Build an 8 Digit Scrolling LED Display

Description

I originally built this design to simplify the rat's nest of wiring up a bunch of individual 8x8 LED Matrix displays (see Figure 1) but ended up with such a cool looking display that I wanted to use it permanently in my shack as an On-Air indicator.

The display consists of 8 8x8 LED Matrix displays, each controlled by a MAX7219 display driver. As a frugal ham, my hint would be to purchase the complete assembly as shown in Figure 1 from an overseas vendor rather than ordering the display and display driver separately. You will likely find it less expensive that way.

Figure 1 Individual LED Matrix PCB

To drive the display drivers, I used a 5 Volt Arduino Pro Mini, which is the least expensive version of the Arduino and is more than sufficient to handle the I/O, memory and processing power required to drive the display.

The schematic and PCB also have an option to use a bare-bones ATMega328 instead of the Pro-Mini. I have not tested that part of the circuit, but since it is the same as many other similar circuits that I have built, I see no reason for it not to work (famous last words?).

Figure 2 shows what the completed PCB looks like. This is Version A and it has some errors, mainly due to my forgetting to check all connections before sending the Gerbers off to a PCB house. Figure 3 shows the mods made to the version A PCB. On Figure 2, you will see the Pro-Mini in the bottom right corner.

Lastly, Figure 5 shows the finished project in a custom-made enclosure. This was made from some 1x4 Oak boards from the local Home Depot, stained and with red transparent Plexiglass added to make the display stand out. The Plexiglass is held in place by 1/8" grooves cut using a table saw.

Power

Power for the display is taken from the USB-B connector shown on the bottom left corner. Connect this to any 1 Ampere (or better) 5 Volt DC power adapter, which are common for most phones these days.

The PCB also has a provision for a 12 to 15 Volt DC power input on the bottom right side (see Figure 4). This input is connected to J1 (make sure you have polarity correct! Version A PCB's have the + marked incorrectly.) and is smoothed by Capacitor C9, a 100uFd 20WVDC electrolytic capacitor, and regulated by U18 which is a LM7805 regulator. 1 Ampere of DC power is more than enough to handle the entire display.

Figure 2 Completed PCB (Version A)

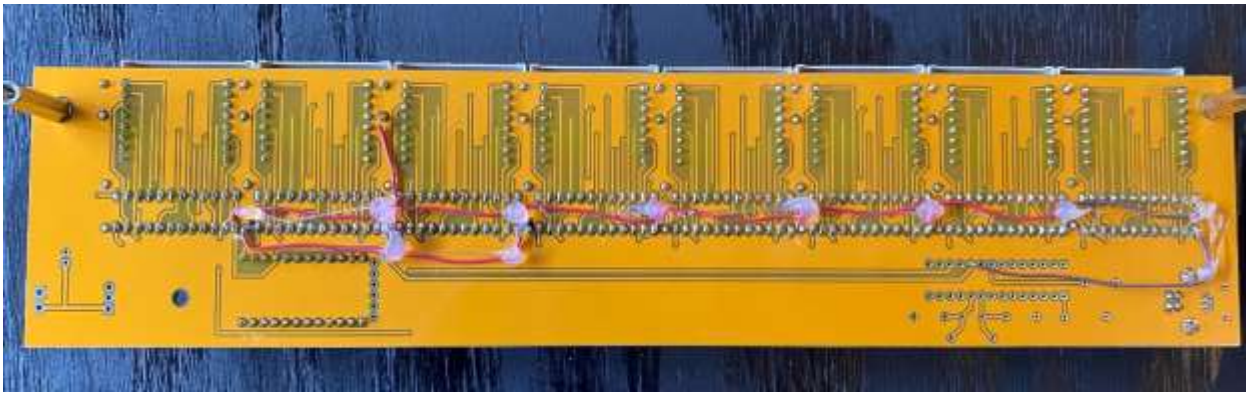

Figure 3 Figure 3 Back side of PCB (Version A)

A Version B Gerber set is available (see Appendix 2) and it corrects the issues found with Version A.

