Hướng dẫn viết code cho LED ma trận 8x8 bằng thanh ghi dịch 74HC595

Sử dụng PIC 16F887

Nhóm 18 – Giảng đường 205

Nguyễn Tùng Sơn

Nghiêm Minh Sơn

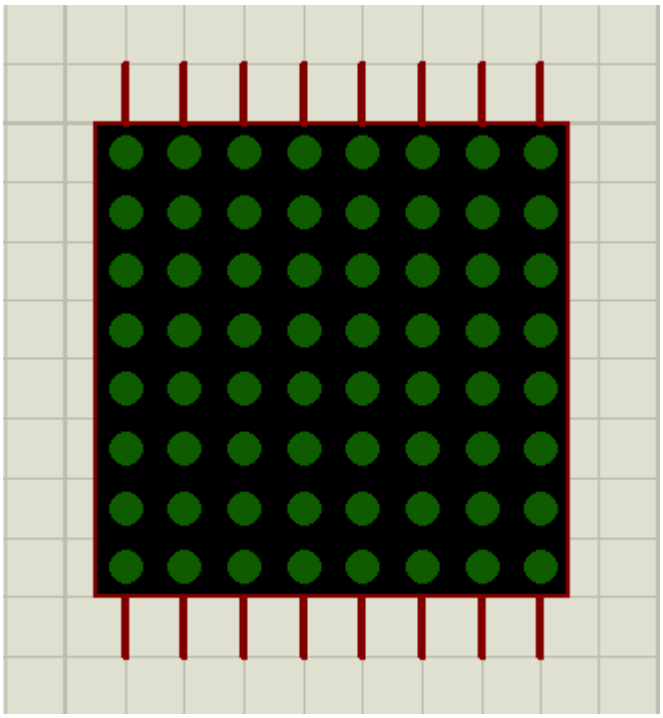
Trần Văn Tam

Hoàng Sỹ Tân

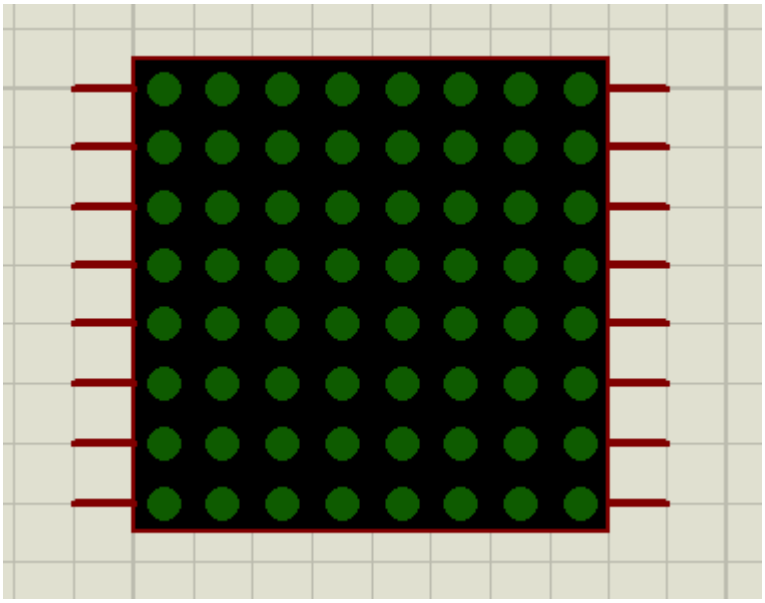
Phạm Anh Tài

1> LED ma trận 8x8:

