Rainbow Cube

From Wiki 来自痴汉的爱

In this box, you’ ll find everything you need to build a ready to shine **Rainbow Cube**!
Have some fun and start creating a masterpiece today with this 4x4x4 RGB LED cube kit from Seeed Studio: The “Rainbow Cube – Ready to Shine” requires no soldering and comes pre-assembled with a Rainbowduino (Arduino-compatible) LED driver controller. Just plug it in to your PC or Mac, grab a copy of the free design software, and you’ ll discover just how easy it is to program this spectacular device and see the results in real time.
The Pack consist of:

- 1 x Rainbow Cube Kit (Assembled) (http://www.seeedstudio.com/wiki/Rainbow_Cube_kit_-_RGB_4*4*4_%28Rainbowduino_Compatible%29)
- 1 x Rainbowduino v3.0
- 1 x Power adaptor
- 1 x USB cable

Seeed Studio has made a video to show how to use Rainbow Cube and program to it. Watch the video, you'll know how cool this product is.

- Curing program makes Rainbow Cube Kit PnP
- Diaphanous Acrylic sheet makes the shining LEDs looks so cool
- Programmable property let the Rainbow Cube shining in your style

Here is the link in YouTobe: http://youtu.be/v44i73zmwA4

Great thanks to Riley Porter @Synthetos, for the excellent job of Acrylic Case design.

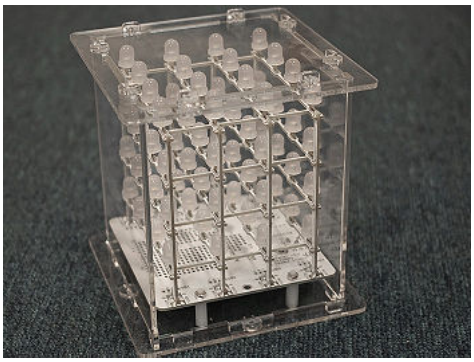
Contents

Rainbow Cube Kit (Assembled)

Rainbow Cube is a **3D RGB** LED Cube useful for creating colorful design. The 3D Cube is artistically crafted with sixty-four 8mm **RGB** LEDs arranged in a 4 x 4 x 4 manner. **Rainbow Cube** can be used to create many beautiful visual effects with A **Rainbowduino**. The **Rainbow Cube** comes with an inbuilt 3.3V / 1 Amp LDO useful for powering the independently. A XBee compatible socket is provided as well, this can be used to connect Rainbowduino with a PC or an Arduino wirelessly.

Model: KIT101E1P (http://www.seeedstudio.com/depot/rainbow-cube-kit-rgb-4x4x4-rainbowduino-compatible-p-596.html?cPath=100)

cubeani.gif



Features

- **8mm RGB** diffused LEDs.
- 4 x 4 x 4 arrangement
- **XBee** compatible **socket**
- Provides a **3.3V/1 Ampere** LDO for power the Cube with unregulated DC 6-9V. Useful when not powered by USB.
- Provides 8 Red, 8 Green and 8 Blue common cathode pins along with 8 Vcc pins in a **2 x 16 header pin**.
    - Controllable with a 8 x 8R,8G,8B multiplexed PWM LED driver like **Rainbowduino**.

Application Ideas

- Colorful LED display : Mix various intensities of RED, GREEN and BLUE channels to produce different colors
- Bright mood Lamp / Night lamp
- Useful for artistic application.

Schematic

- Rainbow Cube is made up of two parts :
    - Panel (board) which supports the cube structure.
    - The Cube structure (8 slim pillars, 8 side panels).

- Click here to download Rainbow Cube
    - Complete schematic in PDF (http://seeedstudio.com/wiki/images/d/dc/Rainbow_Cube_Kit_-_RGB_4x4x4_LED_schematic_board.pdf)
    - Eagle CAD files (http://seeedstudio.com/wiki/images/b/b1/RainbowCube_v1.1_Panel_v1.2_Eaglefiles.zip)