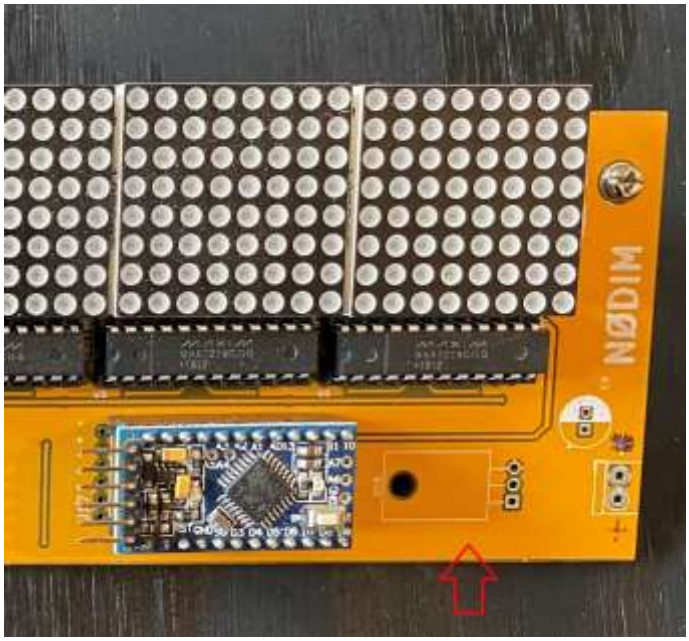

Figure 4 Location of Alternate Powering

Figure 5 The Completed Project in its new enclosure

Software

Rather than re-invent the wheel, I started with the software published by CarterW16 – see Appendix 2 for the link to their Instructables article, which includes the original code as well as a link to the Arduino library required for this project.

The completed code is shown in appendix 3. I've highlighted the code that you will need to change to customize the message displayed.

Other Material

If you want build your own display using the PCB, I have included a link to the Gerber Zip file in Appendix 2. Note: your browser may balk at downloading a Zip file. If this concerns you, contact me.

You can order PCB's using this file from just about any PCB house. I use JLPCB in China and OshPark in the USA for PCB's.

Parts sources for the other components are in Appendix 2. I've included USA sources, but I would encourage you to consider using alternative sources to save some (many) bucks.

The rest of the parts are very common and you should have them in your junk box.

There are 2 footprints for USB connectors on the PCB. The USB-B connector is quite common and available from multiple sources. The alternate footprint is for a micro-USB connector, which is less common, but still available.

73

Mitch N0DIM

Appendix 1 – Schematic (see also Appendix 2)