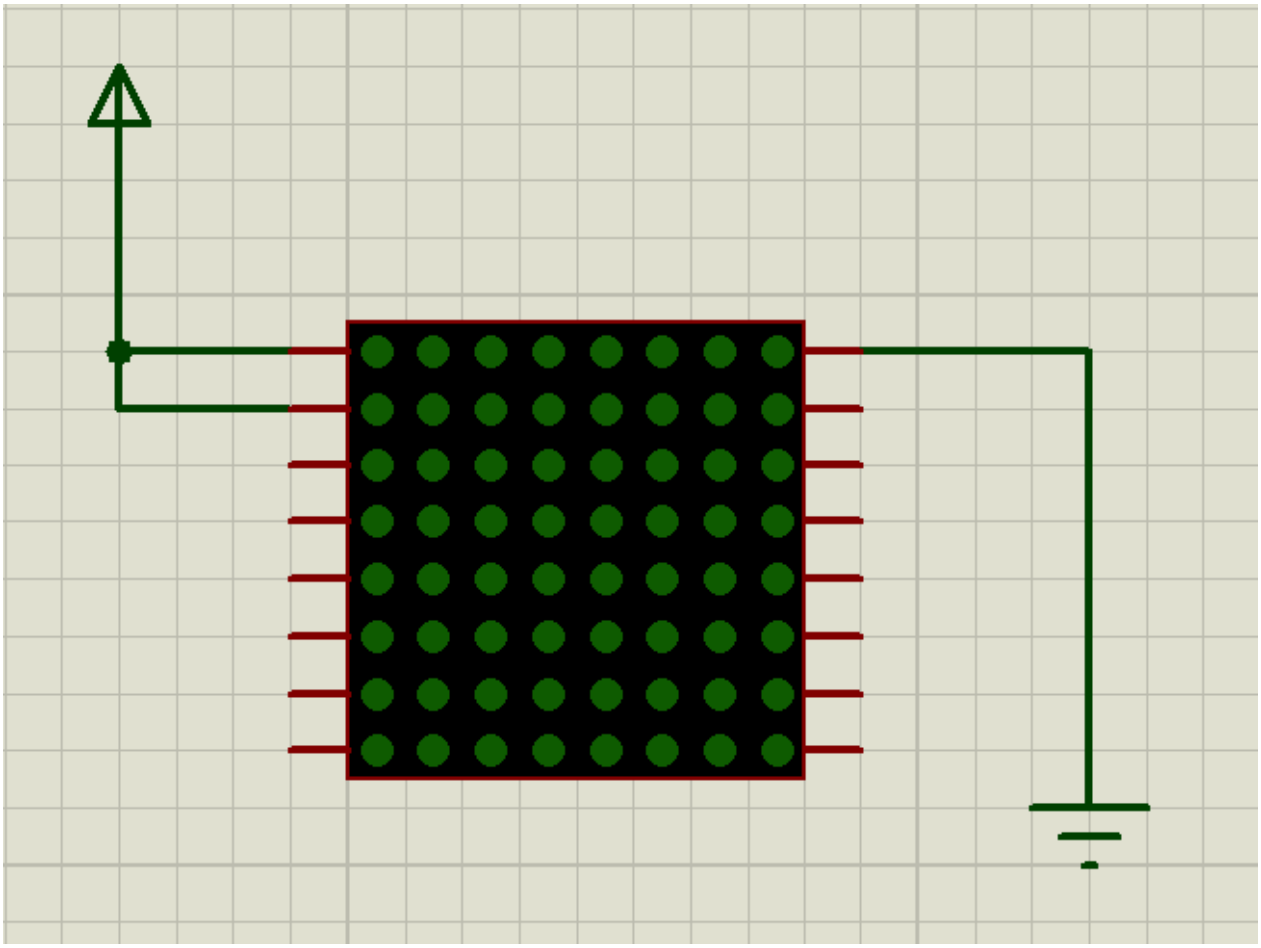
ở đây mình sử dụng led ma trận 8x8 màu xanh trong proteus:



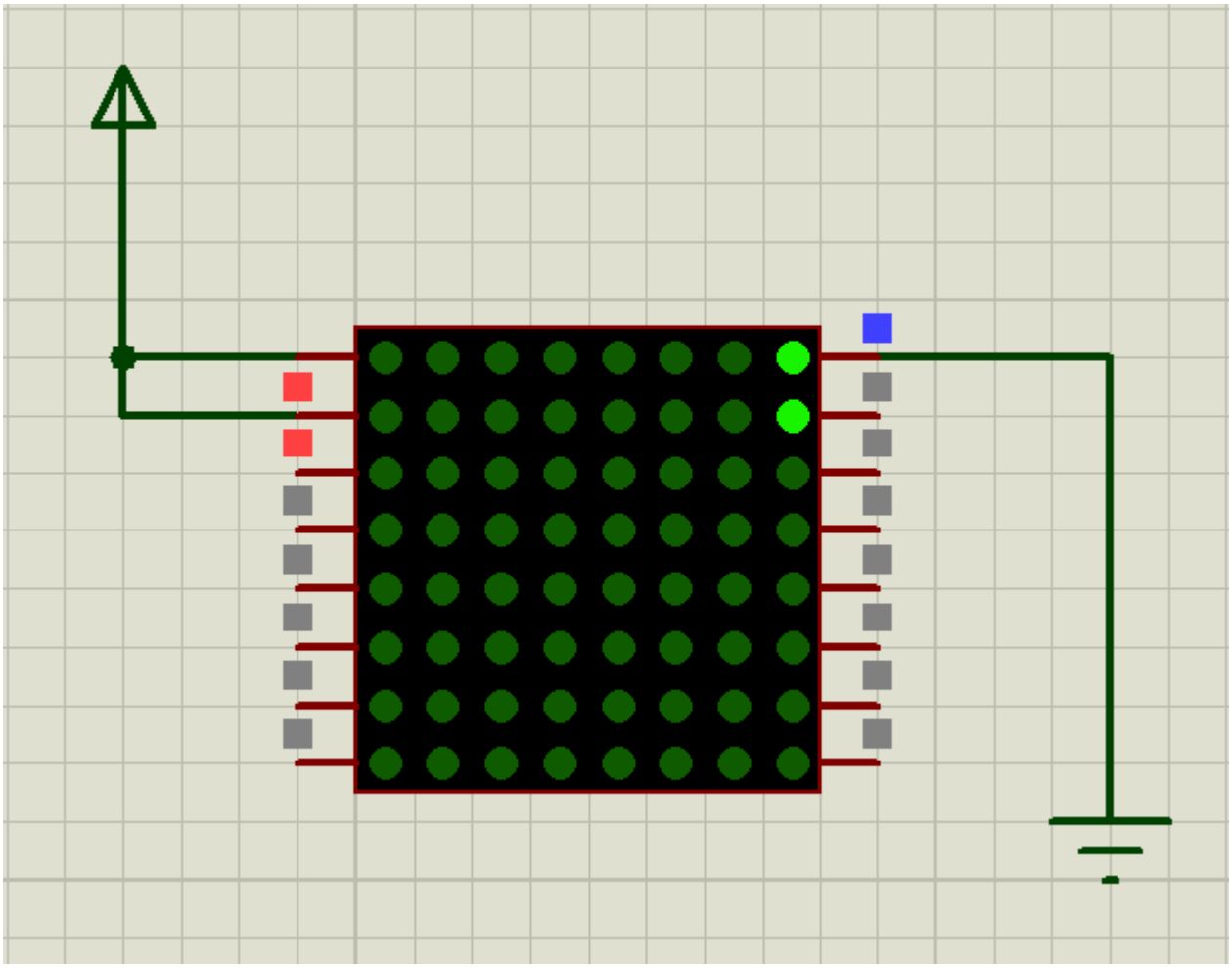
Xoay linh kiện theo chiều  để nó xoay ngang:



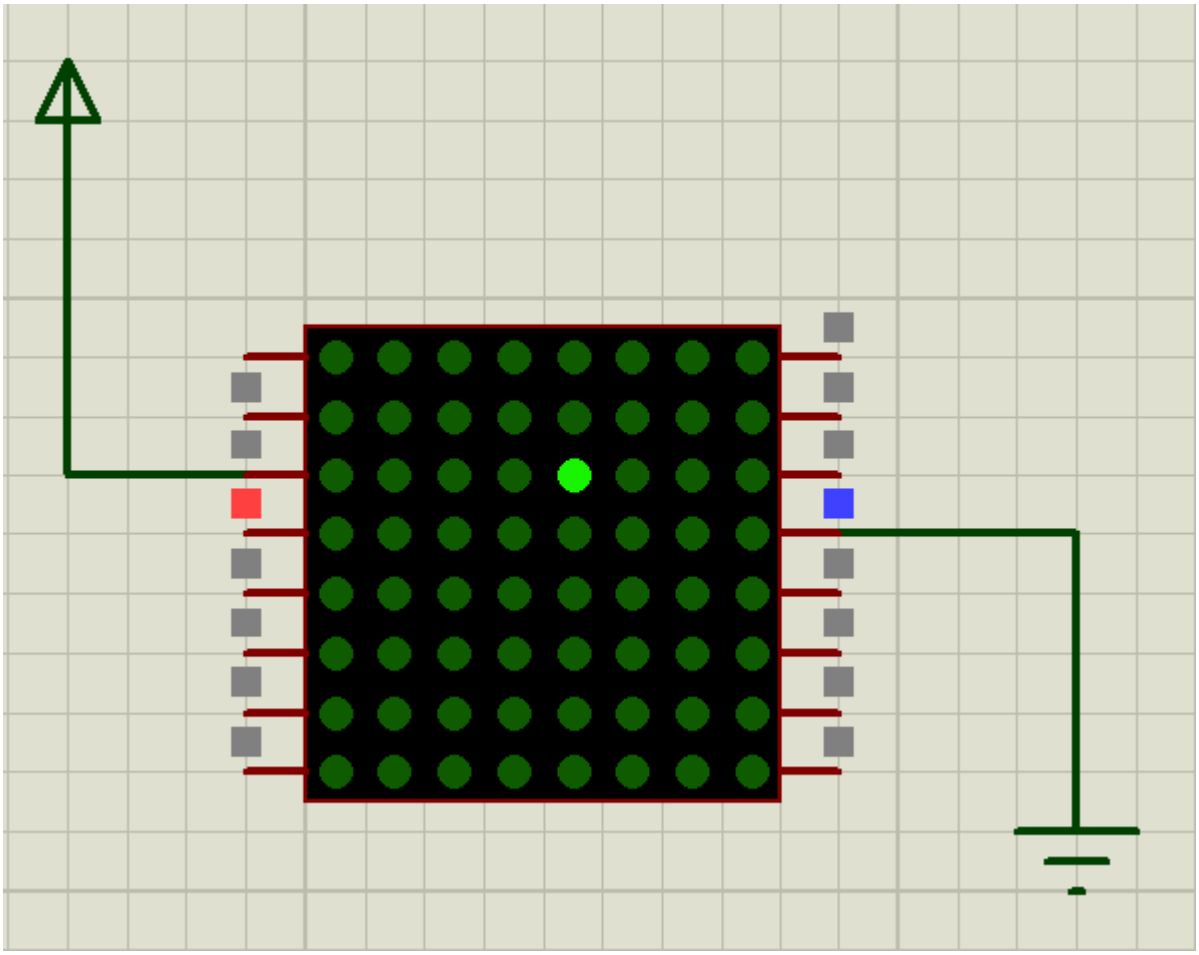
Sau đó, để test chân linh kiện, giả sử ta cho cực dương và đất vào các chân như sau:



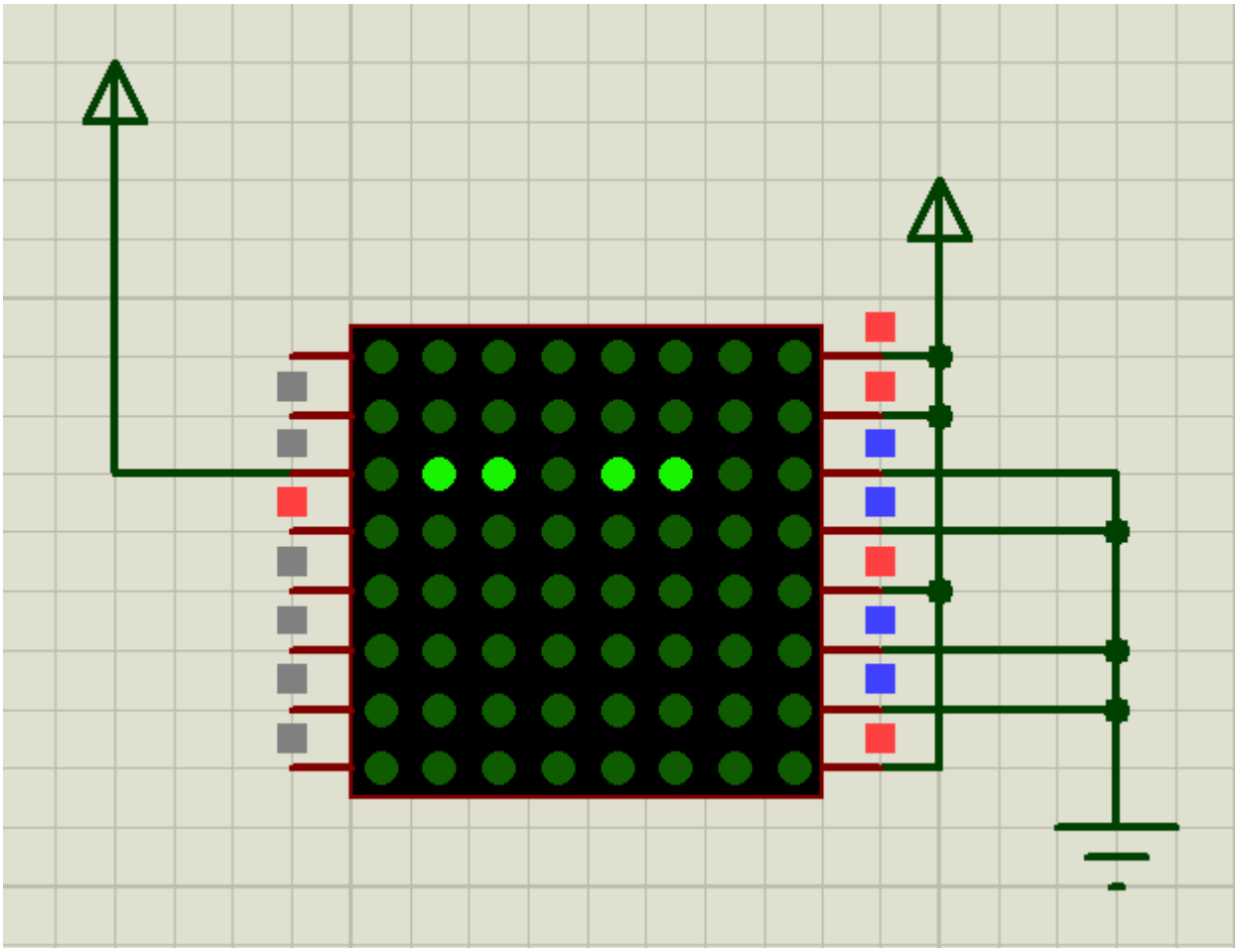
Chạy mô phỏng được kết quả:



Suy ra là 2 chân đầu tiên bên trái là cấp nguồn cho 2 hàng trên, và chân đầu tiên bên phải là đất của cột thứ 8 kể từ bên trái sang. Do đó, giả sử muốn sáng ở ô có hàng là 3, cột là 5 thì cần đưa nguồn vào chân số 3 bên trái, và chân số 4 ở bên phải:



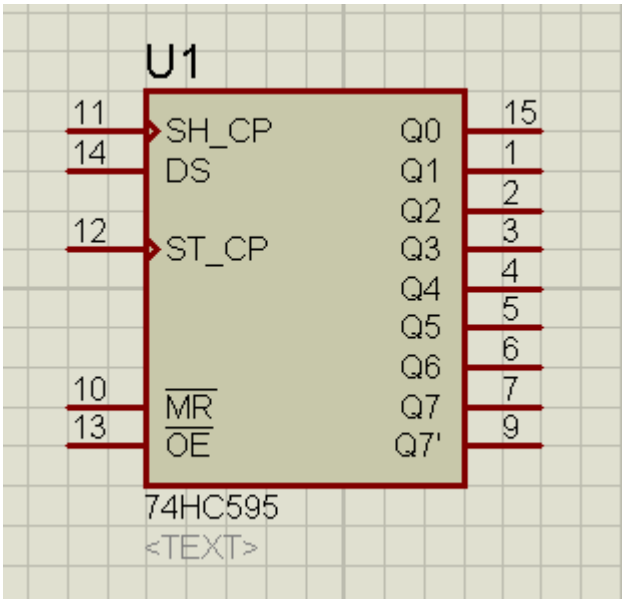
Vậy là khi lấy linh kiện led matrix xanh này ra, nếu quay nó sang phải thì nó sẽ có các chân bên trái là dùng nguồn cấp vào, và số thứ tự chân ứng với số hàng; còn dãy chân bên phải sẽ là những chân mình đưa tín hiệu vào, nếu tín hiệu vào là 1 thì led ở cột nó điều khiển tắt, còn nếu tín hiệu vào là 0 thì cột đó sáng. VD:



Đó là cơ sở để ta quét LED theo hàng, mình sẽ nói sau.

