عینکRGB

این پروژه یک عینک RGBرا با استفاده از LEDهای WS2812Bو آردوینو نانو توضیح می دهد .عینک دارای انیمیشن های متعددی است که می توان آن را از از طریق یک برنامه کنترل کرد .این عینک سبک وزن است و می تواند از طریق بلوتوث با آردوینو ارتباط برقرار کند.

by Alireza Hayat Davodi

فهرست

اجزای سخت افزاری

1 **آردوینو نانو R3** 

مغز پروژه که کنترل LEDها را بر عهده دارد

2 **WS2812Bهای LED*88** 

88عدد LED RGBقابل برنامه ریزی برای ایجاد الگوهای نوری

3 **ماژول بلوتوث HC-05** 

برای ارتباط بی سیم با برنامه موبایل

4 **باتری 3.7ولت** 

منبع تغذیه قابل حمل برای عینک

5 سوییچ ضامن ، روشن 

برای ارتباط بی سیم با برنامه موبایل

نرم افزار و ابزارها

برنامه موبایل

برای کنترل انیمیشن های عینک از راه دور

هویه

برای لحیم کاری اتصالات الکترونیکی

Arduino IDE

محیط برنامه نویسی برای آردوینو که برای نوشتن و آپلود کد استفاده می شود

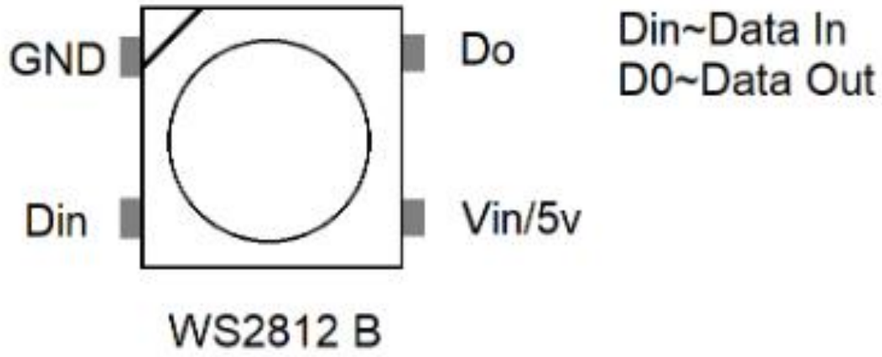
نمودار پین

1_GND(ground):

این پین را به زمین منبع تغذیه یا میکروکنترلر خود متصل کنید.

2_Din (Data In):

این جایی است که شما سیگنال داده (معمولا از یک میکروکنترلر) را به WS2812Bتغذیه می کنید. برای کنترل Ledها ضروری است.



3_Do (Data Out):

اگر چندین ماژول WS2812Bرا با هم زنجیر می کنید ، این پین را به Din ماژول بعدی متصل کنید. سیگنال داده را در امتداد زنجیره عبور می دهد.

4_VIn/5v (Voltage In/5 volts):

منبع تغذیه 5V به این پین. اطمینان حاصل کنید که با ولتاژ مورد نیاز نوار LED شما مطابقت دارد.

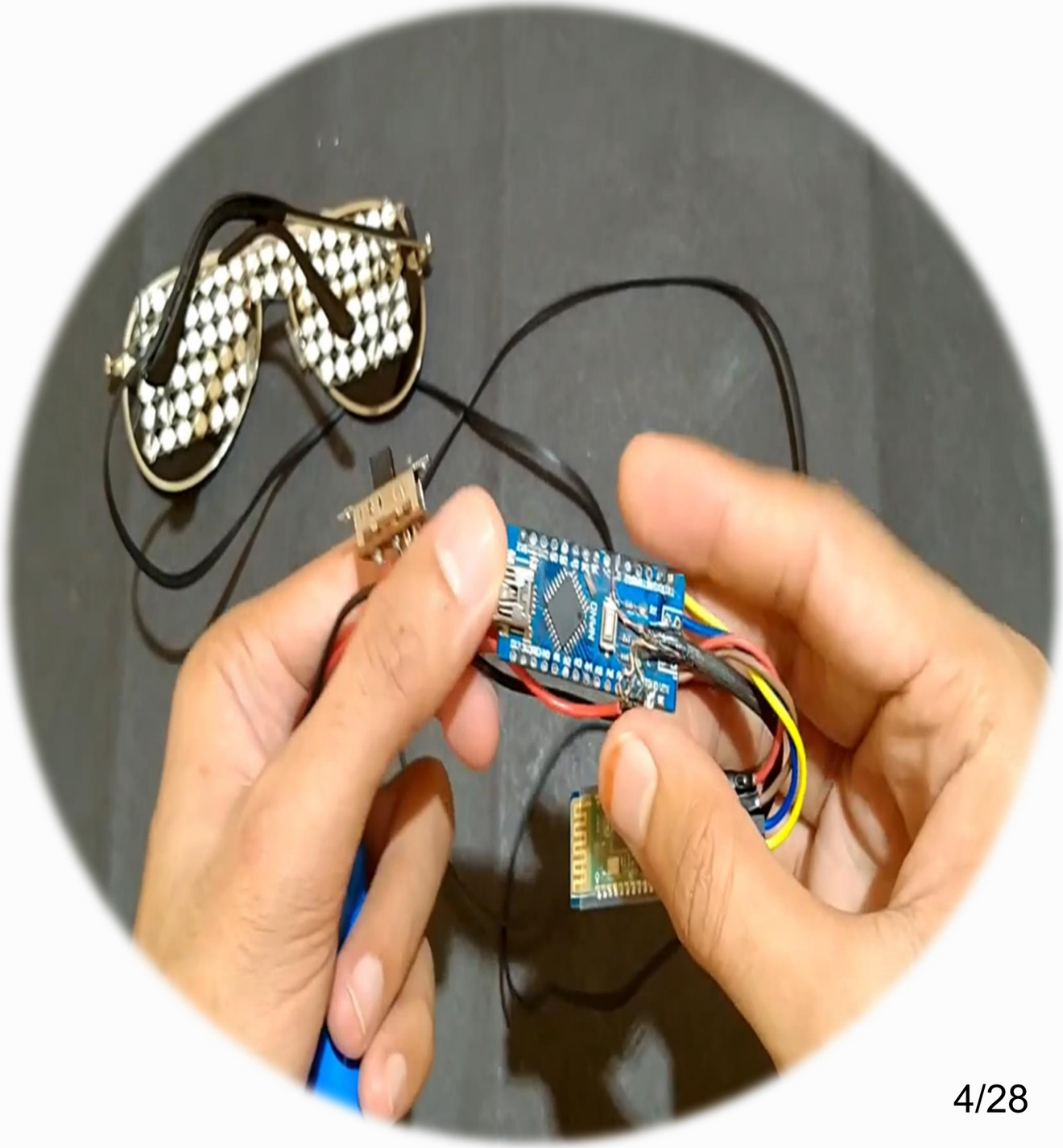
راه اندازی سخت افزار(1)

**آردوینو برد اصلی ما و مغز پروژه که کنترل ال ای دی ها را بر عهده دارد ،
با دستگاه هویه و لحیم اتصالات را به ؛ پین های ال ای دی(در اسلاید قبل دیدیم) ، سر منفی باطری ، و ماژول بلوتوث HC-06 متصل میکنیم**

