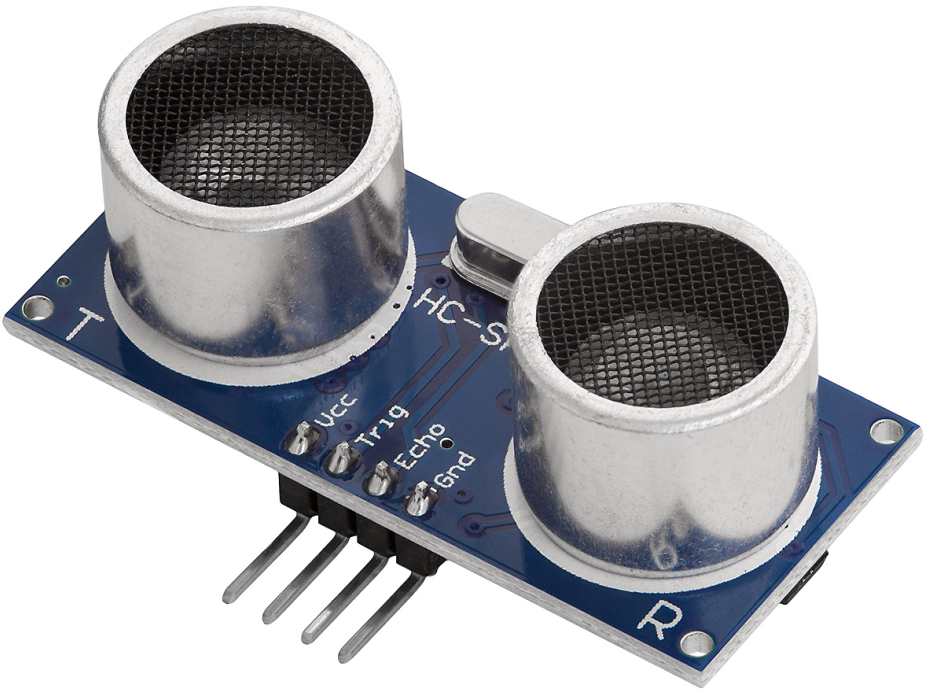
Welcome!

Thank you for purchasing our *AZ-Delivery HC-SR04 Ultrasonic Sensor Module*. On the following pages, you will be introduced to how to use and set up this handy device.

Have fun!



Areas of application
Education and teaching: Use in schools, universities and training institutions 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: Used by electronics enthusiasts and hobbyists to develop and implement DIY projects.

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

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

Environmental conditions
The product should be used in a clean, dry environment to avoid damage caused by 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 to develop, program and prototype electronic projects and applications. The Sensor product is not intended as a finished consumer product, but rather as a tool for technically savvy users, including engineers, developers, researchers and students.

Improper foreseeable use
The product is not suitable for industrial use or safety-relevant applications. Use of the product in medical devices or for aviation and space travel purposes is not permitted

disposal
Do not discard with household waste! Your product is according to the European one Directive on waste electrical and electronic equipment to be disposed of in an environmentally friendly manner. The valuable raw materials contained therein can be recycled become. The application of this directive contributes to environmental and health protection. Use the collection point set up by your municipality to return and Recycling of old electrical and electronic devices. WEEE Reg. No.: DE 62624346

electrostatic discharge
Attention: Electrostatic discharges can damage the product. Note: Ground yourself before touching the product, such as by wearing an anti-static wrist strap or touching a grounded 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 above the permitted limits, residues may still be present. Observe the following safety instructions to avoid chemical hazards: Caution: Soldering can produce fumes that can be harmful to health. Note: Use a solder fume extractor or work in a well-ventilated area. If necessary, wear a respirator mask. Caution: Some people may be sensitive to certain materials or chemicals contained in the product. Note: If skin irritation or allergic reactions occur, stop use and, if necessary, consult a doctor. Caution: Keep the product out of the reach of children and pets to avoid accidental contact and swallowing of small parts. Note: Store the product in a safe, closed container when not in use. Attention: Avoid contact of the product with food and drinks. Note: Do not store or use the product near food to prevent contamination. Although our product complies with the requirements of the RoHS Directive (2011/65/EU) and does not contain any hazardous substances in quantities above the permitted limits, residues may still be present. Observe the following safety instructions to avoid chemical hazards: Caution: Soldering can produce fumes that can be harmful to health. Note: Use a solder fume extractor or work in a well-ventilated area. If necessary, wear a respirator mask. Caution: Some people may be sensitive to certain materials or chemicals contained in the product. Note: If skin irritation or allergic reactions occur, stop use and, if necessary,

