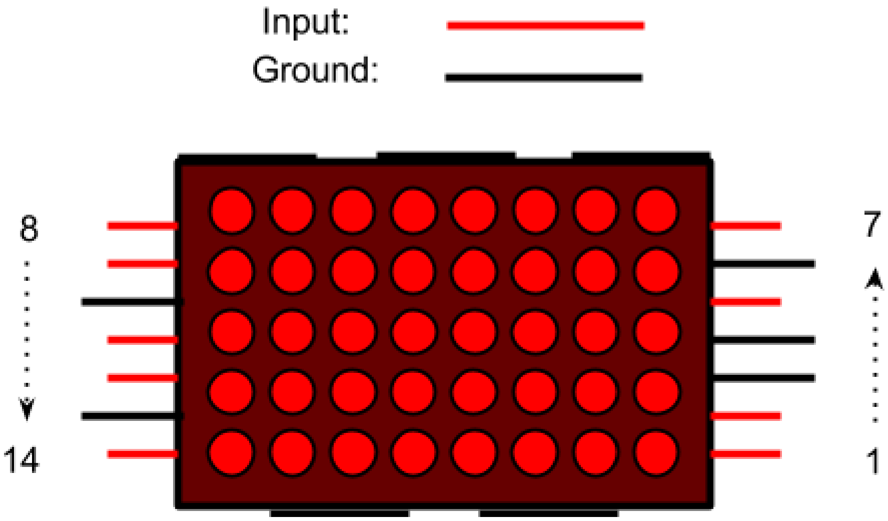
LED Matrix (Shifting Hollow Rectangle)

(1 Day)

LED matrixes are fun and compact components that allow a designer to display various shapes and colors.

How it works?

The LTP-2158AHR is a 5x8 LED matrix. The orientation of the LED matrix in the following diagrams and exercises has been chosen for simplicity.



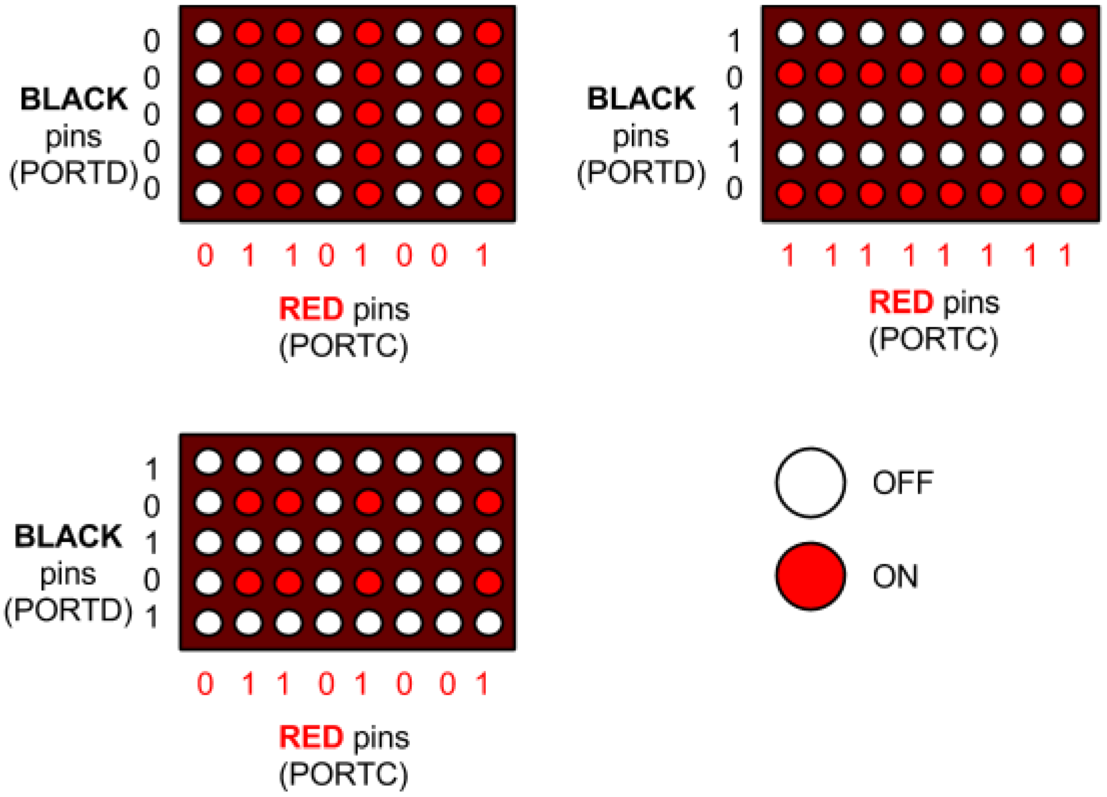
Two sets of pins, represented by **RED** and **BLACK** in the above diagram**,** are used to determine which LEDs are illuminated. You can determine the LED matrix orientation by finding the '1' on the back of the matrix next to pin 1.

