Welcome!

Thank you for purchasing our *AZ-Delivery LED Traffic Light Module*. On the following pages, you will be introduced to how to use and set up this handy device.

Have fun!

Table of Contents

Introduction

LED traffic light module is a mini-traffic light, easy to manage and suitable for the production of traffic light system model.

The module consist of single board with three 8mm LEDs (Light Emitting Diodes) - red, yellow and green - and built-in current limiting resistors. There are only four pins on the device, one for the anode of each LED and a common cathode. This way the connection to microcontrollers is simplified.

When using PWM (pulse width modulation) the LEDs can produce fade-in, fade-out effect or can be dimmed to the desired brightness level from the respective microcontroller pin.

The module is developed for educational purposes (road safety training as well as basic electronics and physical computing/coding), the traffic light system may also be useful for the moderation of a discussion, or filling or charging indicator, etc.

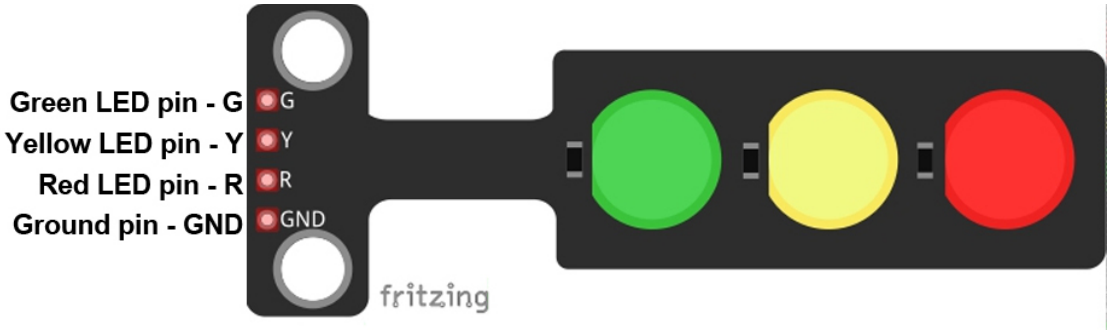
Specifications

Operating voltage range	3.3V to 5V
Current consumption	4mA at 3.3V, 9mA at 5V
Single LED diameter	8mm
Mounting holes diameter	3mm
Dimensions	56x21x11 mm(2.2x0.8x0.4in)

The module is shipped with presoldered pins for simplified prototyping on the breadboard.

The pinout

The LED Traffic Light Module has 4 pins. The pinout is shown in the following image:



How to set-up Arduino IDE

If the Arduino IDE is not installed, follow the *link* and download the installation file for the operating system of choice. The Arduino IDE version used for this eBook is **1.8.12.**

For *Windows* users, double click on the downloaded *.exe* file and follow the instructions in the installation window.

For *Linux* users, download a file with the extension *.tar.xz*, which has to be extracted. When it is extracted, go to the extracted directory and open the terminal in that directory. Two *.sh* scripts have to be executed, the first called *arduino-linux-setup.sh* and the second called *install.sh*.

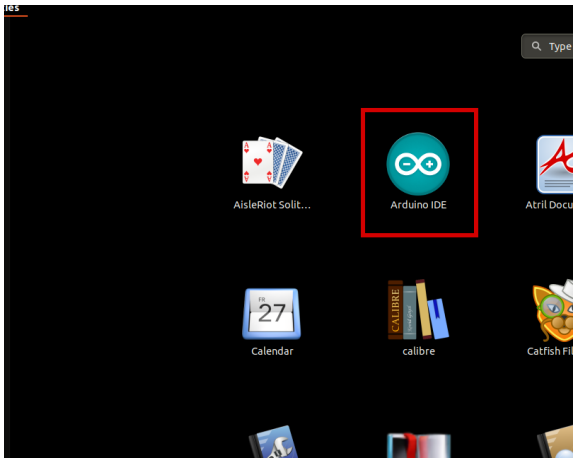
To run the first script in the terminal, open the terminal in the extracted directory and run the following command:

sh arduino-linux-setup.sh user_name

user_name - is the name of a superuser in Linux operating system. A password for the superuser has to be entered when the command is started. Wait for a few minutes for the script to complete everything.

