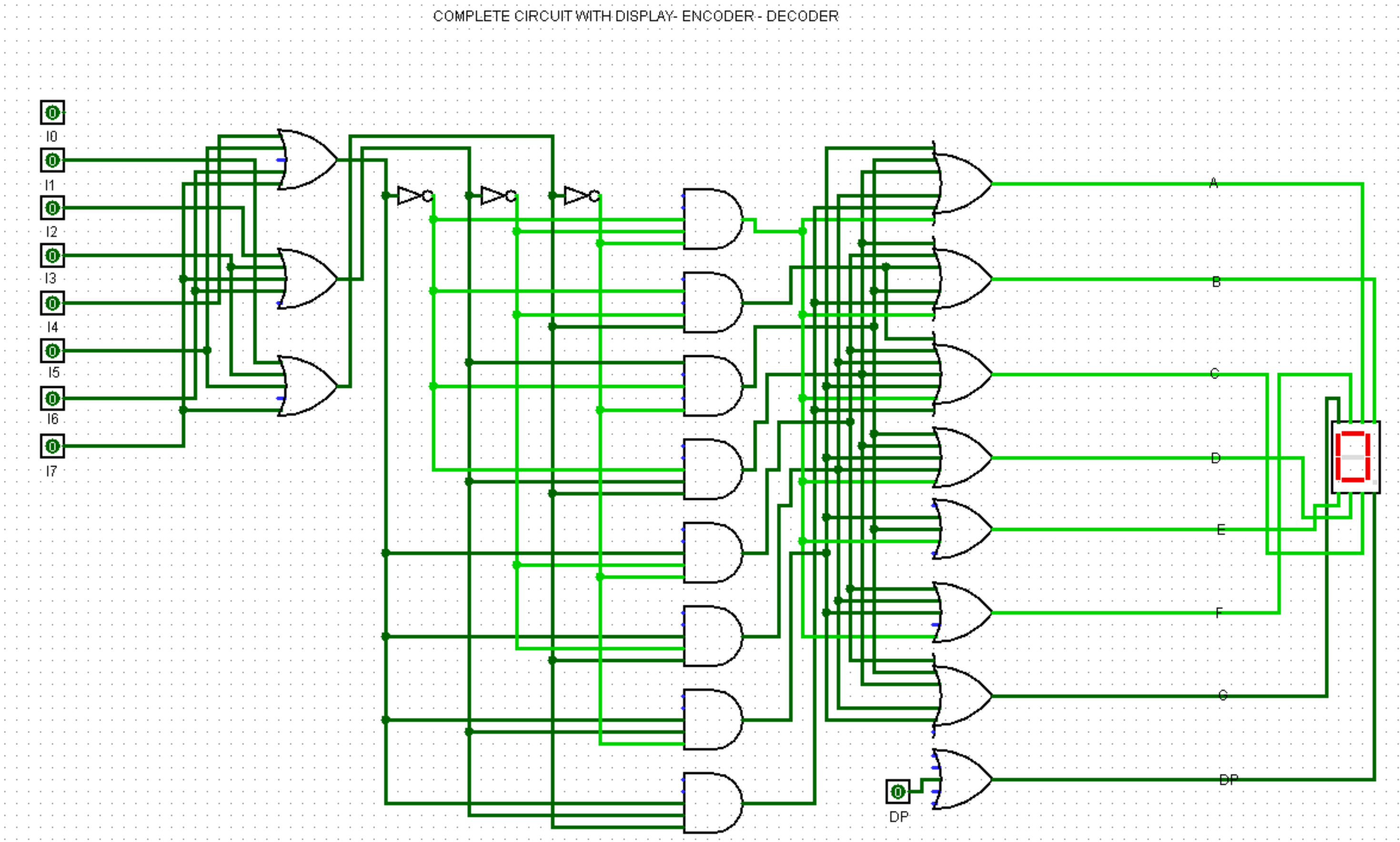
1. Quad 2- input OR gate (SN74LS32N) – 22 ns
2. Quad 2 – input AND gate (SN74LS08N) – 25 ns
3. Hex Inverter gate (SN74LS04) – 22 ns
4. MAX DELAY: NOT AND AND OR OR = 116 ns
5. I used the propagation delay time to calculate the delay because that is the time that it takes the circuit to get inputs and then return an output. To calculate the delay, I calculated the max delay by finding the longest and most gates that a single path could go.

