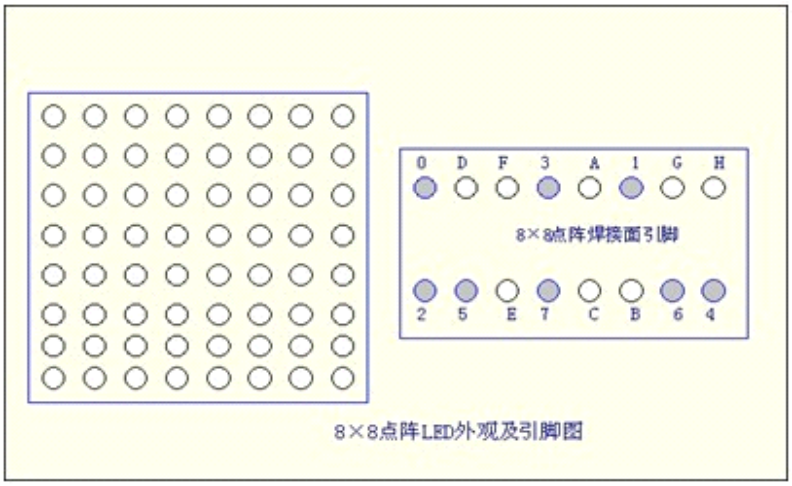
An Arduino control board module as saying that the experiment used a specialized Colorduino RGB RGB full color dot matrix design, this module in addition to the RGB matrix drive independently, but also can be used as a control panel to use Arduino Oh. Very convenient and practical, on-board chip is ATmega 328 chip, the board also raises above the ISP interface, you can always give 328 chip programmed bootloader. The disadvantage is that now you do not want to see other Arduino Uno board, there is no lead to the corresponding I / O out of the mouth, but raises a number of IIC interface and the power supply interface. Follow-up time will be to develop an I / O ports of the board with the purchasers to buy this board does not need to buy more piece Uno board or what the board of MEGA 2560. I hope everyone can support us! Thank you!

Need to explain this Colorduino, shall be divided into two steps terms.

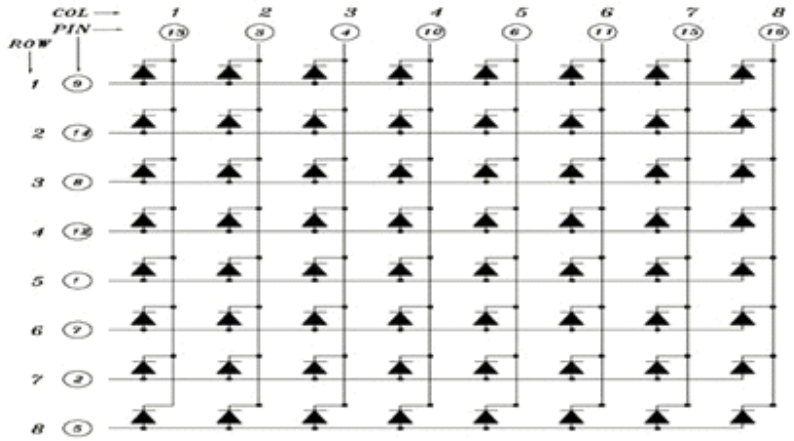
The first is to explain the use of RGB matrix principle.

We first look at the need to use the RGB lattice

RGB Simply put, that is, R, G, B (red, green, blue) three colors abbreviation. She did not come into contact with lattice friends, this is the best start of 8X8 dot matrix monochrome start learning, the driving principle is almost the same. But, RGB multi two colors to be displayed, it will be a little more trouble with respect. So now we describe a simple dot matrix monochrome works:



8×8点阵LED外观及引脚图



Chinese character display is used to display Chinese characters, the character and image information, in buses, banks, hospitals and outdoor advertising and other places have a wide range of applications. Here is the simple production of Chinese characters display, controlled by the microcontroller characters display. To reduce costs, the use of four 8 × LED dot matrix light emitting tube 8 modules, consisting of a 16 × 16 LED dot matrix display, as shown in (3) below. Here only twenty-five characters do show that in the actual use of Chinese characters can be displayed on their own extensions based on this principle, the following is to introduce the principle of Chinese characters display.