1: اتصال به ماژول بلوتوث ؛
-پین TX در آردوینو به پین RXD در ماژول بلوتوث
-پین RXD آردوینو به پین TXD در ماژول بلوتوث

2:اتصال به باتری ؛
-پین GND در آردوینو به سز منفی باتری

3: اتصال به LED ها ؛
- پین D3 در آردوینو به پین Din در LED ها



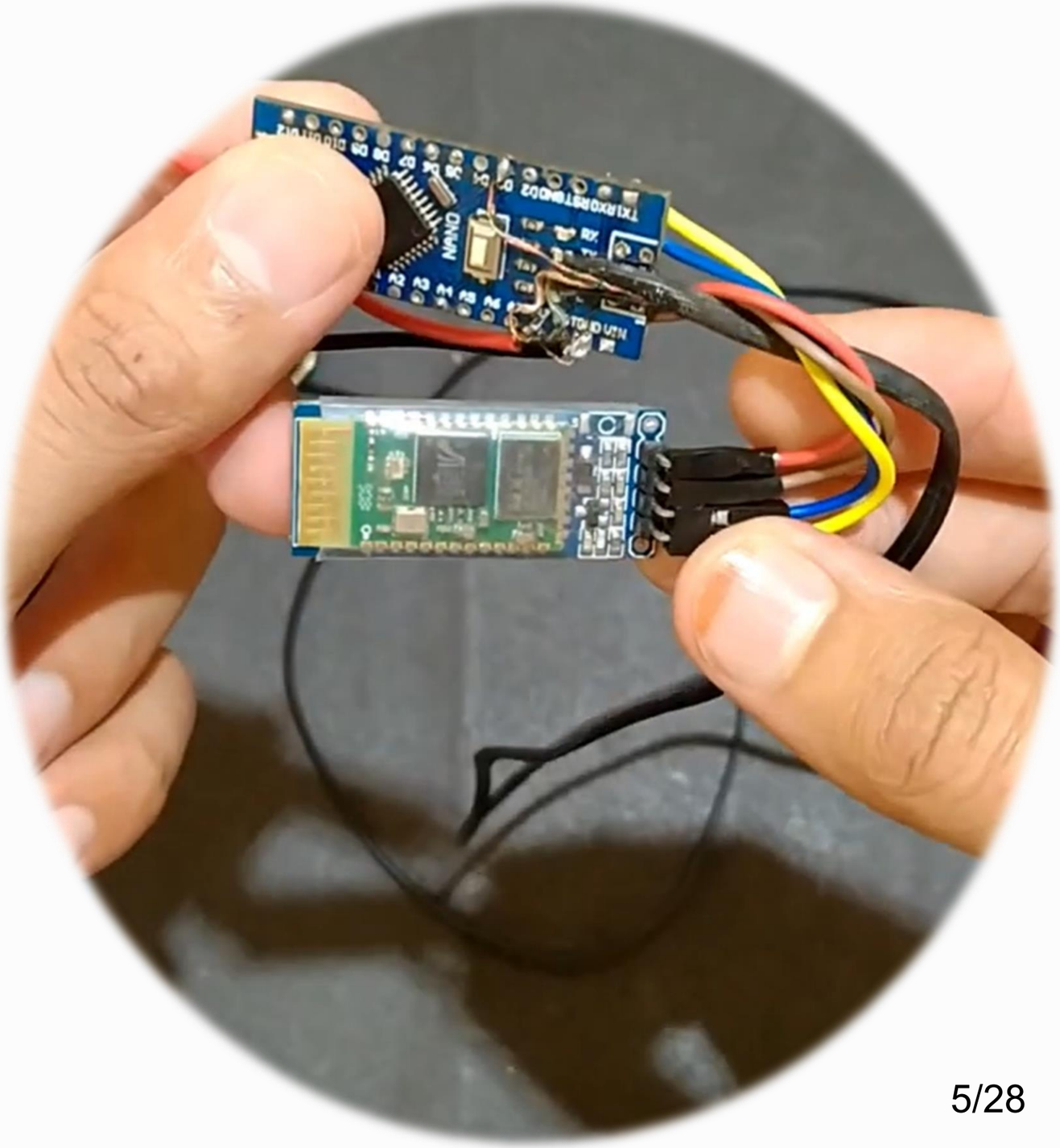
راه اندازی سخت افزار(2)

ماژول بلوتوث ما که برای ارتباط بیسیم و کنترل ال ای دی ها از طریق موبایل است

1: اتصال به آردوینو ؛
-پین RXD در ماژول بلوتوث به پین TX در آردوینو
-پین TXDدر ماژول بلوتوث به پین RXD در آردوینو

2:اتصال به باتری ؛
-پین GND در ماژول بلوتوث به سر منفی باتری

3: اتصال به سوییچ روشن ؛
- پین VCC در ماژول بلوتوث به پین on-on در سوییچ



راه اندازی سخت افزار(3)

باتری ، منبع تغذیه تمام قطعات

1: اتصال به سوییچ ؛
- خروجی مثبت باتری به سوییچ جهت تحت کنترل داشتن انرژی الکتریسیته ورودی

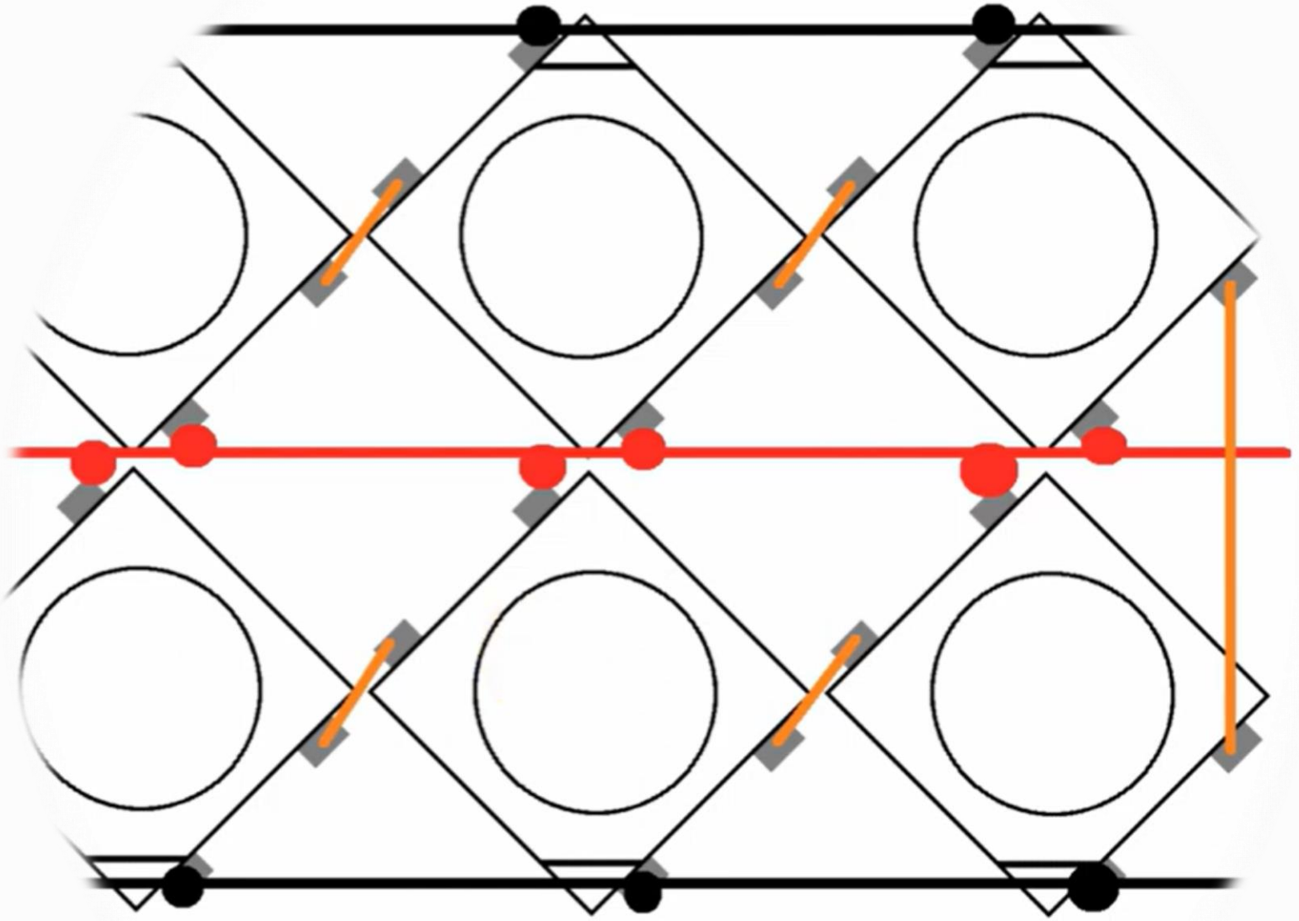
2:اتصال به GND(ground)؛
در هر سه سخت افزار دیگر: آردوینو ،ماژول بلوتوث و ال ای دی ها ، اتصال سر منفی باتری به پین GND در تمام محصولات