consult a doctor. Caution: Keep the product out of the reach of children and pets to avoid accidental contact and swallowing of small parts. Note: Store the product in a safe, closed container when not in use. Attention: Avoid contact of the product with food and drinks. 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 assembly can result in injury or damage. Observe the following safety instructions to avoid mechanical hazards: Attention: The product's circuit board and connectors may have sharp edges. Use caution to avoid cuts. Note: Wear appropriate protective gloves when handling and assembling the product. Caution: Avoid excessive pressure or mechanical stress on the board and components. Note: Only mount the product on stable and flat surfaces. Use appropriate spacers and housings to minimize mechanical stress. Attention: Make sure the product is securely fastened to prevent accidental slipping or falling. Note: Use appropriate support or secure mounting in enclosures or on mounting plates. Caution: Make sure all cable connections are connected securely and correctly to avoid strain and accidental unplugging. Note: Route cables so that they are not under tension and do not pose a tripping hazard. The product operates with electrical voltages and currents that, if used improperly, can result in electric shocks, short circuits or other hazards. Observe the following safety instructions to avoid electrical hazards: Attention: Use the product only with the specified voltages. Note: The performance limits of the product can be found in the associated data sheet Caution: Avoid short circuits between the connectors and components of the product Note: Make sure that no conductive objects touch or bridge the circuit board. Use insulated tools and pay attention to the arrangement of connections. Caution: Do not perform any work on the product when it is connected to a power source. Note: Disconnect the product from power before making any circuit changes or connecting or removing components. Caution: Do not exceed the specified current ratings for the product's inputs and outputs. Note: The performance limits of the product can be found in the technical specifications or in the data sheet Attention: Make sure that the power sources used are stable and correctly sized. Note: Only use tested and suitable power supplies to avoid voltage fluctuations and overloads. Attention: Maintain sufficient distance from live parts to avoid accidental 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 case to avoid accidental touching and short circuits. The product and the components on it may become warm during operation. Improper handling or overloading the product can result in burns, damage or fire. Observe the following safety instructions to avoid thermal hazards: Caution: Make sure the product is used within 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. Attention: 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. Attention: Make sure the product is well ventilated to avoid overheating. Note: Use fans or heat sinks when operating the product in a closed enclosure or in an environment with limited air circulation. Attention: Mount the product on heat-resistant surfaces and in heat-resistant housings. Note: Use enclosure materials that can withstand high temperatures to avoid damage or fire hazard. Caution: Implement temperature monitoring when using an enclosure and, if necessary, protection mechanisms that shut down the product if it overheats. Note: Note: Use temperature sensors and appropriate software to monitor the temperature of the product and shut down the system if necessary. Caution: Avoid overloads that can cause excessive heating of components. Note: To prevent overheating, do not exceed the specified current and voltage limits. Caution: Short circuits can generate significant heat and cause fires. Note: Make sure that all connections are correct and secure and that no conductive objects can accidentally cause short circuits.

Table of Contents

Introduction

The HC-SR04 ultrasonic sensor module is a device that can measure distance using ultrasonic sound waves. It can determine the distance between the module and other objects in vicinity with high accuracy.

The sensor is commonly used in obstacle avoiding devices, various distance measuring devices, automation projects, car parking systems, proximity alarms, etc.

The module consists of an ultrasonic transmitter, receiver, on-board electronic parts like amplifier chips, a crystal oscillator and other passive components like capacitors and resistors.

