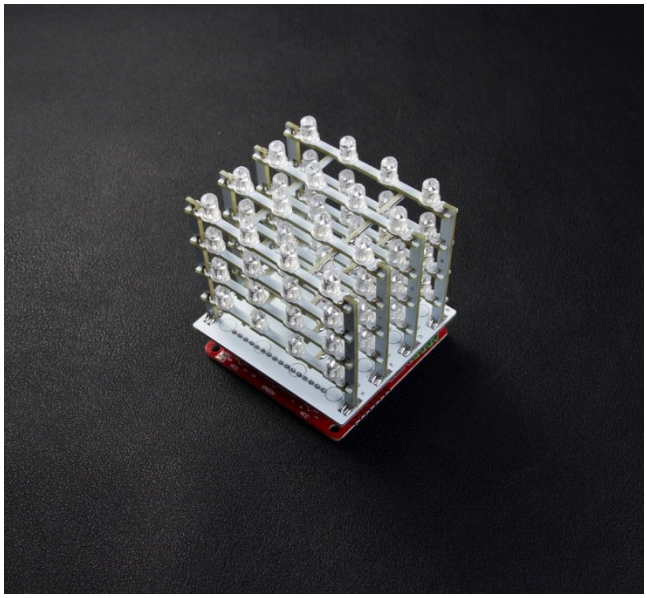
User Manual

for

4×4×4 RGB LED Cube kit(AD060)



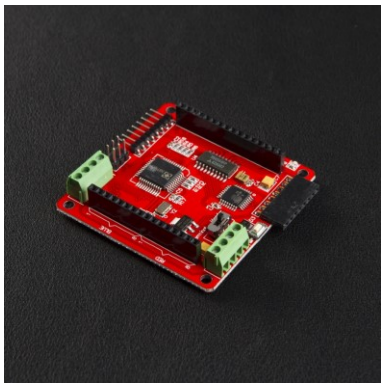
Description

This is a 4*4*4 RGB LED Cube, which is different from most LED cube in the market. Most LED cube projects are based on Rainbow Cube, which works very similar. Our one we design a 4*4*4 LED frame which is suitable for many LED driver boards, you can just buy the frame and program the display through your own led driver board. Of course, we offer an alternative one called colorduino board, the item is ST1149.

Specification

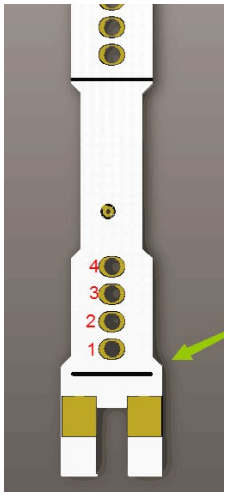
- 4x4x4 matrix of individually addressable 8mm RGB LED
- Compatible different LED driver board
- Dimensions: 106(W) x 130(H) x 106(D)mm
- Assembled or not by your mind, default is a kit

Note: This product needs to work with LED driver board, any colorduino compatible board can fit the size perfectly. Our one as below(Item no. ST1149):

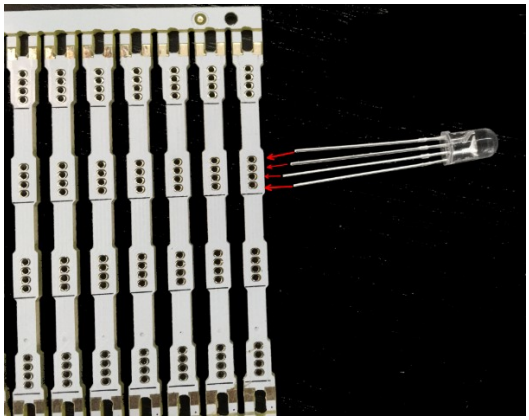


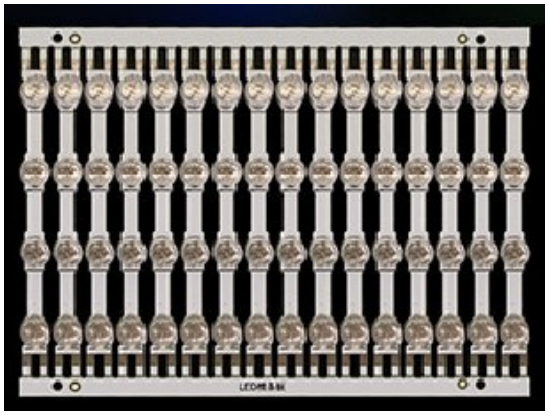
1How to solder the LED ?

There are 64 LEDs, every LED has 4 pins, like the below picture,



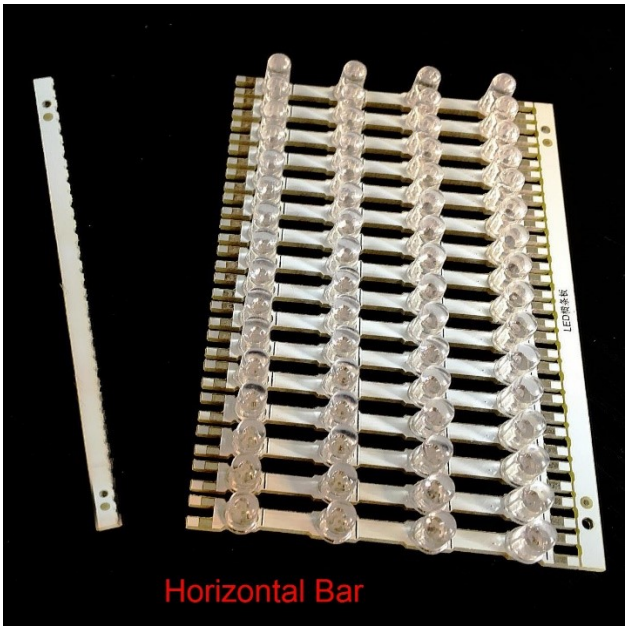
The LED pin1 should be solder the hole 1 on the plate bar. Pin2~4 should be hole 2~4. The final picture as follow:



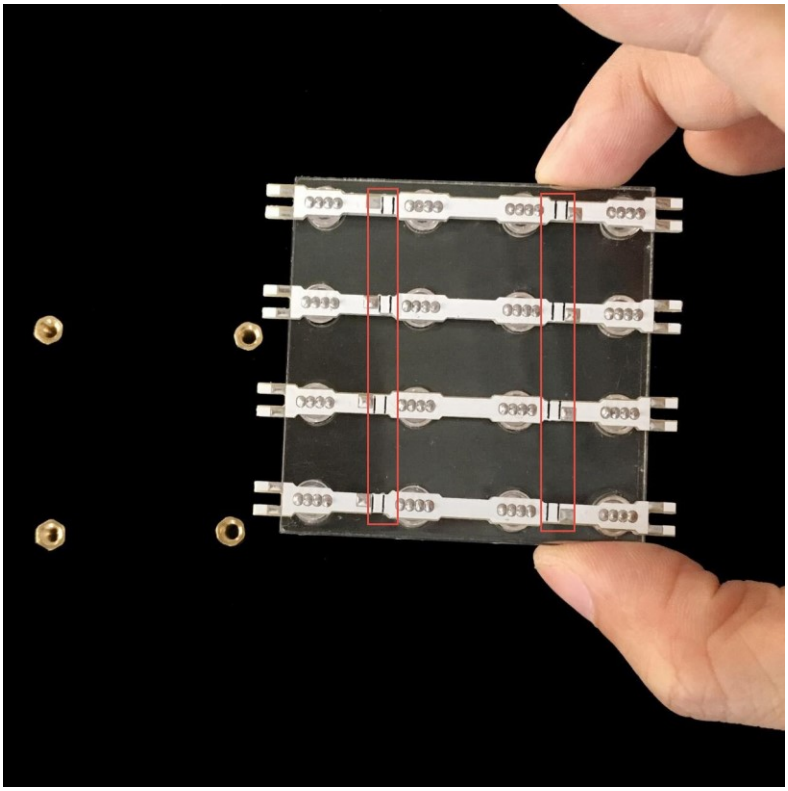


2How to solder the Frame?

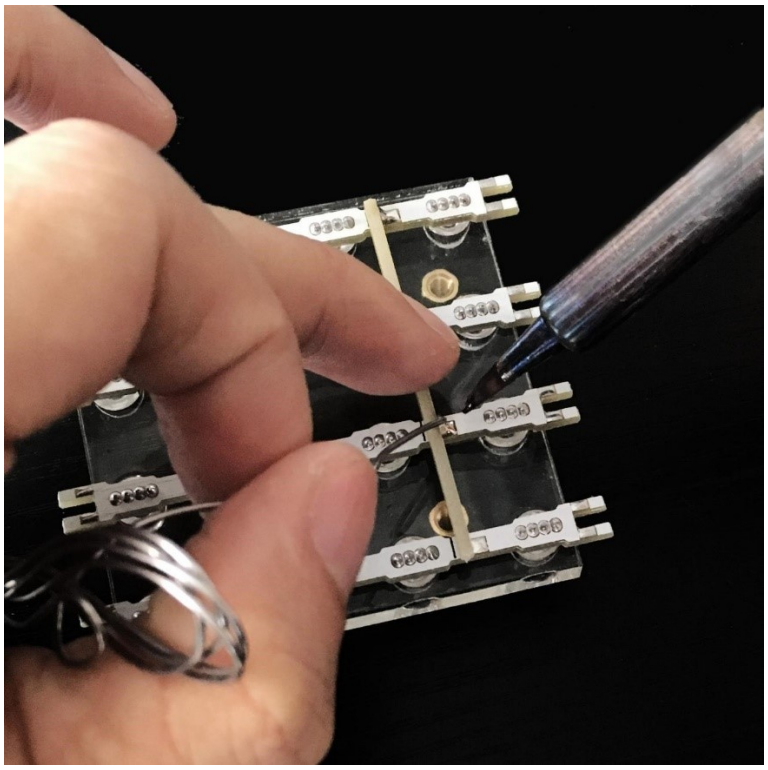
2.1 Unpick the plate bars like the following picture:



In following soldering progress, you can use the soldering helper. As a skillful engineer, I use a simple soldering helper(four nuts and an acrylic plate).