LED Driver display dynamic scanning method, the dynamic is progressive scan mode turns on, so that the scan drive circuit can be achieved among multiple lines share the same set of column drivers. To 16 × 16 dot matrix, for example, the arc tube cathode all on the same line together, all the same column of the arc tube anode together (co Yin connection), corresponding to the first to send a light emitting tube On Off of data and latch, and then gating the first one to make it to kindle a certain time, and then goes

out; then send the first two data and latches, and then select the first two to make it through to kindle the same time, then off; ... After the first 16, again to kindle the first one, repeated reincarnation. When such reincarnation fast enough (more than 24 times per second), because the human visual persistence, you can see a stable display graphics. The method is capable of driving more LED, control more flexible, and save MCU resources.

Display data by microcontroller P0,, P2 mouth then transmitted to the drive circuit pin dot matrix line.

There are two ways LED dot matrix display modules for:

1) in the horizontal direction (X direction) scans, which by-column scanning mode (referred to as a column scan mode): this time with a P port output column code can decide which column light (equivalent yards), with another P port output row code (column data), decide which LED lights on the column (equivalent to section code). AlliGHT scanned from left to right columns 16 (corresponding to 16-bit code circulating movement) that is showing a complete image.

(2) vertical (Y-direction) scan, that progressive scan mode (referred to as line scan mode): this time with a P port output can decide which line light (equivalent yards), another P port output column code (rows of data, line data is the dot column data rotated 90 degrees data) decide on the line which LED lights (equivalent to section code). Can the highlighted line down from the scanning finish line 16 (equivalent to 16-bit code cyclic shift) will show a complete image frame.

The design application is the first scan method, i.e., in the horizontal direction (X direction) scan.

Each word is formed by 16 rows of 16 dot matrix display, that is, each word is represented by 256 dot matrix, we can understand each point one pixel. Usually we use dot matrix font Times New Roman 16 × 16, namely the so-called 16 × 16, is in the region of each character of each aspect 16:00 display. Store from 32 bytes of information from the records of the location of the word

Font information. In fact, this screen can display Kanji characters, it can display any graphics in the range of 256 pixels.

We in the horizontal direction (x direction) scan showed the Chinese character "Young" as an example to illustrate its scanning principle, every word consists of 16 rows of 16 dot matrix display, as shown below, and if eight of AT89S51 microcontroller to control,

because the MCU bus 8, a word needs to be split into two parts. Generally, we break it down into the upper portion and a lower portion, the upper part by a dot matrix consisting of 8 * 16, also from the lower part of the 8 * 16 dot matrix composition. In this example, the first single-chip display is part of the upper-left corner of the first column, the first 0 of P00 ~ P07 port. Directions to P07 to P00, to display Chinese characters "Yang" when, P00 to P02 are accursed, P03 light, because the line attached to a cathode, that binary 11110111, converted to hexadecimal is F7H, as shown in (4) below.

After the top half of the first row is completed, continue to scan the bottom half of the first column, ie scanning direction from P27 to P20, can be seen from the chart, this column P2.2 bright, all the rest off, so the code is 11111011, hexadecimal to FBH, then turned to the second column of the upper half of the microcontroller, in addition to P03 bright, the other is not bright, that is 11110111,16 hex is F7H, this column after the scan is complete proceed with the lower half of the scan, In addition to P20 \ P21 bright, the other is not lit, the binary 11111100, or 16 hexadecimal FCH.