Specification

Operating Voltage

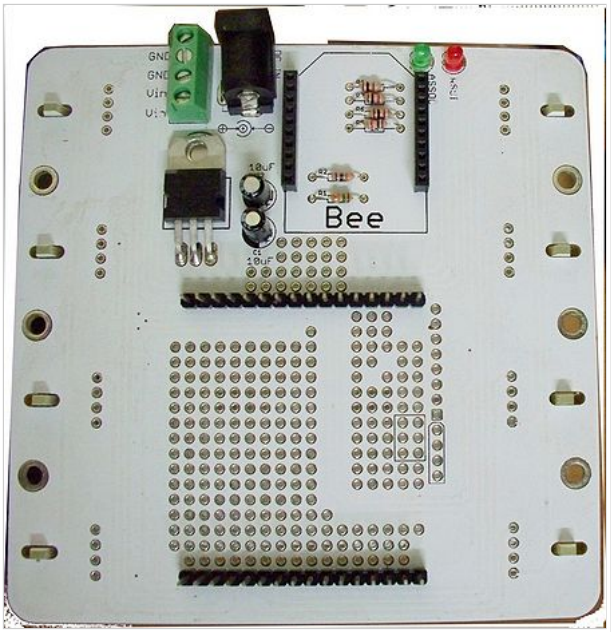
- 3.3V

LEDs

- **8mm** common anode RGB LED.
- **4 Leads** (Blue-shortest lead,Green, Anode-longest lead, Red).
- **Max forward current** IF = **20mA**.

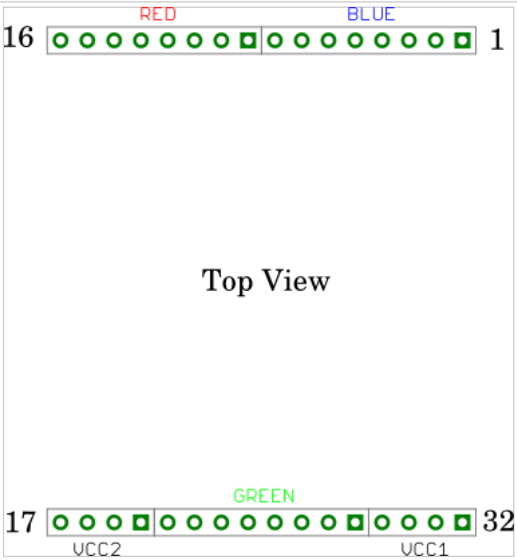
Pin definition and Rating

All pins are accessible from the Panel board show below.



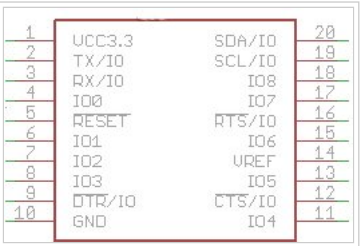
Panel board (bottom view)

- **Rainbow Cube** provides **2 x 16 pin header** for connecting to RGB LEDs driver board like **Rainbowduino**.



2 x 16 pin header connector

- **xBee Socket**



2 x 16 pin header connector

- **DC Jack Pin**
  - Middle Pin VIN (6-9DCV)
  - Side barrel: GND
- 4 pin **Green terminal**.
  - 2 GND pins, 2 VIN pins

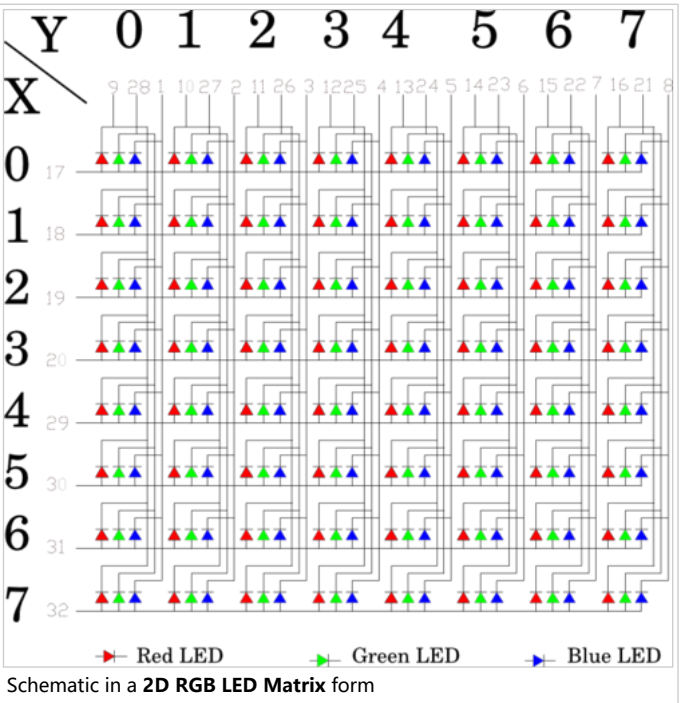
Mechanical Dimensions

Assembled cube is of approx. of 10cm (l) X 10cm (b) X 12cm (h) dimension.

