Bangladesh University of Engineering and Technology
Department of Computer Science and Engineering

CSE 316 January 2021

Microprocessors, Microcontrollers, and Embedded Systems Sessional

Experiment 2

Basic LED matrix control with ATmega32: static and rotating display.

Goal:

To understand basic LED matrix control in order to generate different characters/symbols.

Fig. 1: 8x8 LED Matrix

Experimental Tools And Materials:

ATmega32, Red LED matrix, Push Buttons.

Experiment:

The experiment has the following parts.

- First, you have to show a symbol in the LED matrix. **The symbol will be assigned by the lab teachers.** This is the static mode.
- Second, you will left/right/up/down rotate the symbol. **The direction will be assigned by the lab teachers.** This is the rotating mode.
- If the button is pushed, you have to toggle between static mode and rotate mode. You have to use Interrupt for that. **The interrupt specs will be assigned by the lab teachers.**

Your Specification: MC Experiment 2 - Individual Spec
You can connect the LED matrix to any of the ports of your choice.

LED Matrix Basics:

There are 8 pins in total to select which LED(s) we want to light up. 8 of the pins are used to select the row(s), while the remaining 8 pins are used to select the column(s) (The notion of row and columns are only artificial and depends on the way we consider the orientation of the LED matrix. Rotating the matrix by 90 degrees will alter the orientation. We will be using the default orientation of the Proteus).

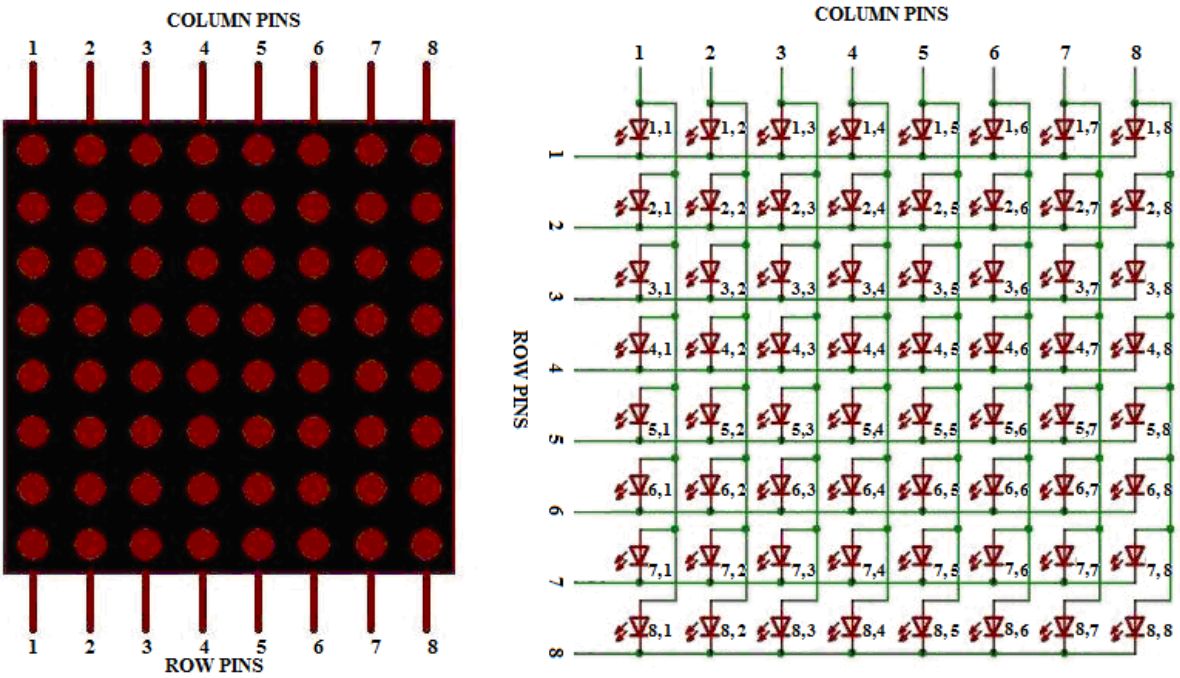
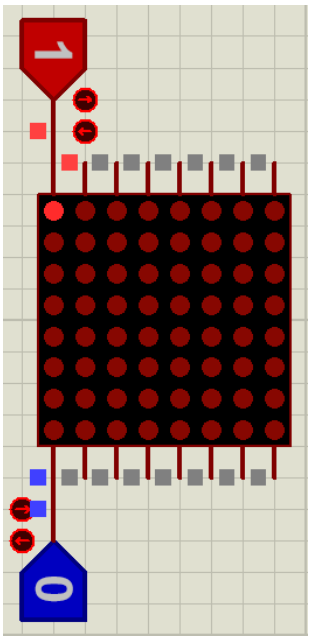


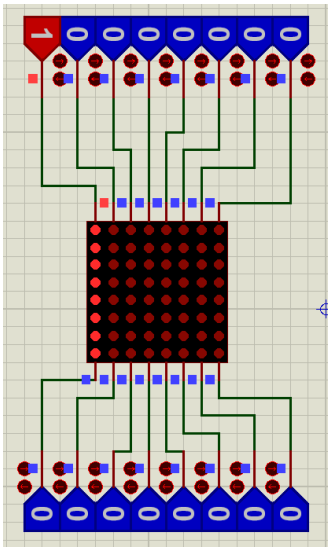
Fig. 2: 8x8 LED matrix display and its internal structure

Let's say, columns pins are C1-C8 and row pins are R1-R8. Let's name 64 LEDs as LED(row,column), e.g. LED(1,1)-LED(8,8).