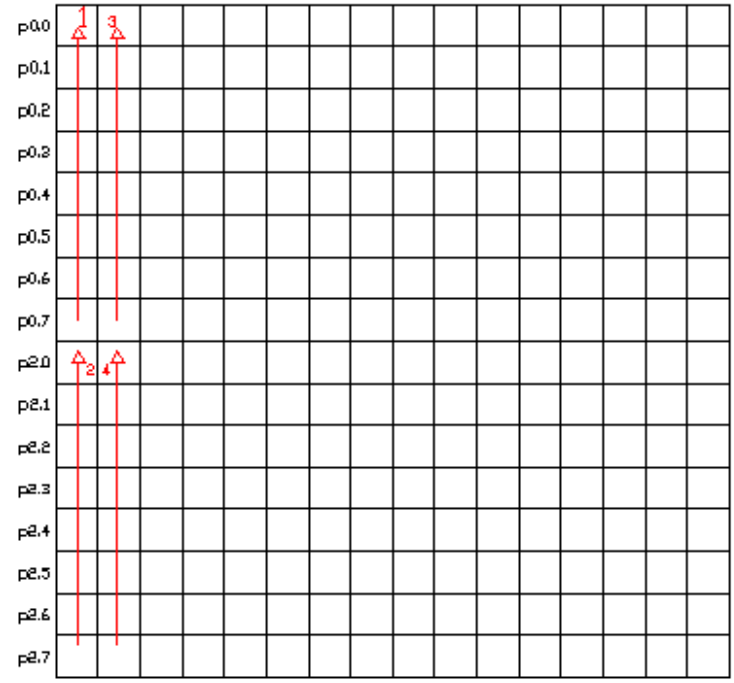
According to this method, continue with the scanning, scanning a total of 32 eight, we can draw the Chinese character "Yang" scan code:

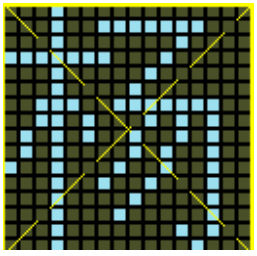
F7H FBH F7H FCH 37H FFH 00H 00H

B7H FFH 77H FEH F5H F7H BDH DBH;

9DH ECH 2DH F7H B5H F9H 39H BEH

BDH 7FH 3FH 80H FFH FFH FFH FFH;





(4) shows the schematic

It can be seen from this principle, despite what the font or image, you can use this method to analyze the scan codes to display it on the screen. Understanding the principles of Chinese characters display after, then how to get font information kanji it? There are some ready-made character font generation software can be downloaded from the Internet Chinese character font extraction procedure extracted directly, as shown as a font generation software (5), enter the characters after the software is open, click on the "seizure", the That hex data kanji code can be generated automatically, copy vertical data we need into our procedures.

Monochrome dot-matrix driver is above works to drive the display of a digital or Chinese, even RGB LED color dot matrix works well.

First, it explains driving this experiment RGB matrix board --Colordunio V2.0

Give us a picture to see:

Fancy graph, we can see both sides of the socket leads to the two rows of RGB, RGB is reserved for the dedicated slot. RGB directly to the above you can plug in to use, simple, easy to use!

Next to the board also raises a number of USB to TTL communication interface. So we need to update the new program board USB to TTL module directly; on-line connection can be updated in real time to update the program. This eliminates a lot of unnecessary trouble downloading program.

