

AZ-Delivery

Welcome!

Thank you for purchasing our *AZ-Delivery HC-SR501 PIR Module*. On the following pages, we will introduce you to how to use and set-up this handy device.

Have fun!



Az-Delivery

Table of Contents

Introduction	3
Features	4
Adjusting hardware features	7
Specifications	9
The pinout	10
How to set-up Arduino IDE	11
How to set-up the Raspberry Pi and Python	16
Connecting the module with ATmega328p microcontroller	17
Sketch example	17
Connecting the module with Raspberry Pi	21
Python script	22

Az-Delivery

Introduction

The HC-SR501 PIR module is a device which can detect change in the infrared light. For example a human activity when is in the scanning range of the module. The module is based on infrared technology and uses pyroelectric passive infrared sensor, PIR for short. It consist of a pyroelectric sensor and few aditional electronic components. To increase the sensing angle a lense is used to focus surrounding light.

When the crystalline element is exposed to the infrared light, the output vottage is changed directly with the infrared light intensity. The voltage output from the sensor is in range of mV . This voltage get amplified via the on-board amplifier so that the microcontroller can use it.

The module is equipped with a so-called *Fresnel* lense which is a special kind of filter that focuses the infrared emissions onto the sensor.

The module can be used with various types of devices such as light devices, intercoms, electric fans, and other home devices. It is widely used in homes, enterprises, hotels, stores, sensitive areas where light automation and security systems are needed.

It has a high sensitivity, high reliability and ultra-low-voltage operation. - 3 -

Az-Delivery

Features

- Automatic induction
- Photosensitive control
- Temperature compensation
- Two way trigger
- Induction blocking time
- Wide operating voltage range
- Micro power consumption
- High signal output

The *Automatic induction* feature turns the output *HIGH* when object is in the sensing range. When the object is out of the sensing range the output is in *LOW* state.

The *Photosensitive control* feature is used to detect day or night. This feature is not set by default.

The *Temperature compensation* is a feature which can be used in the summertime when ambient temperature is higher. In these situations the detection range is slightly shorter. Compensation has to be used to adjust better detection performance.

Az-Delivery

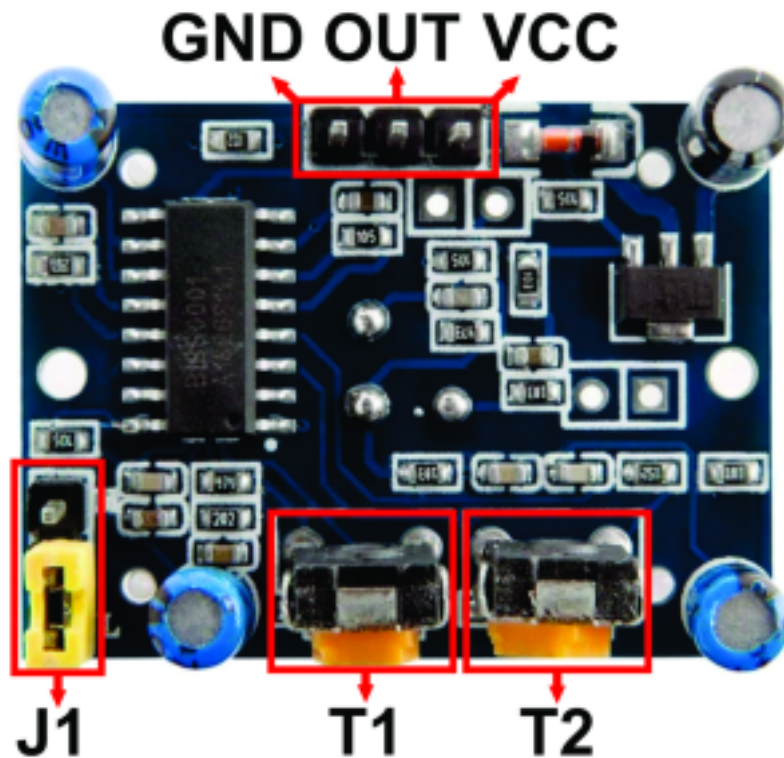
The *Two way trigger* feature has two modes which can be set by the on board jumper. The first mode is called *non-repeatable trigger*. In this mode, when the output is in the *HIGH* state after some delay the output is automatically changed to *LOW* state. The second mode is called *repeatable trigger*. If there is activity in the sensing range, the output remains in *HIGH* state until the objects leave the scanning range. When delay period ends, the output changes its state to *LOW* if there is no activity in the sensing range. The output of detected activity is prolonged because of the preset delay.