راه اندازی سخت افزار(4)

نمودار پین ها و اتصالات پین های ال ای دی ها به صورت سری

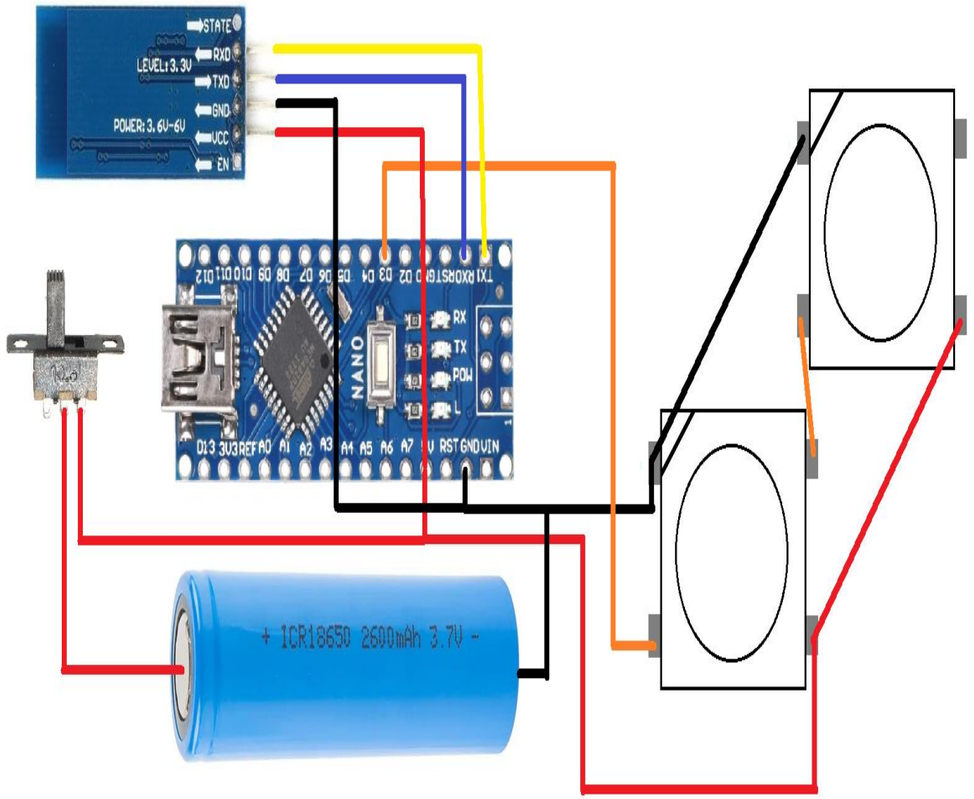
نمودارهای مدار 88 ال ای دی WS2812bبردارید و آنها را روی نوار ویلو یا نوار دو طرفه مرتب کنید.
Ledها باید به گونه ای مرتب شوند که تمام پین های GNDو Vccباید در همان خطوط مربوطه باشند.
خطوط LEDمتناوب باید معکوس شوند تا GND/Vccبرای دو خط LEDرایج شود.
GND~GND
5v / 3v ~Vin / 5v
پین داده ~ پین 3

خط سیاه ؛ خط GND(ground)
خط قرمز ؛ خط منبع تغذیه که از سوییچ میآید
خط نارنجی ؛ برای اتصال چندین LED به هم است که پین Do هر ال ای دی به پین Din ال ای دی بعدی متصل میشود



مدار الکترونیکی کلی

مدار الکترونیکی این پروژه شامل اتصال آردوینو نانو به LEDهای WS2812B، ماژول بلوتوث HC-06و باتری است LED .ها به صورت سری به هم متصل شده و از پین دیجیتال 3آردوینو کنترل می شوند .ماژول بلوتوث نیز به پین های سریال آردوینو متصل می شود.

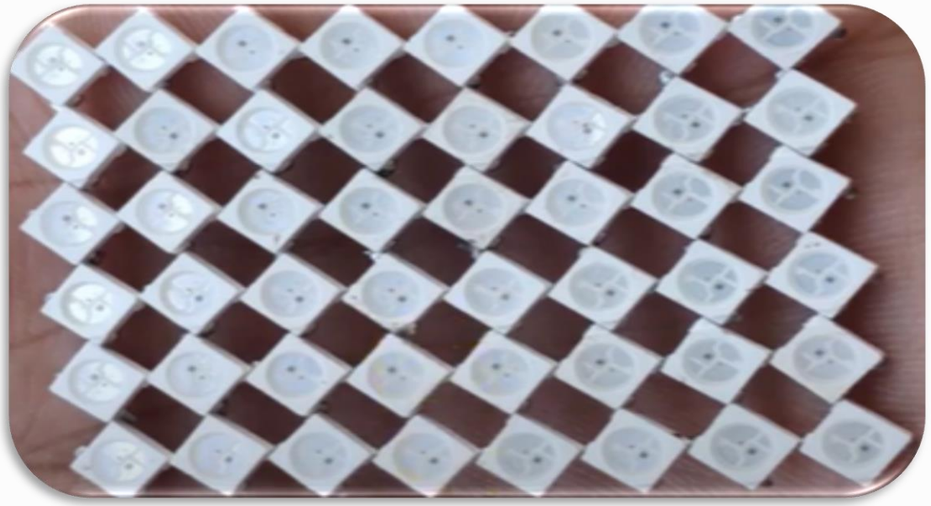


تمام اتصالات خطوط LED را به عنوان خطوط مار گونه ایجاد کنید.