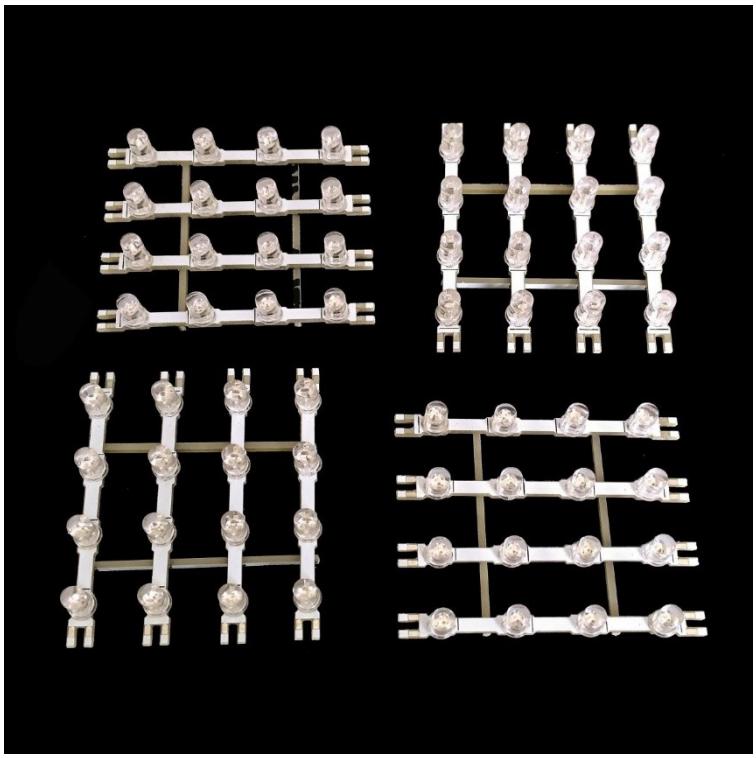


Please note the double black line, next step we will solder vertical bar on there.

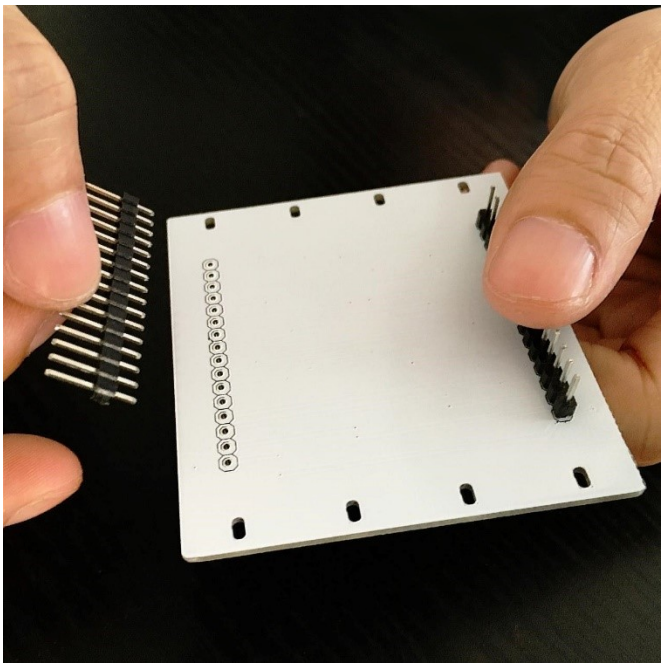


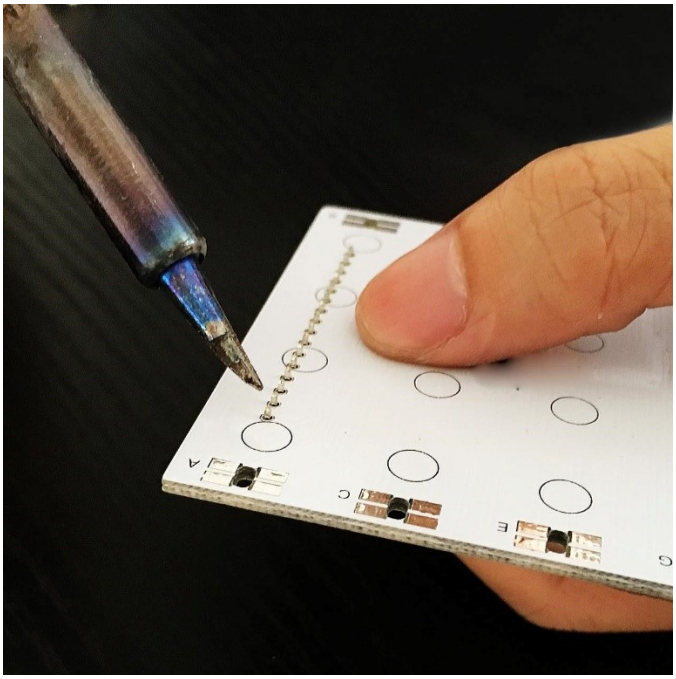
When to finish one plate, the picture as below:

Then repeat above procedure 3 times, we will get 4 plates, as below ;