The *Induction blocking time* feature has a block delay between two measurements. The default measure delay is 2.5s. The value of this delay can be set with value from range of 0s up to a few tens of seconds. During the delay, sensor data can not be read.

The *Micropower consumption* feature is suitable for battery-powered devices and automatic control products. This feature puts the module in the stand-by mode. In this mode current consumption is less than $50\mu\text{A}$.

The *Output high signal* feature enables an easy integration with various types of circuits. With 3.3V at the output pin, it is easily interfaced with other devices.

Adjusting hardware features



The HC-SR501 module has two trimpots, T1 and T2. The trimpot T1 is used to adjust sensitivity and the trimpot T2 is used to adjust output timing delay.

The jumper J1 is used for temperature compensation settings.

To use the module with a 3.3V there is a simple mod which bypasses the on-board voltage regulator. The purpose of this mod is to enable the module to work with the devices that only have 3.3 V power supply pins.

Az-Delivery

To use the module in this mod simply remove the jumper J1 and connect the power supply cable to the H pin (pin 1 of J1).

There are two trimpots on-board the module and they are used for sensitivity and delay adjustment. Sensitivity adjustment can be done by rotating the trimpot T1 clockwise (CW) where distance sensing range increases to around 7 meters. Rotating the trimpot T1 counter clockwise (CCW), reduces the distance sensing range to about 3 meters. Delay adjustment is done by rotating the trimpot T2 clockwise which increases the delay time up to 300 seconds and by rotating the trimpot counter clockwise, it shortens the delay to about 5 seconds.

When the sensor module is powered up, it needs one minute to get operational. This is called initialization time interval. In this interval, the module outputs signal 0-3 time and after this interval it enters the stand-by mode.

Other light sources should be avoided when working with the module. This is because it can cause interference if close to the module. The working environment of the module should be without wind because it can cause temperature differences which interferes with the measurements.

Az-Delivery

Specifications

Power supply voltage:	5V DC
Operational temperature:	from -15 to +70°C
Lock time:	200 milliseconds
Sensing range:	110 degrees
Sensing distance:	up to 7 meters
Block time:	from 2.5sec. (default) up to 10 sec.
Trigger methods:	L - disable, H - enable repeat trigger
TTL output:	LOW – 0V, HIGH – 3.3V (logic levels)
Quiescent current:	less than 50µA
Power consumption:	65mA
Lens diameter:	23mm
Dimensions:	33 x 25 x 30mm [1.3 x 1 x 1.2in]

