

# AZ-Delivery

## A warm welcome!

Thank you very much, for choosing in favour of our *AZ-Delivery HC-SR501 PIR module*. On the following pages, we will show you how to use and set up this practical device .

**Have fun!**



## Areas of application

Education and teaching: Use in schools, universities and training centres to teach the basics of electronics, programming and embedded systems. Research and development: Use in research and development projects to create prototypes and experiments in the fields of electronics and computer science. Prototype development: Use in the development and testing of new electronic circuits and devices. Hobby and maker projects: Use by electronics enthusiasts and hobbyists for the development and realisation of DIY projects.

## Required knowledge and skills

Basic understanding of electronics and electrical engineering. Knowledge of programming, especially in the programming language C/C++. Ability to read circuit diagrams and design simple circuits. Experience in working with electronic components and soldering.

## Operating conditions

The product may only be operated with the voltages specified in the data sheet in order to avoid damage. A stabilised direct current source is required for operation. When connecting to other electronic components and circuits, the maximum current and voltage limits must be observed in order to avoid overloads and damage.

## Environmental conditions

The product should be used in a clean, dry environment to avoid damage from moisture or dust. Protect the product from direct sunlight (UV)

## Intended use

The product is designed for use in educational, research and development environments. It is used for the development, programming and prototyping of electronic projects and applications. The sensor product is not intended as a finished consumer product, but as a tool for technically skilled users, including engineers, developers, researchers and students.

## Non-intended foreseeable use

The product is not suitable for industrial use or safety-relevant applications. The product is not authorised for use in medical devices or for the purposes of air and space technology

## Waste disposal

Do not dispose of with household waste! Your product must be disposed of in an environmentally friendly manner in accordance with the European directive on waste electrical and electronic equipment. The valuable raw materials it contains can then be reused.

are supplied. The application of this directive contributes to environmental and health protection. Use the collection centre set up by your local authority for the return and recycling of old electrical and electronic equipment. WEEE-Reg. No.: DE 62624346

## Electrostatic discharge

Caution: Electrostatic discharges can damage the product. Note: Earth yourself before touching the product, for example by wearing an antistatic wristband or touching an earthed metal surface.

## Safety instructions

Although our product complies with the requirements of the RoHS Directive (2011/65/EU) and does not contain any hazardous substances in quantities exceeding the permitted limits, residues may still be present. Observe the following safety instructions to avoid chemical hazards: Caution: Soldering can produce vapours that can be harmful to health. Note: Use a soldering fume extractor or work in a well-ventilated room.

area. Wear a respirator if necessary. Caution: Some people may be sensitive to certain materials or chemicals contained in the product. Note: If skin irritation or allergic reactions occur, discontinue use and consult a doctor if necessary. Caution: Keep the product out of the reach of children and pets to avoid accidental contact and ingestion of small parts. Note: Store the product in a secure, closed container when not in use. Caution: Avoid contact of the product with food and drink. Note: Do not store or use the product near food to prevent contamination. Although our product fulfils the

If the product complies with the requirements of the RoHS Directive (2011/65/EU) and does not contain any hazardous substances in quantities exceeding the permitted limits, residues may still be present. Observe the following safety instructions to avoid chemical hazards: Caution: Soldering can produce vapours that can be harmful to health. Note: Use a soldering vapour extractor or work in a well-ventilated area. Wear a breathing mask if necessary. Caution: Some people may be sensitive to certain materials or chemicals contained in the product. Note: If skin irritation or allergic reactions occur, discontinue use and consult a doctor if necessary. Caution: Keep the product out of the reach of children and pets to avoid accidental contact and ingestion of small parts. Note: Keep

Store the product in a secure, closed container when not in use. Caution: Avoid contact of the product with food and drink. Note: Do not store or use the product near food to prevent contamination. The product contains sensitive electronic components and sharp edges. Improper handling or installation can lead to injury or damage. Observe the following safety instructions to avoid mechanical hazards: Caution: The circuit board and the connections of the product may have sharp edges. Proceed with caution to avoid cutting injuries. Note: Wear suitable protective gloves when handling and installing the product. Caution: Avoid excessive pressure or mechanical stress on the circuit board and components. Note: Only mount the product on stable and level surfaces. Use suitable spacers and housings to minimise mechanical stress. Caution: Ensure that the product is securely fastened to prevent it from accidentally slipping or falling. Note: Use a suitable base or secure fastening in housings or on mounting plates. Caution: Ensure that all cable connections are securely and correctly connected to prevent tensile loads and accidental disconnection. Note: Route cables so that they are not live and do not pose a tripping hazard. The product works with electrical voltages and currents that can lead to electric shocks, short circuits or other hazards if used improperly. Observe the following safety instructions to avoid electrical hazards: Caution: Only use the product with the specified voltages. Note: The power limits of the product can be found in the corresponding data sheet Caution: Avoid short circuits between the connections and components of the product Note: Ensure that no conductive objects touch or bridge the circuit board. Use insulated tools and observe the arrangement of the connections.

