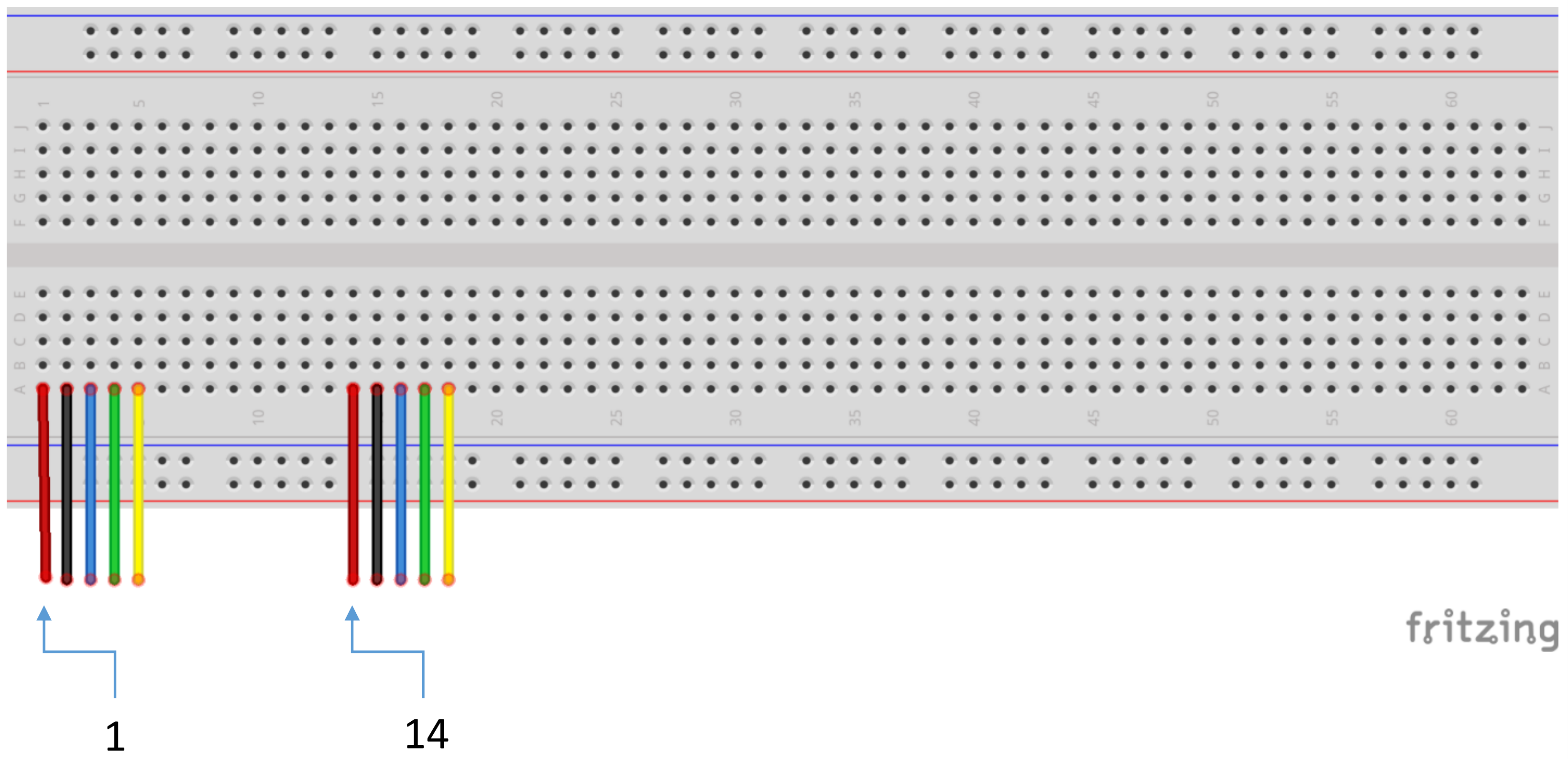
LED 矩陣跑馬燈

材料

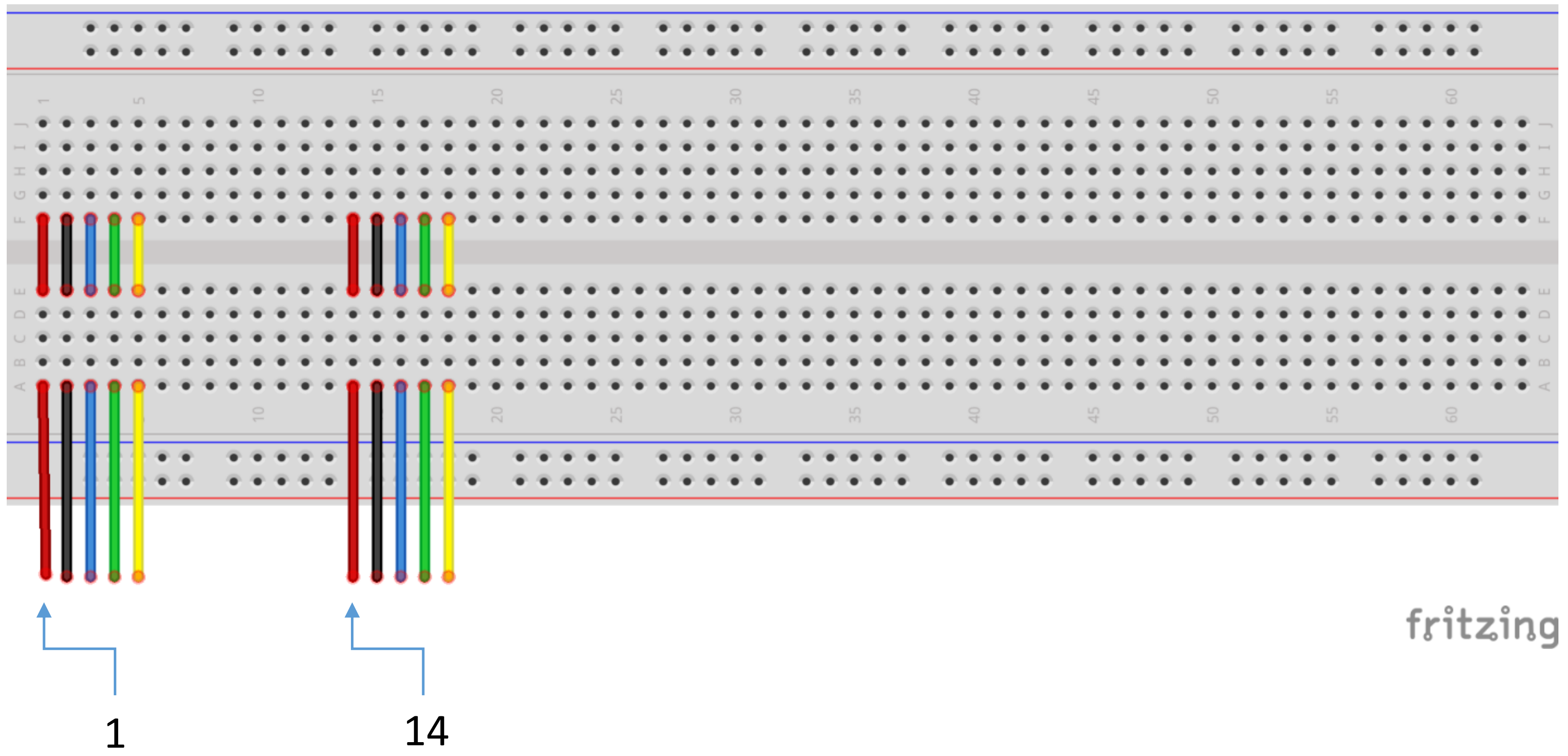
- Arduino UNO x 1
- MAX7218 LED 點矩陣模組 x 2
- 麵包板 x 1
- 杜邦線 公對公 x 5 x 2
- 杜邦線 公對公 x 1 x 11
- 杜邦線 母對公 x 1

