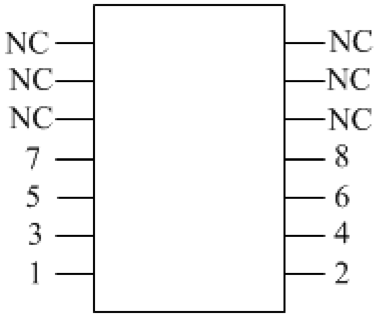
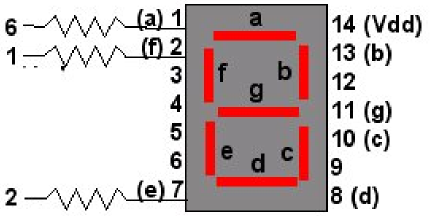
Interfacing LED Display

In this exercise we will interface a 7-segment LED display to port 2. The schematic for the hardware is given below. All resistors are 150 Ohms. The purpose of this resistor is to limit the amount of current sunk by the chip. Be sure to pay attention to details such as the wire colors and orientation of the plug-in connectors.

It is confusing that the Jxx numbers pins starting at 1 whereas this is really bit 0 of port 0, 2 is bit 1, etc. Use the C8051F02x Development Kit User's guide to make sure you have things connected correctly. **Ignore the number on the ribbon cable – they do not correspond to the numbers given below which are the port bit numbers.** For example, we want to connect port bit 0.0 to segment g so we connect Jxx.1 through a resistor to pin 11 on the common anode chip. Pin 10 from the block Jxx provides Vdd for the common anode chip. Make sure that you are supplying the correct voltage by using the probes.

We only have 7 segments so 1 bit is not used. Below this is bit 7 of port 0 which is denoted by X ("don't care") and can be set to 0 or 1 arbitrarily. It is not connected to the output chip.





Common Anode Display

Port Assignment to Segments
7 6 5 4 3 2 1 0 (Port 2)
x a b c d e f g

Write a program, in C, to write the HEX numbers 0 to F to the LED with a delay of about one second between numbers. Here is an outline of what is need to complete this assignment:

- Use a look up table Note that the LED segments are ACTIVE LOW because a common anode display is used – this means that outputting a 0 turns on the LED.
- The best way to do this is to use a lookup table to get the bit pattern to be placed on the port and then transfer this to port 0. We used lookup tables in Week 1 in the Indirect Addressing exercise so you can reuse this code. You can save the bit patterns as hex numbers in code memory.
- Send the converted value to port 0. (Don't forget to configure port 0 as an output, and enable the crossbar switch). Break the program at the point before the write so you can ensure that the value you are reading from the timer is the value that is being displayed.

Hand in your algorithm and a print out of your working C program.