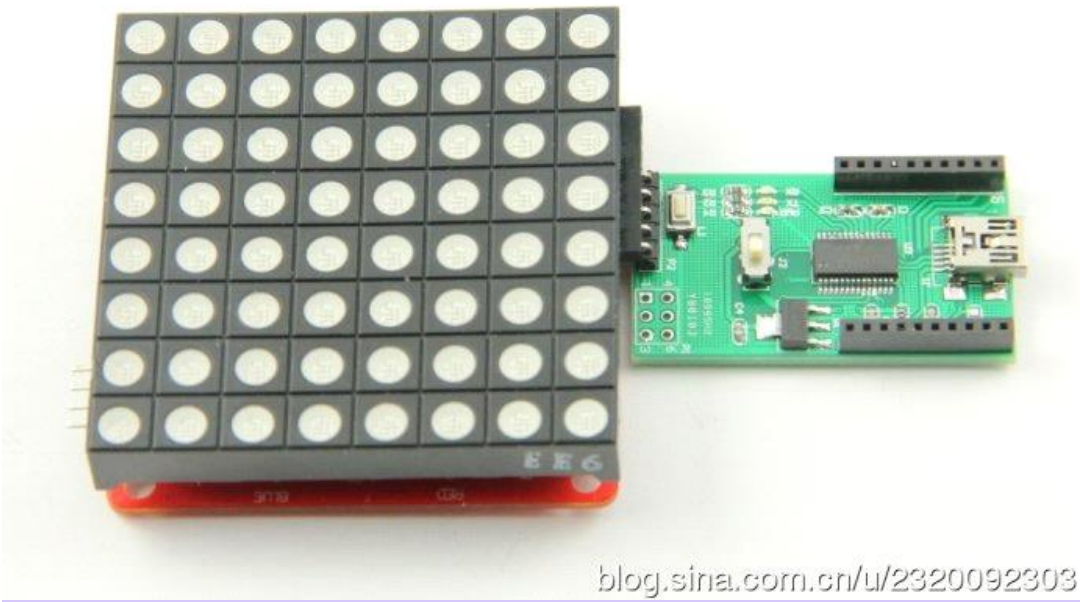
For now the focus of what is Arduino to control RGB dot matrix process.

We look at the definition of a line connection diagram:

Arduino Colordunio V2.1	FT232RT-USB TO TTL
DTR	DTR

GND	GND
RXD	TXD
TXD	RXD
VDD	VDD

The above description of the hardware connection.
Here you can look at the hardware physical connection look.



Very convenient connection, if there is no specific pairing FT232RT modules can also be connected with Dupont line can Oh! Port definition is the same.

After the hardware connection is completed, the software program that needs to be downloaded. You can select various versions of the software version, we use the Arduino IDE 0023 or Arduino IDE 1.0 release!

