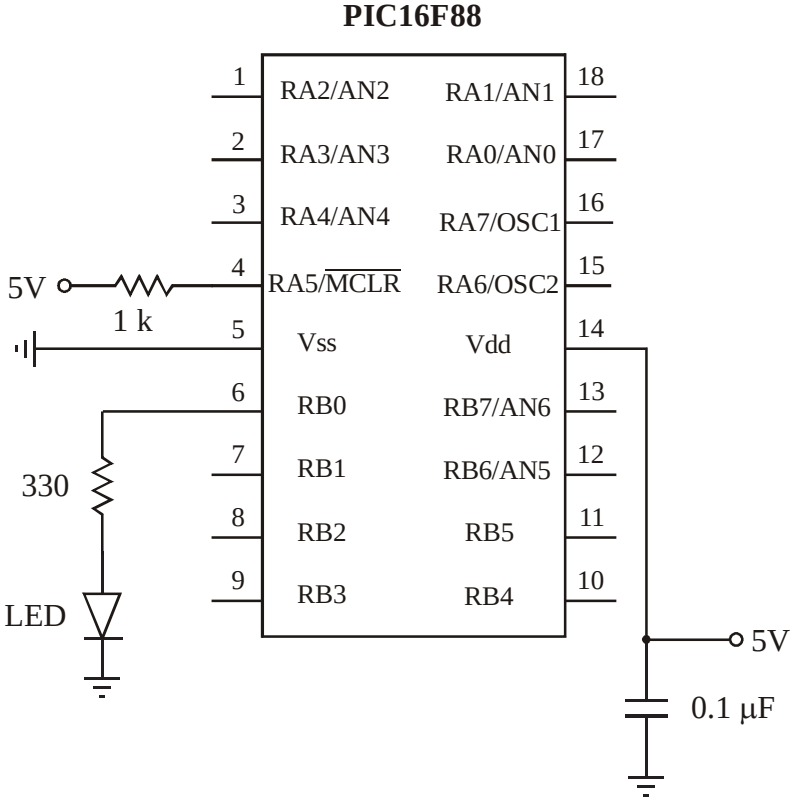
10.5 Procedure

(1) Watch the video demonstration on the Lab website of the original hexadecimal counter circuit described in Section 10.2. **Do not build this circuit.** The code and circuit in Section 10.2 is for demonstration only, and it serves as an additional example. Study the program listed in Section 10.2 and observe the functionality in the video, including the effects of holding down buttons.

(2) Using the figure below as a starting point, draw a complete and detailed wiring diagram required to implement the alternative counter design described in Section 10.3. Figure 10.3 shows useful information from the MAN6910 datasheet. Have your TA check your diagram before you continue. **PLEASE COMPLETE THIS BEFORE COMING TO LAB. NOTE: Your diagram will be very different from the one shown in Figure 10.1.**

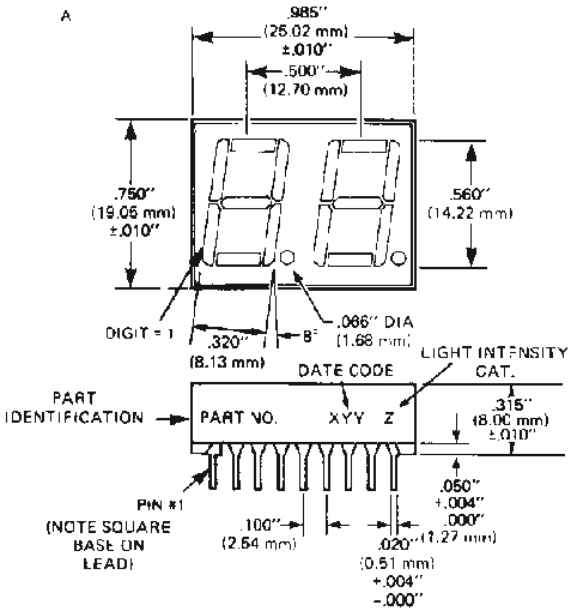


(3) Use an ASCII editor (e.g., Windows Notepad or MS Word - Text Only), or use Microcode Studio in Lab, to create the program necessary to control the alternative design. Name it "counter.bas". Save the file in a folder in your network file space named "counter." **PLEASE COMPLETE THIS BEFORE COMING TO LAB.**

(4) Follow the procedure in the previous laboratory exercise to compile the program and load it onto a PIC.

