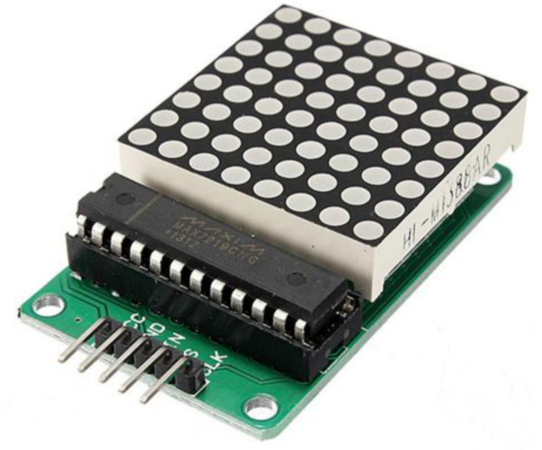
Handson Technology

User Guide

MAX7218 8x8 Dot Matrix Display Module

This is 8x8 (row by column 64x LED) dot matrix LED displays module based on MAX7219 IC. The displays are designed so that they can be mounted in a horizontal chain and can also be expanded in a vertical plane allowing versatile displays panel to be built. A convenient 3-wire serial interface connects to all common controller board like Arduino or Raspberry. Individual dot may be addressed and updated without rewriting the entire display. This module can be daisy chain to form a display panel for scrolling message board.

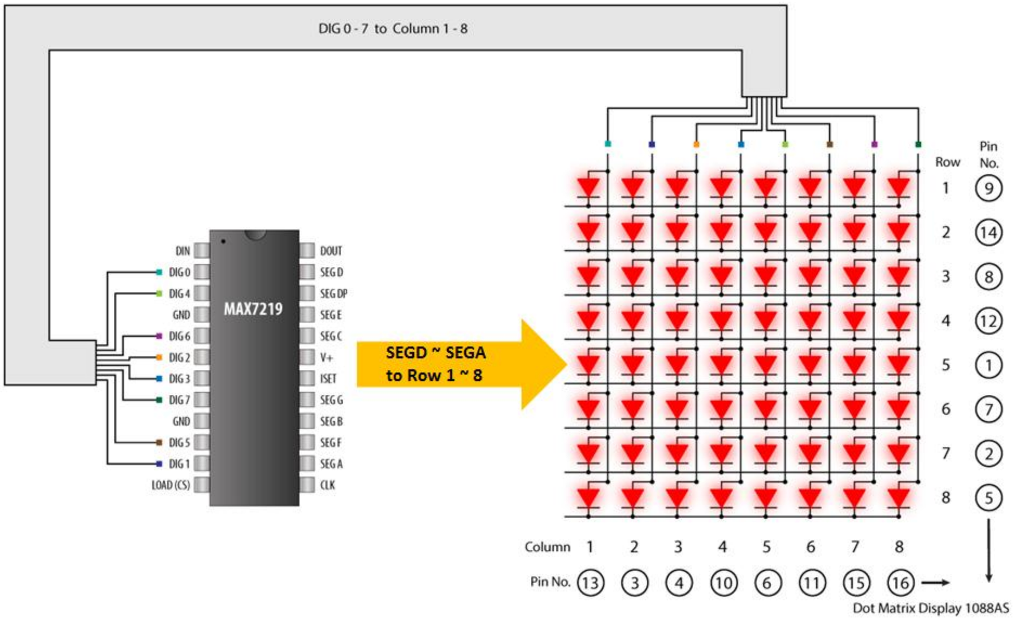


SKU: DSP-1172

Brief Data:

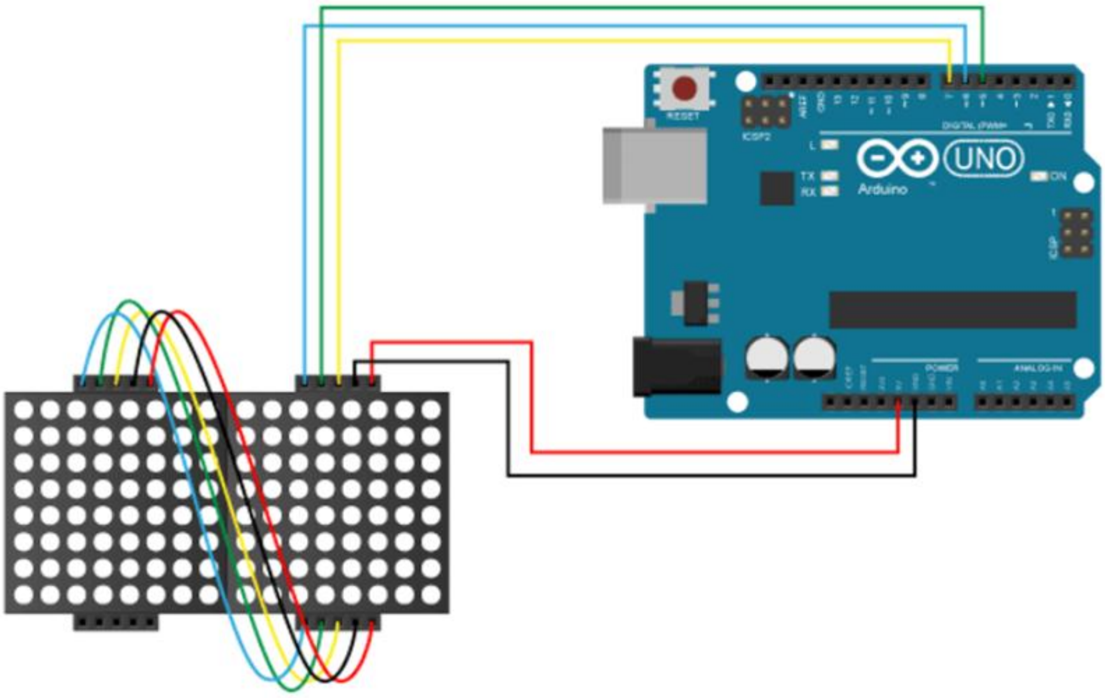
- Matrix Size: 8x8 (64-Dots)
- Display Size: 1.3".
- Display Color: Red.
- Interface: 3-Wires Serial Interface.
- Daisy chain for multiple modules.
- Operating voltage: 4.5 ~ 5V.
- Module size: 5 x 3.2 x 1.5cm (L x W x H).
- Mounting Hole: M3.

MAX7291 Module Matrix Connection:



Arduino Connection Examples:

Now let’s connect the 8×8 LED Matrix module to the Arduino Board. Here’s the circuit schematic:



Connecting Table:

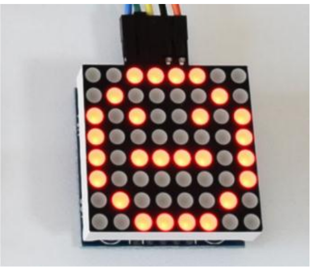
Arduino	MAX7219 Module
5V	VCC
GND	GND
D7	DIN
D5	CS
D6	CLK

Once we wire up the modules we are ready to take a look at the Arduino code of the first example. We will use the MaxMatrix library which can be downloaded from GitHub.

