EE 347

Microcontroller System Design

Project 2 Report

Interactive User Interface

with PIC18 Microcontroller

Written by: Nathan Genetzky and Drake Jeno

Due date: May 1st, 2015

Instructor: Dr. Robert Fourney

## Introduction

This project involves various hardware components as well as the PIC18 with Development Board. The ability for the PIC to obtain virtually simultaneous readings of multiple analog signals was used to determine the position of a joystick. The Liquid Crystal Display built into the development board was a critical portion of this project, and provided a way to communicate with the user. The 8x8 RGB LED Matrix provided a colorful way to demonstrate the capability of the interactive user interface.

## Theory

Interfacing with the joystick was rather intuitive because it was designed to have two analog signals that relate to the position of the joystick. The 8x8 RGB LED matrix was a little more complicated. The theory behind each component will be explored below.

Each row of LED’s in the matrix is connected to a common cathode and each column of LED’s in the matrix is connected to a common anode. An LED illuminates if its row’s common cathode is connected to +5 and its column’s common anode is connected to ground. This creates an issues if multiple rows are pulled high and multiple columns are grounded. This limitation can be overcome by only pulling a single row high at a time and then pulling the desired LEDs’ column low. If this is repeated for all columns in a quick fashion then it will seem as if all LED’s were illuminated simultaneously.

The concept above is considering a monochromatic LED matrix; however the same description can be expanded to an RGB matrix by only a slight modification. Instead of a single common anode column there are separate columns for each color. This modification increases the inputs required for the columns from 8 to 24. Figure 1 below shows the internal wiring to the light emitting diodes in the RGB matrix.

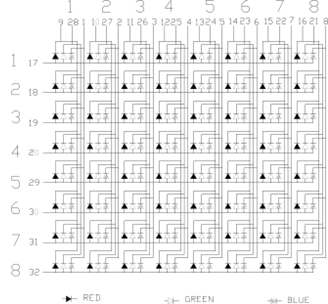


Figure Internal wiring diagram for RGB 8x8 LED Matrix [1]

A specific LED can be lit using the procedure described above; however this can only be done for one of the three primary colors at a time. The secondary colors and varying hues can be produced by cycling through the primary colors quick enough that the individual colors are not perceivable by the human eye; which means the scan of primary colors must occur at least 52 times per second. The hue of the color can be changed by varying the ratio of time each primary color is on. Figure 2 is the basic color wheel for the light spectrum (Additive Color Wheel) and Fig. 3 shows some of the common colors that can be created with a six digit RGB value.

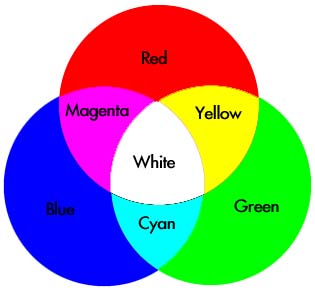


Figure Additive Color Wheel - shows colors implemented in project [2]



Figure Six Digit Hex Color Table - shows 81 colors that could be implemented in project [3]

## Procedure and Design

There was a design decision to consider when designing this system. The RGB matrix could be driven by 32 parallel outputs by 3 serial outputs. The 2nd design has the benefit of using less GPIO (General Purpose Input Output) pins, but has more components, increased complexity, and a slower response. The 1st design was chosen because errors arose in the 2nd design; with so many components the circuit was much harder to debug.

The wiring required for the RGB Matrix for the 2nd design is described in Table 1 below. It is valuable to note that the ‘Row’ connection of the RGB Matrix requires current limiting resistors; these protect both the PIC18 and the LED’s from an overflow of current that can result from a sudden change in voltage. The current limiting resistor selected was 200.