(5) Assemble and fully test your circuit with the programmed PIC. **NOTE: Make sure you use the power supply, and not the function generator, to power the circuit.**

Use the 330Ω DIP IC for the current-limiting resistors. If you are having problems, please refer to Section 10.4 for advice on how to get things working. When everything is working properly, demonstrate it to your TA for credit.



PIN NO.	A MAN6910
1	Cathode E 1
2	Cathode D 1
3	Cathode C 1
4	Cathode D.P. 1
5	Cathode E 2
6	Cathode D 2
7	Cathode G 2
8	Cathode C 2
9	Cathode D.P. 2
10	Cathode B 2
11	Cathode A 2
12	Cathode F 2
13	Anode Digit 2
14	Anode Digit 1
15	Cathode B 1
16	Cathode A 1
17	Cathode G 1
18	Cathode F 1

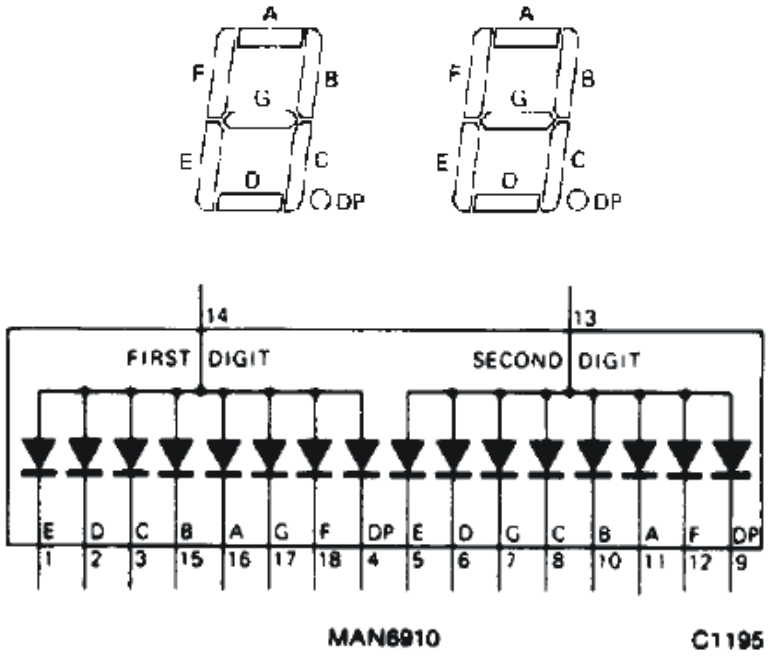


Figure 10.3 MAN6910 Datasheet Information

LAB 10 QUESTIONS

Group: ____ Names: _____ _____

_____ _____

(1) Rewrite "onint.bas" from the previous Lab using polling instead of interrupts.

(2) For the original counter design in Section 10.2, when the 'up' button is held down awhile the PIC will continue to count up. Explain why.

(3) For the original counter design in Section 10.2, explain what happens when the 'up' and 'down' buttons are held down together. Why does this happen?

(4) For the alternative counter design in Section 10.3, why is the 555 and D flip-flop hardware no longer required?

(5) Explain how switch bounce could possibly have a negative impact with the alternative design in Section 10.3 if the 0.01 sec software pause were not included.

(6) Explain why debounce software is not required for the reset button in the alternative design in Section 10.3.

(7) For the original counter design in Section 10.2 that was demonstrated in the video (i.e., not the alternative design in Section 10.3 that you built), how would you create the functionality in the Updatepins subroutine for updating the PORTA and PORTB registers using multiple individual bit references (e.g., PORTA.0 = pins[I].4, PORTA.1 = pins[I].5, ...) instead of single-line assignment statements (e.g., PORTA = ... and PORTB = ...)?
Hint: The comments above the assignment statements in the code explain what is being done.
Hint: The comments below the 7-segment-display illustration in the "counter.bas" program involving the "bit numbers" and "segments" can be helpful.