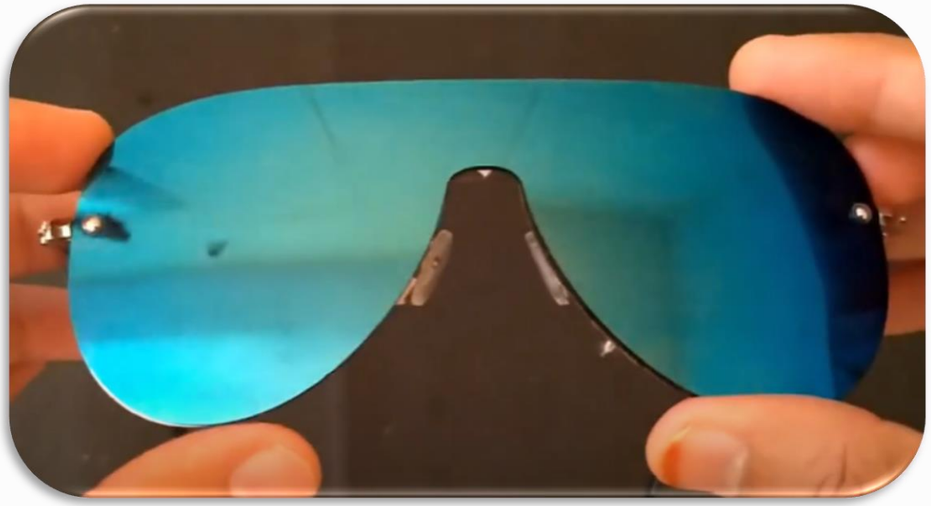
طراحی فیزیکی(1)

نیازمند بدنه عینکی سبک وزن .این طراحی شامل محل قرارگیری LEDها در جلوی عینک است .طراحی به گونه ای است که عینک سبک و راحت باشد.

اتصال ال ای دی ها به هم



بدنه عینک سبک وزن



طراحی فیزیکی(2)

هر دو قسمت را با همدیگه ادغام کرده و ال ای دی هارا با چسب به جلوی شیشه عینک متصل میکنیم

کد آردوینو

1

تنظیمات اولیه

شامل تعریف پین ها، تنظیم سریال و اضافه کردن LEDها

2

حلقه اصلی

خواندن دستورات از بلوتوث و اجرای انیمیشن مربوطه

3

توابع انیمیشن

توابع مختلف برای ایجاد الگوهای نوری متنوع

کد آردوینو

```
#include <FastLED.h>
#define LED_PIN     3
#define NUM_LEDS    88
#define LED_TYPE    WS2812B
#define COLOR_ORDER RGB
#define BRIGHTNESS  255
#define UPDATES_PER_SECOND 100

#define kMatrixWidth  17
#define kMatrixHeight 6
#define kMatrixSerpentineLayout  true

CRGBArray<NUM_LEDS> leds;

uint32_t x,y,v_time,hue_time,hxy;
uint8_t octaves=1;
uint8_t hue_octaves=3;
int xscale=57771;
int yscale=57771;
int hue_scale=1;
int time_speed=1111;
int hue_speed=31;
int x_speed=331;
int y_speed=1111;

CRGBPalette16 currentPalette;
TBlendType    currentBlending;
extern CRGBPalette16 myRedWhiteBluePalette;
extern const TProgmemPalette16 myRedWhiteBluePalette_p PROGMEM;

```

توضیحات

- LED_PIN: شماره پین متصل به LED ها (در اینجا پین 3)
- NUM_LEDS: تعداد کل LED ها (در اینجا 88)
- LED_TYPE: نوع LED ها (در اینجا WS2812B)
- COLOR_ORDER: ترتیب رنگ‌ها (در اینجا RGB)
- BRIGHTNESS: شدت نور LED ها (در اینجا 255)
- UPDATES_PER_SECOND: تعداد بروزرسانی‌ها در ثانیه (در اینجا 100)

کد آردوینو

```
void setup()
{
    Serial.begin(9600);
    delay(1000);
   FastLED.addLeds<NEOPIXEL,LED_PIN>(leds, NUM_LEDS);
    FastLED.setBrightness(BRIGHTNESS);

    currentPalette = RainbowColors_p;
    currentBlending = LINEARBLEND;

  random16_set_seed(8934);
  random16_add_entropy(analogRead(3));
    hxy = (uint32_t)((uint32_t)random16() << 16) + (uint32_t)random16();
  x = (uint32_t)((uint32_t)random16() << 16) + (uint32_t)random16();
  y = (uint32_t)((uint32_t)random16() << 16) + (uint32_t)random16();
  v_time = (uint32_t)((uint32_t)random16() << 16) + (uint32_t)random16()
  hue_time = (uint32_t)((uint32_t)random16() << 16) + (uint32_t)random16
}

void fadeall() { for(int i = 0; i < NUM_LEDS; i++) { leds[i].nscale8(250
```

توضیحات

1. setup():

- در این بخش، تنظیمات اولیه انجام می‌شود.
- Serial.begin(9600): ارتباط سریال با سریال مانیتور را با سرعت 9600 بیت بر ثانیه برقرار می‌کند.
- delay(1000): یک تاخیر ۱ ثانیه‌ای انجام می‌دهد.
- FastLED.addLeds<NEOPIXEL,LED_PIN>(leds, NUM_LEDS): LEDها را به کتابخانه FastLED معرفی می‌کند.
- FastLED.setBrightness(BRIGHTNESS): شدت نورLED ها را تنظیم می‌کند.
- currentPalette = RainbowColors_p: یک پالت رنگی انتخاب می‌کند (در اینجا پالت رنگ‌های رنگین‌کمان).
- currentBlending = LINEARBLEND: نوع ترکیب رنگ‌ها را تعیین می‌کند.
- تعدادی متغیر تصادفی برای تولید نویز و تغییرات مکانی و زمانیLED ها تعریف می‌شوند.

2. fadeall()

- این تابع تمامLED ها را به آرامی خاموش می‌کند.