Caution: Do not carry out any work on the product when it is connected to a power source. Note: Disconnect the product from the power supply before making any changes to the circuit or connecting or disconnecting components. Caution: Do not exceed the specified current levels for the inputs and outputs of the product. Note: The power limits of the product can be found in the technical specifications or in the data sheet Caution: Ensure that the power sources used are stable and correctly dimensioned. Note: Only use tested and suitable power supply units to avoid voltage fluctuations and overloads. Caution: Maintain sufficient distance from live parts to avoid unintentional contact. Note: Ensure that the cabling is arranged safely and clearly according to the voltage used. Caution: Use insulating housings or protective covers to protect the product from direct contact. Note: Place the product in a non-conductive housing to prevent accidental contact and short circuits. The product and the components on it can heat up during operation. Improper handling or overloading of the product can lead to burns, damage or fire. Observe the following safety instructions to avoid thermal hazards: Caution: Ensure that the product is used within the recommended operating temperatures. Note: The recommended operating temperature range is typically between -40°C and

+85°C. Check the specific information in the product data sheet. Caution: Do not place the product near external heat sources such as radiators or direct sunlight. Note: Ensure that the product is operated in a cool and well-ventilated area. Caution: Ensure that the product is well ventilated to prevent overheating. Note: Use fans or heat sinks if the product is operated in a closed housing or in an environment with limited air circulation. Caution: Mount the product on heat-resistant surfaces and in heat-resistant housings. Note: Use materials for enclosures that can withstand high temperatures to avoid damage or fire hazards. Caution: Implement temperature monitoring when using an enclosure and, if necessary, protective mechanisms that switch off the product if it overheats. Note: Use temperature sensors and appropriate software to monitor the temperature of the product and switch off the system if necessary. Caution: Avoid overloading, which can lead to excessive heating of the components. Note: Do not exceed the specified limit values.

The current and voltage are regulated to prevent overheating. Caution: Short circuits can generate considerable heat and cause fires. Note: Ensure that all connections are correct and secure and that no conductive objects can inadvertently cause short circuits.



## **Table of contents**

<b>Introduction</b>	<b>3</b>
<b>Properties</b>	<b>4</b>
<b>Setting the hardware functions</b>	<b>7</b>
<b>Specifications</b>	<b>9</b>
<b>The pin assignment</b>	<b>10</b>
<b>Setting up the Arduino IDE</b>	<b>11</b>
<b>How to set up the Raspberry Pi and Python</b>	<b>16</b>
<b>Connecting the module to the ATmega328p microcontroller</b>	<b>17</b>
<b>Example sketch</b>	<b>17</b>
<b>Connecting the module to the Raspberry Pi</b>	<b>21</b>
<b>Python script</b>	<b>22</b>



## Introduction

The HC-SR501 PIR module is a device that can recognise changes in infrared light. For example, human activity if it is within the detection range of the module. The module is based on infrared technology and uses a pyroelectric passive infrared sensor, or PIR for short. It consists of a pyroelectric sensor and some additional electronic components. A lens is used to increase the detection angle by focussing the ambient light.

When the crystalline element is exposed to infrared light, the output voltage changes directly with the intensity of the infrared light. The voltage output by the sensor is in the range of *mV*. This voltage is amplified by the built-in amplifier so that the microcontroller can utilise it.

The module is equipped with a so-called Fresnel lens, a special filter that focusses the infrared radiation onto the sensor.

The module can be used with various types of devices such as lighting devices, intercoms, electric fans and other household appliances. It is widely used in homes, businesses, hotels, shops and sensitive areas where lighting automation and security systems are required.

It has a high sensitivity, high reliability and works with very low voltage. - 3 -



## Properties

- Automatic induction
- Light-sensitive control
- Temperature compensation
- Two-way release
- Induction blocking time
- Wide operating voltage range
- Micro power consumption
- High signal power

The *automatic* induction function switches the output to *HIGH* if the object is within the detection range. If the object is outside the detection range, the output is *switched* to *LOW*.

The *light-sensitive control* is used to recognise day or night. This function is not set by default.

*Temperature compensation* is a function that can be used in summer when the ambient temperature is higher. In these situations, the detection range is slightly shorter. The compensation must be used to set a better detection performance.



The two-way trigger function has two modes that can be set using the jumper on the circuit board. The first mode is the *non-repeatable trigger*. In this mode, the output is automatically set to the LOW state after a certain delay when it is in the *HIGH state*. The second mode is called *repeatable trigger*. If there is activity in the detection area, the output remains in the *HIGH state* until the objects leave the detection area. After the delay time has elapsed, the output changes to the *LOW state* if there is no activity in the detection area. The output of the detected activity is delayed by the preset delay.