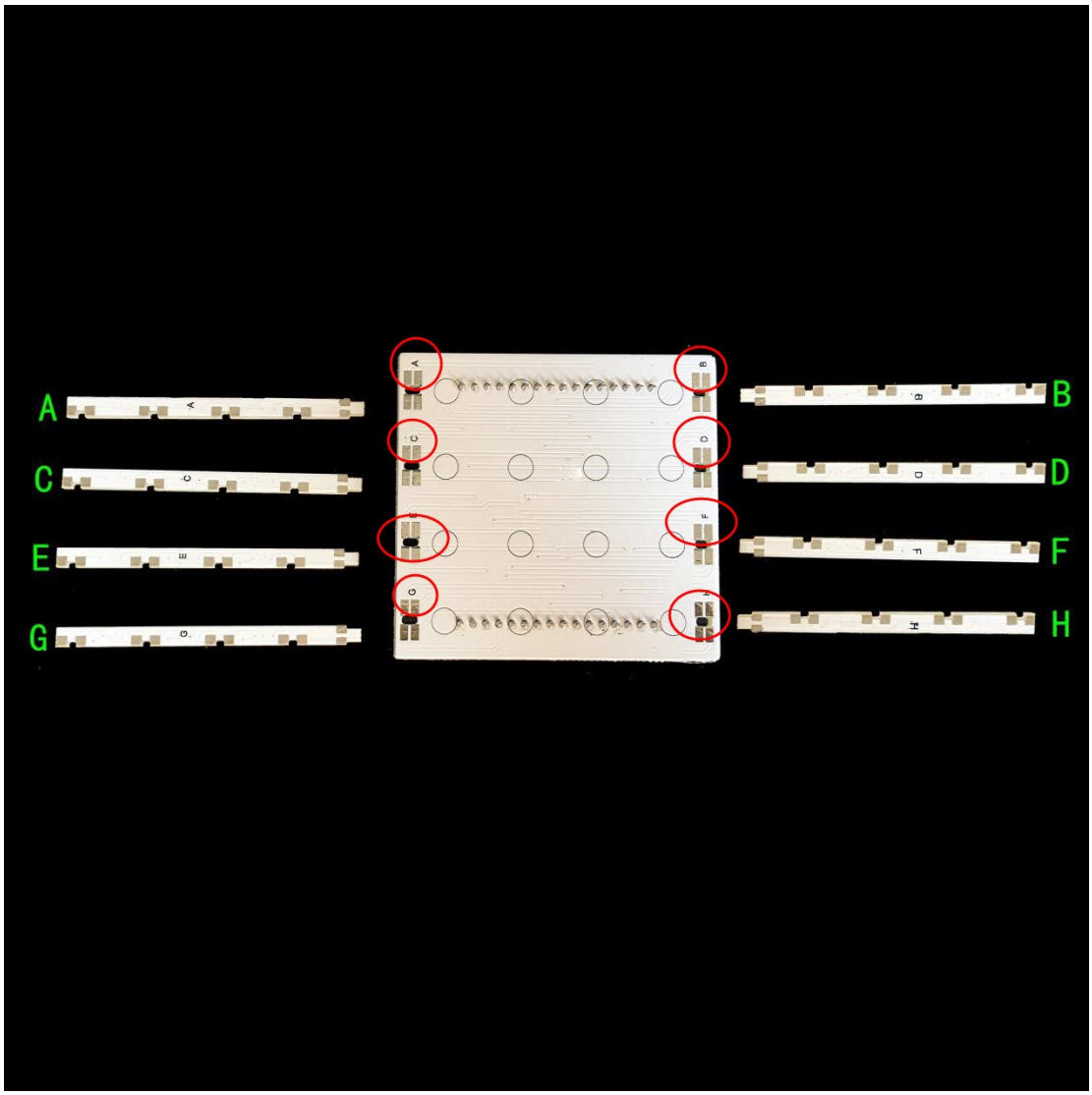
2.2 Solder the base board



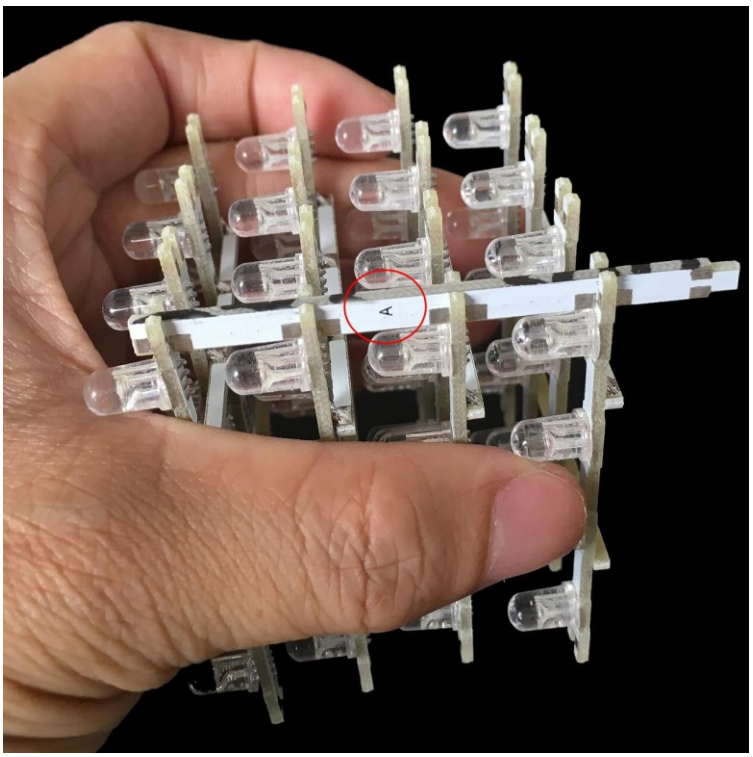


2.3 Assemble the plate

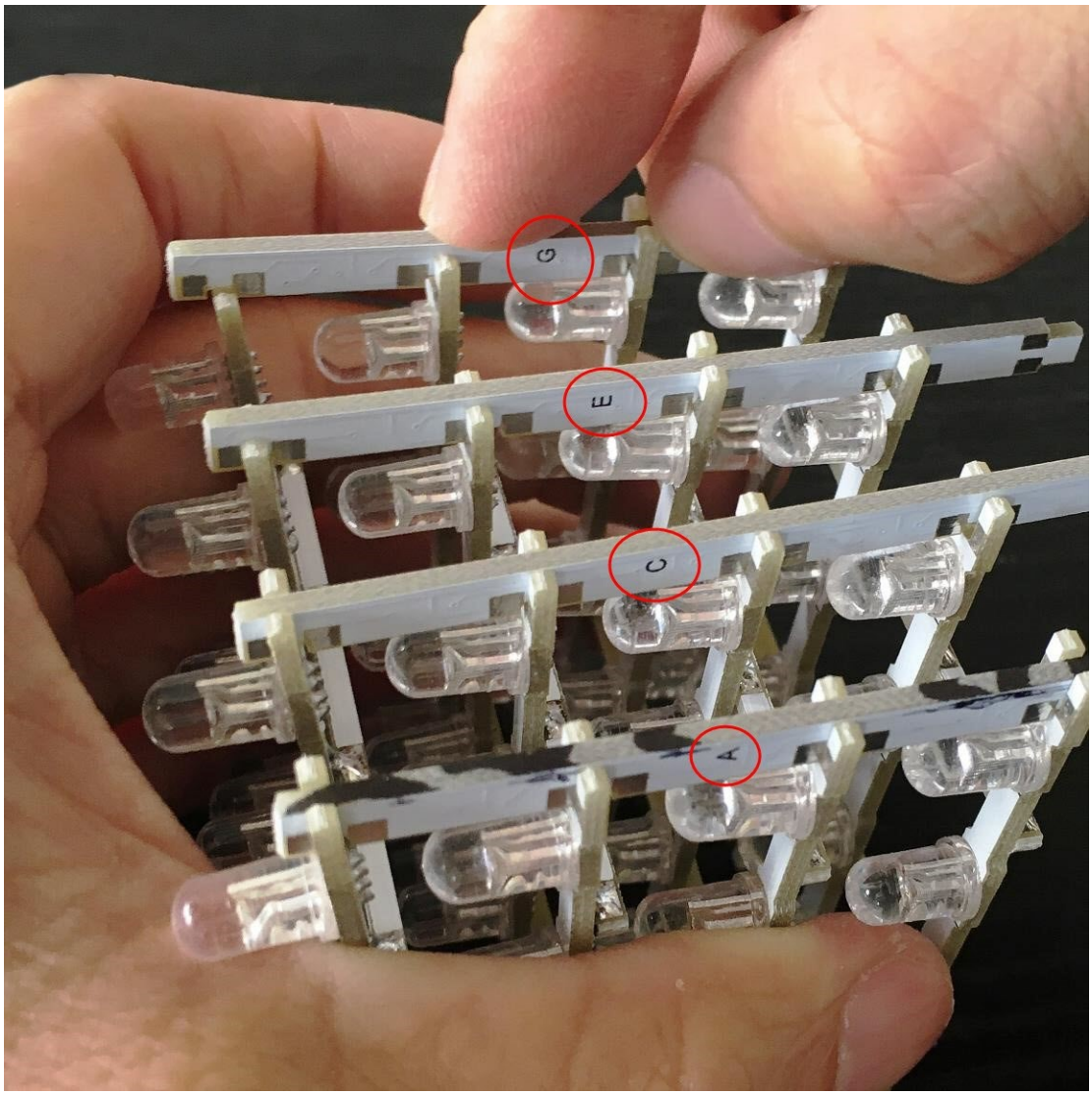
At first, we should see the sequence