Table Description of Pin mapping between RGB Matrix and PIC18

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Row | PORTD |  | Red | PORTF |  | Green | PORTJ |  | Blue | PORTH |
| 0 | D0 |  | 0 | F0 |  | 0 | J0 |  | 0 | H0 |
| 1 | D1 |  | 1 | F1 |  | 1 | J1 |  | 1 | H1 |
| 2 | D2 |  | 2 | F2 |  | 2 | J2 |  | 2 | H2 |
| 3 | D3 |  | 3 | F3 |  | 3 | J3 |  | 3 | H3 |
| 4 | D4 |  | 4 | F4 |  | 4 | J4 |  | 4 | H4 |
| 5 | D5 |  | 5 | F5 |  | 5 | J5 |  | 5 | H5 |
| 6 | D6 |  | 6 | F6 |  | 6 | J6 |  | 6 | H6 |
| 7 | D7 |  | 7 | F7 |  | 7 | J7 |  | 7 | H7 |

The small breadboard that contains the input sources for the user interface is very simple. On the top part of the board are three active-low tactile, momentary switches. The 10 kΩ resistor connects the input pin to a (+5V) source and if the switch is not depressed then it will act as an open and the input will be read as ‘logic 1’; however, if the tactile switch is depressed then the switch acts as a short which pulls the voltage of the input to 0V, which will be read as ‘logic 0’.

The joystick simply contains two 10 kΩ potentiometers that are varied as the joystick is moved left/right or up/down. The analog signal can be converted into a usable digital signal with help of the PIC’s onboard Analog to Digital converter (A2D).