The first set of pins (**RED**) control what pattern is displayed on a row of the LED matrix.

The other set of pins (**BLACK)** control which row(s) display the pattern.

If a **BLACK** pin is grounded, then the pattern present on the **RED** pins will be displayed on the column associated with the grounded **BLACK** pin.

Below are some examples of the behavior of the LED matrix.



Prelab:

NOTE: The final challenge of this lab is to connect your LED Matrix using a shift register. If you feel confident you may complete all the parts using the shift register and get credit for the final challenge to reduce the amount of board rewiring.

1) Make the following connections from the microcontroller to the LED matrix:

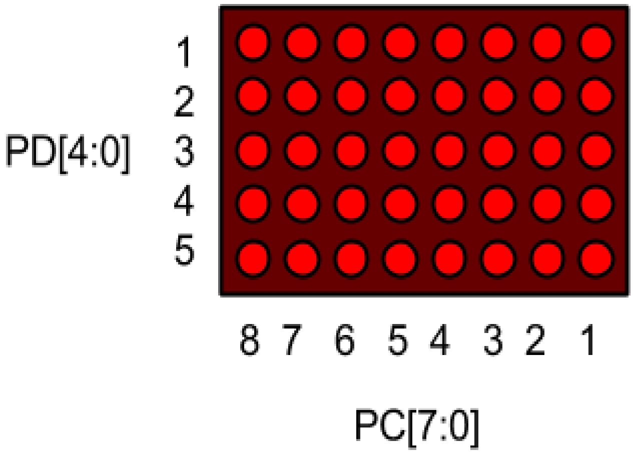
IMPORTANT: Connect resistors between PD[4:0] and the LED matrix pins. Without the resistors in place, there is a risk of burning out the LEDs in the LED matrix.

Note: You may want to flip the orientation of your microcontroller 180 degrees. In the next lab we will be combining the LED matrix with a joystick. It is easier if the matrix is at the top of the breadboard and the joystick is at the bottom. Thus you would want PORTC and PORTD oriented towards the top of your board near the LED matrix and

PORTA (required for analog control) oriented towards the bottom of the breadboard near the joystick.

PC7: pin 9 COL1 PC6: pin 14 COL2 PC5: pin 8 COL3 PC4: pin 12 COL4 PC3: pin 5 COL5 PC2: pin 1 COL6 PC1: pin 7 COL7 PC0: pin 2 COL8	PD4: pin 13 ROW5 PD3: pin 3 ROW4 PD2: pin 4 ROW3 PD1: pin 10 ROW2 PD0: pin 6 ROW1 No connection: pin 11

After making the above connections between the microcontroller and LED matrix, the diagram below shows how the pins of the microcontroller are mapped to the LEDs of the LED matrix.



2) Write a program to run a single synchSM. Copy and paste the code below into the program. This code will act as the synchSM. Set DDRC and DDRD to output. If the microcontroller is connected properly to the LED matrix, the synchSM should display a demo on the LED matrix where a single illuminated LED scrolls from left to right, top to bottom, then resets. The synchSM period should be at least **100 ms** in order for the demo to be visible.

```
//--------------------------------------
// LED Matrix Demo SynchSM
// Period: 100 ms
//--------------------------------------
enum Demo_States {shift};
int Demo_Tick(int state) {

        // Local Variables
        static unsigned char pattern = 0x80;// LED pattern - 0: LED off; 1: LED on
        static unsigned char row = 0xFE;    // Row(s) displaying pattern.
                                                   // 0: display pattern on row
                                                   // 1: do NOT display pattern on row
        // Transitions
        switch (state) {
                case shift:
                        break;
                default:
                        state = shift;
                        break;
        }
        // Actions
        switch (state) {
                case shift:
                        if (row == 0xEF && pattern == 0x01) { // Reset demo
                                pattern = 0x80;
                                row = 0xFE;
                        } else if (pattern == 0x01) { // Move LED to start of next row
                                pattern = 0x80;
                                row = (row << 1) | 0x01;
                        } else { // Shift LED one spot to the right on current row
                                pattern >>= 1;
                        }
                        break;
                default:
                        break;
        }
        PORTC = pattern;      // Pattern to display
        PORTD = row;          // Row(s) displaying pattern
        return state;
}
```

Video Demonstration of Demo: http://youtu.be/EYp3C6MWEU8

Exercise 1 -- Shifting Rows

Design a system where an illuminated row of the LED matrix can be shifted up or down based on button presses.