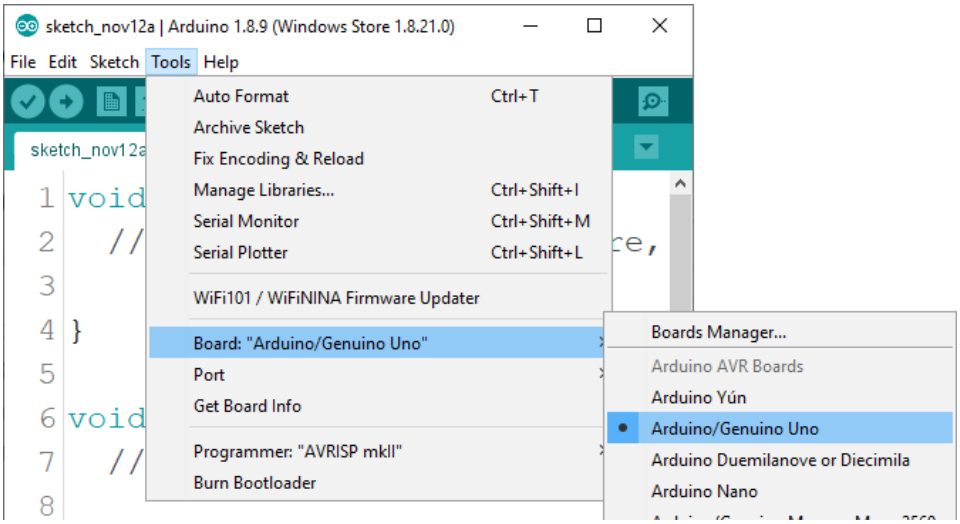
The second script, called *install.sh*, has to be used after the installation of the first script. Run the following command in the terminal (extracted directory): **sh install.sh**

After the installation of these scripts, go to the *All Apps*, where the *Arduino IDE* is installed.



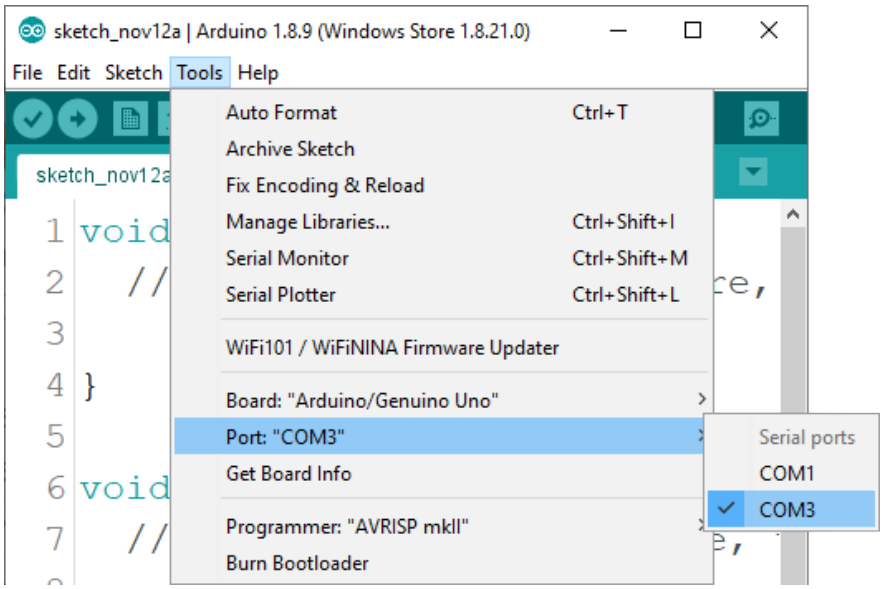
Almost all operating systems come with a text editor preinstalled (for example, *Windows* comes with *Notepad*, *Linux Ubuntu* comes with *Gedit*, *Linux Raspbian* comes with *Leafpad*, etc.). All of these text editors are perfectly fine for the purpose of the eBook.