The operation principle of the module is the following. The ultrasonic sound waves are transmitted by the on-board transmitter. When there is an obstacle in front of the module, the ultrasonic sound waves get reflected by an obstacle back to the module. The reflected ultrasonic sound wave then gets picked up by the on-board receiver. This is then output on the ECHO pin as a digital signal with a frequency that is directly related to the time required for an ultrasonic sound wave to travel from the module to the obstacle and back to the receiver.

Calculating the distance

The module emits ultrasonic sound waves on a frequency that is higher than the human hearing range (more than 20kHz). The speed of the sound waves traveling through the air is approximately *343m/*s at room temperature (*20°C*). This speed value depends on the environmental circumstances like temperature, humidity and variations in air pressure. The speed of sound in the air actually depends much more on the temperature and very little on humidity and the pressure of air. It increases its value approximately *0.6m/s* per degree of Celsius. In most cases at a temperature of *20°C*, the value of *343 m/s* can be used. To accurately calculate the speed of the ultrasonic sound wave in the air, use the following formula:

V (m/s) = 331.3 + (0.606 × T)

where V (m/s) is the speed of ultrasonic sound wave in the air and T (°C) is the temperature of the air.

When the speed of the ultrasonic sound wave is multiplied by the time that the ultrasonic sound wave traveled from the module to the obstacle and back, the result is the distance from the module to the obstacle and back:

Distance = Speed x Time

Because the sound waves travel *343* meters per one second, to measure distances from 20mm up to 4000mm (the range of the module) the measurements are done in microseconds. So, the speed of the ultrasonic sound waves should be converted from the *m/s* into *cm/µs*, which is:

343m/s = 0.0343 cm/µs = 1/29 cm/µs

13503.9in/s = 0.0135 in/µs = 1/74 in/µs (used in the sketch)

Then, to get the actual distance between the module and obstacle, the previous result should be divided by two, because the ultrasonic sound wave travels from the module to the obstacle and back, so calculation goes like this:

Distance (cm) = Speed of sound (cm/µs) × Time (µs) / 2

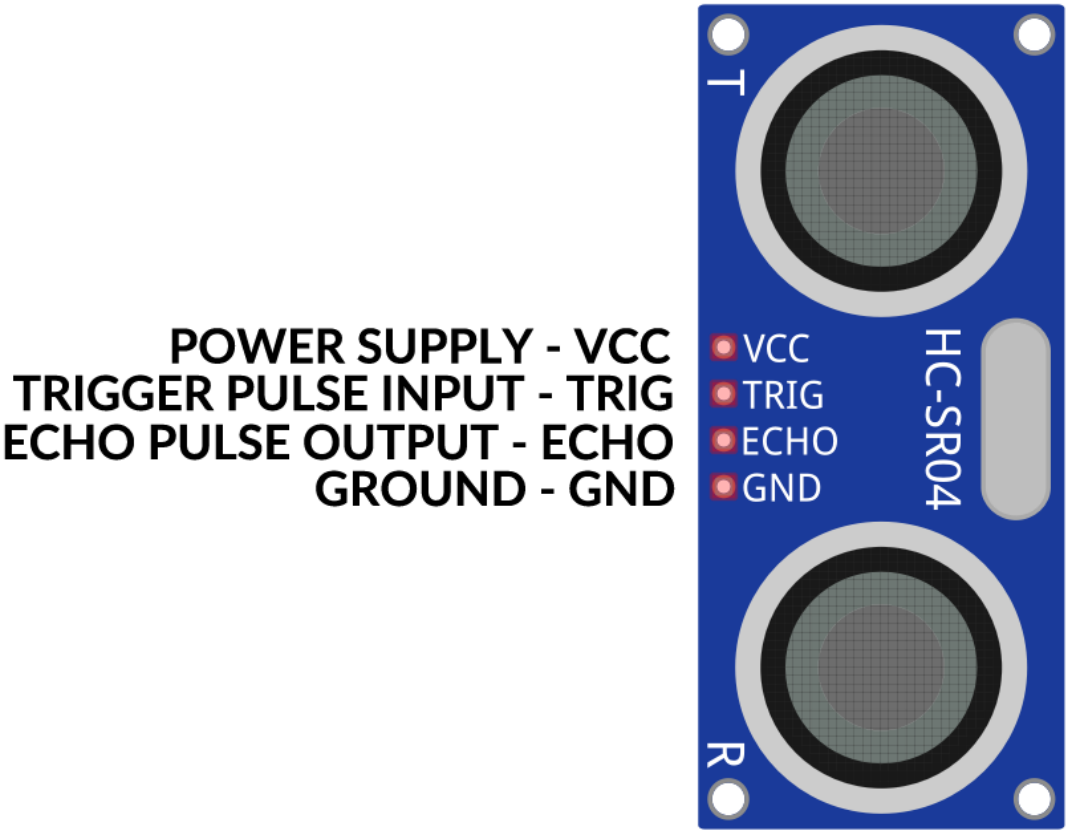
Specifications

Power supply voltage:	up to 5V
Operating voltage:	from 3V to 5V
Output voltage:	5V!
Current consumption:	15mA
Quiescent current:	less than 2mA
Ultrasonic frequency:	40kHz
Trigger input signal:	10µS TTL pulse
Measuring angle:	30 degree
Effectual angle:	less than 15 degrees
Operating distance range:	from 20mm to 4000mm (1in to 13 feet)
Claimed precision:	3mm, realistically 10mm
Dimensions:	45 x 20 x 15mm (1.8 x 0.8 x 0.6in)

The minimal distance on which the measurements are valid is *20mm*. Below this threshold the readings may become unpredictable.

The pinout

The HC-SR04 module has four pins. The pinout is shown on the following image:



Note: The output voltage of the module is in the 5V range. In order to use the module with the Raspberry Pi, the device called logic level converter should be used. Otherwise connecting 5V signals to the GPIO pins may cause damage to the Raspberry Pi. For this purpose use the device called *TXS0108E Logic Level Converter* that AZ-Delivery offers.

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.

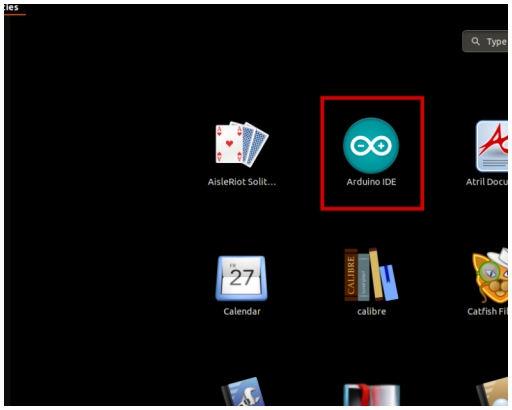
For *Windows* users, double click on the downloaded *.exe* file and follow the instructions in the installation window.

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 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.

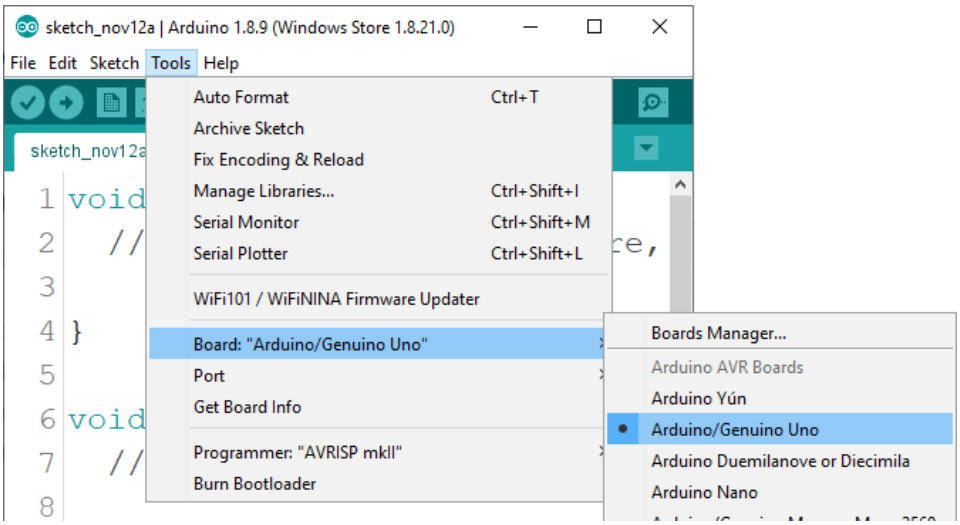
The second script, called *install.sh*, has to be used after the installation of the first script. Run the following command in the terminal (extracted directory): **sh install.sh**