3. char ch

کد آردوینو

```
54  char ch;
55  void loop()
56  {
57    FastLED.clear();
58    FastLED.show();
59    if(Serial.available())
60      {
61        ch=Serial.read();
62        Serial.println(ch);
63         switch (ch)
64          {
```

توضیحات

1. FastLED.clear(); : این دستور تمامLED ها را خاموش می‌کند.
2. FastLED.show(); : این دستور تغییرات را بهLED ها اعمال می‌کند و آن‌ها را روشن می‌کند.
3. if(Serial.available()) : این شرط بررسی می‌کند که آیا داده‌ای از پورت سریال دریافت شده است یا نه.
4. ch=Serial.read(); : این دستور یک کاراکتر از پورت سریال می‌خواند و آن را در متغیر ch ذخیره می‌کند.
5. Serial.println(ch); : این دستور مقدار ch را به پورت سریال ارسال می‌کند تا در مانیتور سریال نمایش داده شود.
6. switch (ch) : این دستور یک ساختار تصمیم‌گیری switch-case است که بر اساس مقدارch ، عملیات مختلفی را انجام می‌دهد

* سپس میرویم سراغ تعریف کیس ها که همان انواع انیمیشن های نوری هستند

کد آردوینو

Case 1

```
 case '1':
        for(int i=0;i<10;i++)
        {
          leds[37] = CRGB(255, 0, 0);
          leds[44] = CRGB(255, 0, 0);
          FastLED.show();
          delay(90);
          FastLED.clear();
          leds[20] = CRGB(255, 0, 0);
leds[30] = CRGB(255, 0, 0);
leds[36] = CRGB(255, 0, 0);
leds[38] = CRGB(255, 0, 0);
leds[43] = CRGB(255, 0, 0);
leds[45] = CRGB(255, 0, 0);
leds[51] = CRGB(255, 0, 0);
leds[58] = CRGB(255, 0, 0);
FastLED.show();
delay(90);
FastLED.clear();

leds[19] = CRGB(255, 0, 0);
leds[21] = CRGB(255, 0, 0);
leds[29] = CRGB(255, 0, 0);
leds[31] = CRGB(255, 0, 0);
leds[35] = CRGB(255, 0, 0);
leds[36] = CRGB(255, 0, 0);
leds[37] = CRGB(255, 0, 0);
leds[38] = CRGB(255, 0, 0);
leds[39] = CRGB(255, 0, 0);
leds[42] = CRGB(255, 0, 0);
leds[43] = CRGB(255, 0, 0);
leds[44] = CRGB(255, 0, 0);
leds[45] = CRGB(255, 0, 0);
leds[46] = CRGB(255, 0, 0);
leds[50] = CRGB(255, 0, 0);
leds[51] = CRGB(255, 0, 0);
leds[52] = CRGB(255, 0, 0);
leds[57] = CRGB(255, 0, 0);
leds[58] = CRGB(255, 0, 0);
leds[59] = CRGB(255, 0, 0);
leds[65] = CRGB(255, 0, 0);
leds[72] = CRGB(255, 0, 0);
FastLED.show();

delay(90);
FastLED.clear();

leds[1] = CRGB(255, 0, 0);
leds[2] = CRGB(255, 0, 0);
leds[4] = CRGB(255, 0, 0);
leds[5] = CRGB(255, 0, 0);
leds[6] = CRGB(0, 0, 0);
leds[11] = CRGB(255, 0, 0);
leds[12] = CRGB(255, 0, 0);
leds[14] = CRGB(255, 0, 0);
leds[15] = CRGB(255, 0, 0);
leds[17] = CRGB(255, 0, 0);
leds[18] = CRGB(255, 0, 0);
leds[19] = CRGB(255, 0, 0);
leds[20] = CRGB(255, 0, 0);
leds[21] = CRGB(255, 0, 0);
leds[22] = CRGB(255, 0, 0);
leds[23] = CRGB(255, 0, 0);
leds[27] = CRGB(255, 0, 0);
leds[28] = CRGB(255, 0, 0);
leds[29] = CRGB(255, 0, 0);
leds[30] = CRGB(255, 0, 0);
leds[31] = CRGB(255, 0, 0);
leds[32] = CRGB(255, 0, 0);
leds[33] = CRGB(255, 0, 0);
leds[34] = CRGB(255, 0, 0);
leds[35] = CRGB(255, 0, 0);
leds[36] = CRGB(255, 0, 0);
leds[37] = CRGB(255, 0, 0);
leds[38] = CRGB(255, 0, 0);
leds[39] = CRGB(255, 0, 0);
leds[40] = CRGB(255, 0, 0);
leds[41] = CRGB(255, 0, 0);
leds[42] = CRGB(255, 0, 0);
leds[43] = CRGB(255, 0, 0);
leds[44] = CRGB(255, 0, 0);
leds[45] = CRGB(255, 0, 0);
leds[46] = CRGB(255, 0, 0);
leds[47] = CRGB(255, 0, 0);
leds[49] = CRGB(255, 0, 0);
leds[50] = CRGB(255, 0, 0);
leds[51] = CRGB(255, 0, 0);
leds[52] = CRGB(255, 0, 0);
leds[53] = CRGB(255, 0, 0);
leds[56] = CRGB(255, 0, 0);
leds[57] = CRGB(255, 0, 0);
leds[58] = CRGB(255, 0, 0);
leds[59] = CRGB(255, 0, 0);
leds[60] = CRGB(255, 0, 0);
leds[64] = CRGB(255, 0, 0);
leds[65] = CRGB(255, 0, 0);
leds[66] = CRGB(255, 0, 0);
leds[71] = CRGB(255, 0, 0);
leds[72] = CRGB(255, 0, 0);
leds[73] = CRGB(255, 0, 0);
leds[79] = CRGB(255, 0, 0);
leds[84] = CRGB(255, 0, 0);
FastLED.show();
delay(90);
FastLED.clear();

leds[19] = CRGB(255, 0, 0);
leds[21] = CRGB(255, 0, 0);
leds[29] = CRGB(255, 0, 0);
leds[31] = CRGB(255, 0, 0);
leds[35] = CRGB(255, 0, 0);
leds[36] = CRGB(255, 0, 0);
leds[37] = CRGB(255, 0, 0);
leds[38] = CRGB(255, 0, 0);
leds[39] = CRGB(255, 0, 0);
leds[42] = CRGB(255, 0, 0);
leds[43] = CRGB(255, 0, 0);
leds[44] = CRGB(255, 0, 0);
leds[45] = CRGB(255, 0, 0);
leds[46] = CRGB(255, 0, 0);
leds[50] = CRGB(255, 0, 0);
leds[51] = CRGB(255, 0, 0);
leds[52] = CRGB(255, 0, 0);
leds[57] = CRGB(255, 0, 0);
leds[58] = CRGB(255, 0, 0);
leds[59] = CRGB(255, 0, 0);
leds[65] = CRGB(255, 0, 0);
leds[72] = CRGB(255, 0, 0);
FastLED.show();

delay(90);
FastLED.clear();

leds[20] = CRGB(255, 0, 0);
leds[30] = CRGB(255, 0, 0);
leds[36] = CRGB(255, 0, 0);
leds[38] = CRGB(255, 0, 0);
leds[43] = CRGB(255, 0, 0);
leds[45] = CRGB(255, 0, 0);
leds[51] = CRGB(255, 0, 0);
leds[58] = CRGB(255, 0, 0);
FastLED.show();
delay(90);
FastLED.clear();
        }
break;
```