MAX7219 LED 點矩陣模組

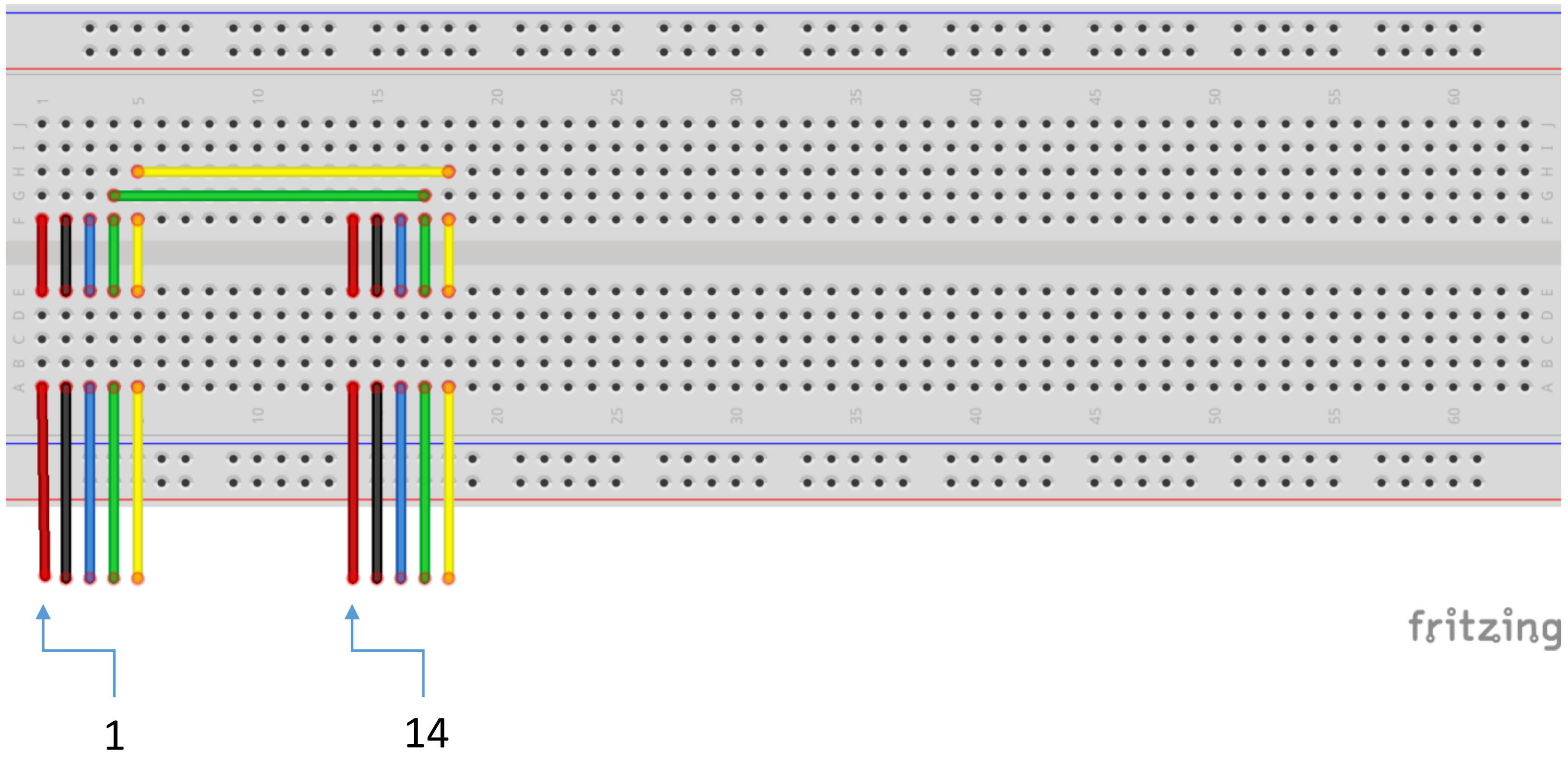
接線



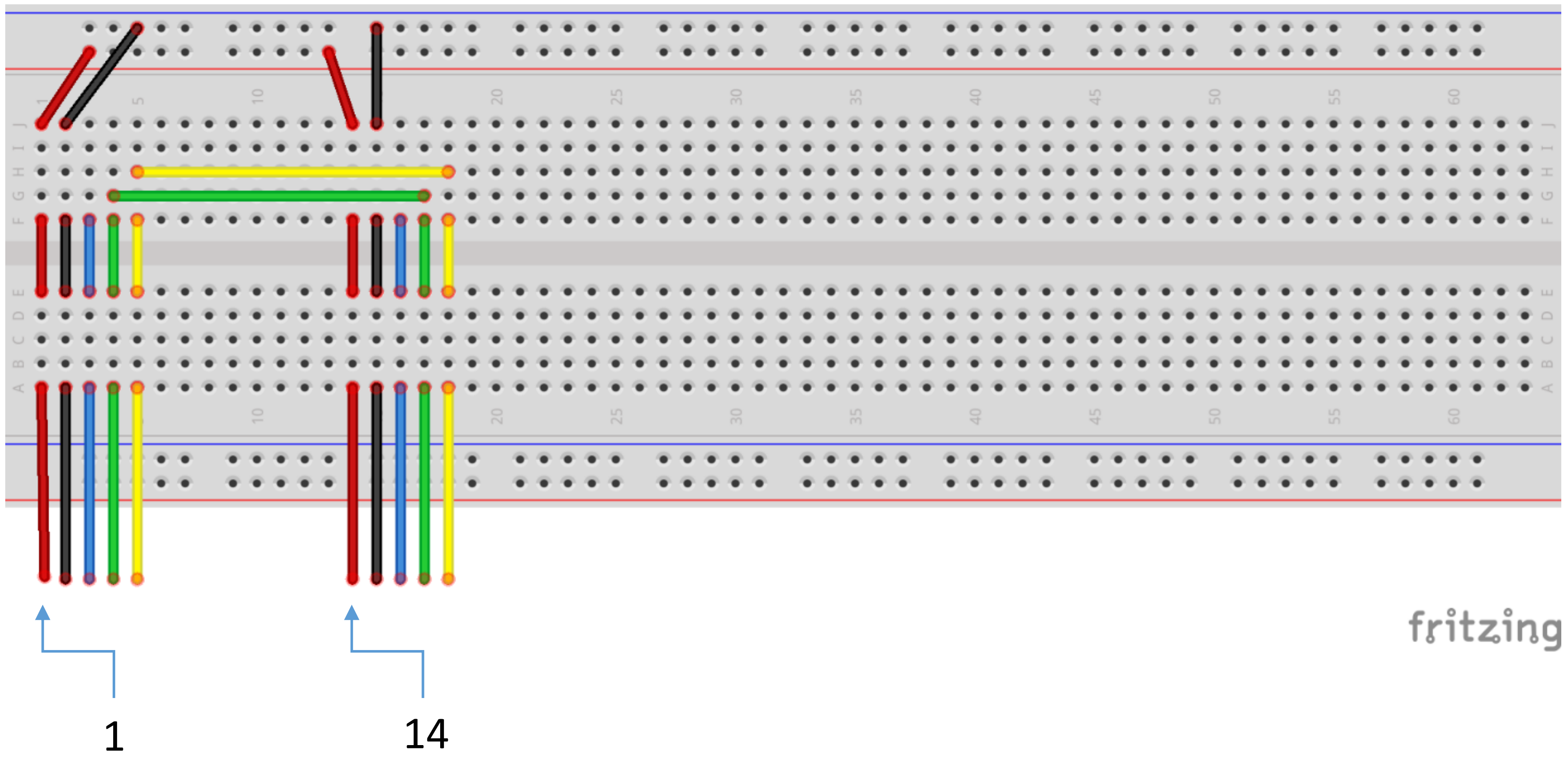
接線



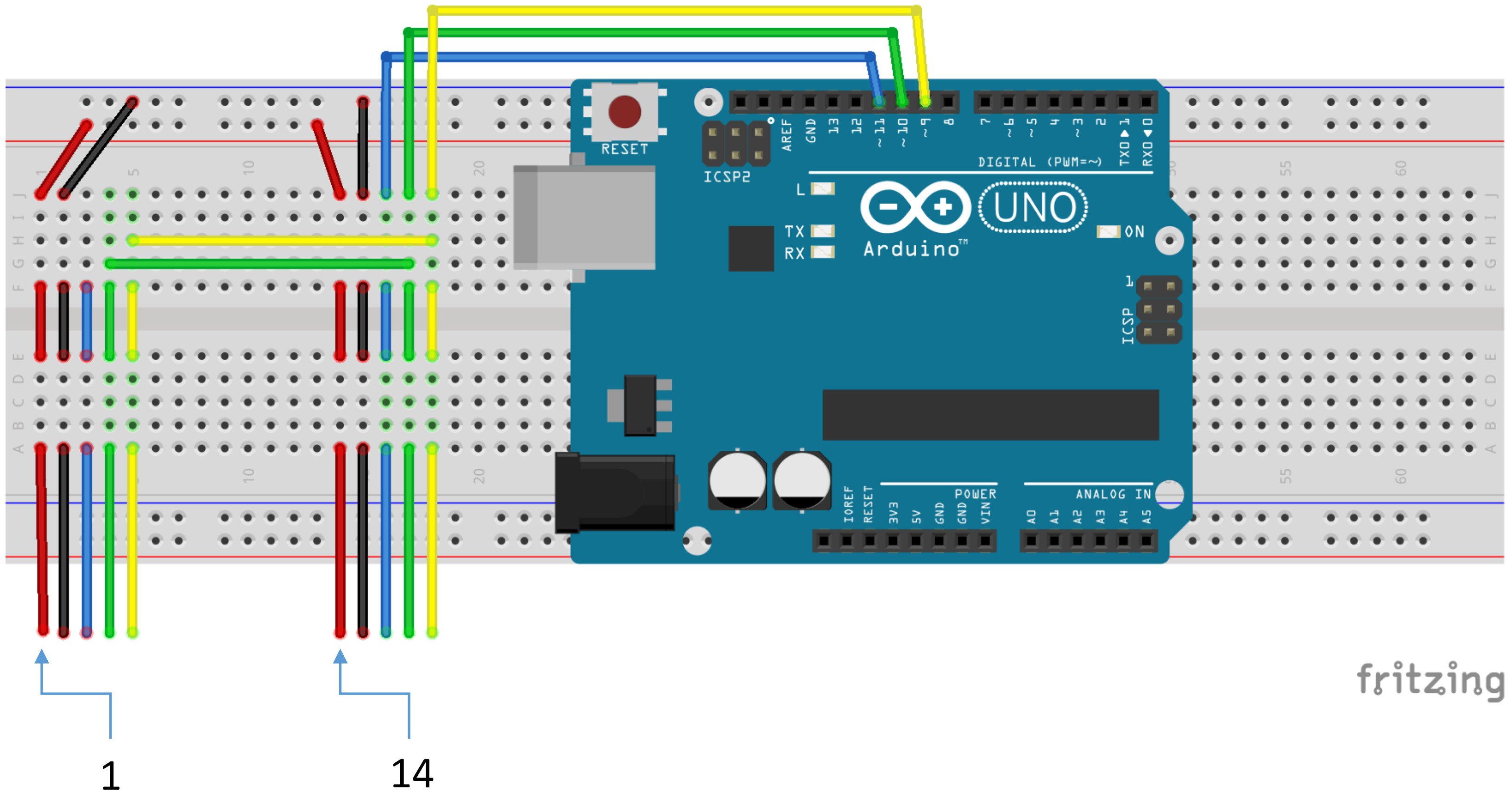
接線



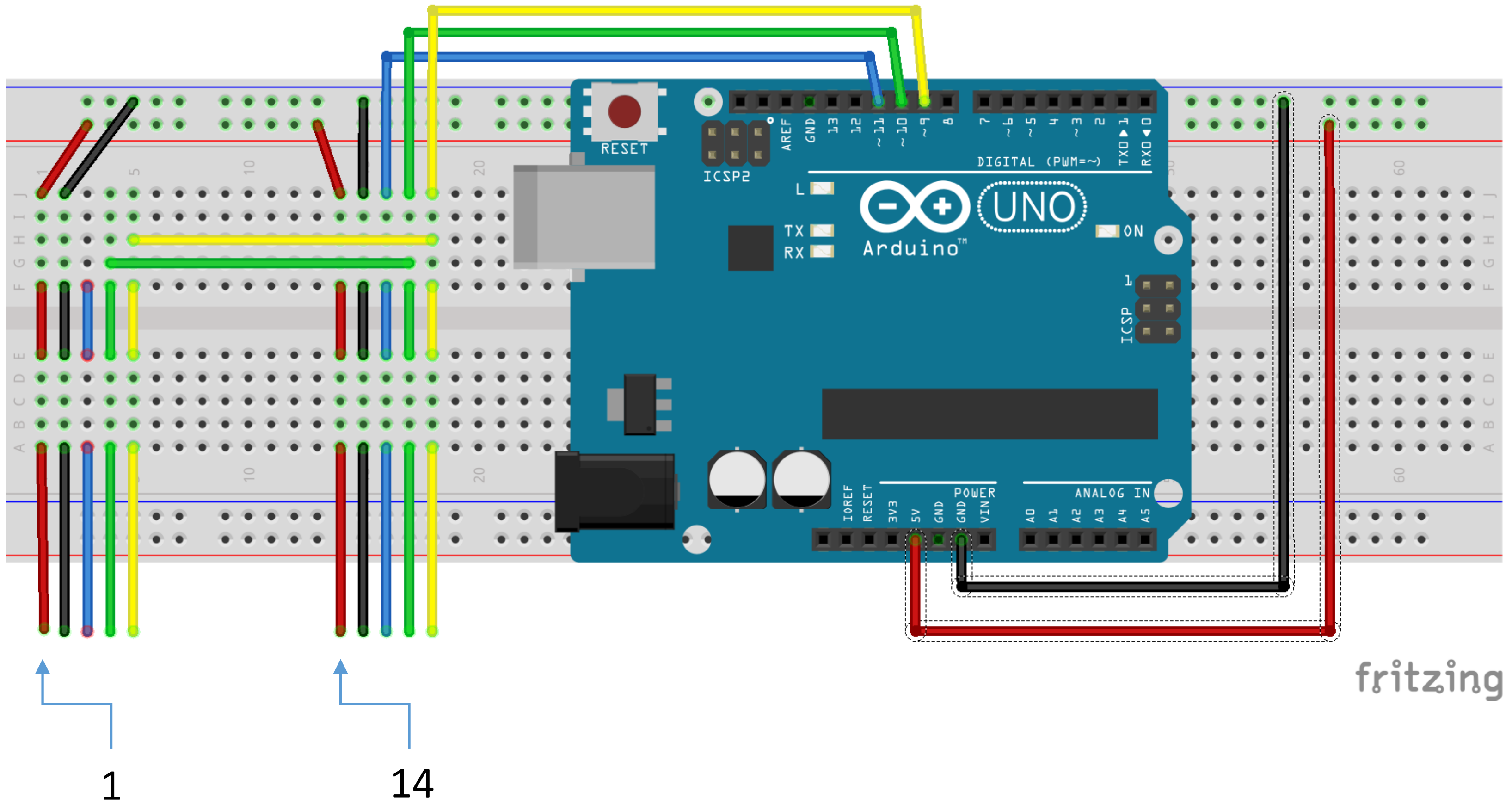
接線



接線


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX▶1
RX◀0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino™
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
fritzing
1
14

接線


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX0▶1
RX0◀0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino™