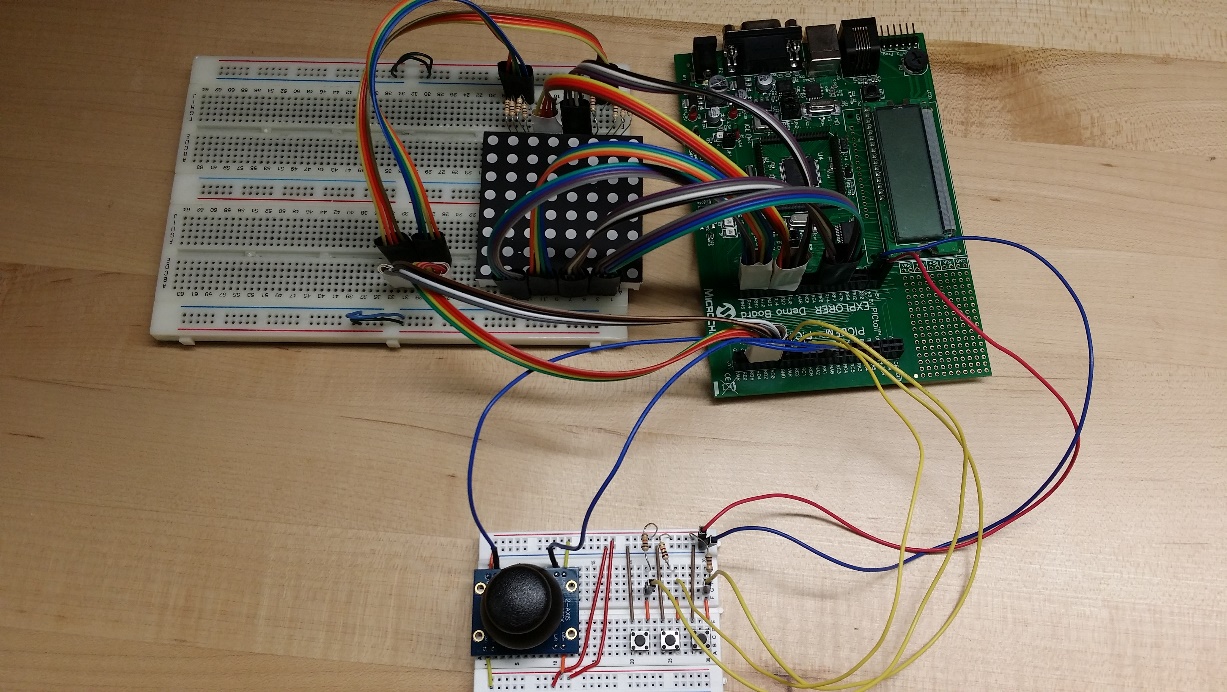


Figure Image of the Project

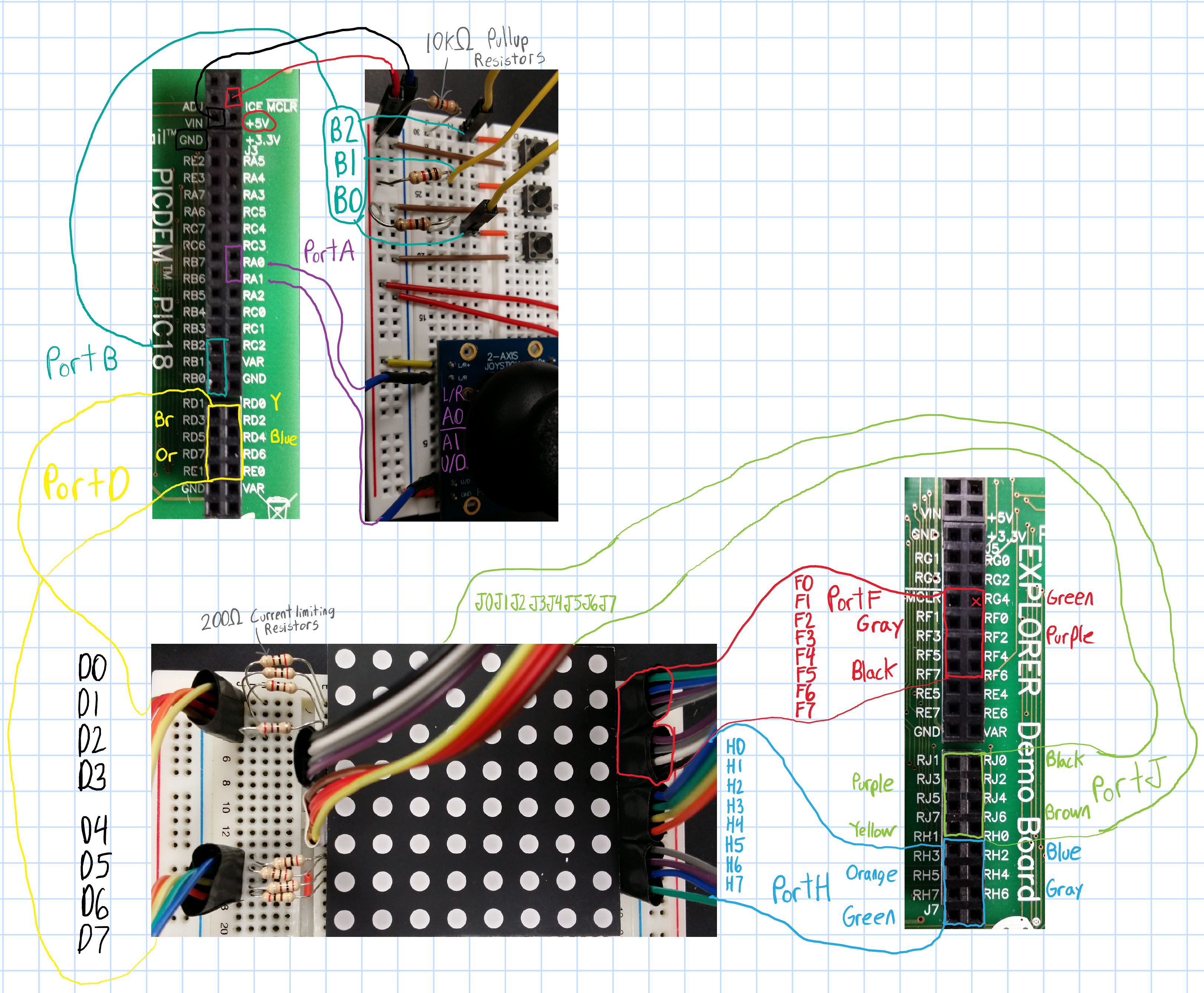


Figure Wiring Diagram for this project. Shows connections between Inputs, PIC18, and RGB 8x8 LED Matrix.