Understanding the Schematic

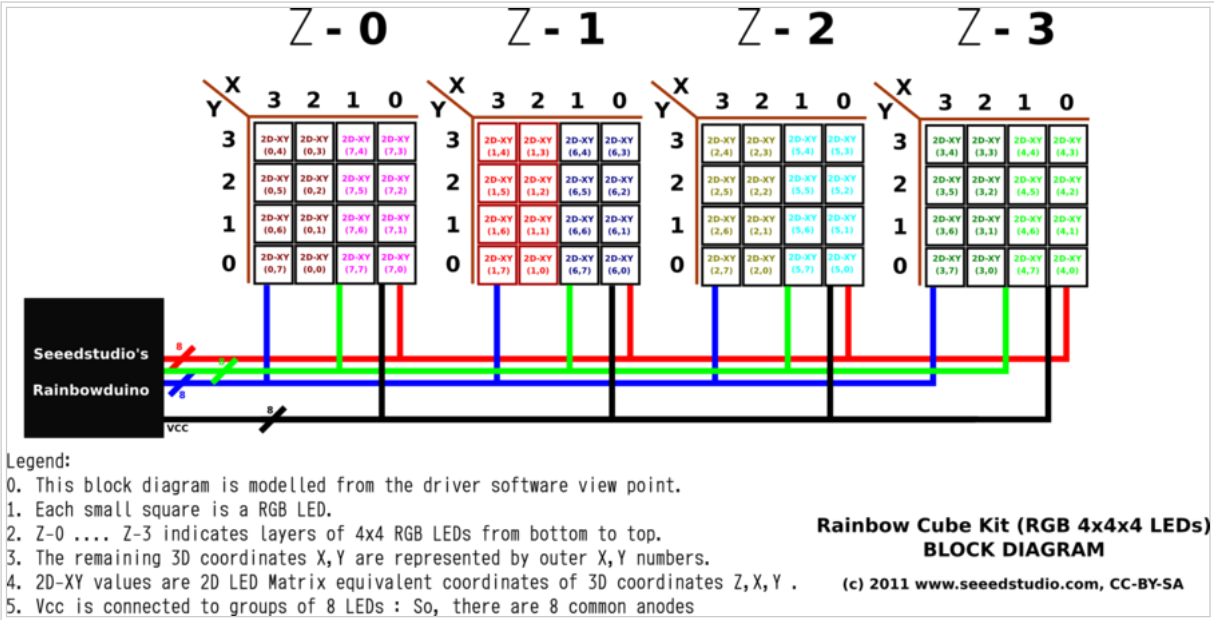
To easily understand the working of **Rainbow Cube**, a very simplified schematic is presented below. In essence, 64 RGB LEDs are arranged in a form consisting of 8 common anodes(positive pins) and 8 common cathodes(gtound pins) for each color Red, Green and Blue.

- The complete schematic of **RGB Cube** is represented in a **2D RGB LED Matrix** form below.
  - Numbers **1 - 32** indicates the pin number of the **2x16** pin header shown above.



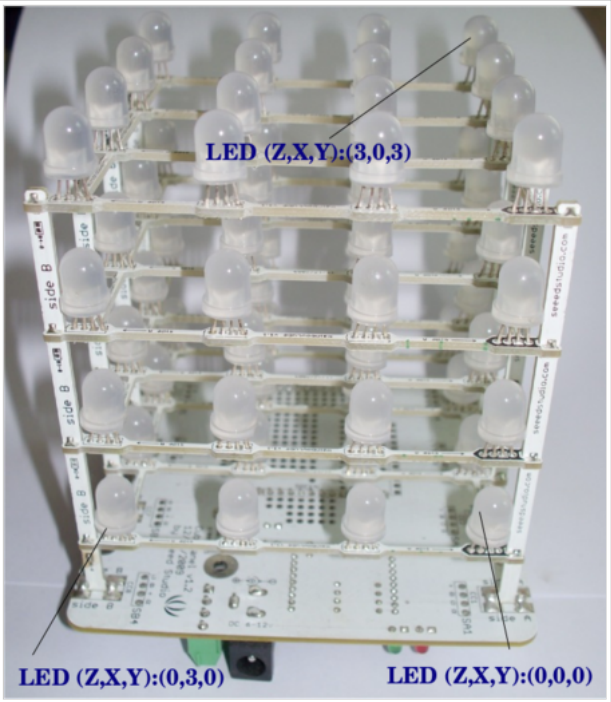
Schematic in a **2D RGB LED Matrix** form

- The **RGB Cube** inter-connection are presented in a block diagram format. This block diagram clearly shows how the 64 LEDs in 2D form are mapped into a 3D Cube form.



RGB Cube from driver software view point

- The actual 3D positions of LEDs are marked in the below photograph.



