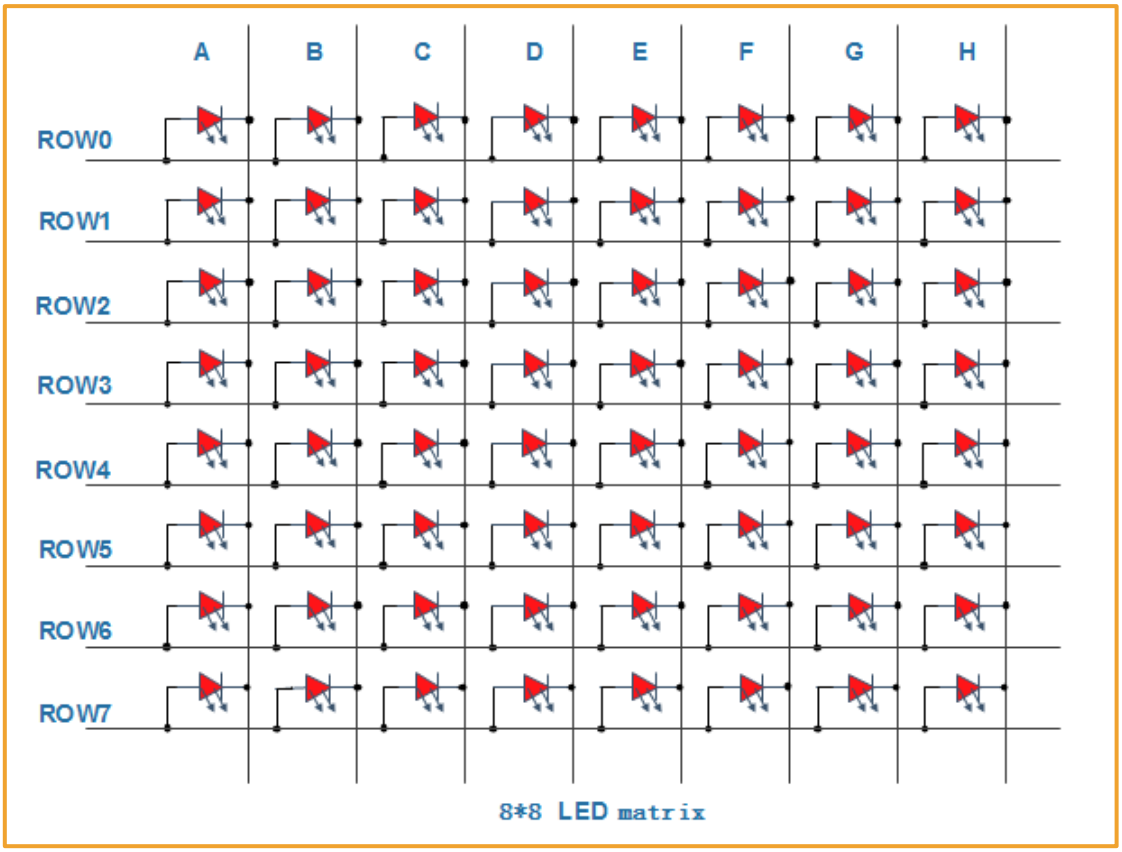
8×8 LED Display Experiment

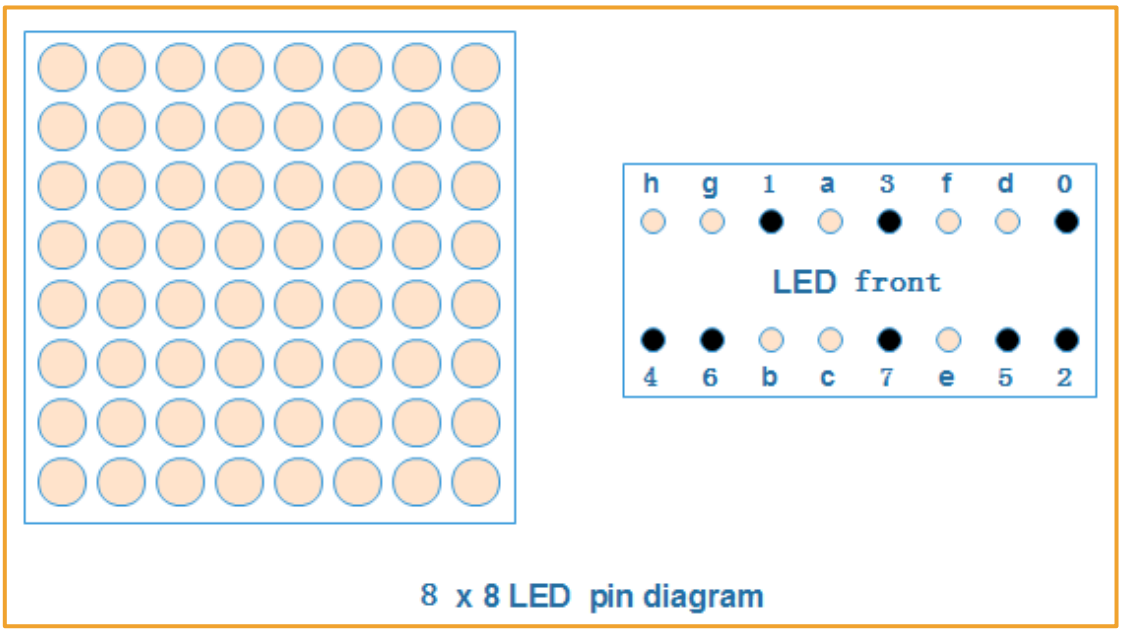
Introduction

Due to we have learned the display of the digital tube, so the digital display is simple. However, if we want to show a variety of patterns in practice, it is obvious that the digital tube has the problem of insufficient capacity and then we need to replace it with the LED matrix. When you walk on the street, you will see a variety of LED neon signs. in fact, they all consist of the **N x N** matrix. Next, we need to learn the internal principle of the **8*8** Matrix.

8*8 Matrix Schematic Diagram

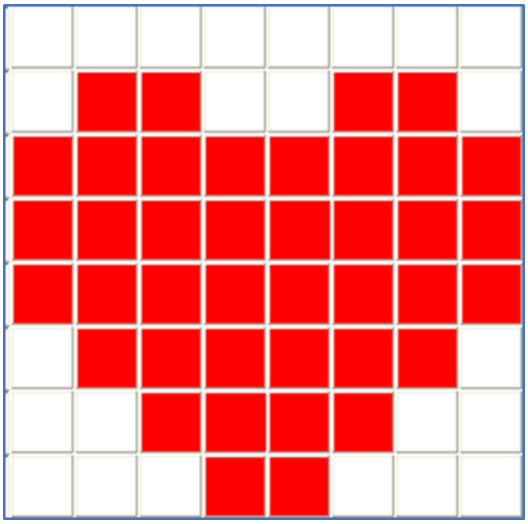


8*8 Matrix Pin



Max7219 chip introduction:

MAX7219/MAX7221 is an integrated serial input/output common-cathode display driver, which connects a microprocessor and an 8-digit 7-segment digital LED display. It can also be connected to a bar graph display or 64 independent LEDs. It includes an on-chip B-type BCD encoder, multiple scan loops, segment word drivers, and an 8*8 static RAM to store each data. Only one external register is used to set the segment current of each LED.



Since this is a dynamic scan, we need to pay attention to the ghosting and flickering of two points. We need to give each column the corresponding level voltage during the scan and pull down the commonly used cathode port. In order to scan the next