2> IC dịch 74HC595:

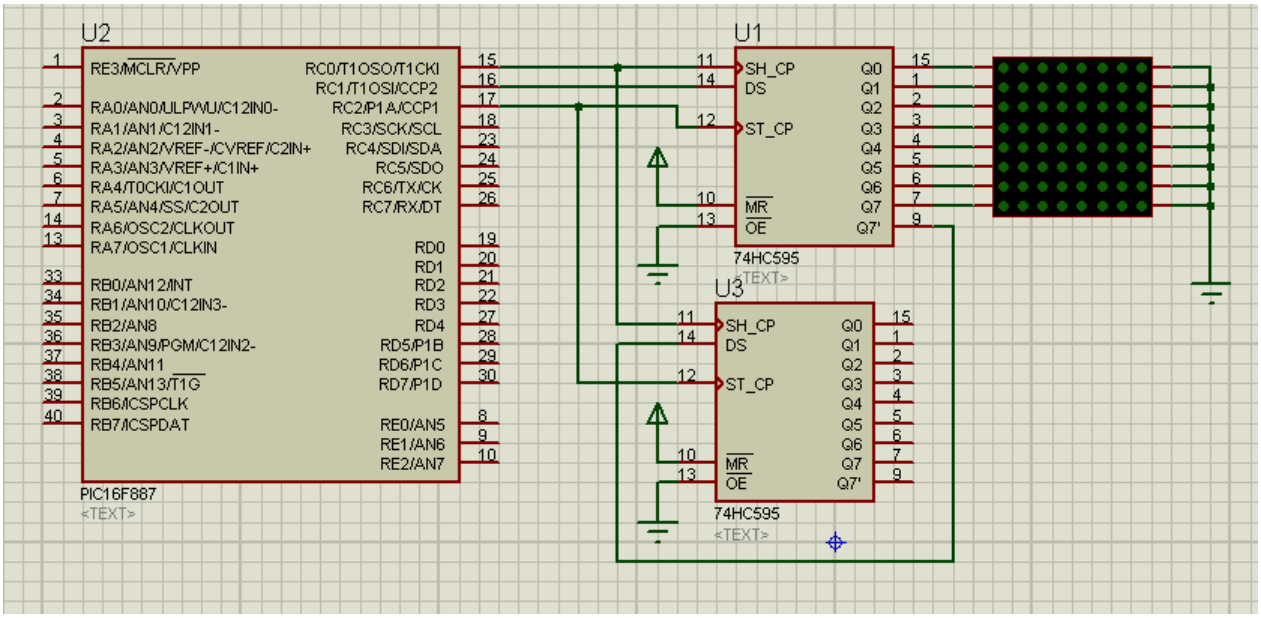
Sơ đồ chân:



Các chân như sau: Mình sẽ nói về chức năng, tí nữa mình sẽ ví dụ cụ thể:

- Các chân từ Q0 đến Q7: các chân này đưa tín hiệu ra ngoài khi có xung xuất, khi chưa có xung xuất thì tương ứng trong đó có thanh ghi 8 bit lưu trữ giá trị sẽ xuất ra.
- Chân Q7’ : dùng để đẩy giá trị của bit dịch khi HC595 đã đầy.
- Chân DS: dữ liệu muốn đưa vào 74HC595 thì đưa tín hiệu vào đây.
- Chân SH_CP: dữ liệu muốn đưa vào 74HC595 thì phải có 1 xung vào chân này để xác nhận.
- Chân ST_CP: khi có 1 xung vào chân này thì giá trị ở các chân Q0 đến Q7 mới xuất điện áp ra ngoài, khi chưa có thì điện áp chưa xuất ra.
- Chân $\overline{MR}$ thì nối nguồn.
- Chân $\overline{OE}$ thì nối đất.

Giả sử nối 74HC595 với PIC như sau:



Giả sử mình muốn đưa tín hiệu Q0=1,Q1=0,Q2=1,Q3=0,Q4=1 vào 595 thứ nhất thì cái nào càng ở cuối thì phải đưa tín hiệu vào đầu tiên:

- Cho chân 16 của PIC =1, đưa 1 xung vào chân 11 của 595 thứ nhất bằng cách cho chân 15 của PIC bằng 1, rồi lại cho chân 15 của PIC bằng 0. Khi đó 595 sẽ xác nhận là có tín hiệu đưa vào chân DS của nó, nên nó đưa tín hiệu 1 vào, và đưa giá trị này vào ô nhớ Q0.
- Cho chân 16 của PIC =0, đưa 1 xung vào chân 11 của 595 thứ nhất bằng cách cho chân 15 của PIC bằng 1, rồi lại cho chân 15 của PIC bằng 0. Khi đó 595 sẽ xác nhận là có tín hiệu đưa vào chân DS của nó, nên nó đưa tín hiệu 0 vào, và đưa giá trị này vào ô nhớ Q0. Giá trị 1 của Q0 lúc trước chuyển sang cho Q1. Vậy sau bước này thì Q0=0 và Q1=1.
- Tương tự như thế, ở chu kì thứ 3, cho chân 16 của PIC bằng 1, cho xung vào chân 11 của 595, thì giá trị 1 này được đưa vào Q0, 2 giá trị kia được đẩy sang Q1 và Q2. Khi đó Q0=1,Q1=0,Q2=1.
- Lần lượt như thế, sau 5 chu kì thì sẽ được giá trị tương ứng như ý muốn nếu mình sử dụng chân 16 của PIC hợp lí