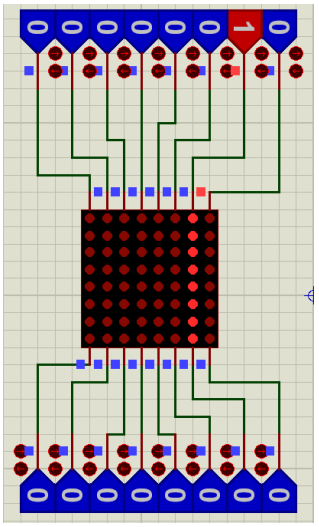


If you want to light up LED(1,1), you have to connect C1 to HIGH and R1 to LOW.

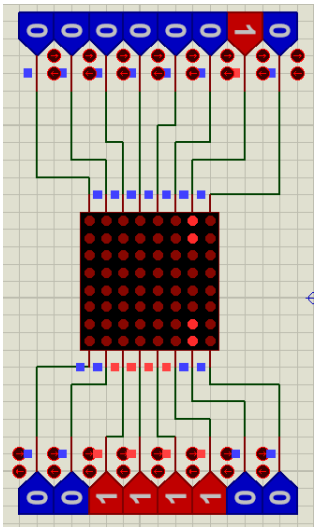
Let's connect all pins to LOGICSTATEs and see some examples.



If you want to light up the first column, set C1 to HIGH and all of R1-R8 to LOW.



Similarly, if you want to light up column 7, set C7 to HIGH and R1-R8 to LOW.



If you set C7 to HIGH, R1-R2 to LOW, R3-R6 to HIGH and R7-R8 to LOW, you will get this. Only LED(1,7), LED(2,7), LED(7,7) and LED (8,7) is on.

Showing a symbol in dot matrix:

Typically multiplexing is used to display arbitrary patterns with led matrices. Multiplexing is sometimes also called scanning. In column multiplexing, it scans columns (usually from left to right) and turns on the selected LEDs only in one row at a time following these steps:

- Turn on only the leftmost column (i.e., set HIGH)
- Turn on the necessary rows of the leftmost column by setting the corresponding row pins to LOW. This lights up the selected LEDs in the leftmost column.
- Do steps 1 and 2 one by one for all columns and keep repeating.

If you do this slowly, you would see the LED rows turning on one after another. However, if it is done fast enough, the human eye will see the whole pattern together. This phenomenon is called persistence of vision.

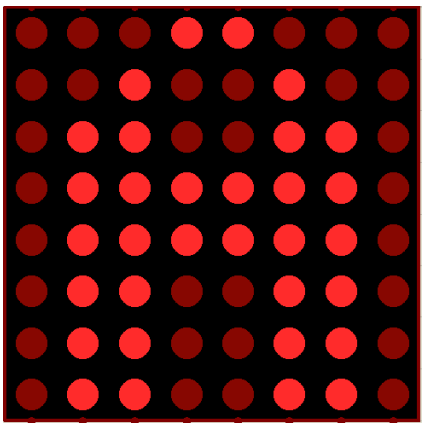
The amount of time you light up one row, before lighting up the next one is very crucial for the symbol to be displayed properly. Usually you will light up one row and then create a delay. After the delay is over you will light up the next row. If this delay is too long, one will see the rows being lit up one after another and the illusion of displaying the whole symbol at once will not work. On the other hand, if the delay is very small, the LEDs will not get enough time to glow fully. Hence, the symbol will look less bright. In the experiment you will have to find a near optimal delay on trial and error basis, so that the symbol is displayed bright and clear.