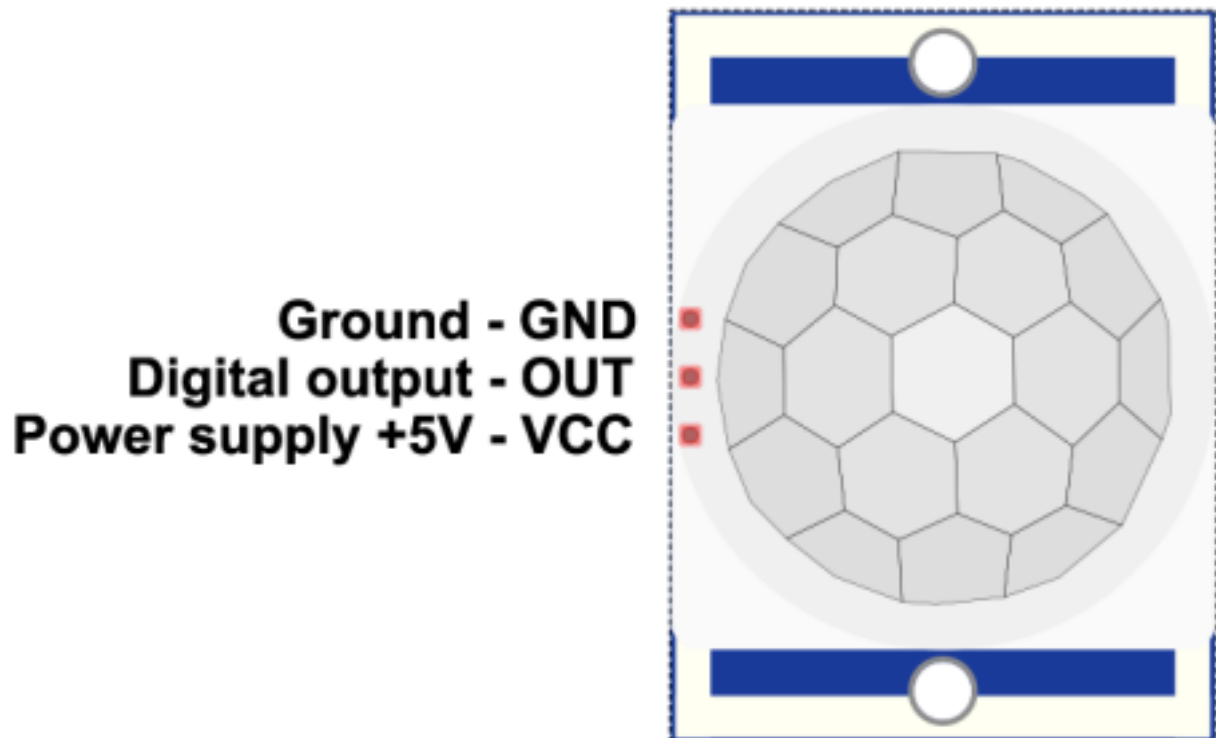
The delay time determines how long the module keeps the output *HIGH* after detecting motion.

The block time is the time interval where the sensor is disabled or do not detect motion. The default block time for the sensor is 2.5 seconds.

Az-Delivery

The pinout

The HC-SR501 PIR module has a three pins. The pinout is shown on the following image:



Note: Preferred operating power supply voltage range is 5V. The output voltage on the digital pin is in 3.3V range.

Note: When the module is connected to the power supply, the sensor takes from 30 up to 60 seconds to warm up, stabilize.

Az-Delivery

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.

Download the Arduino IDE



The screenshot shows the Arduino IDE download page for version 1.8.9. On the left, there is a teal circular logo with a white infinity symbol containing a minus sign on the left and a plus sign on the right. To the right of the logo, the text reads: **ARDUINO 1.8.9**
The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

On the right side of the page, there are several download options listed in a teal box with white text: **Windows** installer, for Windows XP and up
Windows ZIP file for non-admin install
Windows app Requires Win 8.1 or 10
Get (with a Windows logo icon)
Mac OS X 10.8 Mountain Lion or newer
Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits
Below these options are links for **Release Notes**, **Source Code**, and **Checksums (SHA256)**.

For

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

Az-Delivery

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*.

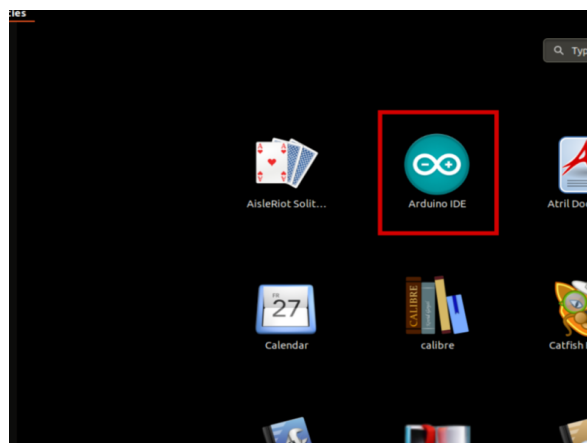
To run the first script in the terminal, open the terminal in the extracted directory and run the following command:

```
sh arduino-linux-setup.sh user_name
```

user_name - is the name of a superuser in the 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* script has to be used after 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.