Copy and paste the below sketch to Arduino IDE and upload to Arduino Uno board:

```
/*
  8x8 LED Matrix MAX7219 Example 01
  Based on the following library:
  GitHub | riyas-org/max7219  https://github.com/riyas-org/max7219
*/

#include <MaxMatrix.h>

int DIN = 7;   // DIN pin of MAX7219 module
int CLK = 6;   // CLK pin of MAX7219 module
int CS = 5;    // CS pin of MAX7219 module
int maxInUse = 1;


MaxMatrix m(DIN, CS, CLK, maxInUse);

char A[] = {4, 8,
            B01111110,
            B00010001,
            B00010001,
            B01111110,
           };

char B[] = {4, 8,
            B01111111,
            B01001001,
            B01001001,
            B00110110,
           };

char smile01[] = {8, 8,
                  B00111100,
                  B01000010,
                  B10010101,
                  B10100001,
                  B10100001,
                  B10010101,
                  B01000010,
                  B00111100
                 };
char smile02[] = {8, 8,
                  B00111100,
                  B01000010,
                  B10010101,
                  B10010001,
                  B10010001,
                  B10010101,
                  B01000010,
                  B00111100
                 };
char smile03[] = {8, 8,
                  B00111100,
                  B01000010,
                  B10100101,
                  B10010001,
                  B10010001,
                  B10100101,
                  B01000010,
                  B00111100
                 };

void setup() {
  m.init(); // MAX7219 initialization
  m.setIntensity(8); // initial led matrix intensity, 0-15
}

void loop() {
  // Seting the LEDs On or Off at x,y or row,column position
  m.setDot(6,2,true);
  delay(1000);
  m.setDot(6,3,true);
  delay(1000);
  m.clear(); // Clears the display
  for (int i=0; i<8; i++){
    m.setDot(i,i,true);
    delay(300);
  }
  m.clear();
  // Displaying the character at x,y (upper left corner of the character)
  m.writeSprite(2, 0, A);
  delay(1000);

  m.writeSprite(2, 0, B);
  delay(1000);

  m.writeSprite(0, 0, smile01);
  delay(1000);

  m.writeSprite(0, 0, smile02);
  delay(1000);

  m.writeSprite(0, 0, smile03);
  delay(1000);

  for (int i=0; i<8; i++){
    m.shiftLeft(false,false);
    delay(300);
  }
  m.clear();

}
```

Once successfully upload, an image will be displayed on the module:



Next let’s take a look at the scrolling text example and see what’s different. Copy and paste the below sketch to Arduino IDE and upload to Arduino Uno board:

```
/*
   8x8 LED Matrix MAX7219 Scrolling Text Example
   Based on the following library:
   GitHub | riyas-org/max7219  https://github.com/riyas-org/max7219
*/

#include <MaxMatrix.h>
#include <avr/pgmspace.h>

PROGMEM const unsigned char CH[] = {
  3, 8, B00000000, B00000000, B00000000, B00000000, B00000000, // space
  1, 8, B01011111, B00000000, B00000000, B00000000, B00000000, // !
  3, 8, B00000011, B00000000, B00000011, B00000000, B00000000, // "
  5, 8, B00010100, B00111110, B00010100, B00111110, B00010100, // #
  4, 8, B00100100, B01101010, B00101011, B00010010, B00000000, // $
  5, 8, B01100011, B00010011, B00001000, B01100100, B01100011, // %
  5, 8, B00110110, B01001001, B01010110, B00100000, B01010000, // &
  1, 8, B00000011, B00000000, B00000000, B00000000, B00000000, // '
  3, 8, B00011100, B00100010, B01000001, B00000000, B00000000, // (
  3, 8, B01000001, B00100010, B00011100, B00000000, B00000000, // )
  5, 8, B00101000, B00011000, B00001110, B00011000, B00101000, // *
  5, 8, B00001000, B00001000, B00111110, B00001000, B00001000, // +
  2, 8, B10110000, B01110000, B00000000, B00000000, B00000000, // ,
  4, 8, B00001000, B00001000, B00001000, B00001000, B00000000, // -
  2, 8, B01100000, B01100000, B00000000, B00000000, B00000000, // .
  4, 8, B01100000, B00011000, B00000110, B00000001, B00000000, // /
  4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // 0
  3, 8, B01000010, B01111111, B01000000, B00000000, B00000000, // 1
  4, 8, B01100010, B01010001, B01001001, B01000110, B00000000, // 2
  4, 8, B00100010, B01000001, B01001001, B00110110, B00000000, // 3
  4, 8, B00011000, B00010100, B00010010, B01111111, B00000000, // 4
  4, 8, B00100111, B01000101, B01000101, B00111001, B00000000, // 5
  4, 8, B00111110, B01001001, B01001001, B00110000, B00000000, // 6
  4, 8, B01100001, B00010001, B00001001, B00000111, B00000000, // 7
  4, 8, B00110110, B01001001, B01001001, B00110110, B00000000, // 8
  4, 8, B00000110, B01001001, B01001001, B00111110, B00000000, // 9
  2, 8, B01010000, B00000000, B00000000, B00000000, B00000000, // :
  2, 8, B10000000, B01010000, B00000000, B00000000, B00000000, // ;
  3, 8, B00010000, B00101000, B01000100, B00000000, B00000000, // <
  3, 8, B00010100, B00010100, B00010100, B00000000, B00000000, // =
  3, 8, B01000100, B00101000, B00010000, B00000000, B00000000, // >
  4, 8, B00000010, B01011001, B00001001, B00000110, B00000000, // ?
  5, 8, B00111110, B01001001, B01010101, B01011101, B00001110, // @
  4, 8, B01111110, B00010001, B00010001, B01111110, B00000000, // A
  4, 8, B01111111, B01001001, B01001001, B00110110, B00000000, // B
  4, 8, B00111110, B01000001, B01000001, B00100010, B00000000, // C
  4, 8, B01111111, B01000001, B01000001, B00111110, B00000000, // D
  4, 8, B01111111, B01001001, B01001001, B01000001, B00000000, // E
  4, 8, B01111111, B00001001, B00001001, B00000001, B00000000, // F
  4, 8, B00111110, B01000001, B01001001, B01111010, B00000000, // G
  4, 8, B01111111, B00001000, B00001000, B01111111, B00000000, // H
  3, 8, B01000001, B01111111, B01000001, B00000000, B00000000, // I
  4, 8, B00110000, B01000000, B01000001, B00111111, B00000000, // J
  4, 8, B01111111, B00001000, B00010100, B01100011, B00000000, // K
  4, 8, B01111111, B01000000, B01000000, B01000000, B00000000, // L
  5, 8, B01111111, B00000010, B00001100, B00000010, B01111111, // M
  5, 8, B01111111, B00000100, B00001000, B00010000, B01111111, // N
  4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // O
  4, 8, B01111111, B00001001, B00001001, B00000110, B00000000, // P
  4, 8, B00111110, B01000001, B01000001, B10111110, B00000000, // Q
  4, 8, B01111111, B00001001, B00001001, B01110110, B00000000, // R
  4, 8, B01000110, B01001001, B01001001, B00110010, B00000000, // S
