AZ-Delivery

Ihr Experte für Mikroelektronik!

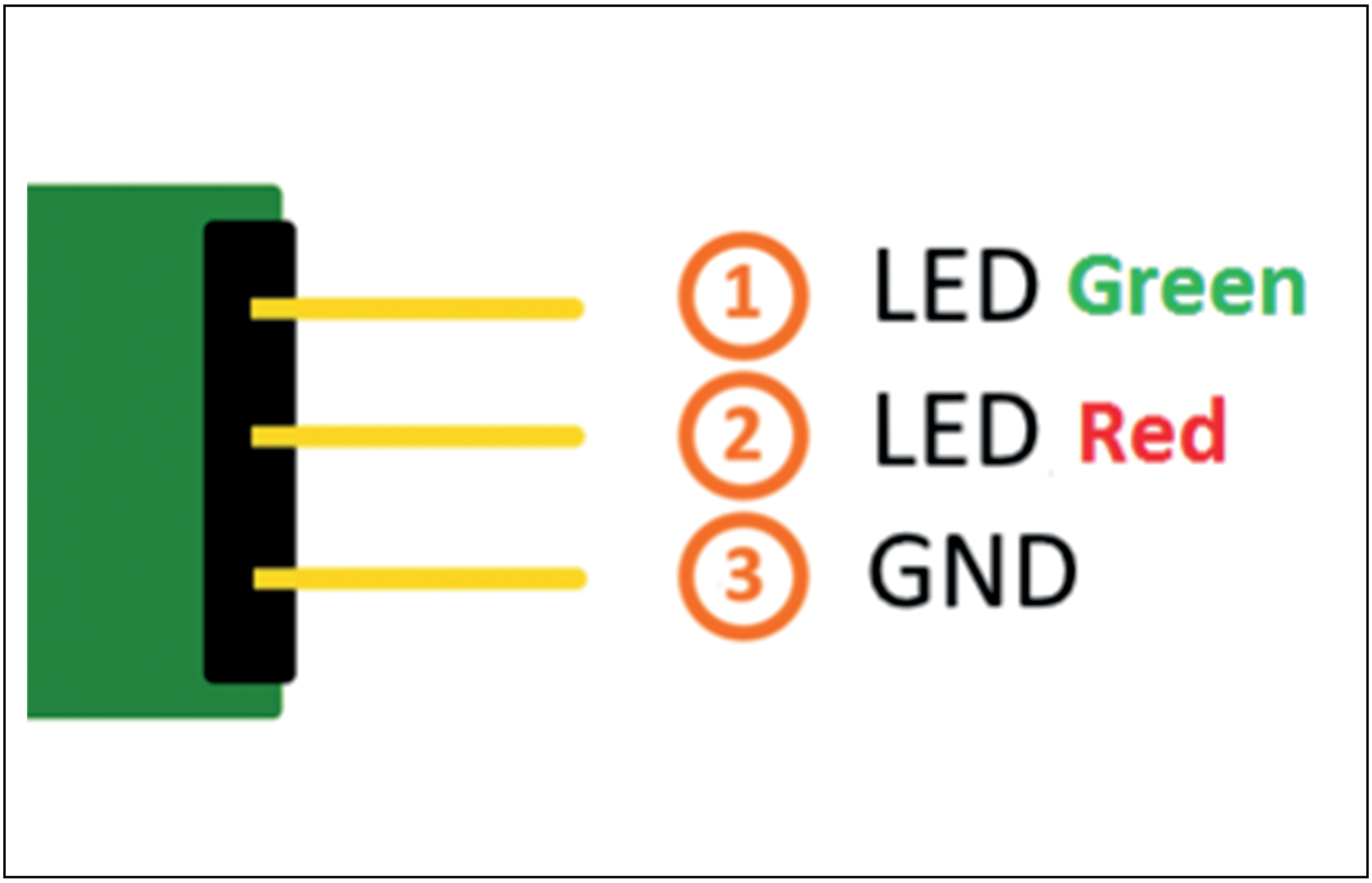
KY-029 Bi-Color LED Modul 3mm Datenblatt

Contents:

1. Technical Data

LED module which includes a red and a green LED. They are connected via cathode.
You need resistors for different voltages.