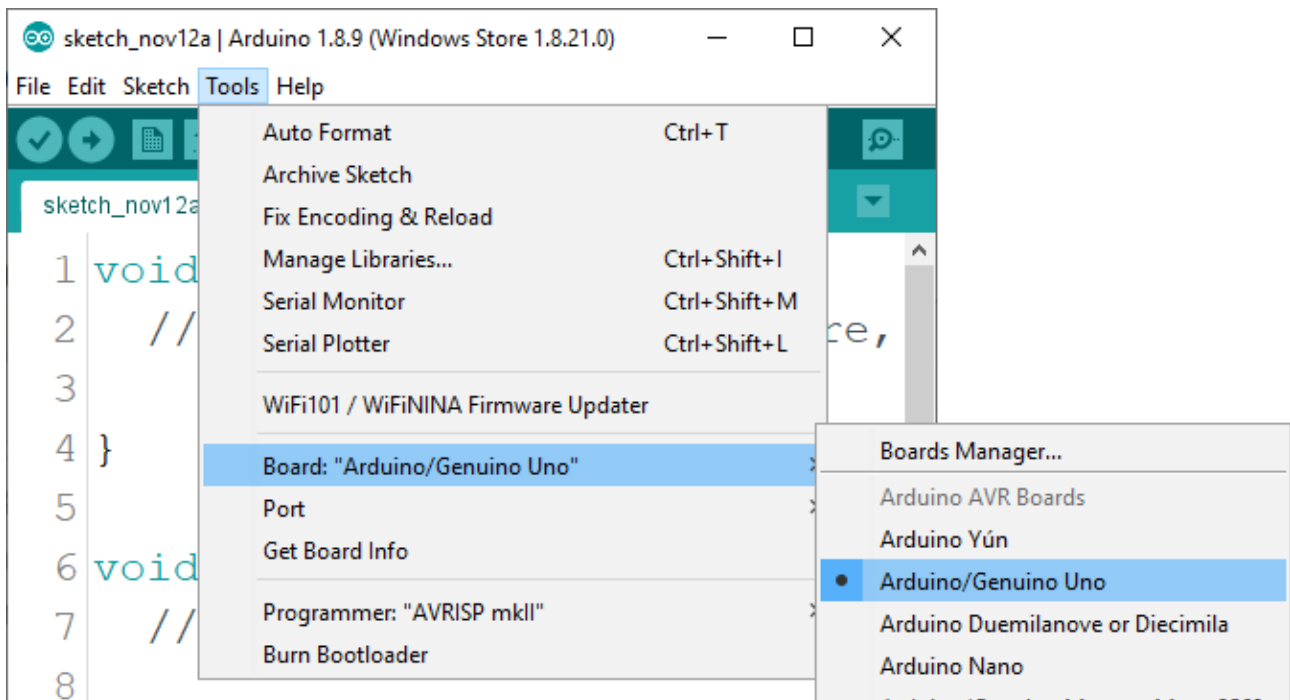
Az-Delivery

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 the microcontroller board. Open freshly installed Arduino IDE, and go to:

Tools > Board > {your board name here}

{your board name here} should be the *Microcontroller/Genuino Uno*, as it can be seen on the following image:



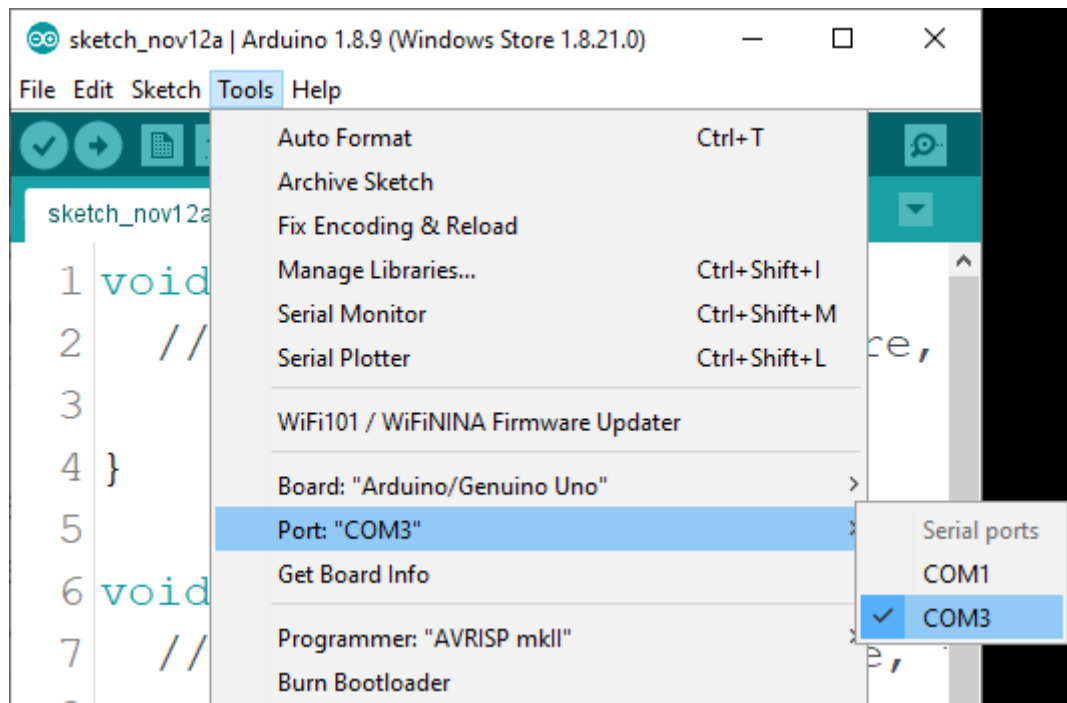
The port to which the board is connected has to be selected. Go to: *Tools > Port > {port name goes here}*

and when the microcontroller board is connected to the USB port, the port name can be seen in the drop-down menu on the previous image.

Az-Delivery

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:

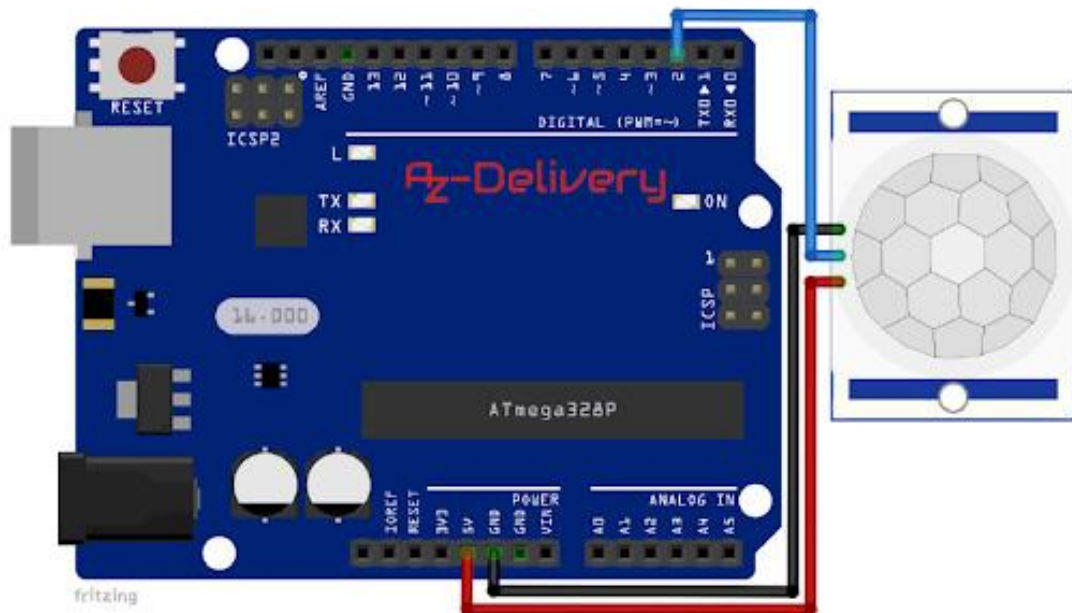
[Raspberry Pi Quick Startup Guide](#)

The *Raspbian* operating system comes with *Python* preinstalled.