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



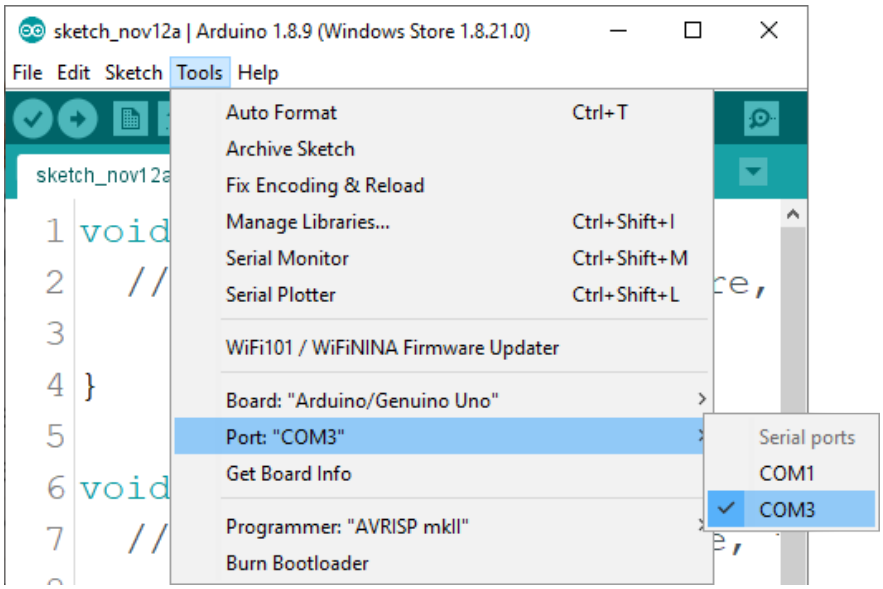
Almost all operating systems come with a text editor preinstalled (for example, *Windows* comes with *Notepad*, *Linux Ubuntu* comes with *Gedit*, *Linux Raspbian* comes with *Leafpad*, etc.). All of these text editors are perfectly fine for the purpose of the eBook.

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*.