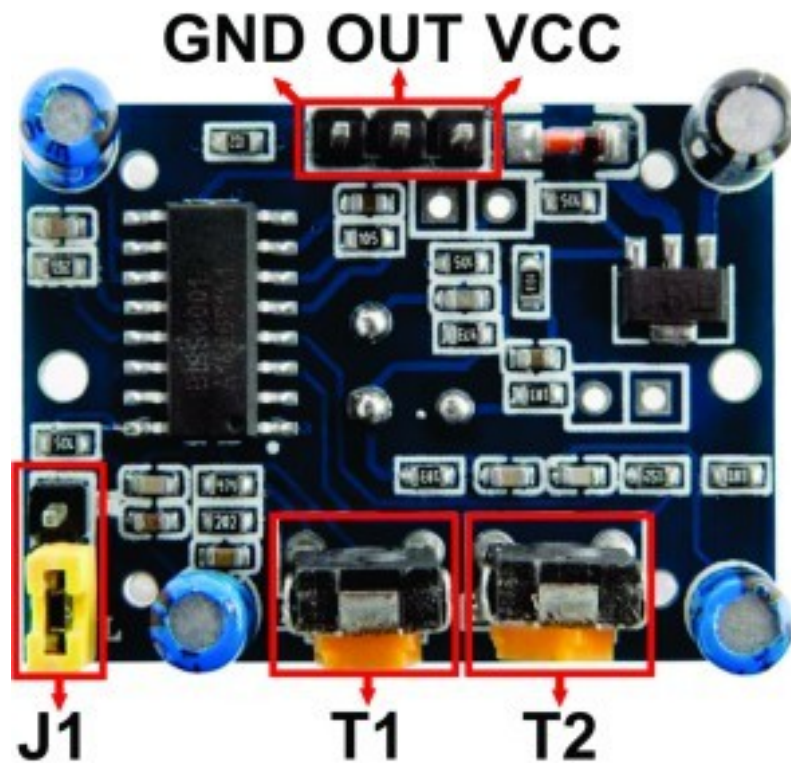
The *induction blocking time* function has a blocking delay between two measurements. The standard measurement delay is 2.5s. The value of this delay can be set in the range from 0s to several tens of seconds. The sensor data cannot be read during the delay.

The micropower consumption function is suitable for battery-operated devices and automatic control products. This function puts the module into stand-by mode. In this mode, the current consumption is less than  $50\mu A$ .

The *output high signal* allows easy integration into various types of circuits. With 3.3 V at the output pin, it can be easily connected to other devices.



## Customisation of hardware functions



The HC-SR501 module has two trim potentiometers, T1 and T2. The sensitivity is set with the trimming potentiometer T1 and the delay of the output with the trimming potentiometer T2.

The Jumper J1 is used for the setting the temperature compensation.

To use the module with 3.3 V, there is a simple mod that bypasses the built-in voltage regulator. The purpose of this mod is to allow the module to work with devices that only have 3.3 V power supply pins.





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

There are two trim potentiometers on the module which are used to adjust the sensitivity and delay. The sensitivity can be adjusted by turning the trim potentiometer T1 clockwise (CW), which increases the distance measuring range to approx. 7 metres. Turning the trim potentiometer T1 anti-clockwise (CCW) reduces the distance range to approx. 3 metres. The delay is set by turning the trim potentiometer T2 clockwise, which increases the delay time to up to 300 seconds, and turning the trim potentiometer anti-clockwise reduces the delay to around 5 seconds.

When the sensor module is switched on, it needs one minute to be ready for operation. This period is called the initialisation time. During this interval, the module outputs the times 0-3, and after this interval it switches to standby mode.

Other light sources should be avoided when working with the module. This is because they can cause interference in the vicinity of the module. The working environment of the module should be windless, as this can lead to temperature differences that affect the measurements.



### Specifications

Voltage of the power supply:	5V DC
Operating temperature:	from -15 to +70°C
Locking period:	200 milliseconds
Detection range:	110 degrees
Detection range:	up to 7 metres
Block time:	from 2.5 sec. (standard) up to 10 sec.
trigger methods:	L - Deactivate, H - Activate trigger repetition
TTL output:	LOW - 0V, HIGH - 3.3V (logic level)
quiescent current:	less than 50µA
Power consumption:	65mA
Diameter of the lens:	23mm
Dimensions:	33 x 25 x 30mm [1.3 x 1 x 1.2in]

The delay time determines how long the module *sets* the output *to HIGH* holds

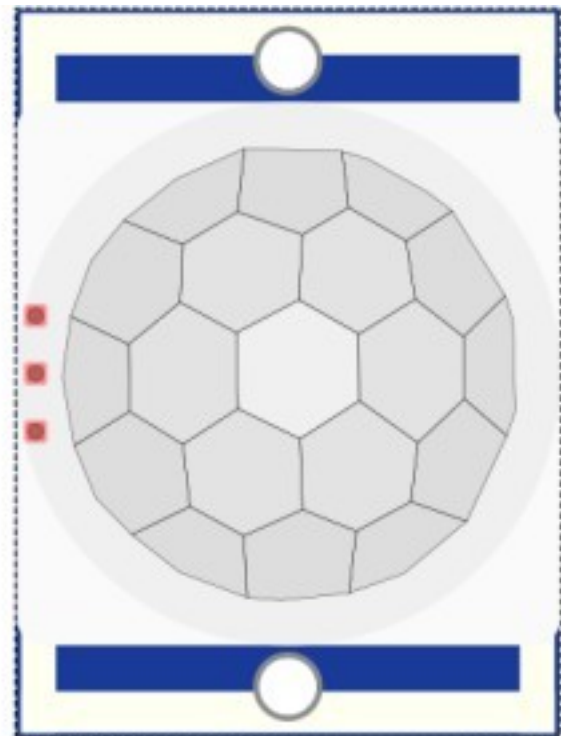
after a movement has been detected.

The blocking time is the time interval in which the sensor is deactivated or does not detect any movement. The standard blocking time for the sensor is 2.5 seconds.