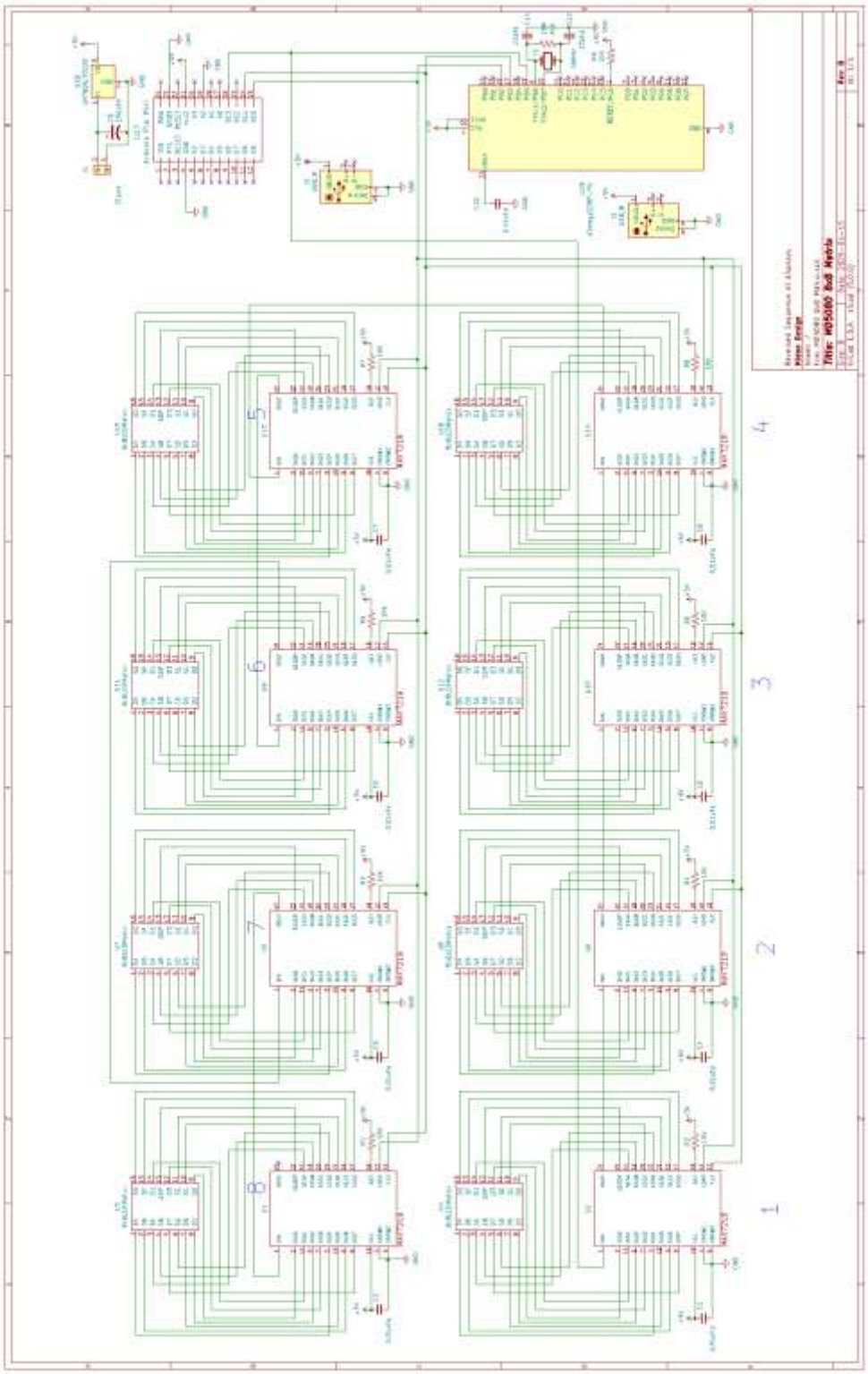

Figure 6 MD5080 Display Schematic

Appendix 2 - Sources

1. Figure 1 from https://www.ebay.com/i/253223466464
2. Original Arduino code from https://www.instructables.com/id/Scrolling-Text-on-a-8x8-LED-Matrix-Using-an-Arduin/
3. PCB House USA: https://oshpark.com/
4. PCB House China: https://jlcpcb.com/
5. Gerbers for Version B PCB: http://n0dim.com/Documents/MD5080%208x8%20MatrixVB.zip
6. Arduino Pro Mini used: https://www.sparkfun.com/products/11113
7. MAX7219 (one example of a USA source): https://www.adafruit.com/product/453
8. 8X8 LED Matrix (one example of a USA source): https://www.adafruit.com/product/454
9. USB-B Connector (one example of a USA source): https://www.jameco.com/z/USB-B-S-RA-CS1-SPCC-WT-Adam-Technologies-Connector-USB-2-0-USB-B-Pcb-Female-Right-Angle_230958.html
10. Micro-USB Connector (one example of a USA source): https://www.jameco.com/z/292303-1-TE-Connectivity-Connector-USB-Recepticle-4-Position-Solder-Right-Angle-Thru-Hole-4-Terminal_921651.html
11. MD5080 Schematic: http://n0dim.com/Documents/MD5080 Schematic.pdf

