```
#include <Wire.h>
#include <math.h>
#define ADDRESS 0x39
#define REG_CTL 0x80
#define REG_TIMING 0x81
#define REG_INT 0x82
#define REG_INT_SOURCE 0x83
#define REG_ID 0x84
#define REG_GAIN 0x87
#define REG_LOW_THRESH_LOW_BYTE 0x88
#define REG_LOW_THRESH_HIGH_BYTE 0x89
#define REG_HIGH_THRESH_LOW_BYTE 0x8A
#define REG_HIGH_THRESH_HIGH_BYTE 0x8B
#define REG_BLOCK_READ 0xCF
#define REG_GREEN_LOW 0xD0
#define REG_GREEN_HIGH 0xD1
#define REG_RED_LOW 0xD2
#define REG_RED_HIGH 0xD3
#define REG_BLUE_LOW 0xD4
#define REG_BLUE_HIGH 0xD5
#define REG_CLEAR_LOW 0xD6
#define REG_CLEAR_HIGH 0xD7
#define CTL_DAT_INIITIATE 0x03
#define CLR_INT 0xE0

//Timing Register
#define SYNC_EDGE 0x40
#define INTEG_MODE_FREE 0x00
#define INTEG_MODE_MANUAL 0x10
#define INTEG_MODE_SYN_SINGLE 0x20
#define INTEG_MODE_SYN_MULTI 0x30
```

```
#define INTEG_PARAM_PULSE_COUNT1 0x00
#define INTEG_PARAM_PULSE_COUNT2 0x01
#define INTEG_PARAM_PULSE_COUNT4 0x02
#define INTEG_PARAM_PULSE_COUNT8 0x03

//Interrupt Control Register
#define INTR_STOP 40
#define INTR_DISABLE 0x00
#define INTR_LEVEL 0x10
#define INTR_PERSIST_EVERY 0x00
#define INTR_PERSIST_SINGLE 0x01
//Interrupt Souce Register
#define INT_SOURCE_GREEN 0x00
#define INT_SOURCE_RED 0x01
#define INT_SOURCE_BLUE 0x10
#define INT_SOURCE_CLEAR 0x03
//Gain Register
#define GAIN_1 0x00
#define GAIN_4 0x10
#define GAIN_16 0x20
#define GANI_64 0x30
#define PRESCALER_1 0x00
#define PRESCALER_2 0x01
#define PRESCALER_4 0x02
#define PRESCALER_8 0x03
#define PRESCALER_16 0x04
#define PRESCALER_32 0x05
#define PRESCALER_64 0x06
int readingdata[20];
int i,green,red,blue,clr,ctl;
double X,Y,Z,x,y,z,b1,b2;
```

```
void setup()
{
 Serial.begin(9600);
 Wire.begin(); // join i2c bus (address optional for master)
   //Bolt Rx-Tx
 pinMode(13, OUTPUT);
}
void set_timing_reg(int x)
{
 Wire.beginTransmission(ADDRESS);
 Wire.write(REG_TIMING);
 Wire.write(x);
 Wire.endTransmission();
 delay(100);
}
void set_interrupt_source_reg(int x)
{
 Wire.beginTransmission(ADDRESS);
 Wire.write(REG_INT_SOURCE);
 Wire.write(x);
 Wire.endTransmission();
 delay(100);
}
void set_interrupt_control_reg(int x)
{
 Wire.beginTransmission(ADDRESS);
 Wire.write(REG_INT);
 Wire.write(x);
 Wire.endTransmission();
 delay(100); }
void set_gain(int x) {
```

```
 Wire.beginTransmission(ADDRESS);
 Wire.write(REG_GAIN);
 Wire.write(x);
 Wire.endTransmission(); }
void set_ADC_EN() {
 Wire.beginTransmission(ADDRESS);
 Wire.write(REG_CTL);
 Wire.write(CTL_DAT_INIITIATE);
 Wire.endTransmission();
 delay(100); }
void clr_interrupt() {
 Wire.beginTransmission(ADDRESS);
 Wire.write(CLR_INT);
 Wire.endTransmission(); }
void read_RGB()
{
 Wire.beginTransmission(ADDRESS);
 Wire.write(REG_BLOCK_READ);
 Wire.endTransmission();
 Wire.beginTransmission(ADDRESS);
 Wire.requestFrom(ADDRESS,8);
 delay(500);
 if(8<= Wire.available()) // if two bytes were received
 {
 for(i=0;i<8;i++)
 {
 readingdata[i]=Wire.read();
 //Serial.println(readingdata[i],BIN);
 }
 }
 green=readingdata[1]*256+readingdata[0];
```

```
red=readingdata[3]*256+readingdata[2];
blue=readingdata[5]*256+readingdata[4];

if (green>red && green>blue)
{
if (red>blue)
{
green = 1.1*green - 0.1*green; red = .85*red; blue = .65*blue;
}
else
{
green = 1.1*green - 0.1*green; red = .65*red; blue = .85*blue;
}
}

else if (red>green && red>blue)
{
if (green>blue)
{
red = 1.1*red; green = .85*green - 0.1*green; blue = .65*blue;
}
else
{
red = 1.2*red; green = .65*green - 0.1*green; blue = .85*blue;
}
}

else
{
if (red>green)
{
```

```
 blue = 1.2*blue; red = .85*red; green = .65*green - 0.1*green;
 }
 else
 {
 blue = 1.2*blue; red = .65*red; green = .85*green - 0.1*green;
 }
 }
 clr=readingdata[7]*256+readingdata[6];
 Serial.println("The RGB value and Clear channel value are");
 //}
Serial.println(red,DEC);
 Serial.println(green,DEC);
 Serial.println(blue,DEC);
 Serial.println(clr,DEC);
}
void calculate_xy()
{
 X=(2.768830875)*red+(1.751709329)*green+(1.130135051)*blue;
 Y=(1)*red+(4.590608578)*green+(0.060066678)*blue;
 Z=(0)*red+(0.056506753)*green+(5.594168503)*blue;
 x=X/(X+Y+Z);
 y=Y/(X+Y+Z);
 if((X>0)&&(Y>0)&&(Z>0))
 {
 Serial.println("The x,y value is");
 Serial.println(x,2);
 Serial.println(y,2);
 }
else {
Serial.println("Error,the value overflow");
}
```

```
}
void loop()
{
 set_timing_reg(INTEG_MODE_FREE);//Set trigger mode.Including free mode,manually mode,single synchronizition jjmode or so.
 set_interrupt_source_reg(INT_SOURCE_GREEN); //Set interrupt source
 set_interrupt_control_reg(INTR_LEVEL|INTR_PERSIST_EVERY);//Set interrupt mode
  set_gain(GAIN_1|PRESCALER_4);//Set gain value and prescaler value
 set_ADC_EN();//Start ADC
 while(1)
{
 read_RGB();
 calculate_xy();
 delay(100);
 clr_interrupt();
 if(x < 0.46&&x > 0.42&&y < 0.41&&y > 0.37){
  b1 = 1;
  digitalWrite(13,HIGH);
 }
 else {
  b1 = 0;
  digitalWrite(13,LOW);
 }

 }
}
```