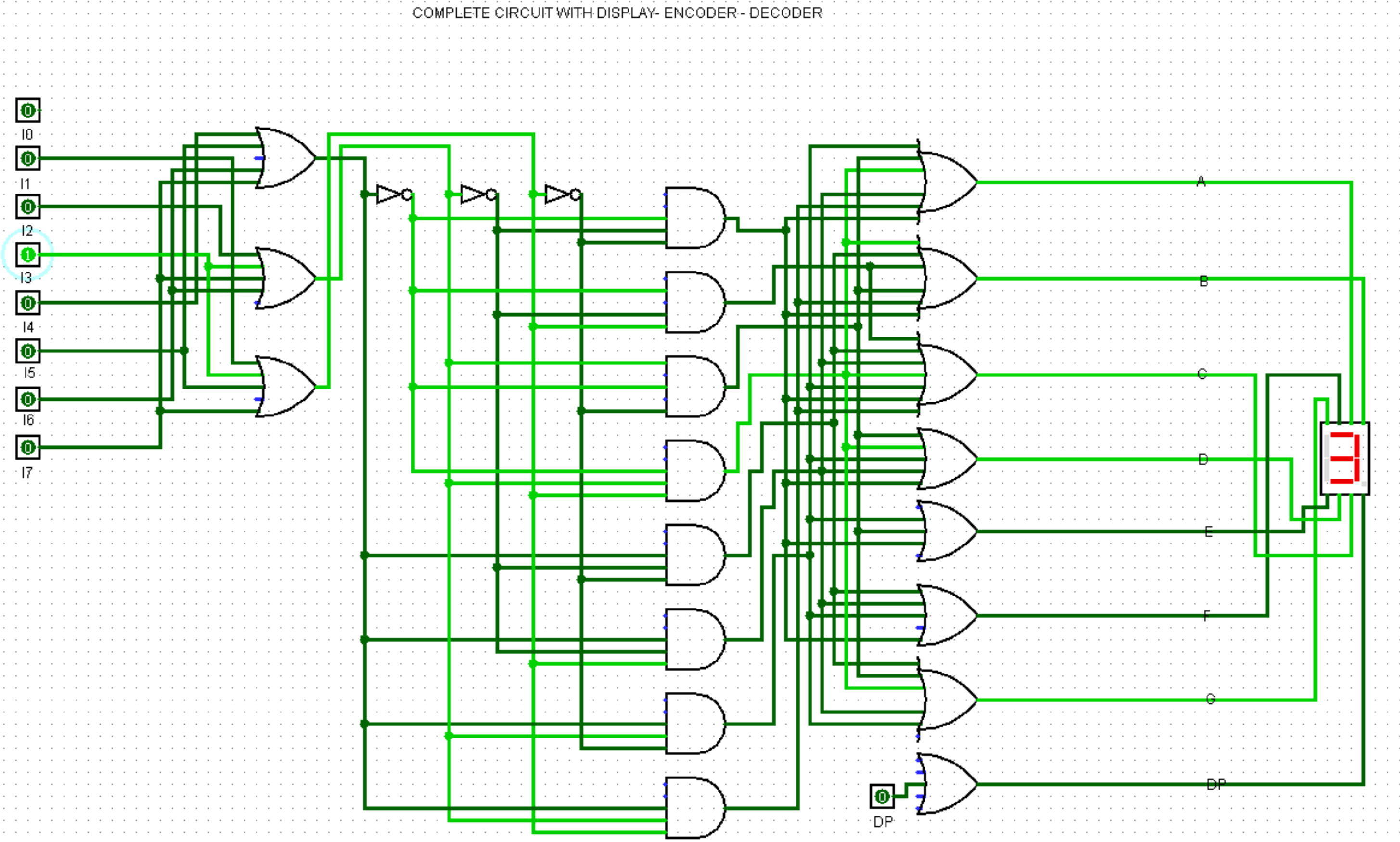
Problem 2:

1. Design Parameters
   1. Number of switches: 7 total
   2. Bits on Data Bus: 3 )
   3. Size of Encoder: 8x3

Size of Decoder: 3x8

1. 8 sides each one lights up a certain place on the display
2.   





1. The way this logic works is if you say car 3 has an issue. You would press the button in 3 (I3 because I0 is the case in which no button is pressed). It would then go through the encoder which will turn it into a 3 bit 011 (which is 3 in binary). Each input is turned into its binary equivalent, which for 8 inputs only requires a maximum of 3 bits. It will then go through the decoder which will return 00010000. A 1 in the place of the o4 assuming o1 is the case in which none are pressed and o2 is the case in which car1 is pressed and so on. You then need to make the number 3 visible on the 7-segment display this means that A B C D and G need to be lit and that is so through the OR gates. Each OR gate goes to a specific segment so that if one of the inputs requires that segment on then it will illuminate. I based my letters off of the given example in the Lab Manual.