## The pin assignment

The HC-SR501 PIR module has three pins. The pin assignment is shown in the following illustration:

**Ground - GND**  
**Digital output - OUT**  
**Power supply +5V - VCC**

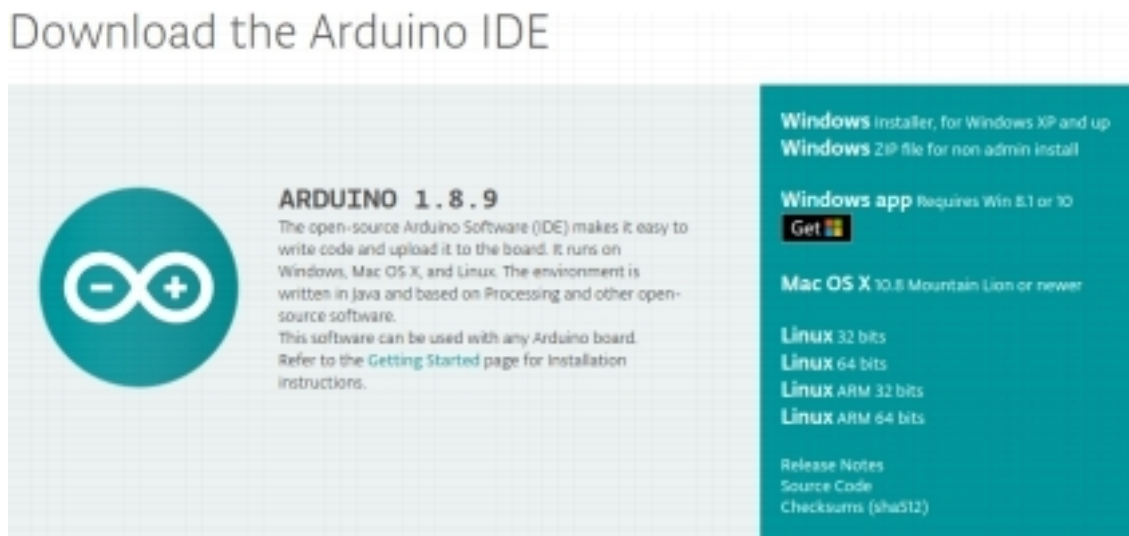


**Note:** The preferred operating voltage range is 5 V. The output voltage at the digital pin is in the range of 3.3 V.

**Note:** When the module is connected to the power supply, the sensor requires between 30 and 60 seconds to warm up and stabilise.

## To set up the Arduino IDE

If the Arduino IDE is not installed, follow the [link](#) and download the installation file for the operating system of your choice.



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 must be unpacked. After unpacking, change to the unpacked directory and open the terminal in this directory. Two *.sh* scripts must be executed, the first is called *arduino-linux-setup.sh* and the second is called *install.sh*.

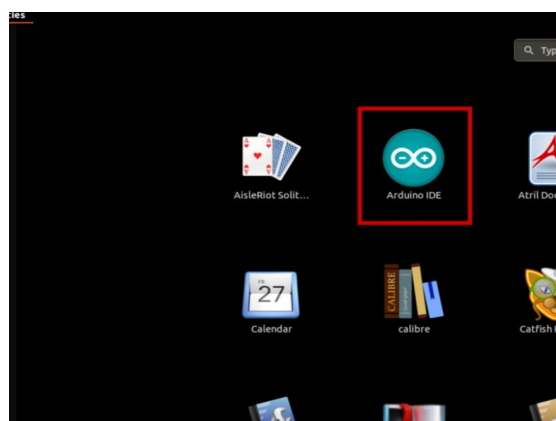
To execute the first script in the terminal, open the terminal in the extracted directory and execute the following command:

**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 must be entered when the command is issued. Wait a few minutes until the script has completed everything.

The second script named *install.sh* must be used after the installation of the first script. Execute the following command in the terminal (extracted directory): **sh install.sh**

After installing these scripts, go to *All apps* where the *Arduino IDE* is installed.