  5, 8, B00000001, B00000001, B01111111, B00000001, B00000001, // T
  4, 8, B00111111, B01000000, B01000000, B00111111, B00000000, // U
  5, 8, B00001111, B00110000, B01000000, B00110000, B00001111, // V
  5, 8, B00111111, B01000000, B00111000, B01000000, B00111111, // W
  5, 8, B01100011, B00010100, B00001000, B00010100, B01100011, // X
  5, 8, B00000111, B00001000, B01110000, B00001000, B00000111, // Y
  4, 8, B01100001, B01010001, B01001001, B01000111, B00000000, // Z
  2, 8, B01111111, B01000001, B00000000, B00000000, B00000000, // [
  4, 8, B00000001, B00000110, B00011000, B01100000, B00000000, // \ backslash
  2, 8, B01000001, B01111111, B00000000, B00000000, B00000000, // ]
  3, 8, B00000010, B00000001, B00000010, B00000000, B00000000, // hat
  4, 8, B01000000, B01000000, B01000000, B01000000, B00000000, // _
  2, 8, B00000001, B00000010, B00000000, B00000000, B00000000, // `
  4, 8, B00100000, B01010100, B01010100, B01111000, B00000000, // a
  4, 8, B01111111, B01000100, B01000100, B00111000, B00000000, // b
  4, 8, B00111000, B01000100, B01000100, B00101000, B00000000, // c
  4, 8, B00111000, B01000100, B01000100, B01111111, B00000000, // d
  4, 8, B00111000, B01010100, B01010100, B00011000, B00000000, // e
  3, 8, B00000100, B01111110, B00000101, B00000000, B00000000, // f
  4, 8, B10011000, B10100100, B10100100, B01111000, B00000000, // g
  4, 8, B01111111, B00000100, B00000100, B01111000, B00000000, // h
  3, 8, B01000100, B01111101, B01000000, B00000000, B00000000, // i
  4, 8, B01000000, B10000000, B10000100, B01111101, B00000000, // j
  4, 8, B01111111, B00010000, B00101000, B01000100, B00000000, // k
  3, 8, B01000001, B01111111, B01000000, B00000000, B00000000, // l
  5, 8, B01111100, B00000100, B01111100, B00000100, B01111000, // m
  4, 8, B01111100, B00000100, B00000100, B01111000, B00000000, // n
  4, 8, B00111000, B01000100, B01000100, B00111000, B00000000, // o
  4, 8, B11111100, B00100100, B00100100, B00011000, B00000000, // p
  4, 8, B00011000, B00100100, B00100100, B11111100, B00000000, // q
  4, 8, B01111100, B00001000, B00000100, B00000100, B00000000, // r
  4, 8, B01001000, B01010100, B01010100, B00100100, B00000000, // s
  3, 8, B00000100, B00111111, B01000100, B00000000, B00000000, // t
  4, 8, B00111100, B01000000, B01000000, B01111100, B00000000, // u
  5, 8, B00011100, B00100000, B01000000, B00100000, B00011100, // v
  5, 8, B00111100, B01000000, B00111100, B01000000, B00111100, // w
  5, 8, B01000100, B00101000, B00010000, B00101000, B01000100, // x
  4, 8, B10011100, B10100000, B10100000, B01111100, B00000000, // y
  3, 8, B01100100, B01010100, B01001100, B00000000, B00000000, // z
  3, 8, B00001000, B00110110, B01000001, B00000000, B00000000, // {
  1, 8, B01111111, B00000000, B00000000, B00000000, B00000000, // |
  3, 8, B01000001, B00110110, B00001000, B00000000, B00000000, // }
  4, 8, B00001000, B00000100, B00001000, B00000100, B00000000, // ~
};

int DIN = 7;   // DIN pin of MAX7219 module
int CLK = 6;   // CLK pin of MAX7219 module
int CS = 5;    // CS pin of MAX7219 module
int maxInUse = 2;

MaxMatrix m(DIN, CS, CLK, maxInUse);

byte buffer[10];

char text[]= "Handson Technology  "; // Scrolling text

void setup() {
  m.init(); // module initialize
  m.setIntensity(15); // dot matix intensity 0-15
}

void loop() {

  printStringWithShift(text, 100); // (text, scrolling speed)
```

```
}
// Display=the extracted characters with scrolling
void printCharWithShift(char c, int shift_speed) {
  if (c < 32) return;
  c -= 32;
  memcpy_P(buffer, CH + 7 * c, 7);
  m.writeSprite(32, 0, buffer);
  m.setColumn(32 + buffer[0], 0);

  for (int i = 0; i < buffer[0] + 1; i++)
  {
    delay(shift_speed);
    m.shiftLeft(false, false);
  }
}
// Extract the characters from the text string
void printStringWithShift(char* s, int shift_speed) {
  while (*s != 0) {
    printCharWithShift(*s, shift_speed);
    s++;
  }
}
```

Once successfully upload, a message will be scrolled across the modules.