Appendix 3 – Arduino Code

```
//--------------------------------------------------------------------------------------------
-------------------------
//  Max7219Demo2
//  19-10-16 created from https://www.instructables.com/id/Scrolling-Text-on-a-8x8-
LED-Matrix-Using-an-Arduin/
//  Uses MaxMatrix Library successfully
//
//  20-01-16 Modified by N0DIM Mitch Matteau
//    Hardware in use = MD5080 PCB VA with mods (reversed DIN/DOUT - see current
schematic)
//                                        (was missing connections to +5 for R3, R5
and R7)
//    Hardware compatible = MD5080 PCB VB (no mods required)
//--------------------------------------------------------------------------------------------
-------------------------
//

// maxmatrix-disp-scroll-text-7219
#include <MaxMatrix.h>
#include <avr/pgmspace.h>

#define maxDisplays 8 // Number of MAX7219's in use.

byte Buf7219[7]; // "width,height,data[5]" single character buffer.

//For use with Arduino Pro-mini:
const int data  = 13;     // DIN or MOSI
const int load  = 10;     // CS
const int clock = 11;     // SCK

//original pin config
//const int data  = 11;     // DIN or MOSI
//const int load  = 10;     // CS
//const int clock = 13;     // SCK

MaxMatrix m(data, load, clock, maxDisplays);
// Data array is stored in program memory (see memcpy_P for access).
// Parameters are width, height, character data...
// There is a speed improvement for characters with height 8 bits see lib.

PROGMEM const unsigned char CH[] = {
3, 8, B0000000, B0000000, B0000000, B0000000, B0000000, // space
1, 8, B1011111, B0000000, B0000000, B0000000, B0000000, // !
3, 8, B0000011, B0000000, B0000011, B0000000, B0000000, // "
5, 8, B0010100, B0111110, B0010100, B0111110, B0010100, // #
4, 8, B0100100, B1101010, B0101011, B0010010, B0000000, // $
5, 8, B1100011, B0010011, B0001000, B1100100, B1100011, // %
5, 8, B0110110, B1001001, B1010110, B0100000, B1010000, // &
1, 8, B0000011, B0000000, B0000000, B0000000, B0000000, // '
3, 8, B0011100, B0100010, B1000001, B0000000, B0000000, // (
3, 8, B1000001, B0100010, B0011100, B0000000, B0000000, // )
5, 8, B0101000, B0011000, B0001110, B0011000, B0101000, // *
5, 8, B0001000, B0001000, B0111110, B0001000, B0001000, // +
2, 8, B10110000, B1110000, B0000000, B0000000, B0000000, // ,
4, 8, B0001000, B0001000, B0001000, B0001000, B0000000, // -
2, 8, B1100000, B1100000, B0000000, B0000000, B0000000, // .
4, 8, B1100000, B0011000, B0000110, B0000001, B0000000, // /
4, 8, B0111110, B1010001, B1001001, B0111110, B0000000, // Ø
3, 8, B1000010, B1111111, B1000000, B0000000, B0000000, // 1
```

```
4, 8, B1100010, B1010001, B1001001, B1000110, B0000000, // 2
4, 8, B0100010, B1000001, B1001001, B0110110, B0000000, // 3
4, 8, B0011000, B0010100, B0010010, B1111111, B0000000, // 4
4, 8, B0100111, B1000101, B1000101, B0111001, B0000000, // 5
4, 8, B0111110, B1001001, B1001001, B0110000, B0000000, // 6
4, 8, B1100001, B0010001, B0001001, B0000111, B0000000, // 7
4, 8, B0110110, B1001001, B1001001, B0110110, B0000000, // 8
4, 8, B0000110, B1001001, B1001001, B0111110, B0000000, // 9
2, 8, B01010000, B0000000, B0000000, B0000000, B0000000, // :
2, 8, B10000000, B01010000, B0000000, B0000000, B0000000, // ;
3, 8, B0010000, B0101000, B1000100, B0000000, B0000000, // <
3, 8, B0010100, B0010100, B0010100, B0000000, B0000000, // =
3, 8, B1000100, B0101000, B0010000, B0000000, B0000000, // >
4, 8, B0000010, B1011001, B0001001, B0000110, B0000000, // ?
5, 8, B0111110, B1001001, B1010101, B1011101, B0001110, // @
4, 8, B1111110, B0010001, B0010001, B1111110, B0000000, // A
4, 8, B1111111, B1001001, B1001001, B0110110, B0000000, // B
4, 8, B0111110, B1000001, B1000001, B0100010, B0000000, // C
4, 8, B1111111, B1000001, B1000001, B0111110, B0000000, // D
4, 8, B1111111, B1001001, B1001001, B1000001, B0000000, // E
4, 8, B1111111, B0001001, B0001001, B0000001, B0000000, // F
4, 8, B0111110, B1000001, B1001001, B1111010, B0000000, // G
4, 8, B1111111, B0001000, B0001000, B1111111, B0000000, // H
3, 8, B1000001, B1111111, B1000001, B0000000, B0000000, // I
4, 8, B0110000, B1000000, B1000001, B0111111, B0000000, // J
4, 8, B1111111, B0001000, B0010100, B1100011, B0000000, // K
4, 8, B1111111, B1000000, B1000000, B1000000, B0000000, // L