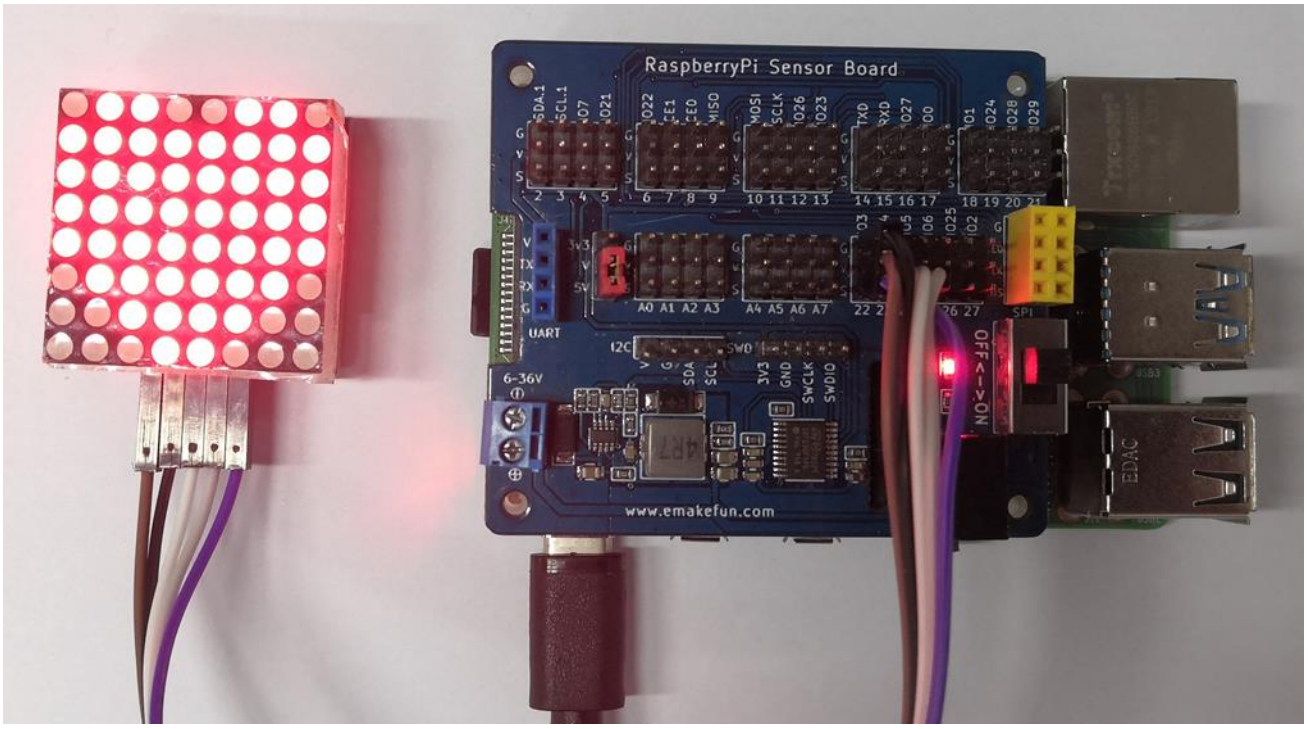
column of the digital tube, we need to pull down the previous column. Due to the column scanning and the residual effect of the human eye is 25hz. However, the frequency sweep of 50Hz is generally better, so the delay time of each column cannot exceed 1000/50/8 = 2.5ms. The effect would be even better if we set the delay to 2ms.

Component List

- Raspberry Pi main board
- Raspberry Pi expansion board
- 8x8 Matrix * 1
- 1k Resistor * 8
- Several jumper wires

Wiring

RaspberryPi	8x8 Dot-matrix
IO4(wiringPi)\23(BCM)	DIN
IO5(wiringPi)\24(BCM)	CS
IO3(wiringPi)\22(BCM)	CLK



C++ program

```
#include <wiringPi.h>
#include <stdio.h>
```

```
#define uchar unsigned char
#define uint unsigned int
#define DecodeMode 0x09
#define Intensity 0x0a
#define ScanLimit 0x0b
#define ShutDown 0x0c
#define DisplayTest 0x0f
#define ShutdownMode 0x00
#define NormalOperation 0x01
#define ScanDigit 0x07
#define DecodeDigit 0x00
#define IntensityGrade 0x0a
#define TestMode 0x01
#define TextEnd 0x00
#define DIN 4
#define CS 5
#define CLK 3

uchar buffer[8]=
{
    0x78,0xFC,0xFE,0x7F,0x7F,0xFE,0xFC,0x78,
};

void senduchar(uchar ch){
    uchar i, tmp;
    for(i = 0; i < 8; i++){
        tmp = ch & 0x80;
        if(tmp)
            digitalWrite(DIN, HIGH);
        else
            digitalWrite(DIN, LOW);
        ch = ch << 1;
        digitalWrite(CLK, HIGH);
        digitalWrite(CLK, LOW);
    }
}

void writeWord(uchar addr, uchar num){
    digitalWrite(CS, HIGH);
    digitalWrite(CS, LOW);
    digitalWrite(CLK, LOW);
```

```
    senduchar(addr);
    senduchar(num);
    digitalWrite(CS, HIGH);
}

void write(uchar (&buff)[8]){
    uchar i;
    for(i = 0; i < 8; i++){
        printf("%d %d \n", i, buff[i]);
        writeWord(i + 1, buff[i]);
        delay(20);
    }
}

void init(){
    writeWord(DecodeMode, 0x00);
    writeWord(Intensity, 0x08);
    writeWord(ScanLimit, 0x07);
    writeWord(ShutDown, 0x01);
    writeWord(DisplayTest, 0x00);
}

int main(){
    wiringPiSetup();
    pinMode(DIN, OUTPUT);
    pinMode(CS, OUTPUT);
    pinMode(CLK, OUTPUT);
    init();
    while(1){
        write(buffer);
    }
return 0;
}
```