RGB Cube LED's 3D Positions illustrated

- The X,Y coordinates of **2D RGB LED Matrix** is mapped to the **RGB Cube** block diagram as follows:
  - Locate the 2D-XY coordinate (X,Y) from RGB Cube block diagram
  - Use these (X,Y) co-ordinates with (X,Y) coordiantes of RGB Matrix to know how the LED is controlled (i.e locating VCC and Cathode pins)
  - For example: LED **(Z,X,Y):(1,0,3)** 's 2D-XY is **(6,3)**. This LED's VCC is Pin 31. The R,G and B LED's cathode are connected to 12,25 and 4 pins respectively.

Rainbowduino v3.0

The Rainbowduino board is an Arduino compatible controller board with professional multiplexed LED driver. It can drive a 8x8 RGB Led Matrix or a 4x4x4 RGB LED Cube in common Anode mode. Rainbowduino v3.0 uses two MY9221 chips which is a 12-channels (R/G/B x 4) constant current Adaptive Pulse Density Modulation (APDM). Rainbowduino v3.0 has provisions for cascading more such boards with I2C interface.

Rainbowduino v3.0 is flashed with Arduino boot-loader and this makes it easy to program sketches using Arduino IDE. Unlike other LED drivers, this comes with a USB to UART (FT232RL) inbuilt for programming the sketches.

Rainbowduino v3.0

Rainbowduino v3.0 bottom

Features

- Provides 2 x 16 pin header for connecting multiplexed LEDs
- Constant current(**20.8mA**) LEDs driver.
- Can drive 4x4x4 RGB LED Cube or 8x8 RGB LED Matrix (i.e **192 LED**)
- Built in **USB to UART** chip (FT232RL)
- Built in **5V / 1 Ampere** voltage regulator

Application Ideas

- General Purpose LED driver
  - Connect 4x4x4 RGB Cube
  - Connect 8x8 RGB Matrix
- Create LED sign boards by chaining more than one Rainbowduino v3.0

Specifications

Constant current output
    20.8mA
Maximum LEDs driving capability
    192 (i.e 8x8x3)

More information, please check the Rainbowduino v3.0 datasheet in wiki.

Power Adapter

High quality switching 'wall wart' AC to DC 6.5V 2A power supply. It has a DC jack connector suitable for for Rainbowduino. Also works well with Arduino / Seeeduino. It can function with 100-240VAC inputs.

Model:TOL113C3O (http://www.seeedstudio.com/depot/wall-adapter-power-supply-65vdc-2a-p-413.html?cPath=178_181)

rainbowpower.jpg

Mini USB Cable

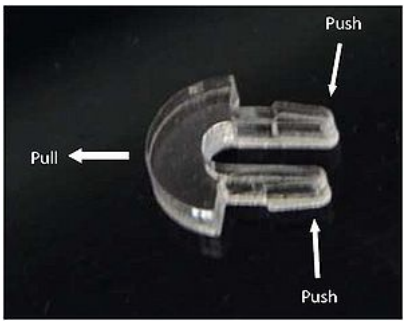
There is a 110cm Mini USB cable, with black color. It can be used to connect Rainbowduino V3.0 with your PC for programming.

Model:ARD112C5B (http://www.seeedstudio.com/depot/mini-usb-cable-110cm-p-252.html?cPath=178_179)

Usage

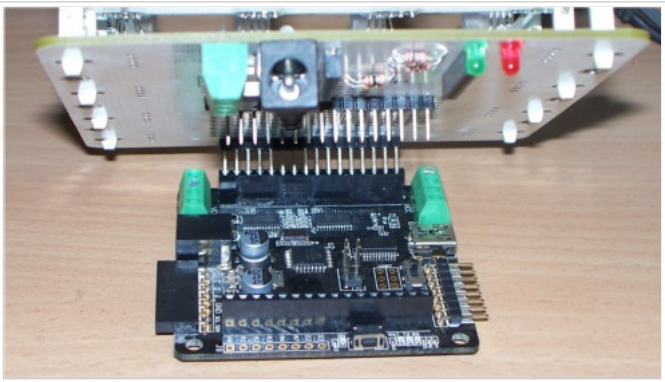
The case is made of Acrylic, to disassemble the box, please pay attention and follow the disassembling note:

The keys are not strong enough to withstand hard stretching. To disassemble the box, you should use your left hand to push the two legs and at the same time use your right hand to pull the key out. This is illustrated in the image below:

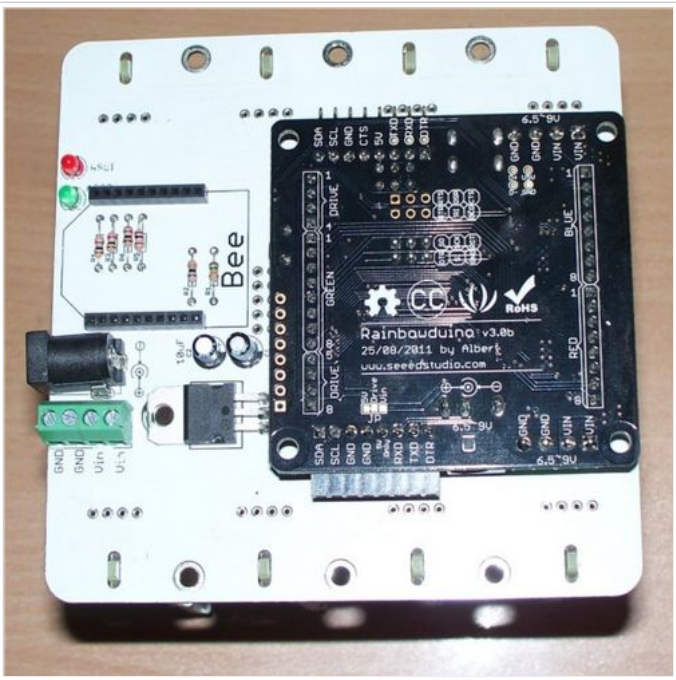


Hardware Installation

- Connect **Rainbow Cube** 2x16 male pin header to **Rainbowduino** as shown below



2x16 pin header positioning



A **Rainbowduino** properly connected to **Rainbow Cube**

- Attach an USB cable to **Rainbowduino** for programming

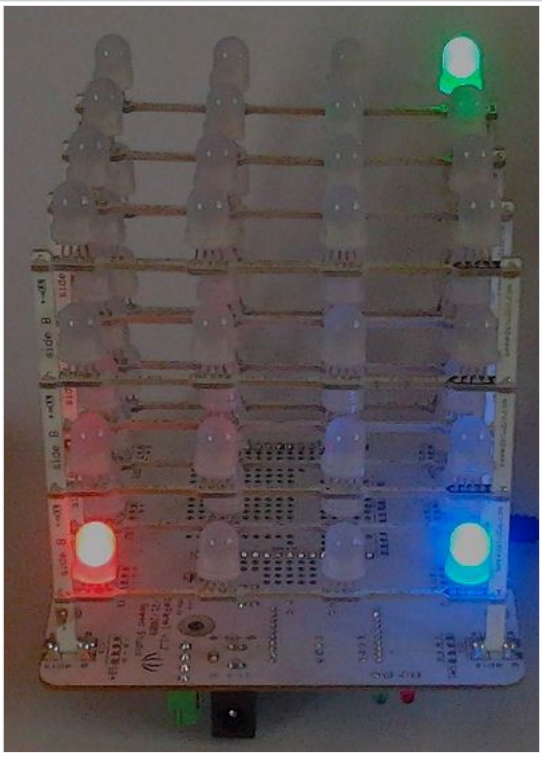
Programming

Let us get started with a simple example:

- Download Rainbowduino v3.0 Library
- Open Cube1.pde sketch (a copy of it is reproduced below):
- Compile and upload the sketch

```
/*
 Rainbowduino v3.0 Library examples:  Cube1

 Sets pixels on 3D plane (4x4x4 cube)
*/

#include <Rainbowduino.h>

void setup()
{
  Rb.init(); //initialize Rainbowduino driver
}

void loop()
{
  //Set (Z,X,Y):(0,0,0) pixel BLUE
  Rb.setPixelZXY(0,0,0,0x0000FF); //uses 24bit RGB color Code

  //Set (Z,X,Y):(0,3,0) pixel RED
  Rb.setPixelZXY(0,3,0,0xFF,0,0); //uses R, G and B color bytes

  //Set (Z,X,Y):(3,0,3) pixel GREEN
  Rb.setPixelZXY(3,0,3,0x00FF00); //uses 24bit RGB color Code
}
```

Output



Cube1.pde Demo

Application Programming Interfaces

In the above example, we have used few of the below **APIs**

init()

First we need to initialize the driver using init()

Usage:

```
Rb.init();//initialize Rainbowduino driver. This should be placed inside setup()
```

To set a LED in the 3D Cube we use the below two **APIs**.

setPixelZXY(Z,X,Y,R,G,B)

To set a LED (Z,X,Y) we use setPixelZXY(Z,X,Y,R,G,B).

Usage:

```
Rb.setPixelZXY(unsigned char x, unsigned char y, unsigned char  colorR,  unsigned char colorG, unsigned char colorB); //This sets the pixel (z,x,y) by
```

setPixelZXY(Z,X,Y,24bRGB)

Alternatively a LED (Z,X,Y) can be set by using setPixelZXY(Z,X,Y,24bRGB).

