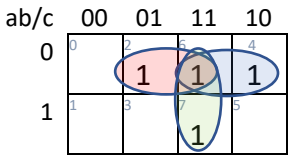
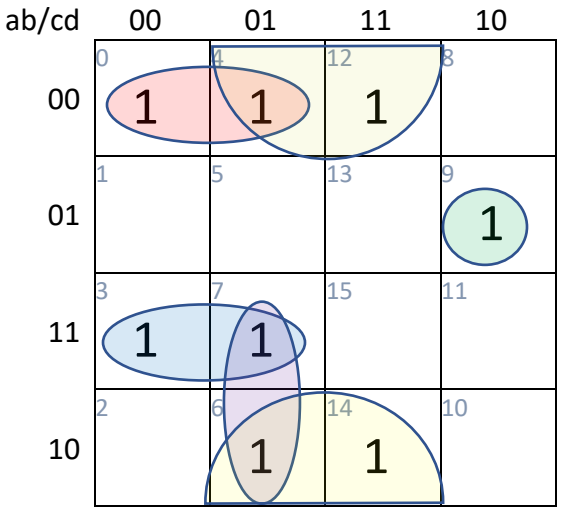
1. F(a,b,c) =ab'c +abc + a'bc + abc'
   a. This is actually pretty annoying. One way:
      = ab'c +abc + a'bc + abc + abc' + abc (Idempotency x3 with some Commutativity)
      =ab + a'bc + abc + abc' + abc (combining)
      = ab + bc + abc' + abc (combining)
      =ab+bc+ac (combining)

   b.



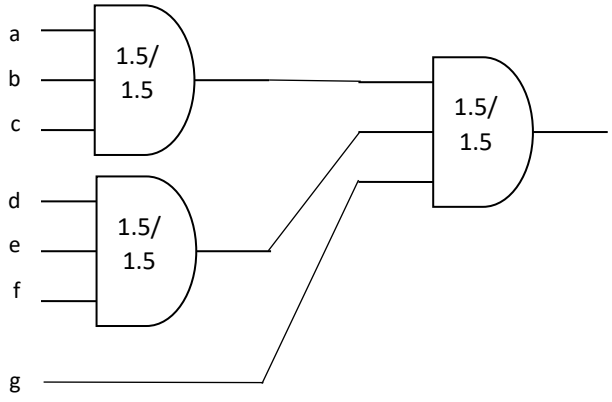
ab + bc + ac

2.