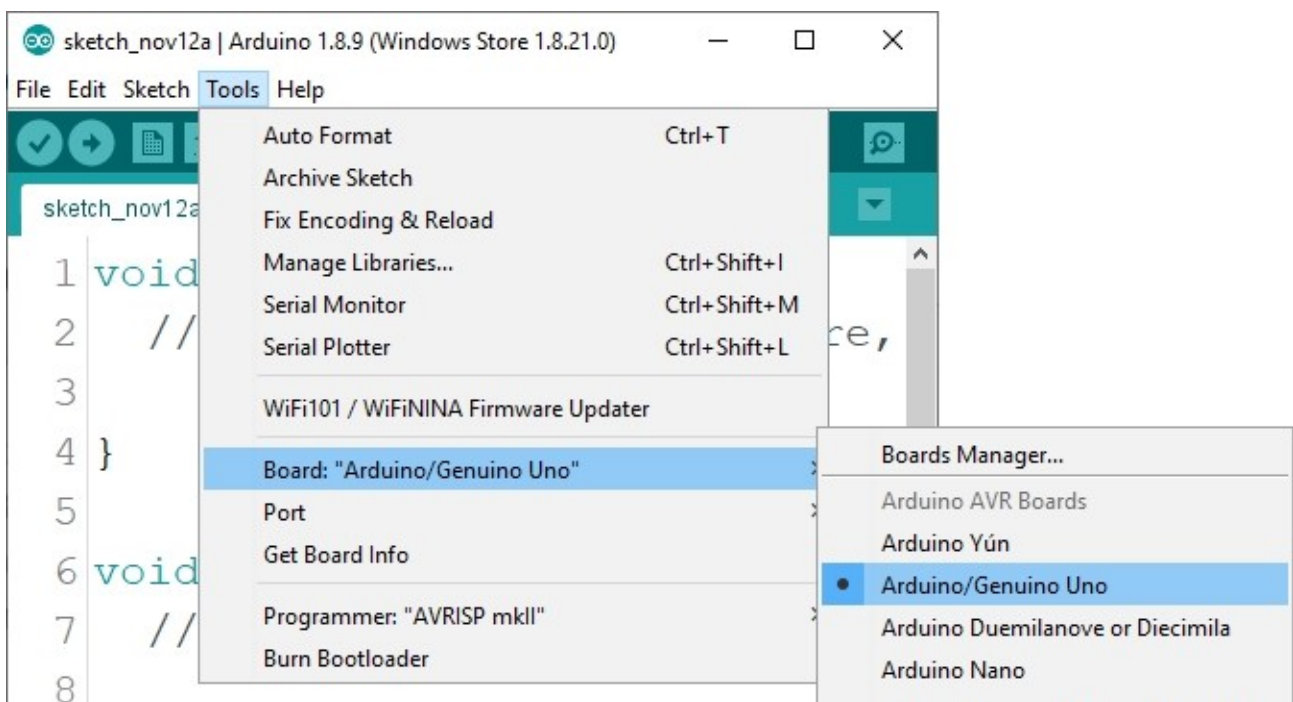
# Az-Delivery

A text editor is pre-installed on almost all operating systems (e.g. *Windows* with *Notepad*, *Linux Ubuntu* with *Gedit*, *Linux Raspbian* with *Leafpad* etc.). All of these text editors are perfectly adequate for the purpose of the e-book.

Next, you need to check whether your PC can recognise the microcontroller board. Open the freshly installed Arduino IDE and go to :

*Tools > Board > {your board name here}*

*{your board name here}* should be the *microcontroller/Genuino Uno* as shown in the following picture:

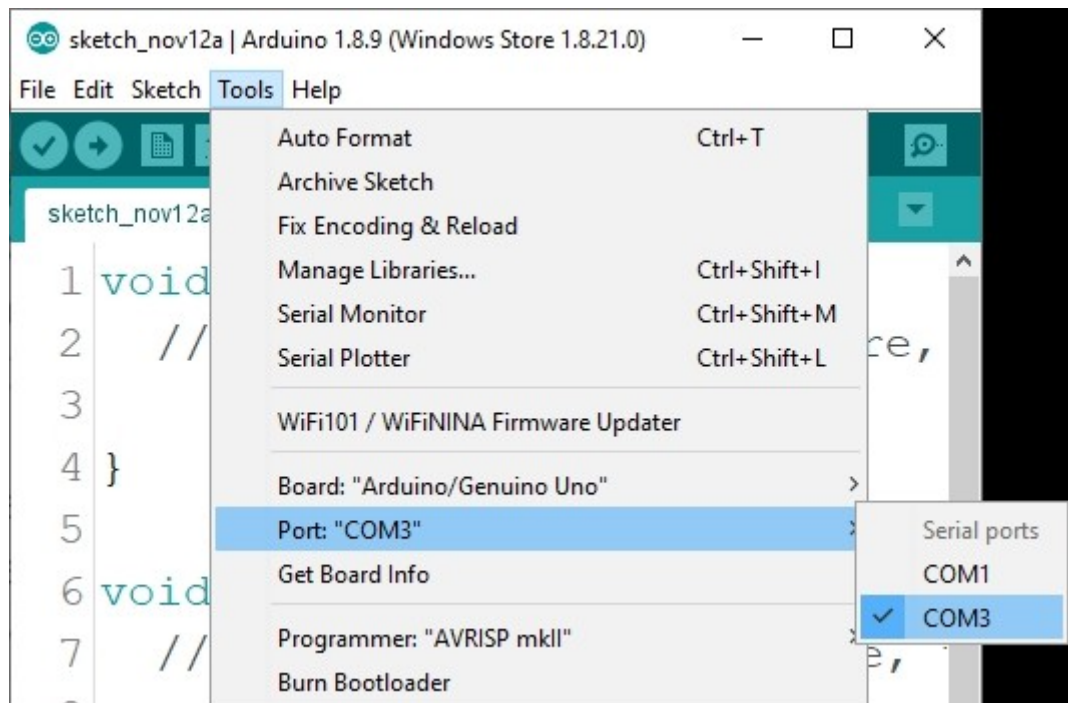


The port to which the card is connected must be selected. Go to: *Tools > Port > {name of the port goes here}*

and if the microcontroller board is connected to the USB port, the name of the port can be displayed in the drop-down menu on the previous screen.

# Az-Delivery

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



For Linux users, the name of the connection is, for example `/dev/ttyUSBx`, where *x* is an integer between 0 and 9.





