EFY TESTED S.C. DWIVEDI

# Arduino Based RGB COLOUR CODE GENERATOR

A. SAMIUDDHIN

Red, green and blue (RGB) are the basic colours for generating various other colours by mixing colours in a particular proportion. This technique is used in TVs, mobiles etc. Similarly, Web designing using HTML or any other language requires the hex code of a particular colour in the design. Presented here is the circuit for learning the concept of generating colours and the respective hex codes of those colours.

## Circuit and working

Circuit diagram of the RGB colour code generator is shown in Fig. 1. It is built around Arduino Uno board (Board1), 16x2 LCD (LCD1), RGB LED (RGB1) and a few other components. Arduino board is the brain of the circuit that performs colour generation using pulse width modulation (PWM). Three potentiometers control intensity of the RGB LED. The 16x2 LCD wired in 4-bit mode is used to display the value of each colour in the first row and the hex value in the second row.

Intensity of each colour is controlled by potmeters VR1, VR2 and VR3. These are connected to analogue inputs A0, A1 and A2 of Arduino board. The program continuously scans the analogue input of the board. The analogue-to-digital converter (ADC) of Arduino converts the analogue value into a 10-bit digital value. Since resolution of the PWM module of the board is 8-bit, the 10-bit analogue value is converted into 8-bit value by the program. Screenshot of the source program is shown in Fig. 2.

Thus, the 8-bit value controls

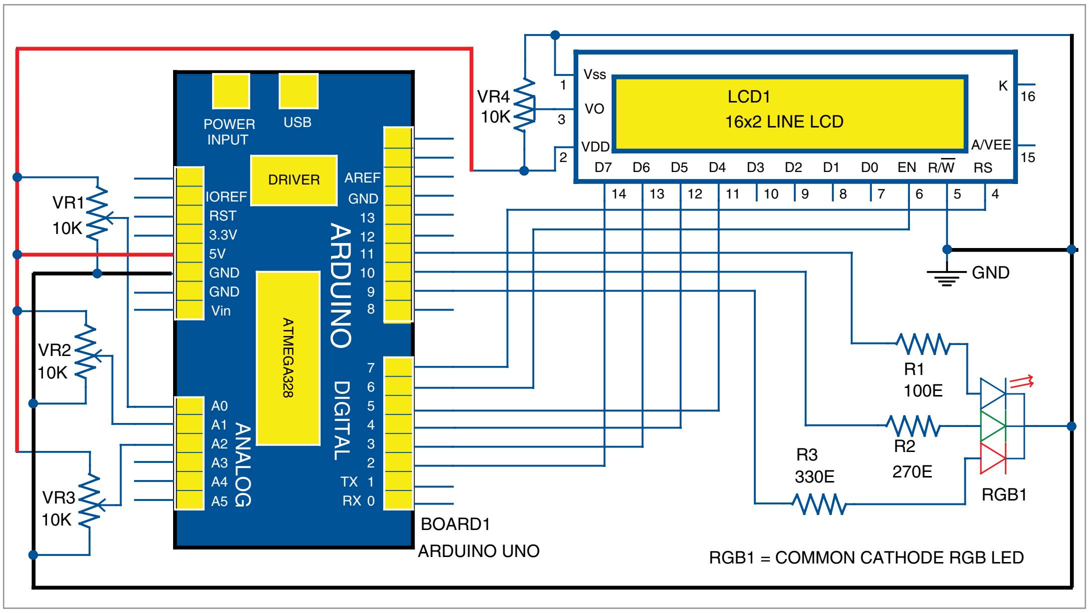
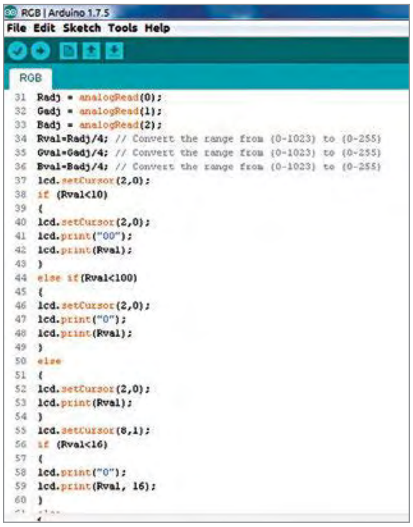
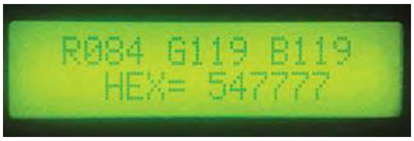