Usage:

```
Rb.setPixelZXY(unsigned char z, unsigned char x, unsigned char y, uint32_t colorRGB /*24-bit RGB Color*/) //This sets the LED (z,x,y) by specifying a
```

blankDisplay(void)

At times, it useful to blank all the LEDs. For this there is an **API blankDisplay(void)**.

Usage:

```
Rb.blankDisplay();
//Clear the LEDs (make all LEDs blank)
```

Demos

setPixelZXY() Demo

- To understand the (Z,X,Y) pixel addressing let us see another example. In this demo, the Layer 0 (i.e Z-0) is painted Green and Layer 3 is painted Blue.

```
/*
 Rainbowduino v3.0 Library examples:  Cube2

 Sets pixels on 3D plane (4x4x4 cube)
*/

#include <Rainbowduino.h>

void setup()
{
  Rb.init(); //initialize Rainbowduino driver
}

unsigned int z,x,y;

void loop()
{
  for(x=0;x<4;x++)
  {
    for(y=0;y<4;y++)
    {
     //Paint layer 0 Green
     Rb.setPixelZXY(0,x,y,0x00FF00); //uses 24bit RGB color Code
    }
  }

  for(x=0;x<4;x++)
  {
    for(y=0;y<4;y++)
    {
     //Paint layer 3 Blue
     Rb.setPixelZXY(3,x,y,0x0000FF); //uses 24bit RGB color Code
    }
  }
}
```

Output

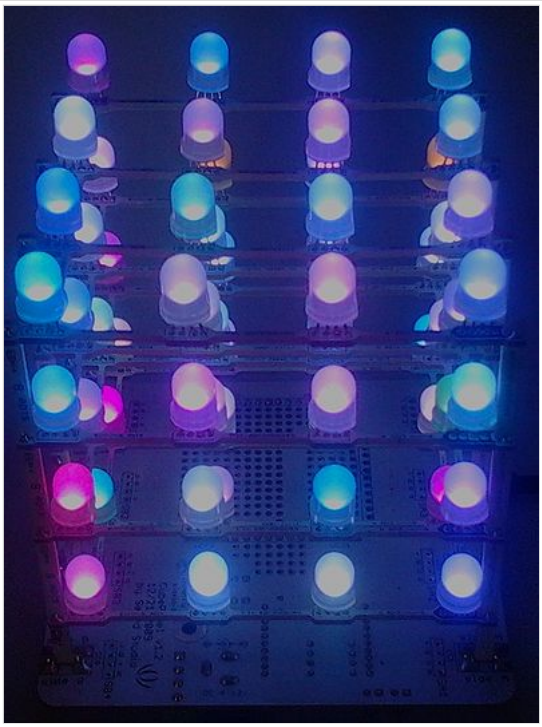
Cube2.pde Demo

setPixelZXY() Random Colors Demo

- In this demo, all LEDs are painted with some random color. After five seconds of delay, the whole cube is repainted with random colors.

```
/*
 Rainbowduino v3.0 Library examples:  Cube3

 Sets pixels on 3D plane (4x4x4 cube)
*/

#include <Rainbowduino.h>

void setup()
{
  Rb.init(); //initialize Rainbowduino driver
}

unsigned int z,x,y;

void loop()
{
 for(z=0;z<4;z++)
 {
  for(x=0;x<4;x++)
  {
    for(y=0;y<4;y++)
    {
     //Paint random colors
     Rb.setPixelZXY(z,x,y,random(0xFF),random(0xFF),random(0xFF)); //uses R, G and B color bytes
    }
  }
 }
delay(5000);
Rb.blankDisplay(); //Clear the LEDs (make all blank)
}
```

Output



Cube3.pde Demo

Night Lamp / Mood Lamp Demo

```
/*

 Rainbowduino v3.0 Library examples : Mood Lamp

*/

#include <Rainbowduino.h>


// HSV to RGB array

unsigned char RED[64] = {255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,238,221,204,188,171,154,137,119,102,85,
68,51,34,17,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,17,35,52};

unsigned char GREEN[64] = {0,17,34,51,68,85,102,119,136,153,170,187,204,221,238,255,255,255,255,255,255,255,255,255,255,255,
255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,238,221,204,188,170,154,136,120,102,86,68,52,34,18,0,0,0,0};

unsigned char BLUE[64] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,18,34,52,68,86,102,120,136,154,170,188,
204,221,238,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255};

void setup()
{
  Rb.init(); //initialize Rainbowduino driver
}

unsigned int z,x,y;
void loop()
{

for(z=0; z<64 ;z++)
{
 for(x=0;x<8;x++)
 {
    for(y=0;y<8;y++)
    {
     //Paint random colors
     //Rb.setPixelZXY(z,x,y,RED[i],GREEN[i],BLUE[i]); //uses R, G and B color bytes
     Rb.setPixelXY(x,y,RED[z],GREEN[z],BLUE[z]); //uses R, G and B color bytes
    }
 }
 delay(100);
}

for(z=63; z > 0 ;z--)
{
 for(x=0;x<8;x++)
 {
    for(y=0;y<8;y++)
    {
     //Paint random colors
     //Rb.setPixelZXY(z,x,y,RED[i],GREEN[i],BLUE[i]); //uses R, G and B color bytes
     Rb.setPixelXY(x,y,RED[z],GREEN[z],BLUE[z]); //uses R, G and B color bytes
    }
 }
 delay(100);
}

}
```