## **How to set up the Raspberry Pi and Python**

The operating system must first be installed on the Raspberry Pi, then everything must be set up so that it can be used in headless mode. Headless mode enables a remote connection to the Raspberry Pi without the need for a PC screen, mouse or keyboard. The only things used in this mode are the Raspberry Pi itself, the power supply and the internet connection. All this is explained in detail in the free eBook:

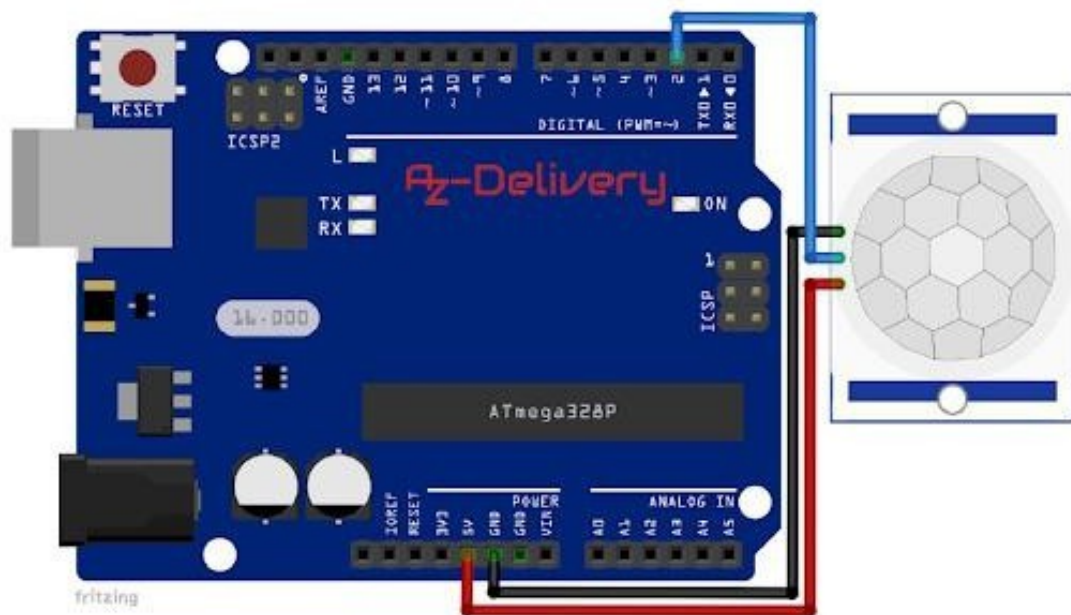
[Raspberry Pi quick start guide](#)

*Python* is already pre-installed on the Raspbian operating system.



## **Connecting the module to the ATmega328p microcontroller**

Connect the HC-SR501 PIR module to the ATmega328p as shown in the following connection diagram:



PIR pin	MC pin	Colour of the cable
VCC	5V	Red cable
OUT	D2	Blue cable
GND	GND	Black cable

# Az-Delivery

## Sketch example

```

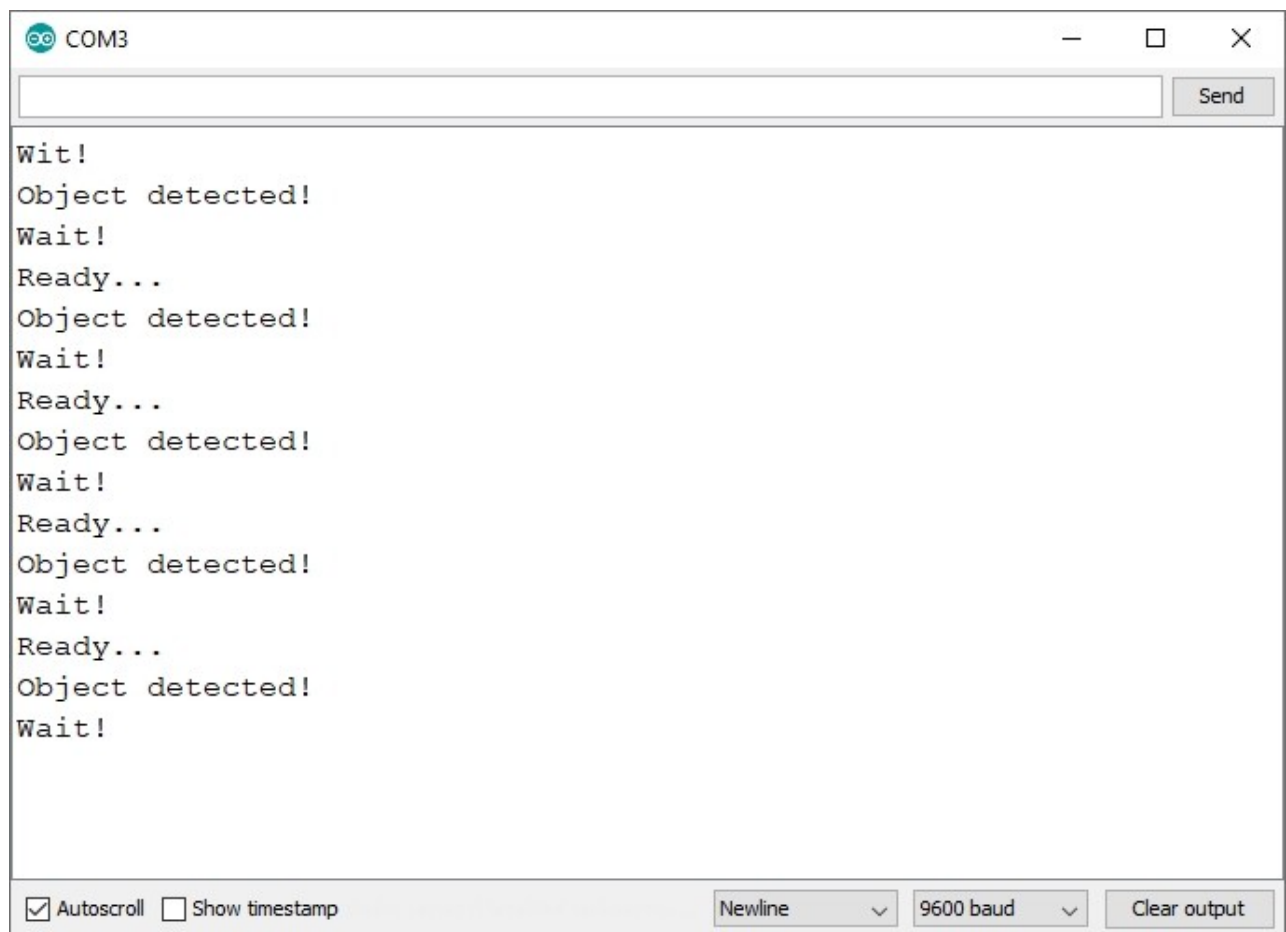
#define OUT_PIN 2
uint8_t output_value = 0;
bool motion_detected = false;
void setup() {
  Serial.begin(9600);
  pinMode(OUT_PIN, INPUT);