how the plate bars install into the base board. Same bar number and hole must be soldered together.



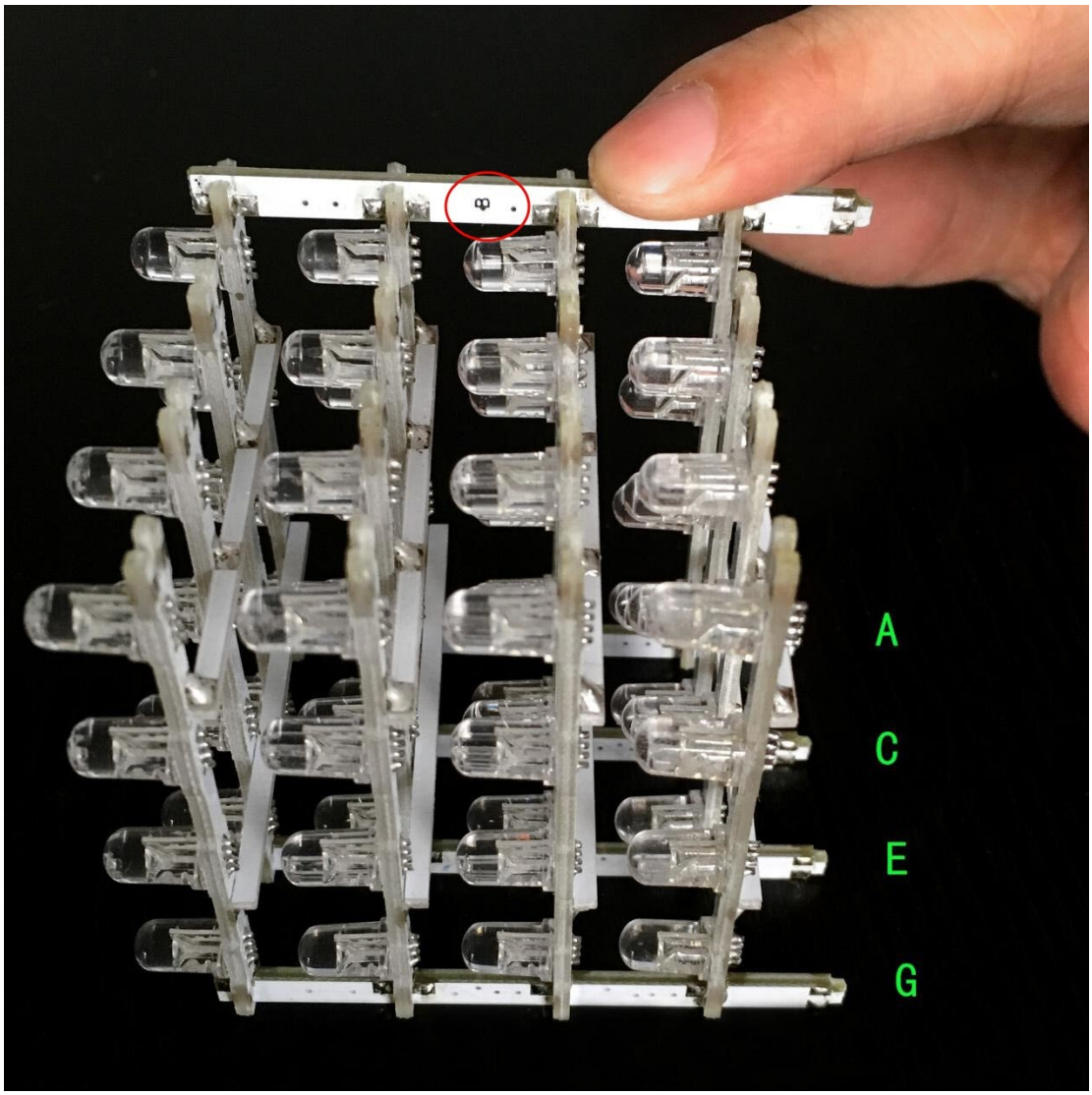
Put four plates in the same direction, and install the plate bar A .