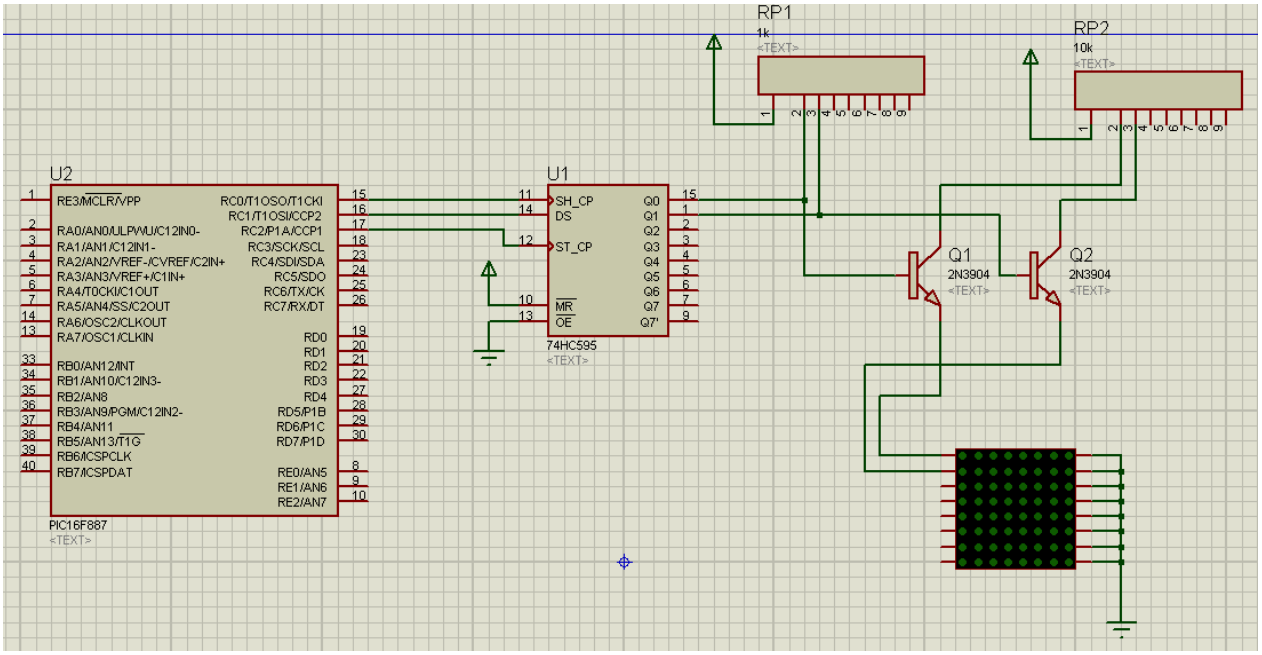
- Giả sử mình đẩy vào 595 thứ nhất lớn hơn 8bit, nghĩa là từ Q0-Q7 không đủ để chửa, thì bit thừa sẽ vào chân Q7', và do chân này nối với chân 14 của 595 thứ 2 nên tính hiệu bit thừa sẽ được đẩy sang con thứ 2 này.

Đó là cách hoạt động của IC dịch bit 74HC595.

3> Mạch điều khiển tín hiệu cho LED ma trận:

Do sử dụng nhiều LED hiển thị nên cần phải có transistor để khuếch đại dòng, tránh sử dụng dòng trực tiếp của PIC sẽ không đủ và không ổn định.

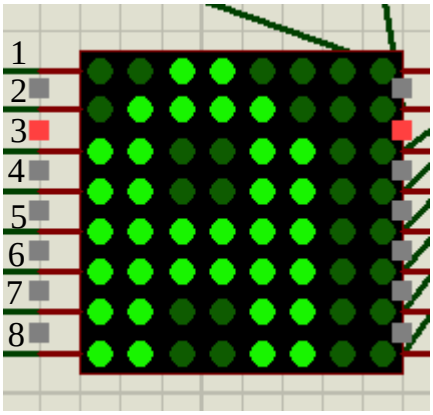
Mạch nối PIC qua 74HC595 và LED ma trận sử dụng transistor 2N3904:



Trong hình trên thì mình chỉ mới nối 2 chân đầu của LED ma trận với 2 transistor, để cần 8 chân điều khiển hàng thì cần 8 transistor nối giống nhau như thế. Nếu có 2 LED ma trận, thì các chân hàng của LED ma trận nối với nhau và nối với Transistor giống như trên.

4> Nguyên lý quét hàng:

Xét cái LED ma trận:



Để hiển thị ra chữ A này, chúng ta quét hàng như sau:

- Cho nguồn vào chân hàng thứ nhất. Các chân hàng khác không có nguồn. Cho tín hiệu các ô từ trái sang phải vào, thứ tự là 11001111 (số 1 thì LED tắt, số 0 thì LED sáng), suy ra tín hiệu đưa vào các chân bên phải là 11110011 (thứ tự ngược lại).
- Cho nguồn vào chân hàng thứ hai. Chân hàng khác không có nguồn. Cho tín hiệu tương tự với tín hiệu cần vào các chân phía bên phải: 11100001 (thứ tự tín hiệu ngược lại với LED sáng).
- Lần lượt như thế đến hàng 8.