```

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

The logo for Az-Delivery, featuring the text "Az-Delivery" in a stylized, bold, red font. The "Az" is in a larger, more prominent font than "Delivery".

Upload the sketch to the ATmega328p and start the Serial Monitor (*Tools > Serial Monitor*). The result should look like the following picture:



# Az-Delivery

Sketch starts with the creation of a macro called *OUT\_PIN*. It represents the digital output pin of the ATmega328p, which is used to connect the sensor output pin.

Next, two variables called *output\_value* and *motion\_detected* are created. The *output\_value* variable saves the status of the output pin. The *motion\_detected* variable is used to display the messages in the serial monitor.

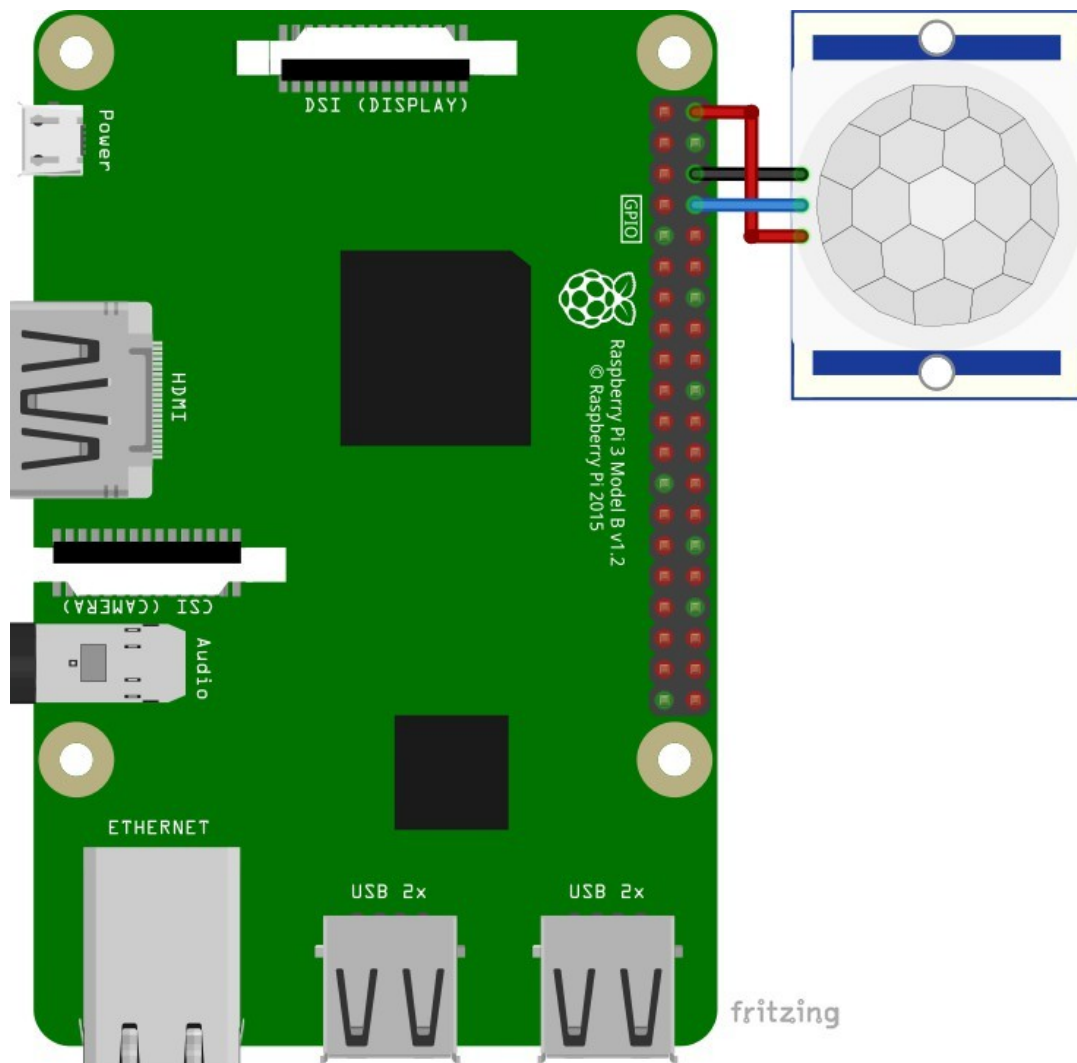
In the *setup()* function, serial communication is started with a baud rate of *9600bps*. Then the pin mode of the *OUT\_PIN* is set to *INPUT*. At the end of the *setup()* function, the delay is set to *60s (60,000ms)*, which is used to warm up the sensor.

