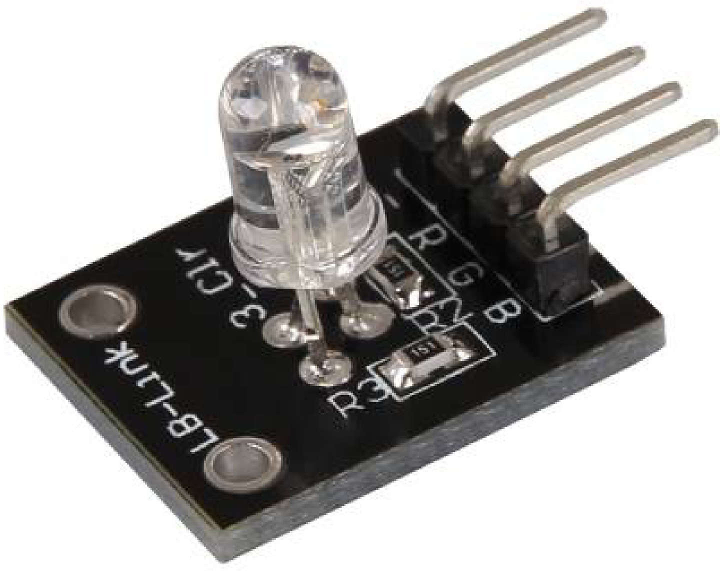
KY-016 RGB 5mm LED module

Contents

Picture



Technical data / Short description

LED-module which includes a red, blue and green LED. These are connected by a common cathode.
An additional resistor might be necessary for some voltages.

Vf [Red]= 1,8V

Vf [Green,Blue]= 2,8V

If= 20mA

Vorwiderstände:

Rf (3,3V) [Green]= 100Ω

Rf (3,3V) [Red]= 180Ω

Rf (3,3V) [Blue]= 100Ω

[e.g. by using with ARM CPU-core based microcontroller like Raspberry-Pi]

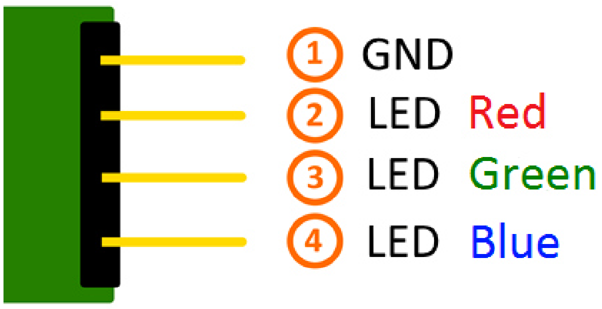
Rf (5V) [Green] = 100Ω

Rf (5V) [Red] = 180Ω

Rf (5V) [Blue] = 100Ω

[e.g. by using with Atmel Atmega based mocrocontroller like Arduino]

Pinout



Code example Arduino

Code example ON/OFF

In this example you will see how the LED is turned on by an output pin, in a 3 second clock pulse.

```
int Led_Red = 10;
int Led_Green = 11;
int Led_Blue = 12;

void setup ()
{
  // Output pin initialization for the LEDs
  pinMode (Led_Red, OUTPUT);
  pinMode (Led_Green, OUTPUT);
  pinMode (Led_Blue, OUTPUT);
}

void loop () //main program loop
{
  digitalWrite (Led_Red, HIGH); // LED will be switched ON
  digitalWrite (Led_Green, LOW); // LED will be switched OFF
  digitalWrite (Led_Blue, LOW); // LED will be switched OFF
  delay (3000); // Waitmode for 3 seconds
  digitalWrite (Led_Rot, LOW); // LED will be switched OFF
  digitalWrite (Led_Gruen, HIGH); // LED wwill be switched ON
  digitalWrite (Led_Blau, LOW); // LED will be switched OFF
  delay (3000); // Waitmode for another 3 seconds in which the LEDs will be shifted.

  digitalWrite (Led_Rot, LOW); // LED will be switched OFF
  digitalWrite (Led_Gruen, LOW); // LED will be switched OFF
  digitalWrite (Led_Blau, HIGH); // LED will be switched ON
  delay (3000); // Waitmode for another 3 seconds in which the LEDs will be shifted.
}
```

Example program ON/OFF download:

KY-016_LED_ON-OFF.zip

Code example PWM

You can regulate the brightness of the LEDs via pulse-width modulation. The LEDs will be switched ON and OFF for specific time periods, in which the relation between ON and OFF leads to a relative brightness, because of the Inertia of the human eyesight, the human eye interprets the ON/OFF as a brightness change. For more information to that theme visit: [Artikel von mikrokontroller.net].