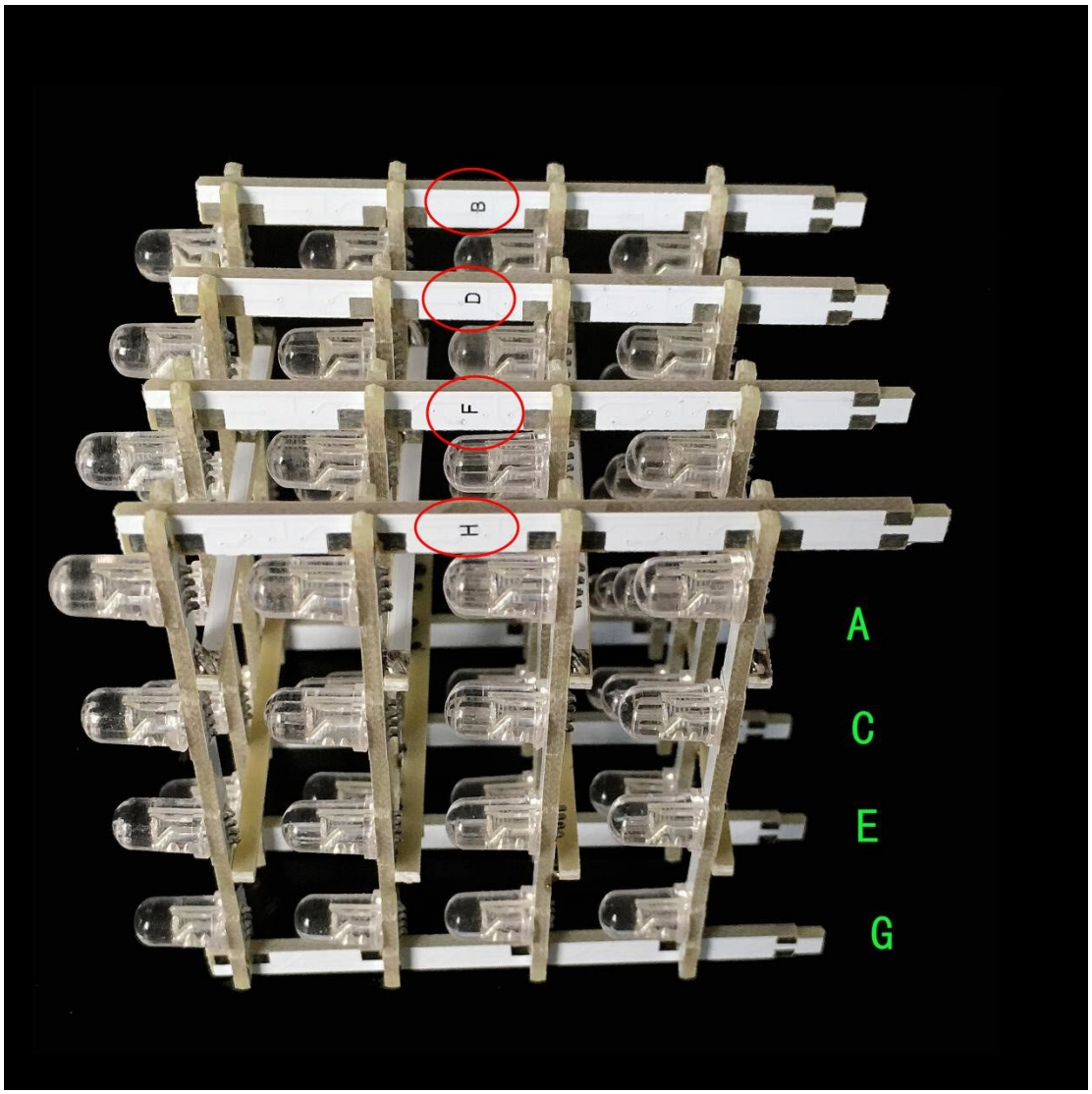


The install the C E G bars, the sequence must be correct.