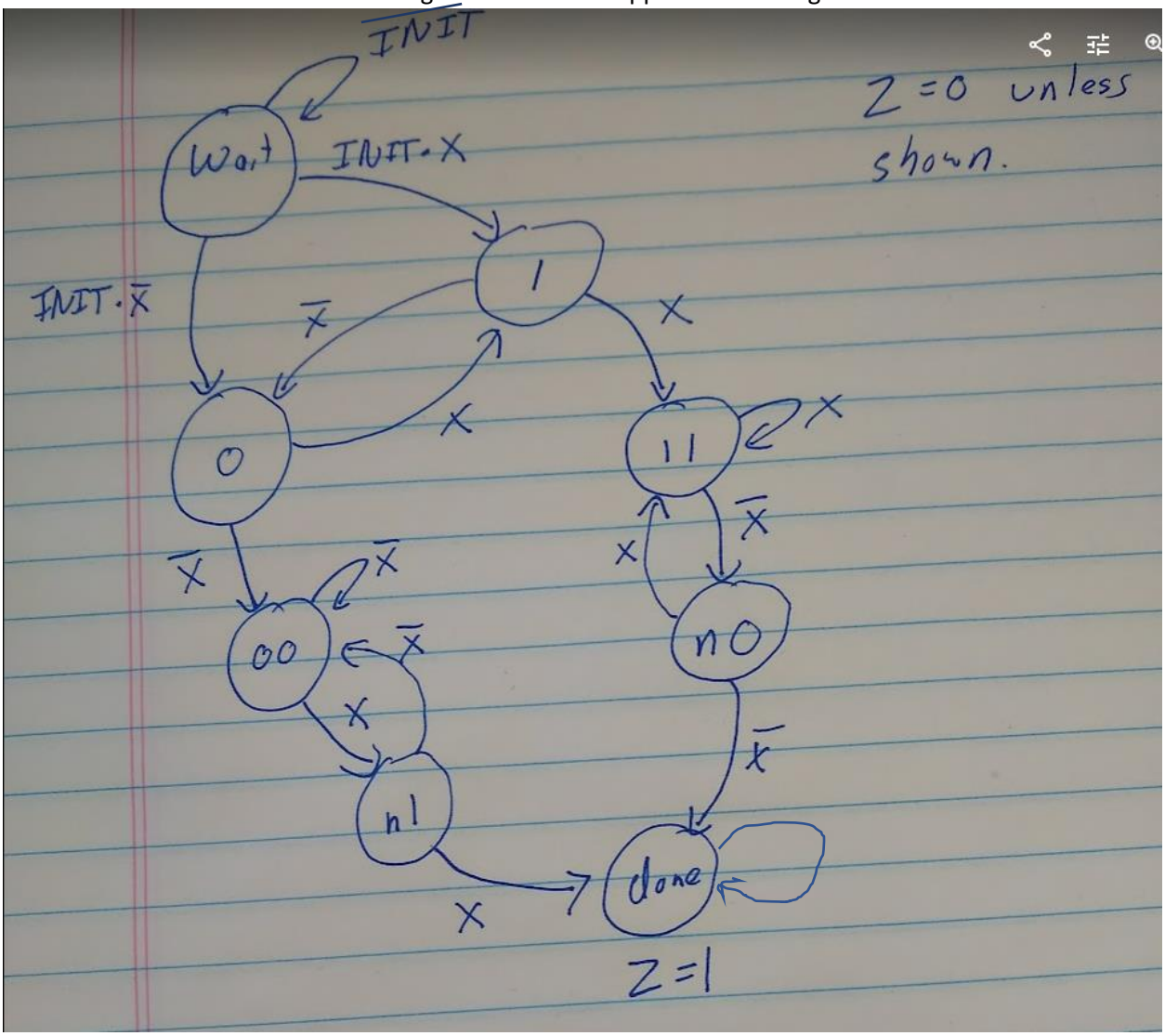


   a. !a!b!c!d, ab!c!d, a!b!cd, !a!bcd, abc!d
   b. !a!c!d, b!d, a!b!cd, !acd, !abc
   c. !a!c!d+b!d+a!b!cd+!acd

3. The question is a bit vague about the “circuit’s delay” in that it’s not immediately clear if that’s worst-case delay for the entire circuit, or the delay from any input. But looking at 6.24 (and realizing what generally matters) it becomes clear that they mean worst-case delay. There are a number of answers, the one below has the same delay (3ns) but uses only 4.5nW rather than 6nW.



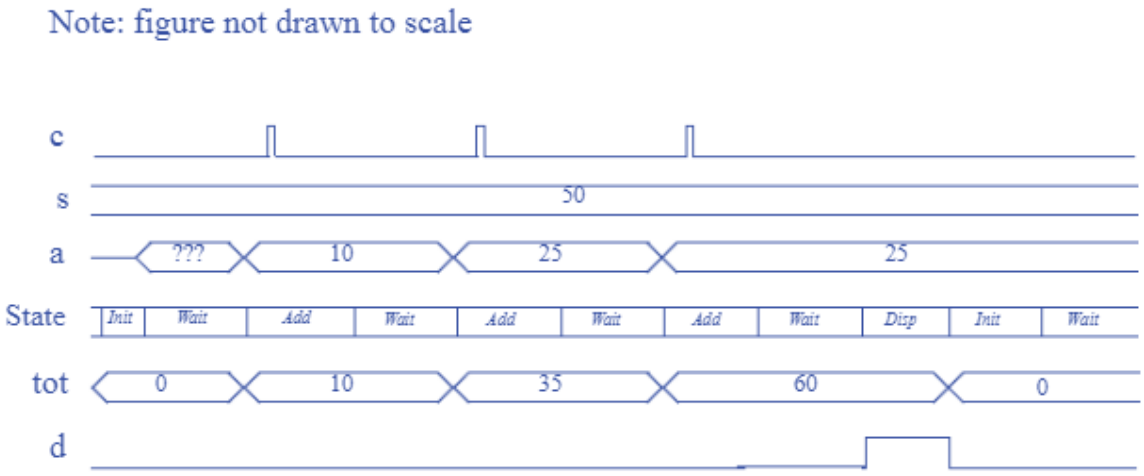
4. The question isn't as clear as it could have been, particularly with respect to INIT so we'll be flexible there. This answer is assuming that once INIT happens once we ignore it.



5. With one hot encoding you can drive each of w, x and y off of a single state bit—no logic needed. For example if we use Q[3:0] such that S0=0001, S1=0010, S2=0100, and S3=1000, then w=Q0, x=Q1, and y=Q2.

   If you use just two bits and the obvious encoding of Q[1:0] such that S0=00, S1=01, S2=10, S3=11 the best you end up with w=!(Q1+Q0)), x=!Q1*Q0, z=Q1*!Q0. So depending on how you count the not gates, something like 1 2-input NOR, 2 2-input ANDs, and 2 NOT gates for a total of 8 gate-inputs. Different assignments will give slightly different results.
   Note: I've treated the question as only asking for the outputs. A similar thing could be done, giving similar results for the next-state logic also.

6.



7. .

a.

- I, sum, a, and average are all registers (sequential)
- <16 outputs a 1 if the input is less than 16. (combinational)
- +1 Outputs its input plus one (combinational)
- + is an adder—that it is outputs the sum of its two inputs (combinational)
- >>4 outputs its input right shifted by 4.

b. Generate an address that is sent externally starting ideally 25 (but note there is no way to get 25 out, the first number would be 26…) to M_addr. That gets the data in the location M_addr sent to M_data. Then add up the first 16 values into sum. Once I is no longer <16, we can stop and grab the value the average register.