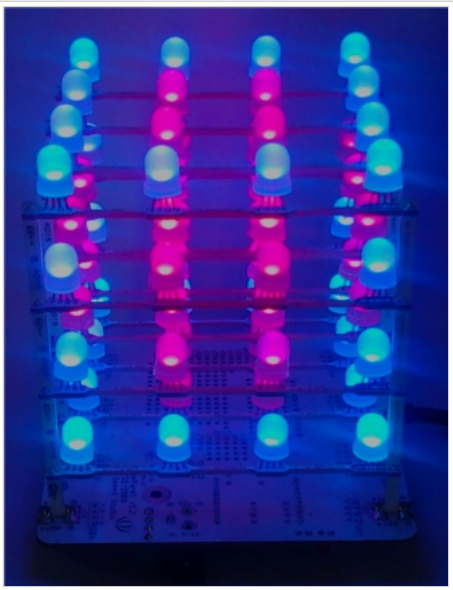
Output

[[Image:|thumb|none|400px| .pde Demo]]

Plasma Cube

```
/*

 Rainbowduino v3.0 Library examples : 3D Plasma

*/

#include <Rainbowduino.h>

// HSV to RGB array

unsigned char RED[64] = {255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,238,221,204,188,171,154,137,119,102,85,
68,51,34,17,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,17,35,52};

unsigned char GREEN[64] = {0,17,34,51,68,85,102,119,136,153,170,187,204,221,238,255,255,255,255,255,255,255,255,255,255,255,
255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,238,221,204,188,170,154,136,120,102,86,68,52,34,18,0,0,0,0};

unsigned char BLUE[64] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,18,34,52,68,86,102,120,136,154,170,188,
204,221,238,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255};

unsigned char plasma[4][4][4];

void setup()
{
  Rb.init(); //initialize Rainbowduino driver

  for(unsigned char x = 0; x < 4; x++)
  {
    for(unsigned char y = 0; y < 4; y++)
    {
      for(unsigned char z = 0; z < 4; z++)
       {
        int color = int(32.0 + (32.0 * sin(x / 1.0))+ 32.0 + (32.0 * sin(y / 1.0)) + 32.0 + (32.0 * sin(z / 1.0))) / 3;
        plasma[x][y][z] = color;
       }
    }
  }
}

unsigned char x,y,z,colorshift=0;

void loop()
{
for(x=0;x<4;x++)
{
 for(y=0;y<4;y++)
 {
  for(z=0;z<4;z++)
    {
     Rb.setPixelZXY(z,x,y,(RED[plasma[x][y][z] + colorshift]) % 256,(GREEN[plasma[x][y][z] + colorshift]) % 256,(BLUE[plasma[x][y][z] + colorshift]) %
    }
 }
}
 delay(100);
 colorshift=  colorshift + 1;
}
```

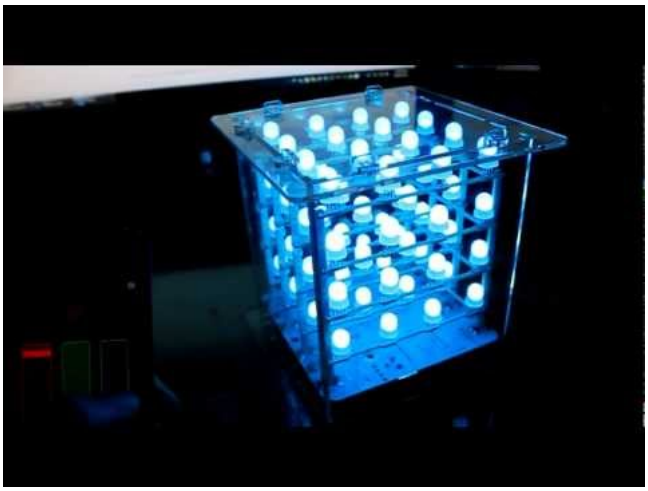
Output



PlasmaCube.pde Demo

Third-party Demos

- RGB iPhone Software (by KonchaTech)
  - Send RGB values to your cube via an iPhone (requires connected computer and TouchOSC software)
  - http://konchatech.blogspot.ca/2012/05/4x4x4-iphone-controlled-rgb-led-cube.html (for sourcecode)
  - http://www.youtube.com/watch?v=JXr-Jgzgigs [Video]