Connecting the module with ATmega328p microcontroller

Connect the HC-SR501 PIR module with the ATmega328p as shown on the following connection diagram:



PIR pin	MC pin	Wire color
VCC	5V	Red wire
OUT	D2	Blue wire
GND	GND	Black wire

AZ-Delivery

Sketch example

```
#define OUT_PIN 2
uint8_t output_value = 0;
bool motion_detected = false;
void setup() {
  Serial.begin(9600);
  pinMode(OUT_PIN, INPUT);
```

```
    delay(60000);  
  }  
  void loop() {  
    output_value = digitalRead(OUT_PIN); if  
(output_value) {  
      Serial.println("Object detected!"); motion_detected  
= true;  
      delay(3000);  
    } else {  
      Serial.println("No object!");  
    }  
    if (motion_detected) {  
      Serial.println("Wait!");  
      delay(6000);  
      Serial.println("Ready...");  
      motion_detected = false;  
    }  
  }  
}
```

AZ-Delivery

Upload the sketch to the ATmega328p and run the Serial Monitor (*Tools > Serial Monitor*). The result should look like as on the following image:


```
COM3
Wit!
Object detected!
Wait!
Ready...
Object detected!
Wait!
Ready...
Object detected!
Wait!
Ready...
Object detected!
Wait!
Ready...
Object detected!
Wait!
Ready...
Object detected!
Wait!
```

Az-Delivery

Sketch starts with creating a macro called *OUT_PIN*. It represents the digital output pin of ATmega328p that is used for connecting the sensor output pin.

Next, two variables called *output_value* and *motion_detected* are created. The variable *output_value* store the state of the output pin. The *motion_detected* variable is used for displaying the messages in the serial monitor.