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
fritzing
1
14

測試程式

- https://github.com/NTUST-Coding-Club/arduino-workshop-2016/tree/master/LEDMatrixForN

MaxMatrix 函式庫

- 下載&安裝函式庫
  - https://code.google.com/archive/p/arudino-maxmatrix-library/

MaxMatrix 函式庫

```
#include "MaxMatrix.h"

#define DIN 11
#define CS   10
#define CLK 9
#define DEVICE_NUMBER 2

MaxMatrix ledController(DIN, CS, CLK, DEVICE_NUMBER);
```

MaxMatrix 函式庫

```
void setup() {
  // 初始設定
  ledController.init();
  // 設定亮度
  ledController.setIntensity(0);
}
```

測試

```
void setup() {
  ….
  ledController.setIntensity(0);
  // test
  for (byte i = 0; i < 8; i++) {
    ledController.setDot(i, i, true);
  }
}
```

MaxMatrix 函式庫

```
void setup() {
  ....
  for (byte i = 0; i < 8; i++) {
    ledController.setDot(i, i, true);
  }
  delay(2000);
  // 清除畫面
  ledController.clear();
}
```

使用 Serial 測試 MaxMatrix 函式庫

```
void setup() {
  ....
  ledController.clear();
  Serial.begin(9600);
  Serial.println("Setting Complete!");
}
```