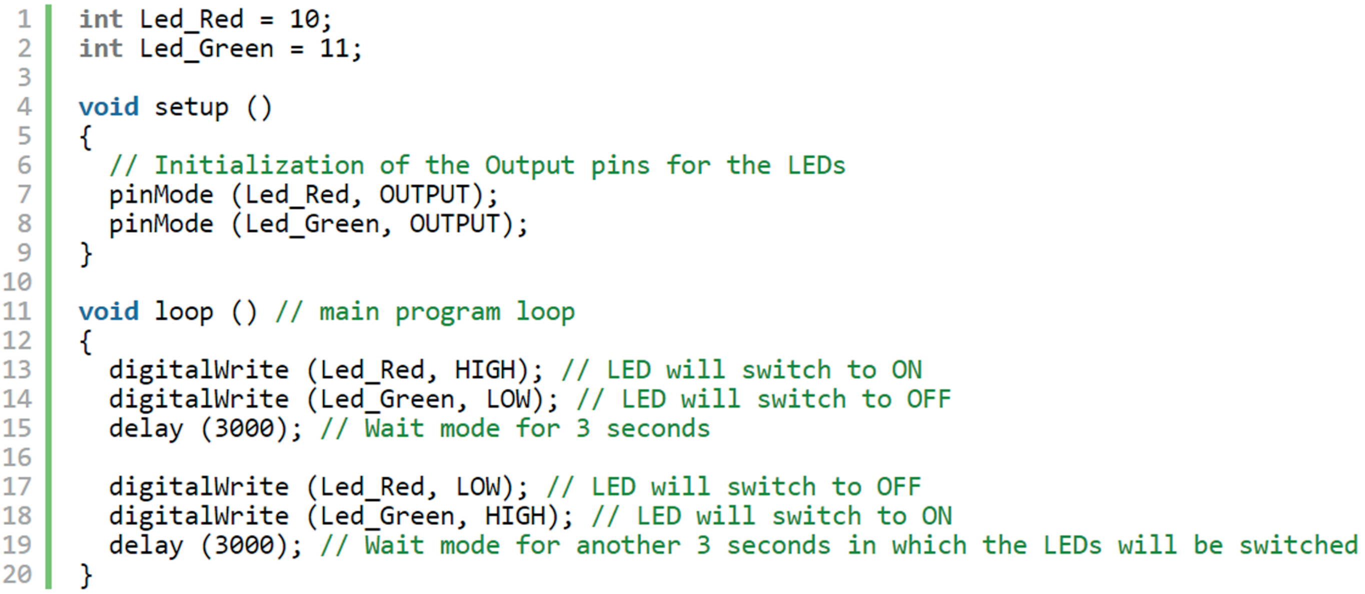
2. Pinout



3. Code Example Arduino

Code example ON/OFF

This example shows how you can switch the LEDs on and off in a 3 seconds time period via defined output pin.