- Ambient Visualizer Software (by KonchaTech)
  - Make your cube change colors in reaction to what is displayed on your computer screen (requires connected computer)
  - http://konchatech.blogspot.ca/2012/05/ambient-led-cube.html (for sourcecode)
  - http://www.youtube.com/watch?v=cC_xqA5irLA [Video 1]
  - http://www.youtube.com/watch?v=ed9FURoNZ6M [Video 2]

0.jpg

Support

If you have questions or other better design ideas, you can go to our forum (http://www.seeedstudio.com/forum) or wish (http://wish.seeedstudio.com) to discuss.

Version Tracker

Revision	Descriptions	Release
v1.0	Initial public release	

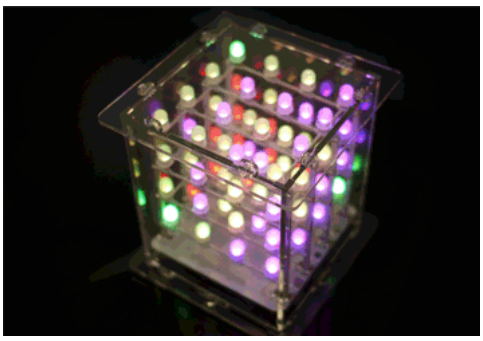
Resources

- Rainbow Cube:
  - Rainbowduino v3.0 Software Library (http://seeedstudio.com/wiki/images/9/9f/Rainbowduino3.0_Library.zip)
  - Rainbowduino3.0 Library for Arduino 1.0 (http://www.seeedstudio.com/wiki/images/4/43/Rainbowduino_for_Arduino1.0.zip)
  - Complete schematic in PDF (http://seeedstudio.com/wiki/images/d/dc/Rainbow_Cube_Kit_-_RGB_4x4x4_LED_schematic_board.pdf)
  - Eagle CAD files (http://seeedstudio.com/wiki/images/b/b1/RainbowCube_v1.1_Panel_v1.2_Eaglefiles.zip)
  - Rainbow cube with Blurtooth Setch (http://www.seeedstudio.com/wiki/images/3/3c/RainbowWithBluetooth.zip)
- Data-sheets
  - MY9221 Datasheet (http://www.seeedstudio.com/wiki/images/9/98/MY9221_DS_1.0.pdf)
  - 8mm RGB LED (http://www.seeedstudio.com/depot/datasheet/8mmLED.pdf)
- Android Apk
  - RainbowBluetooth Apk (http://www.seeedstudio.com/wiki/images/a/a4/RainbowBluetooth.zip)
- Rainbowduino Control Software
  - Rainbowduino Semi - Automater installtion Package Beta (For Windows) (http://diymagicmirror.com/files/Rainbowduino_setup_v1_1.exe)
  - mtXcontrol by rngtng Beta(For MAC) (https://github.com/rngtng/mtXcontrol/downloads)

Related Projects

If you want to make some awesome projects by Rainbow Cube, here's some projects for reference.

Automatic Water Level Controller



The Rainbow Cubes do not have a standard for controlling them or chaining them together. You can look to change that.

I want to make it. (http://www.seeedstudio.com/recipe/205-daisy-chaining-rainbowduinos.html)

Share Your Awesome Projects with Us

Born with the spirit of making and sharing, that is what we believe makes a maker.

And only because of this , the open source community can be as prosperous as it is today.

It does not matter what you are and what you have made, hacker, maker, artist and engineers,

as long as you start sharing your works with others,

you are being part of the open source community and you are making your contributions .

Now share you awesome projects on with us on Recipe (http://www.seeedstudio.com/recipe/) , and win a chance to become the Core User of Seeed.

- Core Users, are those who showing high interests and significant contributions in Seeed products.
- We cooperate with our Core Users in the development of our new product, this, in another word, the Core Users will have the chance to experience any new products of Seeed before its official launch, and in return we expect valuable feedback from them to help us improving the product performance and user experience. And for most of cases if our Core Users have any good ideas for making things, we'll offer hardware pieces, PCBA services as well as technical support. Besides, further commercial cooperation with the Core Users is highly possible.

Get more information about Core User please email to: recipe@seeed.cc

---