Test code is the code:

```
#include <Colorduino.h>

typedef struct              //重新进行数据类型定义
{
    unsigned char r;
    unsigned char g;
```

```
    unsigned char b;
} ColorRGB;

//a color with 3 components: h, s and v    一个颜色由H ,S ,V三个部分组成
typedef struct
{
    unsigned char h;
    unsigned char s;
    unsigned char v;
} ColorHSV;

unsigned char plasma[ColorduinoScreenWidth][ColorduinoScreenHeight];    //定义Colorduino Screen 的宽度和高度
long paletteShift;


//Converts an HSV color to RGB color —HSV颜色转换为RGB颜色
void HSVtoRGB(void *vRGB, void *vHSV)
{
    float r, g, b, h, s, v; //这个函数的工作方式的类型是浮点型0和1
    float f, p, q, t;
    int i;
    ColorRGB *colorRGB=(ColorRGB *)vRGB; //两个RGB颜色相与的公式
    ColorHSV *colorHSV=(ColorHSV *)vHSV;

    h = (float)(colorHSV->h / 256.0);
    s = (float)(colorHSV->s / 256.0);
    v = (float)(colorHSV->v / 256.0);

    //如果饱和度为0，那这个颜色就是为白的颜色，也可以看得出是HSV和RGB的值是相等的!
```

```
    if(s == 0.0) {
        b = v;
        g = b;
        r = g;
    }
    //如果这个饱和度大于 0，那就需要更复杂的计算
    else
    {
        h *= 6.0; //h=h*6.0 把颜色调到 0 和 6 之间，这样就会比较好计算！
        i = (int)(floor(h)); //例如 2.7 变成 2、3.01 变成 3 或者 4.9999 变成 4
        f = h - i;//the fractional part of h — h 为小数部分

        p = (float)(v * (1.0 - s));
        q = (float)(v * (1.0 - (s * f)));
        t = (float)(v * (1.0 - (s * (1.0 - f))));

        switch(i)    //利用 Switch 进行判断
        {
            case 0: r=v; g=t; b=p; break;
            case 1: r=q; g=v; b=p; break;
            case 2: r=p; g=v; b=t; break;
            case 3: r=p; g=q; b=v; break;
            case 4: r=t; g=p; b=v; break;
            case 5: r=v; g=p; b=q; break;
            default: r = g = b = 0; break;
        }
    }
    colorRGB->r = (int)(r * 255.0);//得出 RGB 中 R 的数值
    colorRGB->g = (int)(g * 255.0);//得出 RGB 中 G 的数值
    colorRGB->b = (int)(b * 255.0);//得出 RGB 中 B 的数值
```

```
}

float dist(float a, float b, float c, float d)
{
    return sqrt((c-a)*(c-a)+(d-b)*(d-b));//开平方根
}


void plasma_morph()
{
    unsigned char x,y;
    float value;
    ColorRGB colorRGB;
    ColorHSV colorHSV;

    for(y = 0; y < ColorduinoScreenHeight; y++)
        for(x = 0; x < ColorduinoScreenWidth; x++) {
            {
                value = sin(dist(x + paletteShift, y, 128.0, 128.0) / 8.0)
                    + sin(dist(x, y, 64.0, 64.0) / 8.0)
                    + sin(dist(x, y + paletteShift / 7, 192.0, 64) / 7.0)
                    + sin(dist(x, y, 192.0, 100.0) / 8.0);
                colorHSV.h=(unsigned char)((value) * 128)&0xff;
                colorHSV.s=255;
                colorHSV.v=255;
                HSVtoRGB(&colorRGB, &colorHSV);

                Colorduino.SetPixel(x, y, colorRGB.r, colorRGB.g, colorRGB.b);
            }
    }
```

```
    paletteShift++;

    Colorduino.FlipPage(); // 屏幕显示它交换缓冲区
}

void ColorFill(unsigned char R,unsigned char G,unsigned char B)
{
    PixelRGB *p = Colorduino.GetPixel(0,0);
    for (unsigned char y=0;y<ColorduinoScreenWidth;y++) {
        for(unsigned char x=0;x<ColorduinoScreenHeight;x++) {
            p->r = R;
            p->g = G;
            p->b = B;
            p++;
        }
    }

    Colorduino.FlipPage();
}

void setup()    //此函数是进行初始化的操作
{
    Colorduino.Init(); // 初始化 Colorduino

    // compensate for relative intensity differences in R/G/B brightness
    // array of 6-bit base values for RGB (0~63)
    // whiteBalVal[0]=red
    // whiteBalVal[1]=green
    // whiteBalVal[2]=blue
    unsigned char whiteBalVal[3] = {36,63,63}; // for LEDSEE 6x6cm round matrix
```

```
    Colorduino.SetWhiteBal(whiteBalVal);

    // start with morphing plasma, but allow going to color cycling if desired.
    paletteShift=128000;
    unsigned char bcolor;

    //generate the plasma once
    for(unsigned char y = 0; y < ColorduinoScreenHeight; y++)
        for(unsigned char x = 0; x < ColorduinoScreenWidth; x++)
        {
            //the plasma buffer is a sum of sines
            bcolor = (unsigned char)
            (
                        128.0 + (128.0 * sin(x*8.0 / 16.0))
                    + 128.0 + (128.0 * sin(y*8.0 / 16.0))
            ) / 2;
            plasma[x][y] = bcolor;
        }

  //    调整白平衡的话，你可以取消这一行！
  //    在 loop()函数中注释掉 plasma_morph()
  //    用在 whiteBalVal 上面做实验
  //    ColorFill(255,255,255);
}

void loop()       //执行部分，相当于执行了 plasma_morph()函数即可
{
    plasma_morph();
}
```