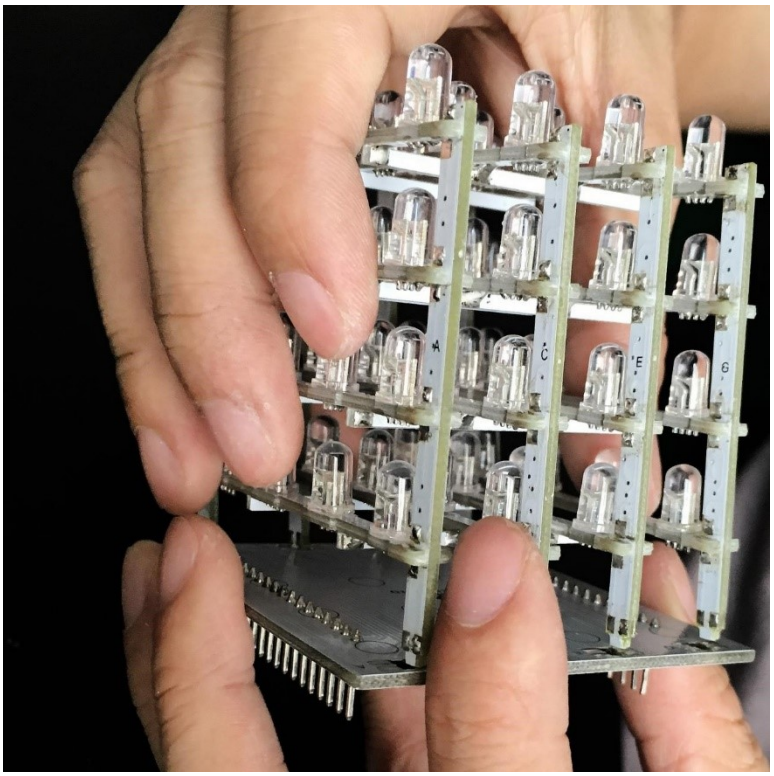


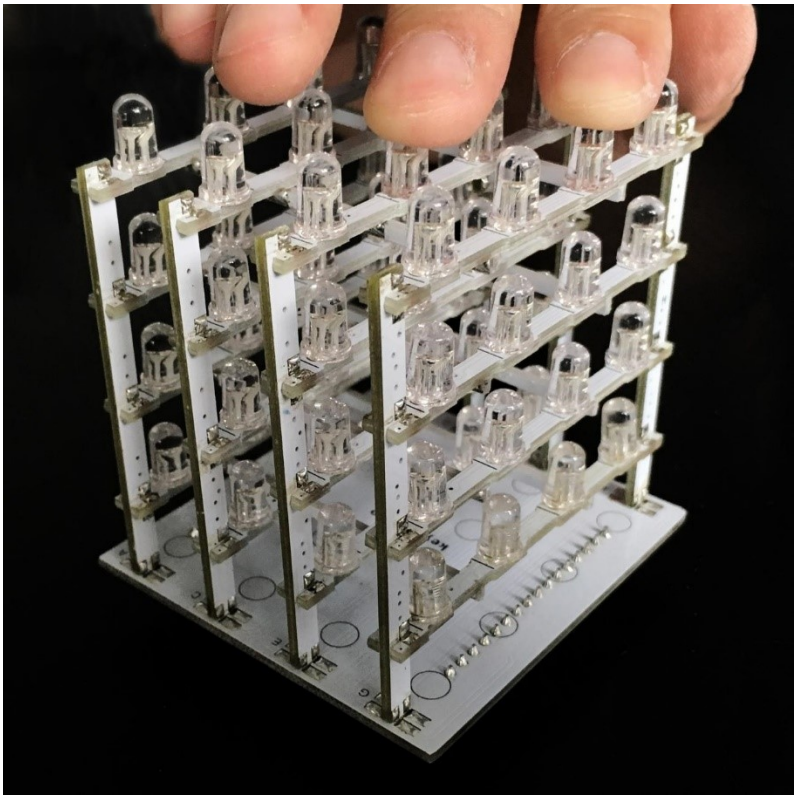
Then install B D F H bars, the sequence must be correct.