The status of the *OUT\_PIN* is read and checked in the *loop()* function. If it is in the *HIGH* state, the message *Object detected!* is displayed in the serial monitor and the status of the *motion\_detected* variable is set to *true*. If the value of *OUT\_PIN* is in the *LOW* state, the message *No object!* is displayed in the serial monitor. If the value stored in the *motion\_detected* variable is *true*, the message *Wait!* is displayed in the serial monitor, followed by a delay of six seconds. After this, the message *Ready...* is displayed in the serial monitor.



### **Connecting the module to the Raspberry Pi**

Connect the HC-SR501 PIR module to the Raspberry Pi as shown in the following connection diagram:



RTC pin	Raspberry Pi pin	Physical needle	Colour of the cable
VCC	5V	2	Red cable
OUT	GPIO14	8	Blue cable
GND	GND	6	Black cable



## Python script

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
PIR_PIN = 4 # Assignment of GPIO4 pin 7 to PIR
GPIO.setup(PIR_PIN, GPIO.IN) # Set up GPIO pin PIR as input print('Sensor
initialised . . .')
time.sleep(60) # Give the sensor 60 seconds to start up print('Active')

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

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

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

except KeyboardInterrupt:
    print('\nScript ended')

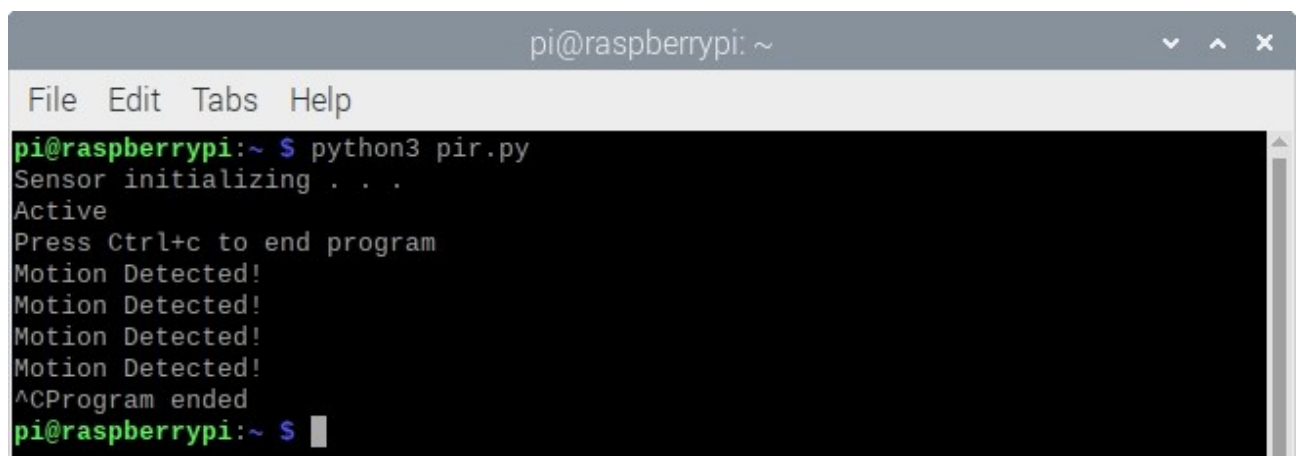
finally:
    GPIO.cleanup()
```



# Az-Delivery

Save the script under the name *pir.py*. To execute the script, open the terminal in the directory in which you saved the script and execute the following command: **python3 pir.py**

The result should look like the following picture:



```
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.



The script starts with the import of two libraries, *Rpi.GPIO* and *time*.

The GPIO pin names are then set to BCM and all warnings relating to the GPIO interfaces are deactivated.

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

A function called *pir()* is then created. The function has one argument and does not return a value. The argument is not used in the script and is therefore not explained. The *Motion Detected!* message, which is displayed when the function is executed, is set in the function.

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

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

The number *4* stands for the GPIO pin; *GPIO.FALLING* specifies at which edge of the digital signal the *pir()* function is executed, in this case at the falling edge of the digital signal; *callback=pir* specifies which function is executed at the falling edge of the signal; *bouncetime=100* specifies how long the pause is between the falling edge and the start of the function execution, in this case *100* milliseconds. With this pause, the bouncing part of the signal is skipped and only the state of the signal after the bounce is read.



Next, the try-except-finally code block is created. An infinite loop is created in the try code block, in which a pause of 100 microseconds is inserted so that the loop itself does not block the Raspberry Pi.

The exception code block is executed when the key combination 'CTRL + C' is pressed on the keyboard. This is known as a keyboard interruption. When the exception code block is executed, the message *Script end! is* displayed in the terminal.

The *final* code block is executed at the end of the script. In this code block, all interfaces and pin modes that were previously set up are deactivated by executing the *cleanup()* function.



Now it's time to learn and create your own projects. You can do this with the help of many example scripts and other instructions that you can find on the Internet.

**If you are looking for high-quality microelectronics and accessories, AZ-Delivery Vertriebs GmbH is the right address for you. You will receive numerous application examples, complete installation instructions, eBooks, libraries and support from our technical experts.**

<https://az-delivery.de>

**Have fun!**

**Imprint**

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