Python program

```
# -*- coding:utf-8 -*-

import RPi.GPIO as GPIO
import time

DecodeMode = 0x09
Intensity = 0x0a
```

```
ScanLimit = 0x0b
ShutDown = 0x0c
DisplayTest = 0x0f
ShutdownMode = 0x00
NormalOperation = 0x01
ScanDigit = 0x07
DecodeDigit = 0x00
IntensityGrade = 0x0a
TestMode = 0x01
TextEnd = 0x00
buffer =[ 0x78, 0xFC, 0xFE, 0x7F, 0x7F, 0xFE, 0xFC, 0x78 ]

class Max7219:

    def __init__(self, CLK, DIN, CS):
        self.DIN = DIN
        self.CS = CS
        self.CLK = CLK
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(self.DIN, GPIO.OUT)
        GPIO.setup(self.CS, GPIO.OUT)
        GPIO.setup(self.CLK, GPIO.OUT)


    def start(self):
        self.writeWord(DecodeMode, 0x00)
        self.writeWord(Intensity, 0x08)
        self.writeWord(ScanLimit, 0x07)
        self.writeWord(ShutDown, 0x01)
        self.writeWord(DisplayTest, 0x00)


    def senduchar(self, ch):
        for i in range(0, 8):
            tmp = ch & 0x80
            if tmp:
                GPIO.output(self.DIN, GPIO.HIGH)
            else:
                GPIO.output(self.DIN, GPIO.LOW)
            ch = ch << 1
            GPIO.output(self.CLK, GPIO.HIGH)
            GPIO.output(self.CLK, GPIO.LOW)
```

```
    def writeWord(self, addr, num):
        GPIO.output(self.CS, GPIO.HIGH)
        GPIO.output(self.CS, GPIO.LOW)
        GPIO.output(self.CLK, GPIO.LOW)
        self.senduchar(addr)
        self.senduchar(num)
        GPIO.output(self.CS, GPIO.HIGH)

    def write(self, buff):
        for i in range(0, 8):
            self.writeWord(i + 1, buff[i])
            time.sleep(0.02)


max7219 = Max7219(22, 23, 24)
max7219.start()
while True:
    max7219.write(buffer)
```

Java program

```
import com.pi4j.wiringpi.Gpio;
import com.pi4j.wiringpi.GpioInterrupt;
import com.pi4j.wiringpi.GpioInterruptListener;
import com.pi4j.wiringpi.GpioInterruptEvent;
import com.pi4j.wiringpi.GpioUtil;


public class Max7219 {
    static int DecodeMode = 0x09;
    static int Intensity = 0x0a;
    static int ScanLimit = 0x0b;
    static int ShutDown = 0x0c;
    static int DisplayTest = 0x0f;
    static int ShutdownMode = 0x00;
    static int NormalOperation = 0x01;
    static int ScanDigit = 0x07;
    static int DecodeDigit = 0x00;
    static int IntensityGrade = 0x0a;
    static int TestMode = 0x01;
    static int TextEnd = 0x00;
```

```
static int DIN = 4, CS = 5, CLK = 3;
static int buffer[] =
                    {
                        0x78,0xFC,0xFE,0x7F,0x7F,0xFE,0xFC,0x78,
                    };

public static void init(){
    Gpio.pinMode(DIN, Gpio.OUTPUT);
    Gpio.pinMode(CS, Gpio.OUTPUT);
    Gpio.pinMode(CLK, Gpio.OUTPUT);
    writeWord(DecodeMode, 0x00);
    writeWord(Intensity, 0x08);
    writeWord(ScanLimit, 0x07);
    writeWord(ShutDown, 0x01);
    writeWord(DisplayTest, 0x00);
}

public static void send_data(int ch){
    int i, tmp;
    for(i = 0; i < 8; i++){
        tmp = ch & 0x80;
        if(tmp == 0x80)
            Gpio.digitalWrite(DIN, Gpio.HIGH);
        else
            Gpio.digitalWrite(DIN, Gpio.LOW);
        ch = ch << 1;
        Gpio.digitalWrite(CLK, Gpio.HIGH);
        Gpio.digitalWrite(CLK, Gpio.LOW);
    }
}

public static void writeWord(int addr, int num){
    Gpio.digitalWrite(CS, Gpio.HIGH);
    Gpio.digitalWrite(CS, Gpio.LOW);
    Gpio.digitalWrite(CLK, Gpio.LOW);
    send_data(addr);
    send_data(num);
    Gpio.digitalWrite(CS, Gpio.HIGH);
}

public static void write(int [] buff){
    int i;
```

```
        for(i = 0; i < 8; i++){
            writeWord(i + 1, buff[i]);
            System.out.println(buff[i]);
            Gpio.delay(20);
        }
    }

    public static void main(String[] args) throws InterruptedException {
        // setup wiring pi
        if (Gpio.wiringPiSetup() == -1) {
            System.out.println(" ==>> GPIO SETUP FAILED");
            return;
        }
        init();
        for (;;) {
            write(buffer);
        }
    }
}
```

Experimental Effect