Mỗi hàng cho delay rất nhỏ, cỡ ms, nhưng lặp lại vòng này nhiều lần, sao cho >24 hình/s thì mắt người sẽ không thấy nháy, sẽ được chữ A hoàn chỉnh.

5> Mạch:

Dựa vào nguyên lí các linh kiện ở trên. Vẽ được mạch đính kèm file.

6> Lập bảng mã chữ:

Dùng Exel tạo bảng:

0	1	2	3	4	5	6	7	8
1								
2								
3								
4								
5								
6								
7								
8								

Giả sử tạo chữ A, những số 0 thì LED ở đó sáng:

0	1	2	3	4	5	6	7	8
1			0	0				
2		0	0	0	0			
3	0	0			0	0		
4	0	0			0	0		
5	0	0	0	0	0	0		
6	0	0	0	0	0	0		
7	0	0			0	0		
8	0	0			0	0		

Thêm các ô còn lại là số 1 vào, chuyển sang mã Hexa theo hàng:

0	1	2	3	4	5	6	7	8
1	1	1	0	0	1	1	1	1
2	1	0	0	0	0	1	1	1
3	0	0	1	1	0	0	1	1
4	0	0	1	1	0	0	1	1
5	0	0	0	0	0	0	1	1
6	0	0	0	0	0	0	1	1
7	0	0	1	1	0	0	1	1
8	0	0	1	1	0	0	1	1

Tương ứng:

Hàng 1: 1100 1111 -> 0xCF

Hàng 2: 1000 0111 -> 0x87…

Suy ra mảng chữ A: {0xCF,0x87,0x33,0x33,0x03,0x03,0x33,0x33}

Làm tương tự để có các chữ khác, mỗi chữ sẽ là 1 mảng 8 phần tử (mỗi phần tử là 1 chuỗi tín hiệu cần đưa vào chân bên phải của LED tương ứng với 1 chu kì quét 8 hàng).

7> Thuật toán:

Gọi 74HC595 ở phía trên, dùng quét hàng là row; 2 con 74HC595 ở phía dưới dùng đưa tín hiệu vào LED ma trận là column.

a. Hàm con đưa tín hiệu vào 74hC595 column:

Đây là hàm con đưa 8 bit của mỗi phần tử của mảng chữ cái, VD ở trên là mảng chữ A vào các con 74HC595.

Chú ý, muốn đưa 1 tín hiệu vào, cần cho tín hiệu vào chân DS, rồi cho 1 xung vào SH_CP. Để đưa 8 bit vào, cần vòng for để lặp 8 lần, kết hợp phép dịch. Sau khi đưa 8 bit vào xong, ta đưa xung vào chân ST_CP thì sẽ xuất điện áp ra các chân từ Q0-Q7.

VD:

```
void sckr1()
  {
  SCKR=1;
  SCKR=0;
  }

//-------------------------------------
//Tao xung SCK cho cot
//-------------------------------------
void sckc1()
  {
  SCKC=1;
  SCKC=0;
  }

//-------------------------------------
//Tao xung xuat du lieu cho 595 hang
//-------------------------------------
void latr1()
  {
  LatR=1;
  LatR=0;
  }

//-------------------------------------
//Tao xung xung xuat du lieu cot
//-------------------------------------
void latc1()
  {
  LatC=1;
  LatC=0;
  }

//-------------------------------------
//Xuat data vao HC595
//-------------------------------------
```

```
void data1(int8 data)
  {
  for(i=0;i<8;i++)
    {
    DataC=data&1;
    data=data>>1;
    sckc1();
    //LatC1();
    }
    LatC1();
  }
```

b. Kết hợp 2 loại HC595 hàng và cột để được chữ như ý muốn.