## Results and Analysis

The color capability of the RGB was used in a relatively limited fashion in this project. The current project used only combinations of red, green, and blue at full magnitude (#FF); full magnitude refers to a color being on for a delay of 255\*10 instruction cycles. By setting red, green and blue, to full magnitude or off, white and the secondary colors (yellow, magenta, and cyan) can be produced. Varying the lengths of these delays to between (#00) and (#FF) would allow for a greater range of colors, such as those shown in Fig 3. Displaying any color other than red, green and blue requires more time sensitive updates to the outputs of the matrix. The PIC 18 micro controller unit (MCU), could not update the LCD while interfacing with the RGB matrix because of conflict of required pins and for this reason special precautions had to be taken for system 3 for updating the analog readings from the joystick.

The connection to the RGB matrix is unreasonable if it desired to implement more systems in the menu of the program. Shift registers could be used in order to increase the number of outputs that are available; these combined with ‘D’ (data/delay) Flip Flops would allow a few high frequency outputs to become many slightly lower frequency outputs. If more time was invested in this project this would be the most valuable modification. By decreasing the number of outputs for the RGB from 32 to approximately 4 it would free up GPIO pins for other uses.

The Menu structure is shown in Table 2 below along with a description of each item. The menu can be navigate by using up and down on the joystick and a menu item can be selected by moving the joystick to the right or hitting RB1. By selecting a menu item the user enters what is considered a “System”.

Each System may have different actions defined for buttons RB1 and RB2, for different joystick directions and for the main while loop. For instance, the RB1 button is used to change the LED color for systems 1, 3, 4, and 5. For systems 1, 3 and 4 the RB2 button will update the LCD display expect; however for system 5 it will start the scan of the board. The Joystick directions are used for system 0 and 1.

Table Description of the Systems in the Menu

|  |  |  |
| --- | --- | --- |
| # | Name | Description |
| 0 | Home | Allows for navigation of the menu. |
| 1 | Display Smiley | Four different emoticon faces are display based on joystick direction. |
| 2 | About Dev. | Displays the software developer’s name. |
| 3 | Joystick A2D | Lights single LED based on joystick direction. |
| 4 | All LED on | Turns on all LEDs on the RGB Matrix. |
| 5 | Scan Board | Lights single LED at a time, but scans entire matrix row by row. |
| 6 | Motor | Not Implemented – Could allow control of stepper motor. |
| 7 | Keypad Entry | Not Implemented – Could allow entry of text or other data entry. |
| 8 | Receive IR | Not Implemented – Could allow control of UI by using IR remote. |

## Conclusion

This project has great potential to be expanded. In order for it to expand to its full potential the process to add a new system should be streamlined. Ideally a system could be created in a separate header/source file and be added into the project by only adding a few lines of code. Parallel to increasing the number of systems would be increasing the output devices connected to the pic, and it would be important to minimize use of GPIO pins for each device. Alternatively the project could be expanded by allowing more sources of input to interact with the menu such as a Keypad, Keyboard, IR remote, computer, or mobile device. The project could also be expanded by providing more input sensors such as a sharp distance sensor, photo resistor, pressure sensor, thermistor, or Hall Effect sensor.

The accuracy of the analog readings from the joystick while controlling the RGB LED matrix was one of these potential significant improvements. Some of the inaccuracy was hardware related, although these errors could be compensated for in software. One instance is that the readings on the two joystick pins was different, when the same position was supplied to each. This could have been resolved by analyzing these voltages and defining position separately for each of the directions, but would require significantly more processing power or sacrifice speed and consequently accuracy when updating the displays. Additional analysis would also be required to control a position on the square LED matrix with the joystick, which used a circular housing. Either intervals 64 A2D intervals would need to be hardcoded to correspond to each of the 64 positions on the LED matrix or the two readings would need to be analyzed using trig level mathematics in addition to compensating for the different readings on each A2D. The best of the two choices would have to be evaluated in terms of the respective processing delays. Reducing the functionality of the display to only the three primary colors would have likely allowed the use of either method while maintaining an accurate response time. In summary improvements to the LED display interface would require comprising either performance or functionality utilizing this configuration and the PIC18 MCU.

## References:

[1] www.seeedstudio.com/depot/datasheet/2088RGBMatrix.pd

[2] http://www.d.umn.edu/~mharvey/colorwheel.jpg

[3] http://jonteaches.com/assets/images/110/color-theory-for-the-web/hex6.jpg