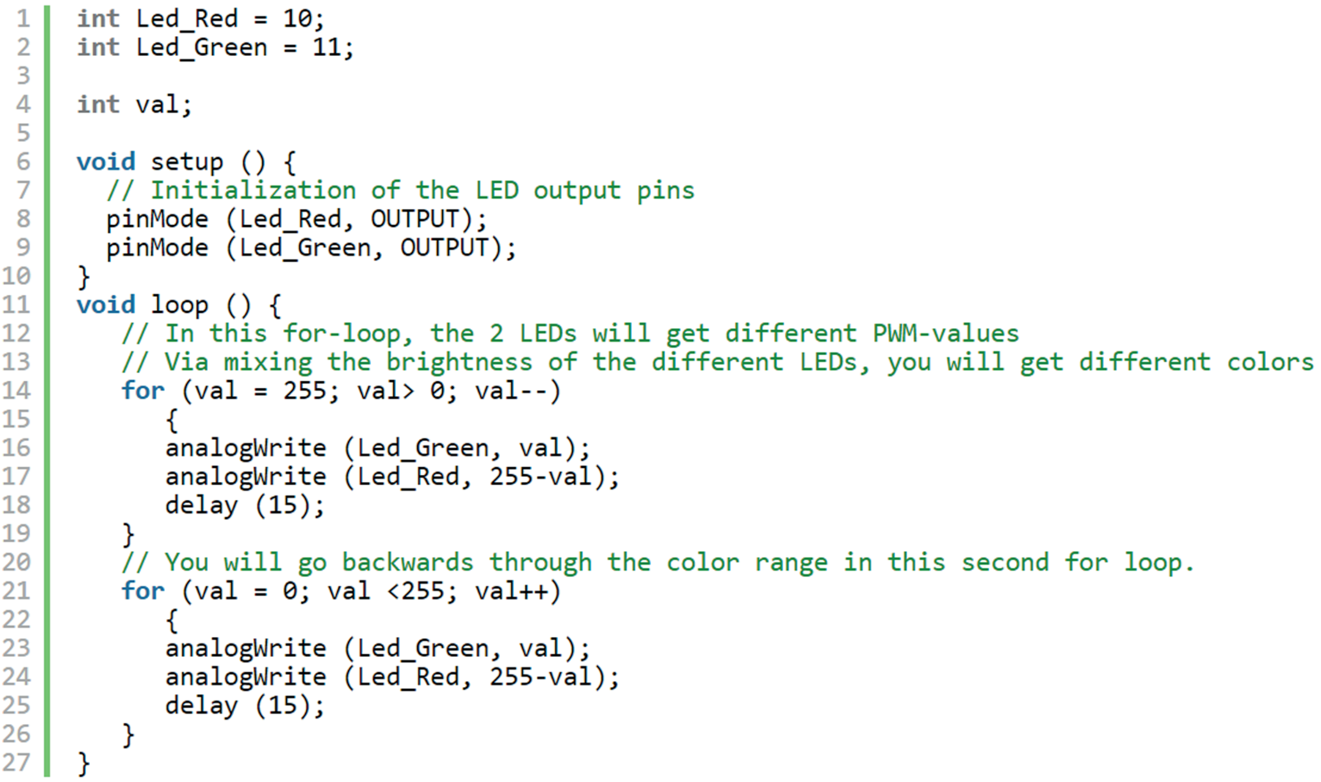


```
int Led_Red = 10;
int Led_Green = 11;

void setup ()
{
  // Initialization of the Output pins for the LEDs
  pinMode (Led_Red, OUTPUT);
  pinMode (Led_Green, OUTPUT);
}

void loop () // main program loop
{
  digitalWrite (Led_Red, HIGH); // LED will switch to ON
  digitalWrite (Led_Green, LOW); // LED will switch to OFF
  delay (3000); // Wait mode for 3 seconds

  digitalWrite (Led_Red, LOW); // LED will switch to OFF
  digitalWrite (Led_Green, HIGH); // LED will switch to ON
  delay (3000); // Wait mode for another 3 seconds in which the LEDs will be switched
}
```

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.

This module provides 2 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 val;