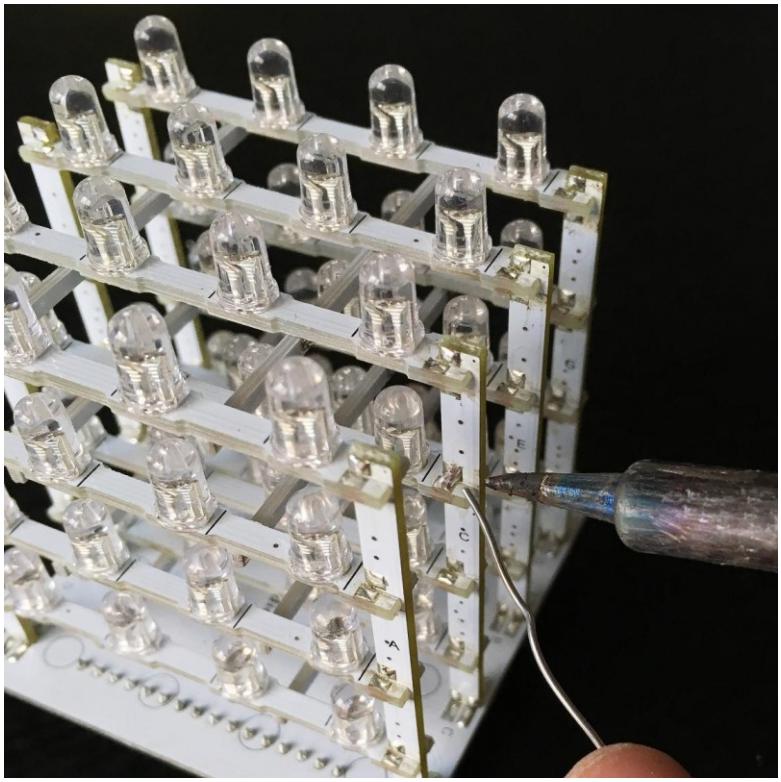

A
C
E
G



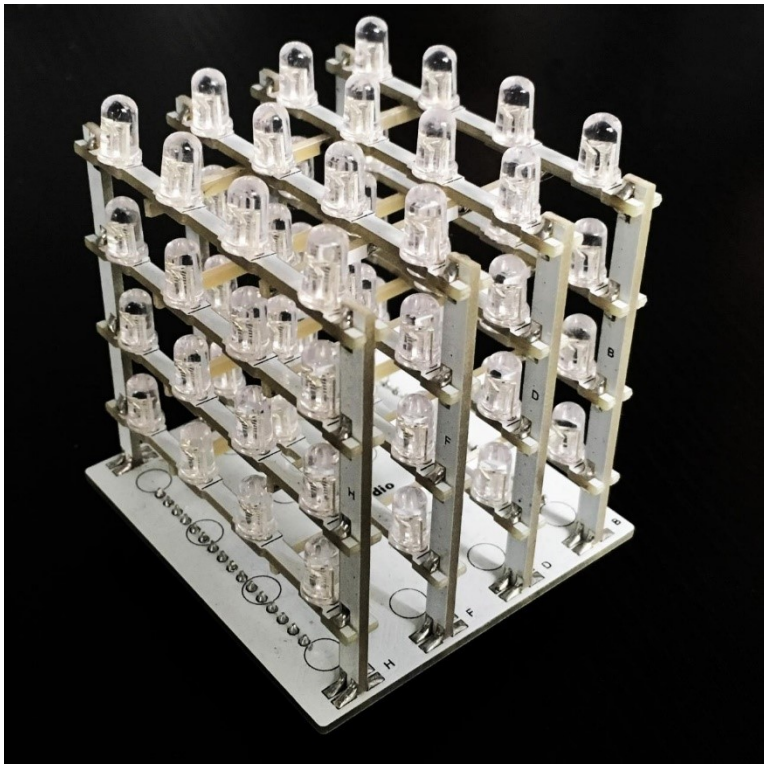
Then plug the plate into the base board and solder:





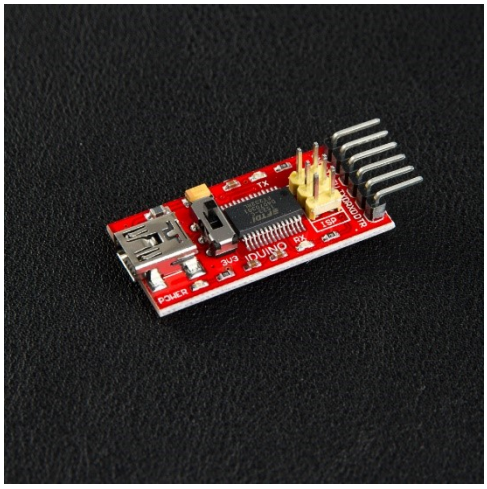


When finished, the picture as below:



3 Burn code to the driver board

In order to upload the code, you may need an FTDI Basic Program Downloader, like the following picture:



Then through the Arduino IDE, upload the code to the driver, and you can change the code to show you wanted a display.

**********Code begin**********

```
#include <Colorduino.h>
typedef struct
{
unsigned char r;
unsigned char g;
unsigned char b;
} ColorRGB;
//a color with 3 components: h, s and v
typedef struct
{
unsigned char h;
unsigned char s;
unsigned char v;
} ColorHSV;
unsigned                                                    char
```

```
plasma[ColorduinoScreenWidth][ColorduinoScreenHeight];
long paletteShift;
//Converts an HSV color to RGB color
void        HSVtoRGB(void        *vRGB,        void        *vHSV)
//////////////////////////////////////////////////////////////////
////////////////
{
float r, g, b, h, s, v; //this function works with floats between
0 and 1
float f, p, q, t;
int i;
ColorRGB *colorRGB=(ColorRGB *)vRGB;
ColorHSV *colorHSV=(ColorHSV *)vHSV;
h = (float)(colorHSV->h / 256.0);
s = (float)(colorHSV->s / 256.0);
v = (float)(colorHSV->v / 256.0);
//if saturation is 0, the color is a shade of grey
if(s == 0.0) {
b = v;
g = b;
r = g;
}
//if saturation > 0, more complex calculations are needed
else
{
h *= 6.0; //to bring hue to a number between 0 and 6, better for
the calculations
i = (int)(floor(h)); //e.g. 2.7 becomes 2 and 3.01 becomes 3 or
4.9999 becomes 4
f = h - i;//the fractional part of h
p = (float)(v * (1.0 - s));
q = (float)(v * (1.0 - (s * f)));
t = (float)(v * (1.0 - (s * (1.0 - f))));
switch(i)
{
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);
colorRGB->g = (int)(g * 255.0);
colorRGB->b = (int)(b * 255.0);
}
float    dist(float    a,    float    b,    float    c,    float
d)////////////////////////////////////////////////////
{
return sqrt((c-a)*(c-a)+(d-b)*(d-b));
}
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(); // swap screen buffers to show it
}
/*****************************************************
Name: ColorFill
Function: Fill the frame with a color
```

```
Parameter:R: the value of RED. Range:RED 0~255
G: the value of GREEN. Range:RED 0~255
B: the value of BLUE. Range:RED 0~255
**************************************************/
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()
{
Colorduino.Init(); // initialize the board
// 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
http://keyes-arduino.taobao.com
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;
}
// to adjust white balance you can uncomment this line
// and comment out the plasma_morph() in loop()
// and then experiment with whiteBalVal above
// ColorFill(255,255,255);
}
void loop()
{
plasma_morph();
}
```

**********Code End**********