This module provides a few LEDs - with the overlay of the different brightness levels, you can create different colors. This will be shown in the following code example.

```
int Led_Red = 10;
int Led_Green = 11;
int Led_Blue = 12;

int val;

void setup () {
  // Output pin initialization for the LEDs
  pinMode (Led_Red, OUTPUT);
  pinMode (Led_Green, OUTPUT);
  pinMode (Led_Blue, OUTPUT);
}
void loop () {
   // In this for-loop, the 3 LEDs will get different PWM-values
   // Via mixing the brightness of the different LEDs, you will get different colors.
   for (val = 255; val> 0; val--)
      {
       analogWrite (Led_Blue, val);
       analogWrite (Led_Green, 255-val);
       analogWrite (Led_Red, 128-val);
       delay (1);
   }
   // You will go backwards through the color range in this second for loop.
   for (val = 0; val <255; val++)
      {
      analogWrite (Led_Blue, val);
      analogWrite (Led_Green, 255-val);
      analogWrite (Led_Red, 128-val);
      delay (1);
   }
}
```

Example program PWM download:

KY-016_PWM.zip

Connections Arduino:

LED Red	=	[Pin 10]
LED Green	=	[Pin 11]
LED Blue	=	[Pin 12]
Sensor GND	=	[Pin GND]

Code example Raspberry Pi

Code example ON/OFF

In this example you will see how the LED is turned on by an output pin, in a 3 second clock pulse.

```
# Needed modules will be imported
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

# The output pins will be declared, which are connected with the LEDs.
LED_RED = 6
LED_GREEN = 5
LED_BLUE = 4

GPIO.setup(LED_RED, GPIO.OUT, initial= GPIO.LOW)
GPIO.setup(LED_GREEN, GPIO.OUT, initial= GPIO.LOW)
GPIO.setup(LED_BLUE, GPIO.OUT, initial= GPIO.LOW)

print "LED-test [press ctrl+c to end]"

# main program loop
try:
        while True:
                        print("LED RED is on for 3 seconds")
                        GPIO.output(LED_RED,GPIO.HIGH) #LED will be switched ON
                        GPIO.output(LED_GREEN,GPIO.LOW) #LED will be switched OFF
                        GPIO.output(LED_BLUE,GPIO.LOW) #LED will be switched OFF
                        time.sleep(3) # waitmode for 3 seconds
                        print("LED GREEN is on for 3 seconds")
                        GPIO.output(LED_RED,GPIO.LOW) #LED will be switched OFF
                        GPIO.output(LED_GREEN,GPIO.HIGH) #LED will be switched ON
                        GPIO.output(LED_BLUE,GPIO.LOW) #LED will be switched OFF
                        time.sleep(3) # waitmode for 3 seconds
                        print("LED BLUE is on for 3 seconds")
                        GPIO.output(LED_RED,GPIO.LOW) #LED will be switched OFF
                        GPIO.output(LED_GREEN,GPIO.LOW) #LED will be switched OFF
                        GPIO.output(LED_BLUE,GPIO.HIGH) #LED will be switched ON
                        time.sleep(3) #waitmode for 3 seconds

# Scavenging work after the end of the program
except KeyboardInterrupt:
        GPIO.cleanup()
```

Example program ON/OFF download

To start, enter the command:

```
sudo python KY016_RPI_ON-OFF.py
```

Code example PWM

You can regulate the brightness of the LEDs via pulse-width modulation. The LEDs will be switched ON and OFF for specific time periods, in which the relation between ON and OFF leads to a relative brightness, because of the Inertia of the human eyesight, the human eye interprets the ON/OFF as a brightness change. For more information to that theme visit: [Artikel von mikrokontroller.net].

This module provides a few LEDs - with the overlay of the different brightness levels, you can create different colors. This will be shown in the following code example. At the Raspberry Pi, only one Hardware-PWM channel is carried out unrestricted to the GPIO pins, why we have used Software-PWM at this example.

```
# Needed modules will be imported and configured
import random, time
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

# Declaration of the output pins, which are connected with the LEDs
LED_Red = 6
LED_Green = 5
LED_Blue = 4

# Set pins to output mode
GPIO.setup(LED_Red, GPIO.OUT)
GPIO.setup(LED_Green, GPIO.OUT)
GPIO.setup(LED_Blue, GPIO.OUT)

Freq = 100 #Hz

# The different colors will be initialized
RED = GPIO.PWM(LED_Red, Freq)
GREEN = GPIO.PWM(LED_Green, Freq)
BLUE = GPIO.PWM(LED_Blue, Freq)
RED.start(0)
GREEN.start(0)
BLUE.start(0)

# This function generate the actually color
# You can change the color with the specific color variable
# After the configuration of the color if finished, you will use time.sleep to
# configure how long the specific color will be displayed

def LED_color(Red, Green, Blue, pause):
    RED.ChangeDutyCycle(Red)
    GREEN.ChangeDutyCycle(Green)
    BLUE.ChangeDutyCycle(Blue)
    time.sleep(pause)

    RED.ChangeDutyCycle(0)
    GREEN.ChangeDutyCycle(0)

print "LED-test [press ctrl+c to end]"

# Main program loop:
# The task of this loop is to create for every single color an own variable
# By mixing the brightness levels of the colors, you will get a color gradient.
try:
    while True:
        for x in range(0,2):
            for y in range(0,2):
                for z in range(0,2):
                    print (x,y,z)
                    for i in range(0,101):
                        LED_color((x*i),(y*i),(z*i),.02)

# Scavenging work after the end of the program
except KeyboardInterrupt:
        GPIO.cleanup()
```

Example program PWM download:

KY-016_RPi_PWM.zip

To start, enter the command:

```
sudo python KY-016_RPi_PWM.py
```

Connections Raspberry Pi:

LED Red	= GPIO6	[Pin 22]
LED Green	= GPIO5	[Pin 18]
LED Blue	= GPIO4	[Pin 16]
Sensor GND	= GND	[Pin 6]