کد آردوینو

Case 2

```
case '2':
for(int i=0;i<5;i++)
{
leds[36] = CRGB(255, 0, 187);
leds[37] = CRGB(255, 0, 187);
leds[44] = CRGB(255, 0, 187);
leds[45] = CRGB(255, 0, 187);
leds[50] = CRGB(255, 0, 187);
leds[51] = CRGB(255, 0, 187);
leds[58] = CRGB(255, 0, 187);
leds[59] = CRGB(255, 0, 187);
FastLED.show();
delay(90);
FastLED.clear();

leds[19] = CRGB(255, 0, 187);
leds[20] = CRGB(255, 0, 187);
leds[30] = CRGB(255, 0, 187);
leds[31] = CRGB(255, 0, 187);
leds[35] = CRGB(255, 0, 187);
leds[38] = CRGB(255, 0, 187);
leds[43] = CRGB(255, 0, 187);
leds[46] = CRGB(255, 0, 187);
leds[49] = CRGB(255, 0, 187);
leds[52] = CRGB(255, 0, 187);
leds[57] = CRGB(255, 0, 187);
leds[60] = CRGB(255, 0, 187);
leds[64] = CRGB(255, 0, 187);
leds[65] = CRGB(255, 0, 187);
leds[72] = CRGB(255, 0, 187);
leds[73] = CRGB(255, 0, 187);
FastLED.show();
delay(90);
FastLED.clear();

leds[0] = CRGB(0, 0, 0);
leds[1] = CRGB(255, 0, 187);
leds[2] = CRGB(255, 0, 187);
leds[3] = CRGB(255, 0, 187);
leds[4] = CRGB(255, 0, 187);
leds[12] = CRGB(255, 0, 187);
leds[13] = CRGB(255, 0, 187);
leds[14] = CRGB(255, 0, 187);
leds[15] = CRGB(255, 0, 187);
leds[17] = CRGB(255, 0, 187);
leds[22] = CRGB(255, 0, 187);
leds[28] = CRGB(255, 0, 187);
leds[33] = CRGB(255, 0, 187);
leds[34] = CRGB(255, 0, 187);
leds[39] = CRGB(255, 0, 187);
leds[42] = CRGB(255, 0, 187);
leds[47] = CRGB(255, 0, 187);
leds[48] = CRGB(255, 0, 187);
leds[53] = CRGB(255, 0, 187);
leds[56] = CRGB(255, 0, 187);
leds[61] = CRGB(255, 0, 187);
leds[62] = CRGB(255, 0, 187);
leds[67] = CRGB(255, 0, 187);
leds[70] = CRGB(255, 0, 187);
leds[75] = CRGB(255, 0, 187);
leds[77] = CRGB(255, 0, 187);
leds[78] = CRGB(255, 0, 187);
leds[79] = CRGB(255, 0, 187);
leds[80] = CRGB(255, 0, 187);
leds[83] = CRGB(255, 0, 187);
leds[84] = CRGB(255, 0, 187);
leds[85] = CRGB(255, 0, 187);
leds[86] = CRGB(255, 0, 187);
FastLED.show();
delay(90);
FastLED.clear();

leds[0] = CRGB(0, 0, 0);
leds[1] = CRGB(255, 0, 187);
leds[2] = CRGB(255, 0, 187);
leds[3] = CRGB(255, 0, 187);
leds[4] = CRGB(255, 0, 187);
leds[12] = CRGB(255, 0, 187);
leds[13] = CRGB(255, 0, 187);
leds[14] = CRGB(255, 0, 187);
leds[15] = CRGB(255, 0, 187);
leds[17] = CRGB(255, 0, 187);
leds[22] = CRGB(255, 0, 187);
leds[28] = CRGB(255, 0, 187);
leds[33] = CRGB(255, 0, 187);
leds[34] = CRGB(255, 0, 187);
leds[39] = CRGB(255, 0, 187);
leds[42] = CRGB(255, 0, 187);
leds[47] = CRGB(255, 0, 187);
leds[48] = CRGB(255, 0, 187);
leds[53] = CRGB(255, 0, 187);
leds[56] = CRGB(255, 0, 187);
leds[61] = CRGB(255, 0, 187);
leds[62] = CRGB(255, 0, 187);
leds[67] = CRGB(255, 0, 187);
leds[70] = CRGB(255, 0, 187);
leds[75] = CRGB(255, 0, 187);
leds[77] = CRGB(255, 0, 187);
leds[78] = CRGB(255, 0, 187);
leds[79] = CRGB(255, 0, 187);
leds[80] = CRGB(255, 0, 187);
leds[83] = CRGB(255, 0, 187);
leds[84] = CRGB(255, 0, 187);
leds[85] = CRGB(255, 0, 187);
leds[86] = CRGB(255, 0, 187);
FastLED.show();
delay(90);
FastLED.clear();

leds[19] = CRGB(255, 0, 187);
leds[20] = CRGB(255, 0, 187);
leds[30] = CRGB(255, 0, 187);
leds[31] = CRGB(255, 0, 187);
leds[35] = CRGB(255, 0, 187);
leds[38] = CRGB(255, 0, 187);
leds[43] = CRGB(255, 0, 187);
leds[46] = CRGB(255, 0, 187);
leds[49] = CRGB(255, 0, 187);
leds[52] = CRGB(255, 0, 187);
leds[57] = CRGB(255, 0, 187);
leds[60] = CRGB(255, 0, 187);
leds[64] = CRGB(255, 0, 187);
leds[65] = CRGB(255, 0, 187);
leds[72] = CRGB(255, 0, 187);
leds[73] = CRGB(255, 0, 187);
FastLED.show();
delay(90);
FastLED.clear();

leds[36] = CRGB(255, 0, 187);
leds[37] = CRGB(255, 0, 187);
leds[44] = CRGB(255, 0, 187);
leds[45] = CRGB(255, 0, 187);
leds[50] = CRGB(255, 0, 187);
leds[51] = CRGB(255, 0, 187);
leds[58] = CRGB(255, 0, 187);
leds[59] = CRGB(255, 0, 187);
FastLED.show();
delay(90);
FastLED.clear();
}
break;
```

کد آردوینو

Case 4

```
case '4':

leds[2] = CRGB(0, 183, 255);
leds[3] = CRGB(0, 183, 255);
leds[13] = CRGB(0, 183, 255);
leds[14] = CRGB(0, 183, 255);
leds[17] = CRGB(0, 183, 255);
leds[18] = CRGB(0, 183, 255);
leds[19] = CRGB(0, 183, 255);
leds[20] = CRGB(0, 183, 255);
leds[21] = CRGB(0, 183, 255);
leds[28] = CRGB(0, 183, 255);
leds[29] = CRGB(0, 183, 255);
leds[30] = CRGB(0, 183, 255);
leds[31] = CRGB(0, 183, 255);
leds[32] = CRGB(0, 183, 255);
leds[34] = CRGB(0, 183, 255);
leds[36] = CRGB(0, 183, 255);
leds[37] = CRGB(0, 183, 255);
leds[42] = CRGB(0, 183, 255);
leds[44] = CRGB(0, 183, 255);
leds[45] = CRGB(0, 183, 255);
leds[48] = CRGB(0, 183, 255);
leds[49] = CRGB(0, 183, 255);
leds[50] = CRGB(0, 183, 255);
leds[51] = CRGB(0, 183, 255);
leds[52] = CRGB(0, 183, 255);
leds[53] = CRGB(0, 183, 255);
leds[56] = CRGB(0, 183, 255);
leds[57] = CRGB(0, 183, 255);
leds[58] = CRGB(0, 183, 255);
leds[59] = CRGB(0, 183, 255);
leds[60] = CRGB(0, 183, 255);
leds[64] = CRGB(0, 183, 255);
leds[65] = CRGB(0, 183, 255);
leds[67] = CRGB(0, 183, 255);
leds[72] = CRGB(0, 183, 255);
leds[73] = CRGB(0, 183, 255);
leds[75] = CRGB(0, 183, 255);
leds[77] = CRGB(0, 183, 255);
leds[78] = CRGB(0, 183, 255);
leds[79] = CRGB(0, 183, 255);
leds[80] = CRGB(0, 183, 255);
leds[81] = CRGB(0, 183, 255);
leds[83] = CRGB(0, 183, 255);
leds[84] = CRGB(0, 183, 255);
leds[85] = CRGB(0, 183, 255);
leds[86] = CRGB(0, 183, 255);
leds[87] = CRGB(0, 183, 255);
FastLED.show();
delay(3000);
FastLED.clear();
break;
```