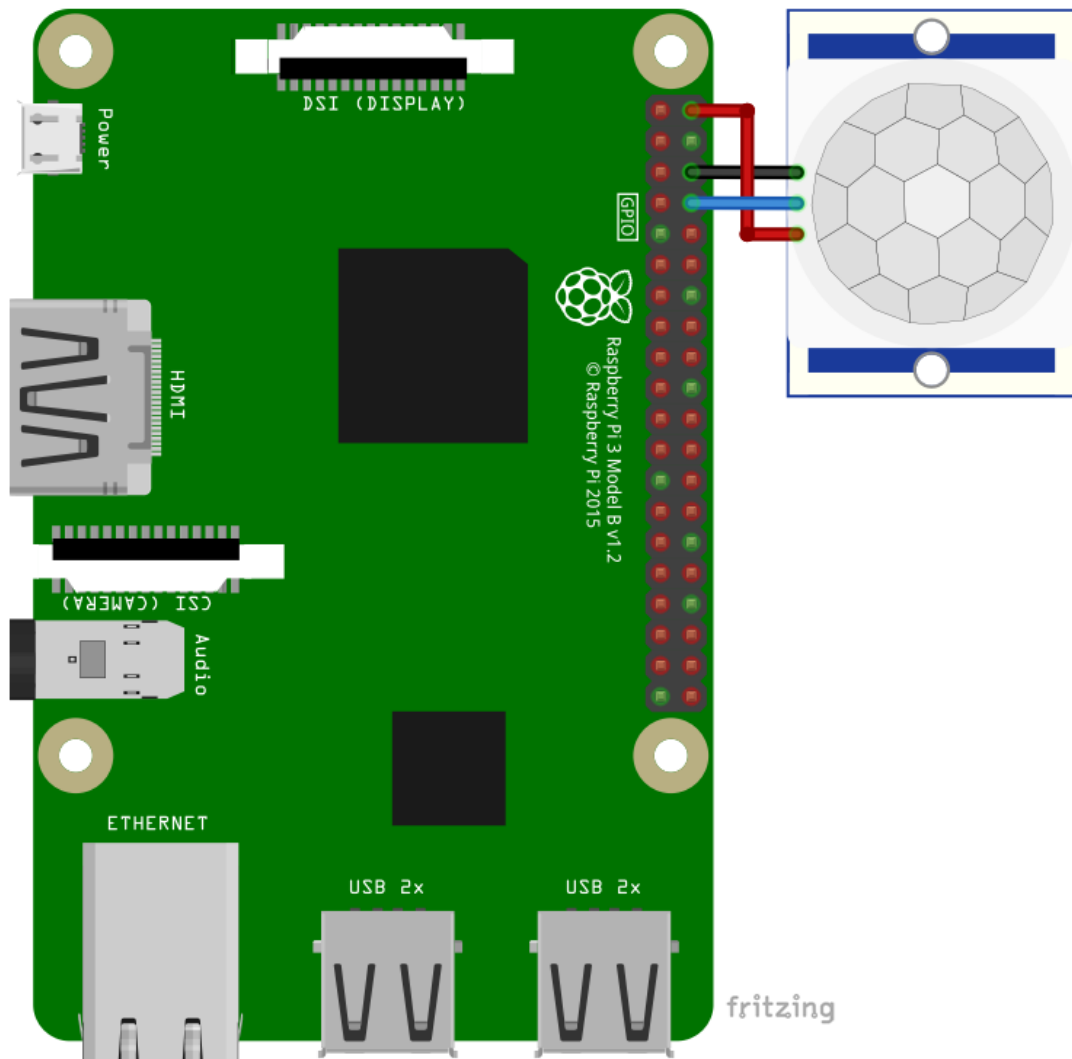
In the *setup()* function the serial communication is started with a baud rate of *9600bps*. Then the pin mode of the *OUT_PIN* is set to *INPUT*. At the end of the *setup()* function the delay is set to *60s (60.000ms)* which is used for warm up of the sensor.

In the *loop()* function the state of the *OUT_PIN* is read and checked. If it is in *HIGH* state, the message *Object detected!* is displayed in the serial monitor and the state of the *motion_detected* variable is set to *true*. If the value of *OUT_PIN* is in *LOW* state, message *No object!* is displayed in the serial monitor. When the value stored in the *motion_detected* variable is equal to *true*, the message *Wait!*, followed by delay of six seconds, is displayed in the serial monitor. After this, the message *Ready...* is displayed in the serial monitor.

The logo for AZ-Delivery, featuring the text "AZ-Delivery" in a bold, red, sans-serif font. The "AZ" is stylized with a horizontal line through the middle of the letters.

Connecting the module with Raspberry Pi

Connect the HC-SR501 PIR module with the Raspberry Pi as shown on the following connection diagram:



RTC pin	Raspberry Pi pin	Physical pin	Wire color
VCC	5V	2	Red wire
OUT	GPIO14	8	Blue wire
GND	GND	6	Black wire

Az-Delivery

Python script

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
PIR_PIN = 4 # Assign GPIO4 pin 7 to PIR
GPIO.setup(PIR_PIN, GPIO.IN) # Setup GPIO pin PIR as input print('Sensor
initializing . . .')
time.sleep(60) # Give sensor time to start-up, 60 seconds print('Active')

def pir(pin):
    print('Motion Detected!')