使用 Serial 測試 MaxMatrix 函式庫

```
void loop() {
  if (Serial.available() > 0) {
    byte x = Serial.parseInt();
    byte y = Serial.parseInt();
    ledController.setDot(x, y, true);
  }
}
```

MaxMatrix 函式庫

```
void setup() {
    ....
    Serial.println("Setting Complete!");
    ledController.setColumn(0, 1);
    ledController.setColumn(1, 2);
    ledController.setColumn(2, 3);
    ledController.setColumn(3, 4);
    ....
}
```

使用 Serial 測試 MaxMatrix 函式庫

```
void loop() {
    if (Serial.available() > 0) {
        byte column = Serial.parseInt();
        byte value = Serial.parseInt();
        ledController.setColumn(column, value);
    }
}
```

MaxMatrix 函式庫

```
void loop() {
    ledController.shiftUp(false);
    delay(200);
}
```

MaxMatrix 函式庫

```
void loop() {
    ledController.shiftDown(true);
    delay(200);
}
```

LED 矩陣跑馬燈

```
void loop() {
  ledController.shiftLeft(false, true);
  delay(200);
}
```

LED 矩陣跑馬燈

```
void loop() {
    if (Serial.available() > 0) {
        ledController.setColumn(
            DEVICE_NUMBER * 8 - 1, Serial.parseInt());
    }
    ledController.shiftLeft(true, true); delay(200);
}
```

Arduino 跟 MAX7219 的溝通方式

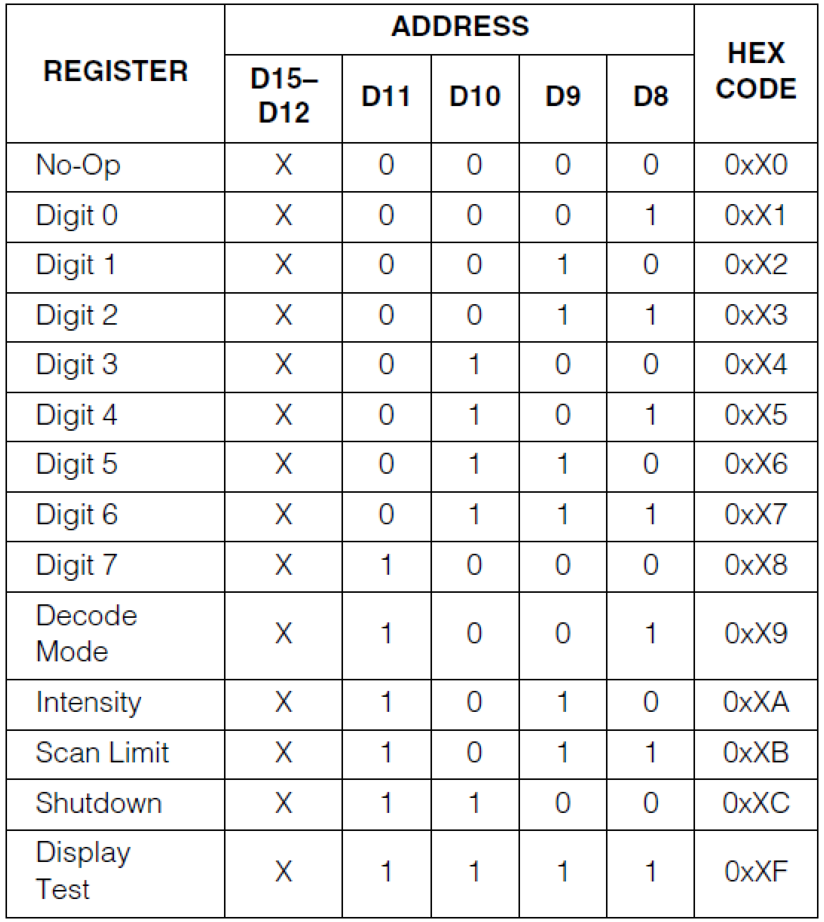
- 以 SPI 協定的方式通訊
  - 資料從最高有效位元(MSB)開始接收 => MSB first
- 每次接收兩個位元組(2 bytes) 的資料
  - 第一個 byte 用來選擇暫存器
  - 第二個 byte 用來傳送要儲存在暫存器的內容

MSB

LSB

0	1	1	0	1	0	1	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

MAX7219 的 暫存器 種類



REGISTER	ADDRESS					HEX CODE
	D15–D12	D11	D10	D9	D8	
No-Op	X	0	0	0	0	0xX0
Digit 0	X	0	0	0	1	0xX1
Digit 1	X	0	0	1	0	0xX2
Digit 2	X	0	0	1	1	0xX3
Digit 3	X	0	1	0	0	0xX4
Digit 4	X	0	1	0	1	0xX5
Digit 5	X	0	1	1	0	0xX6
Digit 6	X	0	1	1	1	0xX7
Digit 7	X	1	0	0	0	0xX8
Decode Mode	X	1	0	0	1	0xX9
Intensity	X	1	0	1	0	0xXA
Scan Limit	X	1	0	1	1	0xXB
Shutdown	X	1	1	0	0	0xXC
Display Test	X	1	1	1	1	0xXF