Next thing is to check if your PC can detect an Atmega328p board. Open freshly installed Arduino IDE, and go to:
Tools > Board > {your board name here}
{your board name here} should be the *Arduino/Genuino Uno*, as it can be seen on the following image:



The port to which the Atmega328p board is connected has to be selected. Go to: *Tools > Port > {port name goes here}*
and when the Atmega328p board is connected to the USB port, the port name can be seen in the drop-down menu on the previous image.

If the Arduino IDE is used on Windows, port names are as follows:



For *Linux* users, for example, port name is */dev/ttyUSBx*, where *x* represents integer number between *0* and *9*.

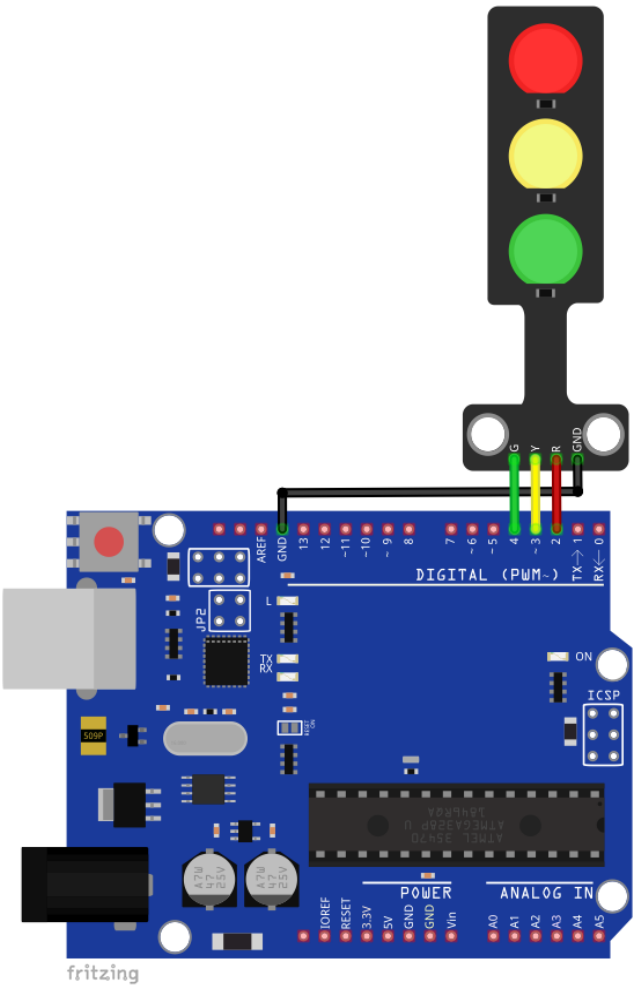
How to set-up the Raspberry Pi and Python

For the Raspberry Pi, first the operating system has to be installed, then everything has to be set-up so that it can be used in the *Headless* mode. The *Headless* mode enables remote connection to the Raspberry Pi, without the need for a *PC* screen Monitor, mouse or keyboard. The only things that are used in this mode are the Raspberry Pi itself, power supply and internet connection. All of this is explained minutely in the free eBook: *Raspberry Pi Quick Startup Guide*

The *Raspberry Pi OS (*operating system), previously known as Raspbian, comes with *Python* preinstalled.

Connecting the module with Atmega328p

Connect the module with the Atmega328p as shown n the following connection diagram:



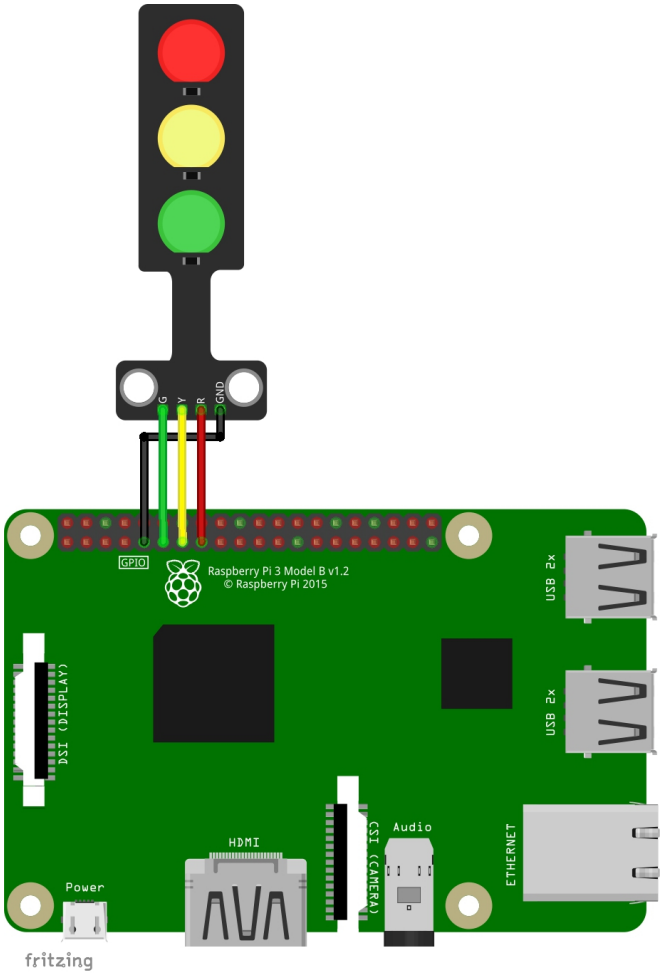
Module pin	Mc pin	Wire color
G	D4	**Green wire**
Y	D3	**Yellow wire**
R	D2	**Red wire**
GND	GND	**Black wire**

Sketch example

```
int GREEN = 4;
int YELLOW = 3;
int RED = 2;

void setup()
{
    pinMode(GREEN, OUTPUT);
    pinMode(YELLOW, OUTPUT);
    pinMode(RED, OUTPUT);
}

void loop()
{
    digitalWrite(GREEN, LOW);
    digitalWrite(YELLOW, LOW);
    digitalWrite(RED, HIGH);
    delay(3000);

    digitalWrite(YELLOW, HIGH);
    delay(1000);

    digitalWrite(RED, LOW);
    digitalWrite(YELLOW, LOW);
    digitalWrite(GREEN, HIGH);
    delay(3000);
    digitalWrite(GREEN, LOW);
    digitalWrite(YELLOW, HIGH);
    delay(1000);
}
```

Connecting the module with Raspberry Pi

Connect the module with the Raspberry Pi as shown on the following connection diagram:



Module pin	Raspberry Pi pin	Physical pin	Wire color
G	GPIO17	11	**Green wire**
Y	GPIO27	13	**Yellow wire**
R	GPIO22	15	**Red wire**
GND	GND	9	**Black wire**

Python script

```
import RPi.GPIO as GPIO
import time
import signal
import sys

GPIO.setmode(GPIO.BCM)
GPIO.setup(17, GPIO.OUT)  # green LED
GPIO.setup(27, GPIO.OUT)  # yellow LED
GPIO.setup(22, GPIO.OUT)  # red LED

def allLightsOff(signal, frame):
   GPIO.output(17, False)
   GPIO.output(27, False)
   GPIO.output(22, False)
   GPIO.cleanup()
   sys.exit(0)

signal.signal(signal.SIGINT, allLightsOff)
print('[Press CTRL + C to end the script!]')

   while True:
      GPIO.output(22, True)  # red on
      time.sleep(3)

      GPIO.output(27, True)  # yellow on, red is still on
      time.sleep(1)
```

```
GPIO.output(27, False) # yellow off
GPIO.output(22, False) # red off
GPIO.output(17, True)  # green on
time.sleep(5)

GPIO.output(17, False) # green off
GPIO.output(27, True)  # yellow on
time.sleep(2)

GPIO.output(27, False) # yellow off
```

Save the script under the name *trafficLight.py*. To run the script open the terminal in the directory where the script is saved and run the following command: **python3 trafficLight.py**

To end the script press 'CTRL + C' on the keyboard.

Now it is the time to learn and make your own projects. You can do that with the help of many example scripts and other tutorials, which can be found on the Internet.

If you are looking for the high quality microelectronics and accessories, AZ-Delivery Vertriebs GmbH is the right company to get them from. You will be provided with numerous application examples, full installation guides, eBooks, libraries and assistance from our technical experts.

https://az-delivery.de

Have Fun!

Impressum

https://az-delivery.de/pages/about-us