Ở bài này dùng 2 LED ma trận, mình sẽ dùng mảng 16 bit để hiển thị 1 lúc 2 chữ cái. VD code của mình:

```
#include <16F887.h>
#device adc=8

#FUSES NOWDT                    //No Watch Dog Timer
#FUSES HS                       //High speed Osc (> 4mhz for PCM/PCH) (>10mhz for PCD)
#FUSES NOPUT                    //No Power Up Timer
#FUSES MCLR                     //Master Clear pin enabled
#FUSES NOPROTECT                //Code not protected from reading
#FUSES NOCPD                    //No EE protection
#FUSES NOBROWNOUT               //No brownout reset
#FUSES IESO                     //Internal External Switch Over mode enabled
#FUSES FCMEN                    //Fail-safe clock monitor enabled
#FUSES NOLVP                    //No low voltage prgming, B3(PIC16) or B5(PIC18) used
for I/O
#FUSES NODEBUG                  //No Debug mode for ICD
#FUSES NOWRT                    //Program memory not write protected
#FUSES BORV40                   //Brownout reset at 4.0V
#FUSES RESERVED                 //Used to set the reserved FUSE bits

#use delay(clock=20000000)

#use fast_io(D)
#use fast_io(C)
#byte PORTD=0x08
```

```
#byte PORTC=0x07
#bit C0=PORTC.0
#bit C1=PORTC.1
#bit C2=PORTC.2
#bit D5=PORTD.5
#bit D6=PORTD.6
#bit D7=PORTD.7

#define SCKR C0
#define LatR C1
#define DataR C2

#define SCKC D7
#define LatC D6
#define DataC D5

int8 i=0;
int8 chu1,chu2;
int8 BangChu[16]={0xCF,0x87,0x33,0x33,0x03,0x03,0x33,0x33, //chu A
            0x1F,0x6F,0x6F,0x1F,0x6F,0x6F,0x6F,0x1F};//chu B

int16 BangChu2[8]={0xCF1F,0x876F,0x336F,0x331F,0x036F,0x036F,0x336F,0x331F};

//-------------------------------------
//Tao xung SCK cho hang
//-------------------------------------
void sckr1()
  {
  SCKR=1;
  SCKR=0;
  }

//-------------------------------------
//Tao xung SCK cho cot
//-------------------------------------
void sckc1()
  {
  SCKC=1;
  SCKC=0;
  }

//-------------------------------------
//Tao xung xuat du lieu cho 595 hang
```

```
//-----------------------------------
void latr1()
  {
  LatR=1;
  LatR=0;
  }

//-----------------------------------
//Tao xung xung xuat du lieu cot
//-----------------------------------
void latc1()
  {
  LatC=1;
  LatC=0;
  }

//-----------------------------------
//Xuat data vao HC595
//-----------------------------------
void data1(int16 data)
  {
  for(i=0;i<16;i++)
    {
    DataC=data&1;
    data=data>>1;
    sckc1();
    //LatC1();
    }
    LatC1();
  }
//-----------------------------------
//--------Hien thi 2 chu--------------
//-----------------------------------
void DispA(int8 chu2)
{
  data1(BangChu2[chu2+0]);DataR=1;sckr1();LatR1();delay_ms(2);
  data1(0xFFFF);data1(0xFFFF);
  data1(BangChu2[chu2+1]);DataR=0;sckr1();LatR1();delay_ms(2);
  data1(0xFFFF);data1(0xFFFF);
  data1(BangChu2[chu2+2]);DataR=0;sckr1();LatR1();delay_ms(2);
  data1(0xFFFF);data1(0xFFFF);
  data1(BangChu2[chu2+3]);DataR=0;sckr1();LatR1();delay_ms(2);
  data1(0xFFFF);data1(0xFFFF);
```

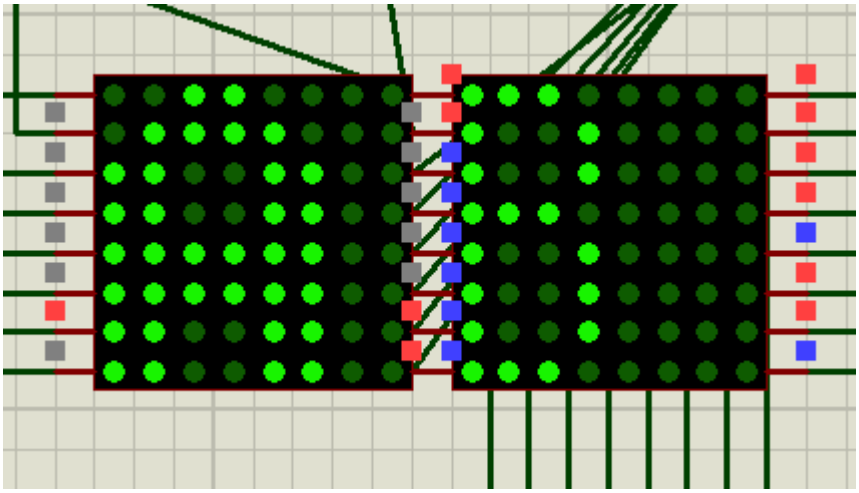
```
  data1(BangChu2[chu2+4]);DataR=0;sckr1();LatR1();delay_ms(2);
  data1(0xFFFF);data1(0xFFFF);
  data1(BangChu2[chu2+5]);DataR=0;sckr1();LatR1();delay_ms(2);
  data1(0xFFFF);data1(0xFFFF);
  data1(BangChu2[chu2+6]);DataR=0;sckr1();LatR1();delay_ms(2);
  data1(0xFFFF);data1(0xFFFF);
  data1(BangChu2[chu2+7]);DataR=0;sckr1();LatR1();delay_ms(2);
  data1(0xFFFF);data1(0xFFFF);

}

void main()
{

int j=0;
set_tris_C(0x00);
set_tris_D(0x00);
while(1)
{
chu2=0;
DispA(chu2);
}
}
```

Kết quả:



Do đang thử nên font không được đẹp lắm.

Hoàng Sỹ Tân – Điện tử 3 – K53

Mình chỉ mới làm đến đây, chưa có hiệu ứng gì. Các bạn có thể thêm bớt, kết hợp để có hiệu ứng chữ chạy.

Có nhiều thuật toán xuất tín hiệu khác nhau nên có thể sử dụng hàm con khác, cần đọc kỹ. Do mình cũng chưa làm được nên cũng chưa giới thiệu ra đây. Mong các bạn tìm hiểu thêm.