MAX7219 的 暫存器 種類 ___

```
#define reg_no_op     0x00
#define reg_digit_0   0x01
#define reg_digit_1   0x02
#define reg_digit_2   0x03
#define reg_digit_3   0x04
#define reg_digit_4   0x05
#define reg_digit_5   0x06
#define reg_digit_6   0x07
#define reg_digit_7   0x08
```

MAX7219 的 暫存器 種類

```
#define reg_decode_mode      0x09
#define reg_brightness       0x0A
#define reg_scan_limit       0x0B
#define reg_shutdown         0x0C
#define reg_test             0x0F
```

設定 SPI 腳位

```
#define DIN 11
#define CS   10
#define CLK 9
```

```
void setup() {
   // 設定 SPI 腳位
   pinMode(DIN, OUTPUT);
   pinMode(CS,  OUTPUT);
   pinMode(CLK, OUTPUT);
   digitalWrite(CS, HIGH);
}
```

傳送資料給 MAX7219

```
void spiTransfer(byte dataPin, byte clockPin, byte value) {
  for (byte i = 0; i < 8; i++) {
    digitalWrite(dataPin, value & (1 << i));
    digitalWrite(clockPin, HIGH);
    digitalWrite(clockPin, LOW);
  }
}
```

傳送資料給 MAX7219

```
void spiTransfer(byte dataPin, byte clockPin, byte value) {
  for (byte i = 0; i < 8; i++) {
    digitalWrite(dataPin, value & (1 << (7 - i)));
    digitalWrite(clockPin, HIGH);
    digitalWrite(clockPin, LOW);
  }
}
```

反轉位元組 程式碼

```
byte reverseByte(byte b) {
    b = b >> 4 | b << 4;
    b = (b & B11001100) >> 2 | (b & B00110011) << 2;
    b = (b & B10101010) >> 1 | (b & B01010101) << 1;
    return b;
}
```

傳送資料給 MAX7219

```
void spiTransfer(byte dataPin, byte clockPin, byte value) {
    byte reversedValue = reverseByte(value);
    for (byte i = 0; i < 8; i++) {
        digitalWrite(dataPin, reversedValue & (1 << i));
        digitalWrite(clockPin, HIGH);
        digitalWrite(clockPin, LOW);
    }
}
```

傳送指令給 MAX7219

```
void setCommand(byte command, byte value) {
    digitalWrite(CS, LOW);
    for (byte i = 0; i < DEVICE_NUMBER; i++) {
        spiTransfer(DIN, CLK, command);
        spiTransfer(DIN, CLK, value);
    }
    digitalWrite(CS, HIGH);
}
```

傳送指令給 MAX7219

```
for (byte i = 0; i < DEVICE_NUMBER; i++) {
    if (i == device) {  // void setCommand(byte device, byte command, byte value)
        spiTransfer(DIN, CLK, command);
        spiTransfer(DIN, CLK, value);
    } else {
        spiTransfer(DIN, CLK, reg_noop);
        spiTransfer(DIN, CLK, 0);
    }
}
```

LED 模組初始化

- setCommand(reg_decode_mode, 0);
- setCommand(reg_brightness, 0);
- setCommand(reg_scan_limit, 7);
- setCommand(reg_shutdown, 1);
- setCommand(reg_test, 0);

測試 001

```
void loop() {
    setCommand(reg_test, 1); delay(1);
    setCommand(reg_test, 0); delay(500);
}
```

合併 setCommand

```
if (i == device || device == 255) {
    spiTransfer(DIN, CLK, command);
    spiTransfer(DIN, CLK, value);
} else {
    spiTransfer(DIN, CLK, reg_noop);
    spiTransfer(DIN, CLK, 0);
}
```

合併 setCommand

```
void setCommand(byte command, byte value) {
  setCommand(255, command, value);
}
```

清除 MAX7219 的所有輸出

```
void clearAllLEDs() {
  for (byte i = 0; i <  8; i++) {
    setCommand(i + 1, 0);
  }
}
```

與 Serial 通訊

```
Serial.begin(9600);
Serial.println("Setting Complete!");
```

與 Serial 通訊 & 測試

```
void loop() {
    if (Serial.available() > 0) {
        Serial.println(Serial.parseInt());
    }
}
```

畫圖

- byte pic00[] = {B00001111, B00001111, B00110011, B00101011,
  B11010100, B11001100, B11110000, B11110000};
- byte pic01[] = {B11110000, B11110000, B11001100, B11010100,
  B00101011, B00110011, B00001111, B00001111};
- byte charA[] = {B00000000, B00000000, B01111110, B00001001,
  B00001001, B01111110, B00000000, B00000000};
- byte charB[] = {B00000000, B00000000, B01111111, B01001001,
  B01001001, B00111110, B00000000, B00000000};