Fig. 1: Circuit of the RGB colour code generator



```
Radj = analogRead(0);
Gadj = analogRead(1);
Badj = analogRead(2);
Rval=Radj/4; // Convert the range from (0-1023) to (0-255)
Gval=Gadj/4; // Convert the range from (0-1023) to (0-255)
Bval=Badj/4; // Convert the range from (0-1023) to (0-255)
lcd.setCursor(2,0);
if (Rval<10)
{
lcd.setCursor(2,0);
lcd.print("00");
lcd.print(Rval);
}
else if(Rval<100)
{
lcd.setCursor(2,0);
lcd.print("0");
lcd.print(Rval);
}
else
{
lcd.setCursor(2,0);
lcd.print(Rval);
}
lcd.setCursor(8,1);
if (Rval<16)
{
lcd.print("0");
lcd.print(Rval, 16);
}
```

Fig. 2: Screenshot of the source code



```
R084 G119 B119
  HEX= 547777
```

Fig. 3: RGB colour display on the LCD

**PARTS LIST**

| | |
|---|---|
| *Semiconductors:* | |
| Board1 | - Arduino Uno board |
| LCD1 | - 16×2 LCD |
| RGB1 | - Common-cathode RGB LED |
| *Resistors (all 1/4-watt, ±5% carbon):* | |
| R1 | - 100-ohm |
| R2 | - 270-ohm |
| R3 | - 330-ohm |
| VR1-VR3 | - 10-kilo-ohm potmeter |
| VR4 | - 10-kilo-ohm preset |
| *Miscellaneous:* | |
| CON1 | - 5-pin connector |
| CON2 | - 9-pin connector |

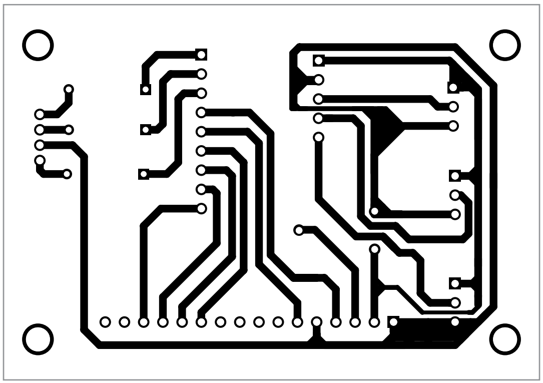


Fig. 4: PCB pattern of the RGB colour code generator

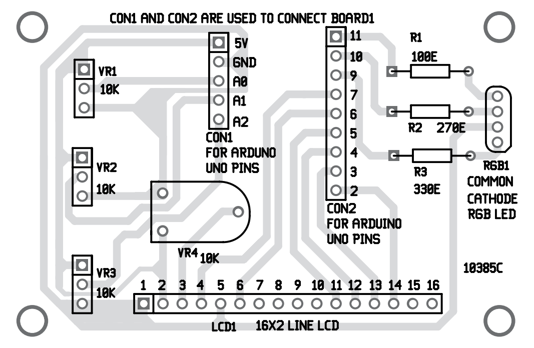


Fig. 5: Component layout of the PCB

the duty cycle of the PWM, thereby controlling the intensity of the colour. The RGB LED emits the light of a colour, which is controlled by the potmeters (VR1 through VR3). The corresponding colour value is displayed in the first row as Rxxx Gxxx Bxxx, where xxx are the numbers for RGB colours, respectively. The second line displays the hex value of the colour as HEXxxxxxx. RGB colours with numbers are displayed on the LCD as shown in Fig. 3.

Resistor values (R1, R2 and R3) are critical. Choose these values after calibration. For calibration, connect each lead of the RGB LED in series with a 1-kilo-ohm potmeter to 5V. Vary the potmeter to low-resistance side until the RGB LED emits a slightly whitish colour. Measure resistance from the arm of the potmeter to the anode of the LED and note down the resistance value, say, 100-ohm. Using the same technique, measure the remaining two leads of the LED. These resistance values were found to be 270-ohm and 330-ohm, respectively, during testing.

Be careful while calibrating as there is a chance of burning the potmeter when the arm moves to very low-resistance side due to more current passing through the arm.

## Construction and testing

An actual-size, single-side PCB for the RGB colour code generator is shown in Fig. 4 and its component layout in Fig. 5. Assemble the circuit on a designed PCB. Connectors CON1 and CON2 in the PCB layout are used to connect Board1 (Arduino UNO board) through external connectors. Arduino board can be powered by an external 9V, 500mA adaptor or USB cable.

Software for the RGB generator is written in Arduino programming language. Arduino UNO is programmed using Arduino IDE. Select the correct board from Board→Tools menu in Arduino IDE, select COM port and upload the program (RGB.ino) through the standard USB port in computer.

***Note.*** For backlight, connect pin 15 of LCD1 to +5V via a 220-ohm resistor and pin 16 to GND. EFY

**EFY Note**

The source code of this project is included in this month's EFY DVD and is also available for free download at *source.efymag.com*

A. Samiuddhin is B.Tech in electrical and electronics engineering. His interests include LED lighting, power electronics, microcontrollers and Arduino programming