کد آردوینو

Case 5

```
case '5':
for(int i=0;i<4;i++)
{
 static uint8_t hue;
  for(int i = 0; i < NUM_LEDS/2; i++) {
    // fade everything out
    leds.fadeToBlackBy(40);

    // let's set an led value
    leds[i] = CHSV(hue++,255,255);

    // now, let's first 20 leds to the top
20 leds,
    leds(NUM_LEDS/2,NUM_LEDS-
1) = leds(NUM_LEDS/2 - 1 ,0);
    FastLED.delay(33);
  }
}
  break;
```

کد آردوینو

Case 6

```
case '6':
{
  for(int i=0;i<3;i++)
  {
    static uint8_t hue = 0;
  Serial.print("x");
  for(int i = 0; i < NUM_LEDS; i++) {
    leds[i] = CHSV(hue++, 255, 255);
    FastLED.show();
    fadeall();
    delay(10);
  }
  Serial.print("x");
  for(int i = (NUM_LEDS)-1; i >= 0; i--
) {
    leds[i] = CHSV(hue++, 255, 255);
    FastLED.show();
    fadeall();
    delay(10);
  }
}
}
break;
```

functions

```
void FillLEDsFromPaletteColors(
uint8_t colorIndex)
{
    uint8_t brightness = 255;

    for( int i = 0; i < NUM_LEDS; i++) {
        leds[i] = ColorFromPalette(
currentPalette, colorIndex,
brightness, currentBlending);
        colorIndex += 3;
    }
}
```

توضیحات

این تابع با استفاده از پالت رنگی (currentPalette) رنگ‌های مختلف را به LEDها اختصاص می‌دهد. برای هر LED، از تابع ColorFromPaletteاستفاده می‌شود و مقدار colorIndexبا 3 افزایش می‌یابد.

functions

```
void ChangePalettePeriodically()
{
    uint8_t secondHand = (millis() / 1000) % 60;
    static uint8_t lastSecond = 99;

    if( lastSecond != secondHand) {
        lastSecond = secondHand;
        if( secondHand ==  0)  { currentPalette =
RainbowColors_p;         currentBlending = LINEARBLEND; }
        if( secondHand == 10)  { currentPalette =
RainbowStripeColors_p;   currentBlending = NOBLEND;  }
        if( secondHand == 15)  { currentPalette =
RainbowStripeColors_p;   currentBlending = LINEARBLEND;
}
        if( secondHand == 20)  { SetupPurpleAndGreenPalette();
currentBlending = LINEARBLEND; }
        if( secondHand == 25)  { SetupTotallyRandomPalette();
currentBlending = LINEARBLEND; }
        if( secondHand == 30)  {
SetupBlackAndWhiteStripedPalette();       currentBlending =
NOBLEND; }
        if( secondHand == 35)  {
SetupBlackAndWhiteStripedPalette();       currentBlending =
LINEARBLEND; }
        if( secondHand == 40)  { currentPalette = CloudColors_p;
currentBlending = LINEARBLEND; }
        if( secondHand == 45)  { currentPalette = PartyColors_p;
currentBlending = LINEARBLEND; }
        if( secondHand == 50)  { currentPalette =
myRedWhiteBluePalette_p; currentBlending = NOBLEND;  }
        if( secondHand == 55)  { currentPalette =
myRedWhiteBluePalette_p; currentBlending =
LINEARBLEND; }
    }
}
```

توضیحات

این تابع به صورت دوره‌ای پالت رنگی را تغییر می‌دهد. بر اساس ثانیه‌ی فعلی (secondHand)، پالت رنگی مختلفی انتخاب می‌شود.

functions

```
void SetupTotallyRandomPalette()
{
    for( int i = 0; i < 16; i++) {
        currentPalette[i] = CHSV(
random8(), 255, random8());
    }
}
```

توضیحات

این تابع یک پالت رنگی کاملاً تصادفی ایجاد می‌کند.

functions

توضیحات

```
void
SetupBlackAndWhiteStripedPalette()
{
    // 'black out' all 16 palette entries...
    fill_solid( currentPalette, 16,
CRGB::Black);
    // and set every fourth one to white.
    currentPalette[0] = CRGB::White;
    currentPalette[4] = CRGB::White;
    currentPalette[8] = CRGB::White;
    currentPalette[12] = CRGB::White;

}
```

این تابع یک پالت رنگی با خطوط سیاه و سفید ایجاد می‌کند.

functions

```
void SetupPurpleAndGreenPalette()
{
    CRGB purple = CHSV( HUE_PURPLE, 255, 255);
    CRGB green  = CHSV( HUE_GREEN, 255, 255);
    CRGB black  = CRGB::Black;

    currentPalette = CRGBPalette16(
                                green,  green,  black,  black,
                                purple, purple, black,  black,
                                green,  green,  black,  black,
                                purple, purple, black,  black );
}
```

توضیحات

این تابع یک پالت رنگی با رنگ‌های بنفش و سبز ایجاد می‌کند.