畫圖

```
void setup() {
  ……
  Serial.println("Setting Complete!");
  for (byte i = 0; i < 8; i++) {
    setCommand(0, i + 1, pic00[i]);
    setCommand(1, i + 1, pic01[i]);
  }
}
```

或是

```
void loop() {
  for (byte i = 0; i < 8; i++) {
    setCommand(0, i + 1, pic00[i]);
    setCommand(1, i + 1, pic01[i]);
  } delay(500);
  for (byte i = 0; i < 8; i++) {
    setCommand(1, i + 1, pic00[i]);
    setCommand(0, i + 1, pic01[i]);
  } delay(500);
}
```

Draw Column

```
void drawColumn(byte column, byte value) {
    byte n = column / 8;
    byte c = column % 8;
    setCommand(n, c + 1, value);
}
```

測試

```
void setup() {
  ……
  Serial.println("Setting Complete!");
   for (byte i = 0; i < 4; i++) {
     drawColumn(i, charA[2 + i]);
     drawColumn(i + 8, charB[2 + i]);
  }
}
```

或是

```
void loop() {
  static byte delta = 4;
  delta = (delta == 0) ? 4 : (delta - 1);
  clearAllLEDs();
  for (byte i = 0; i < 4; i++) {
    drawColumn(i + delta, charA[2 + i]);
    drawColumn(i + delta + 8, charB[2 + i]);
  } delay(500);
}
```

從 Serial 設定 & 測試

```
void loop() {
  if (Serial.available() > 0) {
    byte column = Serial.parseInt();
    byte value = Serial.parseInt();
    drawColumn(column, value);
  }
}
```

Arduino 內建傳送 SPI 的程式碼

```
void shiftOut(byte dataPin, byte clockPin, byte bitOrder, byte value) {
  for (i = 0; i < 8; i++)  {
    if (bitOrder == LSBFIRST) {
      digitalWrite(dataPin, !!(val & (1 << i)));
    } else {
      digitalWrite(dataPin, !!(val & (1 << (7 - i))));
    }
    digitalWrite(clockPin, HIGH);
    digitalWrite(clockPin, LOW);
  }
}
```

修改 setCommand & 測試

```
 if (i == device || device == 255) {
    shiftOut(DIN, CLK, MSBFIRST, command);
    shiftOut(DIN, CLK, MSBFIRST, value);
} else {
    shiftOut(DIN, CLK, MSBFIRST, reg_noop);
    shiftOut(DIN, CLK, MSBFIRST, 0);
}
```

Arduino 內建硬體 SPI 傳送功能

```
#include <SPI.h>
/* Arduino 內建 SPI 腳位
 *  DIN MOSI(11)
 *  CS SS(10)
 *  CLK SCK(13)
 */
```

腳位設定

```
void setup() {
  // 設定 SPI
  SPI.begin();
  // LED 模組初始化
  ....
}
```

修改 setCommand & 測試

```
void setCommand(byte device, byte command, byte value) {
    digitalWrite(SS, LOW);
    for (byte i = 0; i < DEVICE_NUMBER; i++) {
        if (i == device || device == 255) {
            SPI.transfer(command);
           SPI.transfer(value);
        } else {
            SPI.transfer(reg_noop);
            SPI.transfer(0);
        }
    }
    digitalWrite(SS, HIGH);
}
```

Hello, world! font

```
byte columns[16] = {
    0x7F, 0x08, 0x08, 0x08, 0x7F,   // H
    0x00, 0x44, 0x7D, 0x40, 0x00, // i
    0x00, 0x00, 0x5F, 0x00, 0x00   // !
};
```

顯示 Hi!

```
void setup() {
    ....
    Serial.println("Setting Complete!");
    // Hi
    for(byte i = 0; i < 16; i++) {
        drawColumn(i, columns[i]);
    }
}
```

簡化

```
#include "font5x7.h"
char helloString[] = "Hello, world!";
int length = strlen(helloString);
for (int i = 0; i < length; i++) {
    for (int j = 0; j < 5; j++) {
        drawColumn(i * 5 + j, font5x7[(helloString[i] - 32) * 5 + j]);
    }
}
```

測試

```
void loop() {
  static char delta = DEVICE_NUMBER * 8;
  for (int I = 0; I < length; i++) {
    for (int j = 0; j < 5; j++) {
      drawColumn(I * 5 + j + delta, font5x7[(helloString[i] – 32) * 5 + j]);
    }
  } delay(100);
  delta = (delta < -(length * 5)) ? DEVICE_NUMBER * 8 : (delta - 1);
  clearAllLEDs();
}
```

文字/圖形顛倒怎麼辦

- ~~干我P4~~
- **reverseByte**

拿掉 delay

```
bool isTimer1Up(unsigned long period) {
    static unsigned long timer = millis();
    bool timeIsUp = (millis() - timer) > period;
    if (timeIsUp) timer = millis();
    return timeIsUp;
}
```

測試

```
void loop() {
    if (isTimer1Up(100)) updateLEDs();
    static unsigned long t = 0;
    Serial.println(t++);
}
```

LedControl 函式庫

- 來不及做……