Why multiplexing:

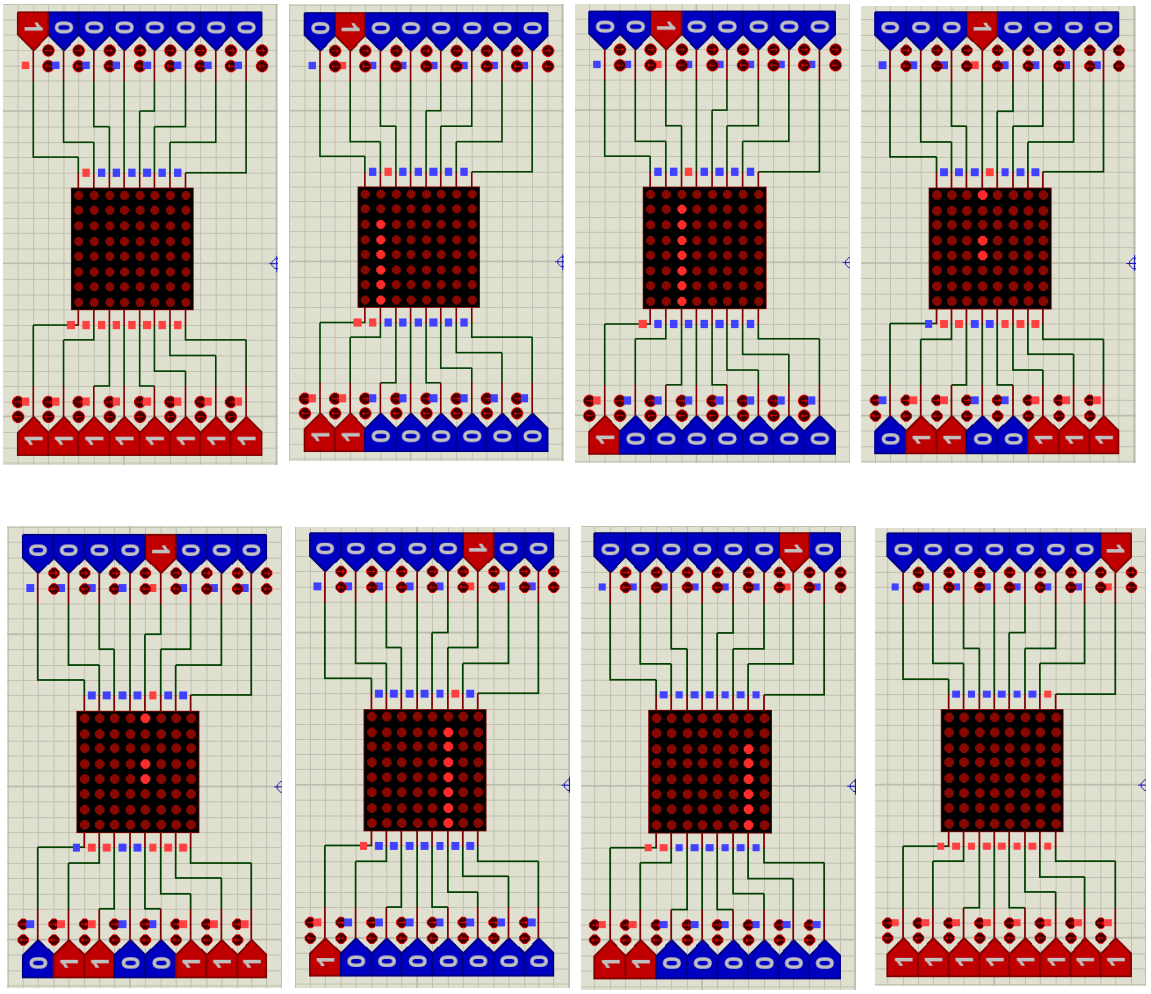
- There are 64 bicolor LEDs in a dot matrix, without multiplexing we would have needed 64 different pins to operate a single dot matrix! Using multiplexing technique we can still operate it with only 16 pins.
- Also, without multiplexing, we would have to turn on all the necessary LEDs together. That would have consumed a lot more energy. In multiplexing, we are only turning on at most 8 LEDs at a time.

Example:

Say, you want to show the following pattern,



You have to start from the leftmost column. Then for each column turn on the necessary LEDs by setting corresponding row pins to LOW.



Rotate a Symbol:

Show a symbol for some time (say, 100 ms). Then according to the direction assigned to you, rotate the symbol. After each button press you have to toggle between static mode and rotating mode. Here's a demo of what we expect. MC Experiment 2 - LED Matrix Demo.mkv . This one displays 'A' and rotates to the right.

Procedure:

1. In Proteus, connect the LED matrix to the ATmega32.
2. Write a C program to implement, **static display, rotating display and mode change by button press**.

Submission:

Open a folder with your 7 digit student ID. Copy these 2 files

1. Proteus Project file (.pdsprj)
2. C Code (.c)

Zip the folder. Submit the 1705xxx.zip.

Special Notes (11-04-2021):

You have to keep the matrix in the default orientation. In the real world, you are absolutely allowed to orient the matrix to your preferences to make your code easier. However, we are imposing the default orientation just to test your understanding.

Miscellaneous:

- Further details on working with LED matrices: How Does Led Matrix Work?

- Some pins of PORTC can not be used for I/O directly. First the JTAG has to be turned off.
  One way is to uncheck the JTAG box while writing the fuse bits, the second is to write a 1 to the JTD bit twice consecutively. MCUCSR = (1<<JTD); MCUCSR = (1<<JTD);
  For more details refer to: JTAG enabling/disabling in ATmega32 and fuse settings-SOLVED