functions

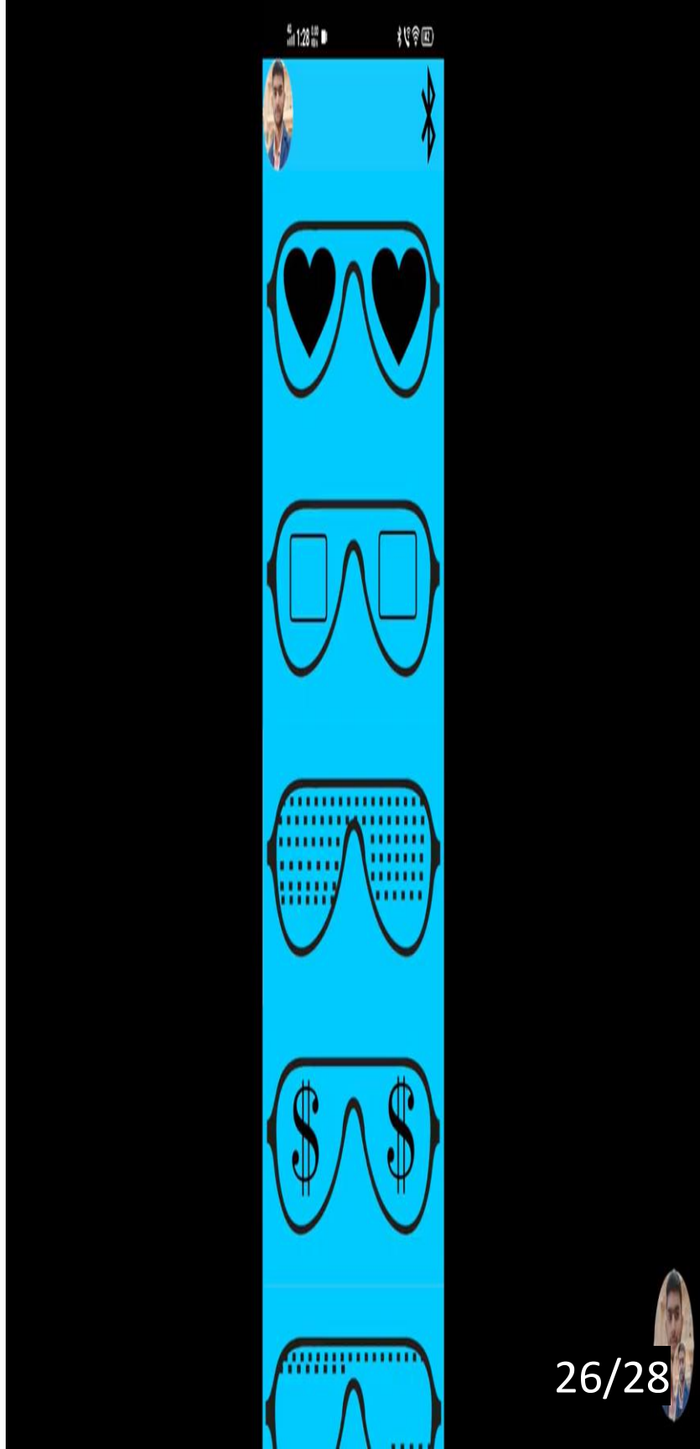
```
const TProgmemPalette16
myRedWhiteBluePalette_p PROGMEM =
{
    CRGB::Red,
    CRGB::Gray, // 'white' is too bright compared to red
and blue
    CRGB::Blue,
    CRGB::Black,

    CRGB::Red,
    CRGB::Gray,
    CRGB::Blue,
    CRGB::Black,

    CRGB::Red,
    CRGB::Red,
    CRGB::Gray,
    CRGB::Gray,
    CRGB::Blue,
    CRGB::Blue,
    CRGB::Black,
    CRGB::Black
};
```

توضیحات

این تعریف یک پالت رنگی ثابت با رنگ‌های قرمز، سفید و آبی است.

نحوه کار با برنامه موبایل

یک برنامه اندروید ساده برای کنترل عینک طراحی شده است .این برنامه از طریق بلوتوث به عینک متصل شده و امکان انتخاب انیمیشن های مختلف را فراهم می کند .کاربر می تواند با لمس دکمه های مختلف، انیمیشن دلخواه را انتخاب کرده و به عینک ارسال کند.



لینک دانلود برنامه برای موبایل

https://hacksterio.s3.amazonaws.com/uploads/attachments/1113873/kingsdiy_eDXfKj3Tk6.apk

چالش ها و راه حل ها

اتصالات الکترونیکی

استفاده از لحیم کاری دقیق و محافظت از اتصالات با چسب حرارتی

گرمای تولیدی

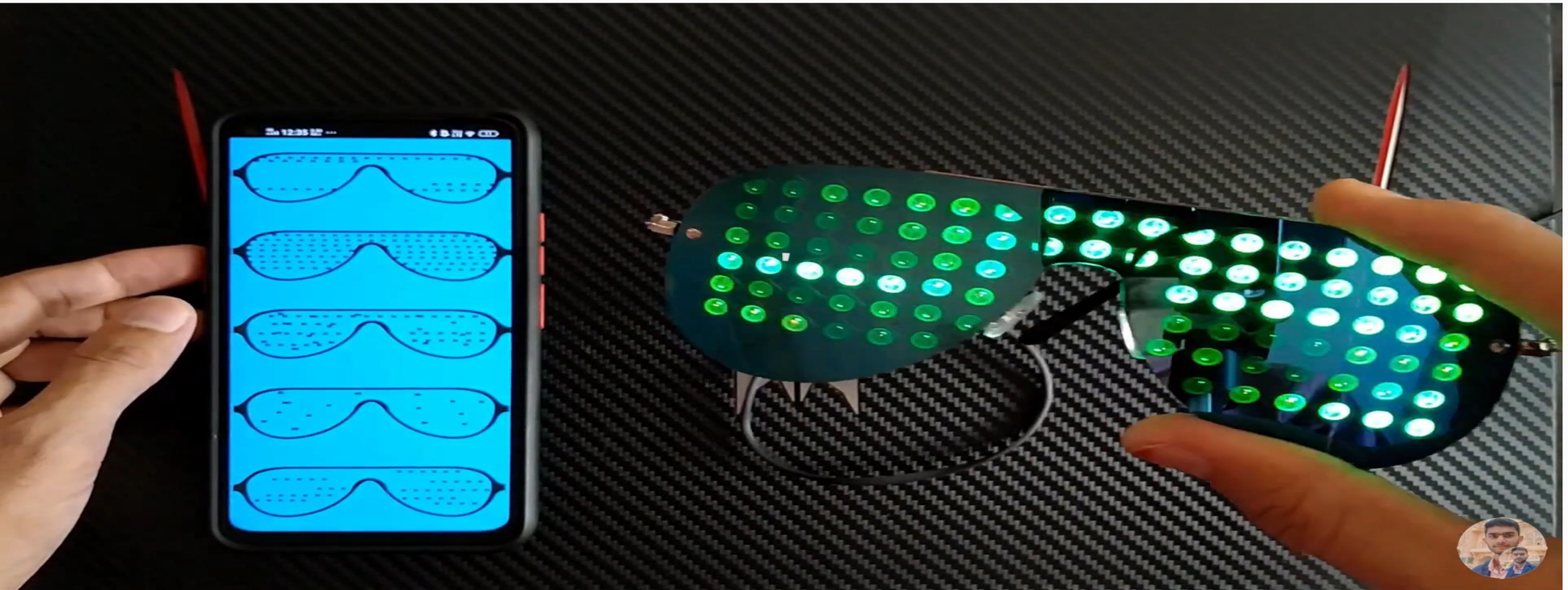
طراحی بدنه با تهویه مناسب و محدود کردن شدت نور LEDها

مصرف باتری

استفاده از LEDهای کم مصرف و بهینه سازی کد برای کاهش مصرف انرژی

نتیجه گیری و کاربردها

این پروژه نشان می دهد که چگونه می توان با استفاده از قطعات الکترونیکی ارزان قیمت، یک وسیله سرگرم کننده و جذاب ساخت .عینک RGBمی تواند در مهمانی ها، نمایش ها و حتی به عنوان یک وسیله تبلیغاتی مورد استفاده قرار گیرد .این پروژه همچنین می تواند الهام بخش ایده های دیگر برای ترکیب الکترونیک و پوشیدنی ها باشد



منابع

-Chat GPT 4o
-Copilot Microsoft AI
-gamma.app website
-Hackster.io project link is :(https://www.hackster.io/Mukesh_Sankhla/rgb-goggles-8d3ef5#overview)
-Wokwi.com aurdino code link is (https://www.hackster.io/code_files/445050/download)
-Translate.Yandex.com website
-Youtube the video URL is (https://youtu.be/yMi15hjzdFM)