Criteria:

- Two buttons control the system: One button shifts the illuminated row up. The other button shifts the illuminated row down.
- The illuminated row cannot be shifted off of the matrix. (i.e. If the illuminated row reaches the top of the LED matrix, if the “up” button is pressed again, the illuminated row will remain at the top of the LED matrix.)

Video Demonstration: **http://youtu.be/DdK-x-z_xD8**

Exercise 2 -- Shifting Columns

Design a system where an illuminated column of the LED matrix can be shifted left or right based on a button press.

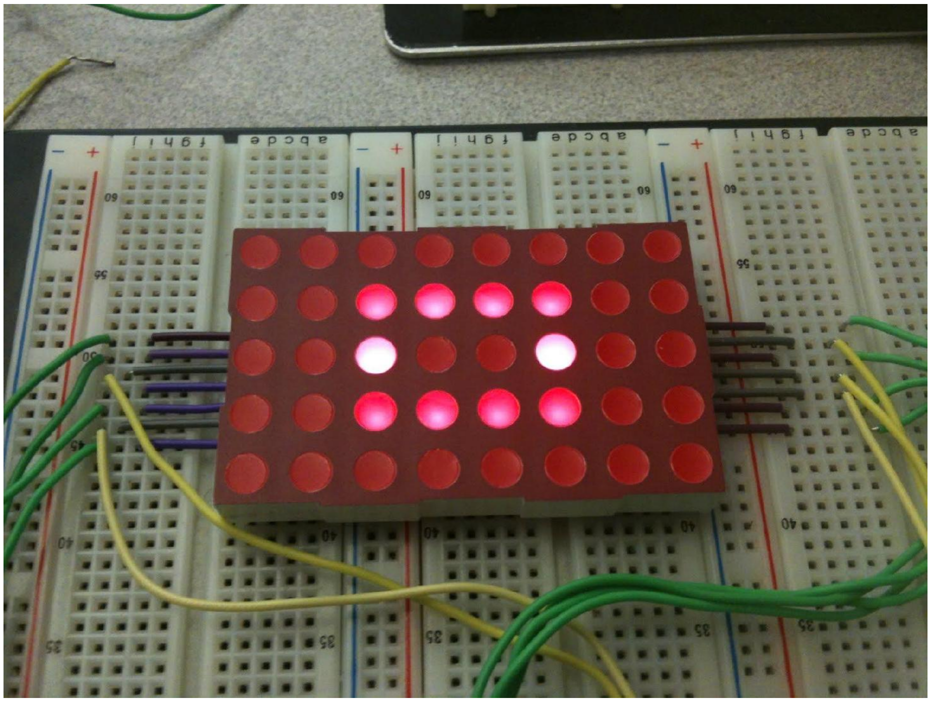
Criteria:

- Two buttons control the system: Once button shifts the illuminated column right. The other button shifts the illuminated column left.
- The illuminated column cannot be shifted off of the LED matrix (i.e. if the illuminated column reaches the far left side of the LED matrix, if the “left” button is pressed again, the illuminated column remains at the far left of the LED matrix.)

Video Demonstration: **http://youtu.be/tF-ieEPy76Q**

Exercise 3 -- Displaying a Rectangle

Design a system where a 3x4 hollow rectangle is displayed in the center of the LED matrix like the photo below: **Note**: The varying brightness of the LEDs is only present in the photo. The completed exercise will not have varying brightness.



Hints:

- Since only one row pattern can be displayed on the LED matrix at a time, to display the rectangle, different patterns need to be displayed at different times
- Every one ms, update the pattern to be displayed and the row that the pattern will be displayed on.
- Use two arrays to store relevant data for the square. One array stores the patterns for each row. The other array stores the row that displays the pattern.
- Ground only one row at a time.

Exercise 4 -- Shifting a Hollow Rectangle

Expand upon exercise 3 of the lab by introducing four buttons that control the position of the hollow rectangle.

Criteria:

- One button shifts the rectangle up
- One button shifts the rectangle down
- The other two buttons shift the rectangle left or right
- The rectangle cannot be shifted off of the LED matrix

Hints:

- Use two synchSMs. One synchSM controls the pattern displayed on the LED matrix. The other synchSM monitors button presses.

Video Demonstration: **http://youtu.be/Jpgkrmy_IZM**

Exercise 5 (challenge) -- Adding Shift Registers

Repeat exercise 4 of the lab by using shift registers instead of directly wiring the microcontroller to the LED matrix.

Each person must submit their .c source files according to instructions in the Lab submission guidelines.