5, 8, B1111111, B0000010, B0001100, B0000010, B1111111, // M
5, 8, B1111111, B0000100, B0001000, B0010000, B1111111, // N
4, 8, B0111110, B1000001, B1000001, B0111110, B0000000, // O
4, 8, B1111111, B0001001, B0001001, B0000110, B0000000, // P
4, 8, B0111110, B1000001, B1000001, B10111110, B0000000, // Q
4, 8, B1111111, B0001001, B0001001, B1110110, B0000000, // R
4, 8, B1000110, B1001001, B1001001, B0110010, B0000000, // S
5, 8, B0000001, B0000001, B1111111, B0000001, B0000001, // T
4, 8, B0111111, B1000000, B1000000, B0111111, B0000000, // U
5, 8, B0001111, B0110000, B1000000, B0110000, B0001111, // V
5, 8, B0111111, B1000000, B0111000, B1000000, B0111111, // W
5, 8, B1100011, B0010100, B0001000, B0010100, B1100011, // X
5, 8, B0000111, B0001000, B1110000, B0001000, B0000111, // Y
4, 8, B1100001, B1010001, B1001001, B1000111, B0000000, // Z
2, 8, B1111111, B1000001, B0000000, B0000000, B0000000, // [
4, 8, B0000001, B0000110, B0011000, B1100000, B0000000, // backslash
2, 8, B1000001, B1111111, B0000000, B0000000, B0000000, // ]
3, 8, B0000010, B0000001, B0000010, B0000000, B0000000, // hat
4, 8, B1000000, B1000000, B1000000, B1000000, B0000000, // _
2, 8, B0000001, B0000010, B0000000, B0000000, B0000000, // `
4, 8, B0100000, B1010100, B1010100, B1111000, B0000000, // a
4, 8, B1111111, B1000100, B1000100, B0111000, B0000000, // b
4, 8, B0111000, B1000100, B1000100, B0000000, B0000000, // c // JFM MOD.
4, 8, B0111000, B1000100, B1000100, B1111111, B0000000, // d
4, 8, B0111000, B1010100, B1010100, B0011000, B0000000, // e
3, 8, B0000100, B1111110, B0000101, B0000000, B0000000, // f
4, 8, B10011000, B10100100, B10100100, B01111000, B0000000, // g
4, 8, B1111111, B0000100, B0000100, B1111000, B0000000, // h
3, 8, B1000100, B1111101, B1000000, B0000000, B0000000, // i
4, 8, B1000000, B10000000, B10000100, B1111101, B0000000, // j
4, 8, B1111111, B0010000, B0101000, B1000100, B0000000, // k
3, 8, B1000001, B1111111, B1000000, B0000000, B0000000, // l
5, 8, B1111100, B0000100, B1111100, B0000100, B1111000, // m
4, 8, B1111100, B0000100, B0000100, B1111000, B0000000, // n
4, 8, B0111000, B1000100, B1000100, B0111000, B0000000, // o
4, 8, B11111100, B0100100, B0100100, B0011000, B0000000, // p
```

```
4, 8, B0011000, B0100100, B0100100, B11111100, B0000000, // q
4, 8, B1111100, B0001000, B0000100, B0000100, B0000000, // r
4, 8, B1001000, B1010100, B1010100, B0100100, B0000000, // s
3, 8, B0000100, B0111111, B1000100, B0000000, B0000000, // t
4, 8, B0111100, B1000000, B1000000, B1111100, B0000000, // u
5, 8, B0011100, B0100000, B1000000, B0100000, B0011100, // v
5, 8, B0111100, B1000000, B0111100, B1000000, B0111100, // w
5, 8, B1000100, B0101000, B0010000, B0101000, B1000100, // x
4, 8, B10011100, B10100000, B10100000, B1111100, B0000000, // y
3, 8, B1100100, B1010100, B1001100, B0000000, B0000000, // z
3, 8, B0001000, B0110110, B1000001, B0000000, B0000000, // {
1, 8, B1111111, B0000000, B0000000, B0000000, B0000000, // |
3, 8, B1000001, B0110110, B0001000, B0000000, B0000000, // }
4, 8, B0001000, B0000100, B0001000, B0000100, B0000000, // ~
};

void setup()
{
  m.init();
  m.setIntensity(1); //change brightness
//  m.setIntensity(4); //change brightness
}

// Scrolling Text
char string[]  = "N0DIM    "; //edit text message
char string1[] = "On the Air  ";
char string2[] = "Mitch";

void loop()        //Send text
{
// delay(1000);//delay between messages
  m.shiftLeft(false, true);
  printStringWithShift(string,1);   //number is message speed (actually it's the
delay!  The bigger the number, the longer the delay.)
  delay(1000);
  m.clear();
  m.shiftLeft(false, true);
  printStringWithShift(string1,1);
  delay(1000);
  m.clear();
//  m.shiftLeft(false, true);
//  printStringWithShift(string2,200);
//  m.clear();
}


// Put text on Display
void printCharWithShift(char c, int shift_speed)
{
  if (c < 32) return;
  c -= 32;
  memcpy_P(Buf7219, CH + 7*c, 7);
  m.writeSprite(maxDisplays*8, 0, Buf7219);
  m.setColumn(maxDisplays*8 + Buf7219[0], 0);


  for (int i=0; i<=Buf7219[0]; i++)
  {
    delay(shift_speed);
    m.shiftLeft(false, false);
  }

}
```

```
void printStringWithShift(char* s, int shift_speed)
{
  while (*s != 0)
  {
    printCharWithShift(*s, shift_speed);
    s++;
  }
}
```