void setup () {
  // Initialization of the LED output pins
  pinMode (Led_Red, OUTPUT);
  pinMode (Led_Green, OUTPUT);
}
void loop () {
   // In this for-loop, the 2 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_Green, val);
      analogWrite (Led_Red, 255-val);
      delay (15);
   }
   // You will go backwards through the color range in this second for loop.
   for (val = 0; val <255; val++)
      {
      analogWrite (Led_Green, val);
      analogWrite (Led_Red, 255-val);
      delay (15);
   }
}
```

Connections Arduino:

LED Green = [Pin 10]

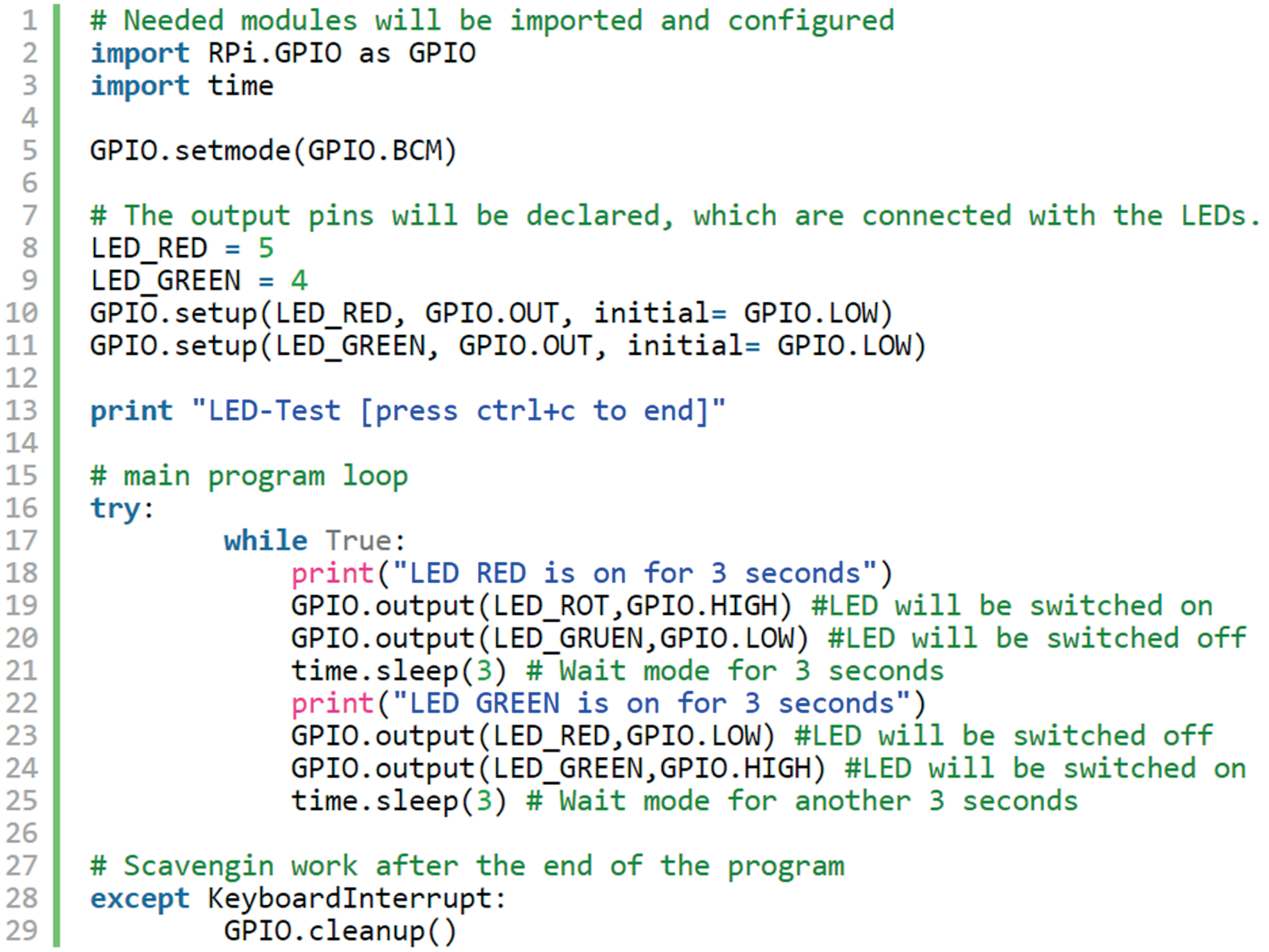
LED Red = [Pin 11]

Sensor GND = [Pin GND]

4. Code Example Raspberry Pi

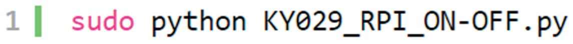
Code example ON/OFF

In this example you will see how the LEDs will be switched on with a defined output pin, in a 3 second clock pulse.



```
# Needed modules will be imported and configured
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 = 5
LED_GREEN = 4
GPIO.setup(LED_RED, GPIO.OUT, initial= GPIO.LOW)
GPIO.setup(LED_GREEN, 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_ROT,GPIO.HIGH) #LED will be switched on
      GPIO.output(LED_GRUEN,GPIO.LOW) #LED will be switched off
      time.sleep(3) # Wait mode 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
      time.sleep(3) # Wait mode for another 3 seconds

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

To start, enter the command:

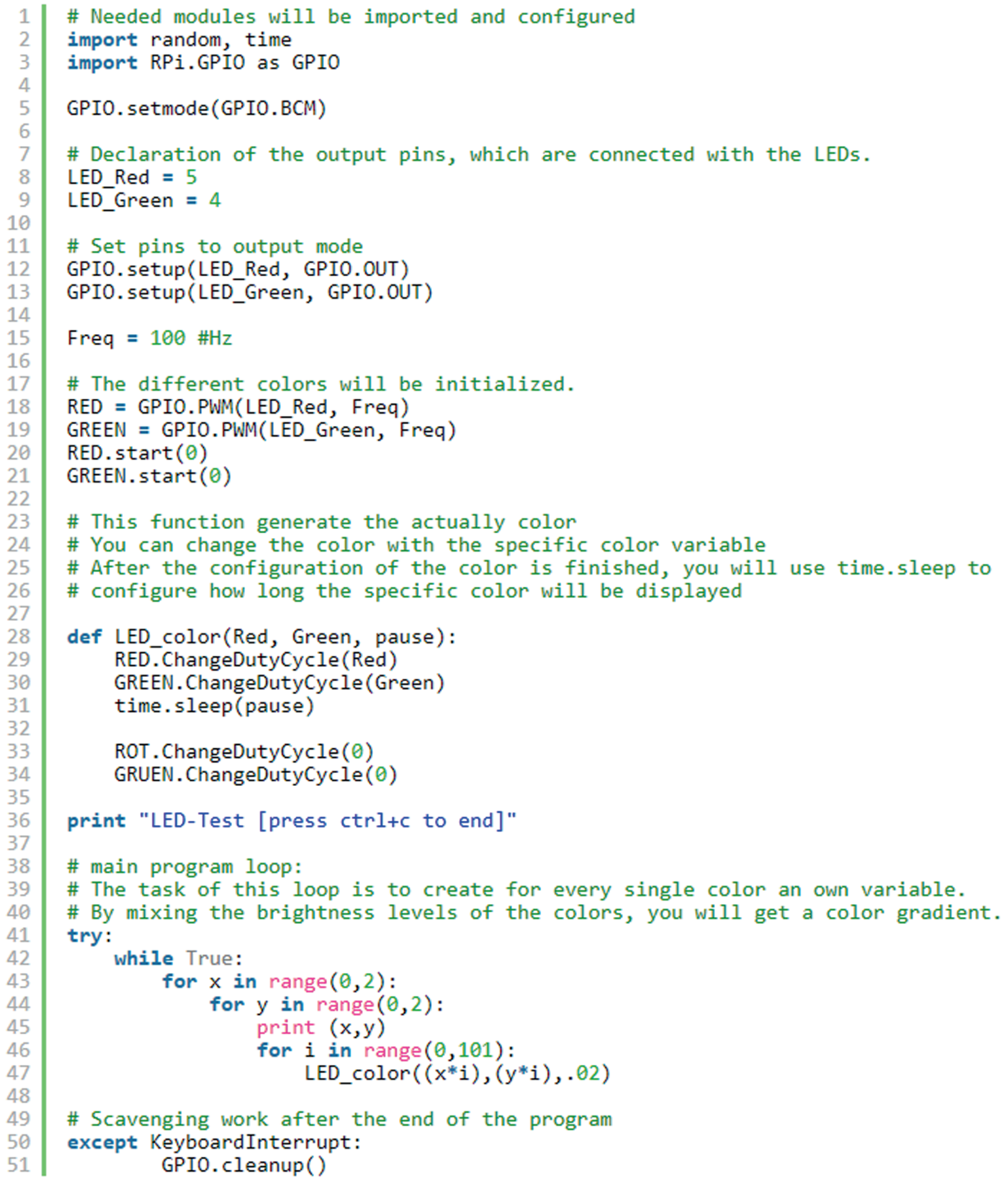


```
sudo python KY029_RPI_ON-OFF.py
```

Code example PWM

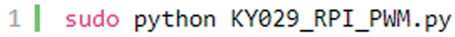
You can regulate the brightness of the LEDs via pulse-width modulation. The LEDs will be switched ON and OFF of for specific time periods, in which the relation between ON and OFF leads to a relative brightness, because of the Inertia of the human eyesight, the human eye interprets the ON/OFF as a brightness 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 = 5
LED_Green = 4

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

Freq = 100 #Hz

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

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

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

    ROT.ChangeDutyCycle(0)
    GRUEN.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):
                print (x,y)
                for i in range(0,101):
                    LED_color((x*i),(y*i),.02)

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

To start, enter the command:



```
sudo python KY029_RPI_PWM.py
```

Connections Raspberry Pi:

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