GPIO.add_event_detect(4, GPIO.FALLING, callback=pir, bouncetime=100)

print('[Press Ctrl + C to end program!])
try:
    while True:
        time.sleep(0.001)

except KeyboardInterrupt:
    print('\nScript ended')

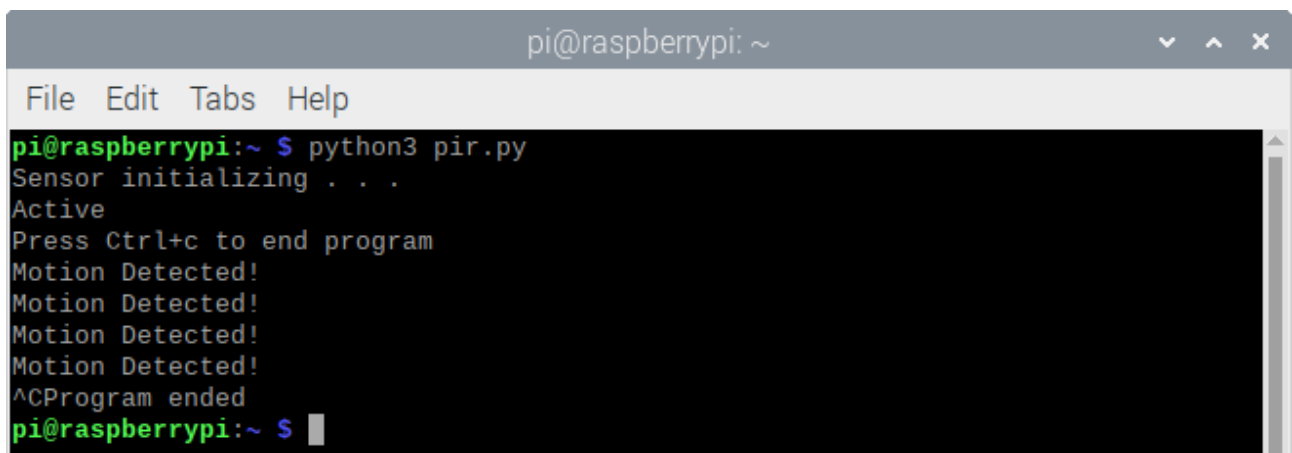
finally:
    GPIO.cleanup()
```

Az-Delivery

Save the script by name *pir.py*. To run the script, open terminal in the directory where you save the script and run the following command:

python3 pir.py

The result should look like the output on the following image:



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ python3 pir.py  
Sensor initializing . . .  
Active  
Press Ctrl+c to end program  
Motion Detected!  
Motion Detected!  
Motion Detected!  
Motion Detected!  
^CProgram ended  
pi@raspberrypi:~ $
```

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

Az-Delivery

The script starts with importing two libraries, *Rpi.GPIO* and *time*.

Then, the GPIO pin names are set to BCM and all warnings regarding GPIO interfaces are disabled.

Next, the pin mode of the GPIO pin 4 is set to input, followed by the delay of 60 seconds for warm up of the sensor.

After this, the function called *pir()* is created. The function has one argument and returns no value. The argument is not used in the script so it is not explained. In the function the message *Motion Detected!* is set to be displayed on the function execution.

The interrupt routine is set with the following line of code:

```
GPIO.add_event_detect(4, GPIO.FALLING, callback=pir, bouncetime=100)
```

Where number *4* represents the GPIO pin; *GPIO.FALLING* represents on which edge of the digital signal the *pir()* function is executed, in this case on falling edge of the digital signal; *callback=pir* sets which function is executed on the falling edge of the signal; *bouncetime=100* represents how long is the pause between the falling edge and the beginning of the function execution, in this case it is *100* milliseconds. With this pause the bouncing part of the signal is skipped and only the state of the signal after bouncing is read.

Az-Delivery

Next, the *try-except-finally* block of code is created. In the *try* block of code the indefinite loop is created where a pause of 100 microseconds is created, so that the loop itself does not block the Raspberry Pi.

The *except* block of code is executed when the 'CTRL + C' on the keyboard is pressed. This is called keyboard interrupt. When the *except* block of code is executed, the message *Script end!* is displayed in the terminal.

The *finally* block of code is executed on the script end. In this block of code all of the interfaces and pin modes that are previously set up are disabled with executing the *cleanup()* function.

AZ-Delivery

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 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>