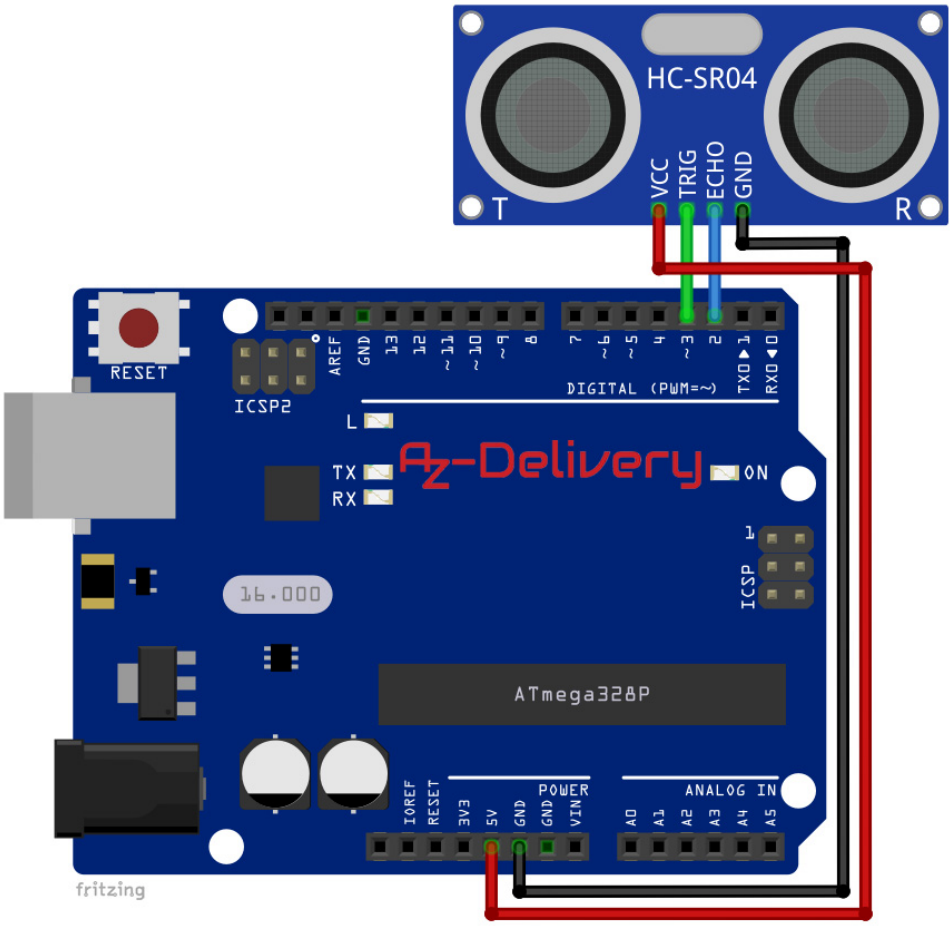
How to set-up the Raspberry Pi and Python

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

Connecting the module with Atmega328P

Connect the module with the Atmega328P as shown on the following image:

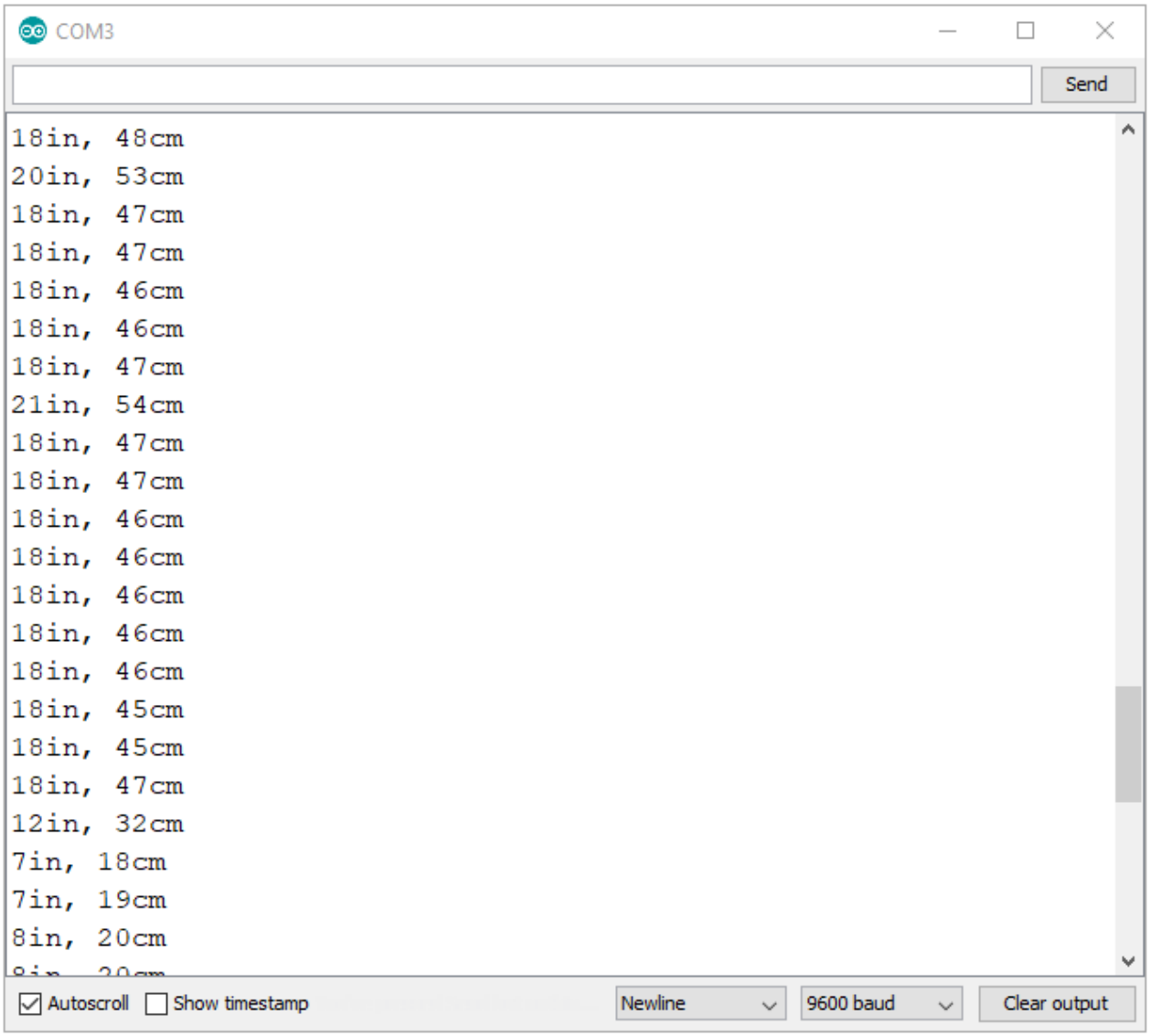


Module pin	Mc pin	Wire color
VCC	5V	**Red wire**
TRIG	D2	**Blue wire**
ECHO	D3	**Green wire**
GND	GND	**Black wire**

Sketch example

```
#define TRIG_PIN 2
#define ECHO_PIN 3
long duration, cm, inches;

void setup() {
  Serial.begin(9600);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
}
void loop() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(5);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  duration = pulseIn(ECHO_PIN, HIGH);

  cm = (duration / 2) / 29.1;   // Divide by 29.1 or multiply by 0.0343
  inches = (duration / 2) / 74; // Divide by 74 or multiply by 0.0135

  Serial.print(inches);
  Serial.print("in, ");
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();

  delay(500);
}
```

Upload the sketch to the Atmega328P and run the Serial Monitor (*Tools* > *Serial Monitor*). The result should look like as on the following image:



The sketch starts with creating two macros *TRIG_PIN* and *ECHO_PIN*. These macros represent the digital pins of Atmega328P to which pins of the module are connected.

Next, three *long* variables are created *duration*, *cm* and *inches*. In the *duration* variable the time interval of the measurement is saved. In the *cm* variable the calculated distance in the centimeters is stored. In the *inches* variable the calculated distance in the inches is stored.

In the *setup()* function, serial communication is started with a baud rate of *9600bps*. After this the modes for digital pins are set: *TRIG_PIN* as *OUTPUT* and *ECHO_PIN* as *INPUT*.

At the beginning of the *loop()* function, *TRIG_PIN* is activated in the following manner:
First, the state of the *TRIG_PIN* is held in *LOW* state for *5* microseconds, next it is held in the *HIGH* state for the 10 microseconds and then again it is set in the *LOW* state. This way, the ultrasonic sound wave is transmitted at the specific time interval so that the measurements can take place.

Next, the function *pulseln()* is used to measure the length of the pulse on the *ECHO_PIN*. The function has two arguments and returns a *long* value. The first argument is the name of the pin on which the measurement takes place.

The second argument of the *pulseln()* function can have two values: *HIGH* or *LOW*. When the value of the second argument is *HIGH*, the function waits for the signal on the *ECHO_PIN* to change its state from *LOW* to *HIGH* to start the measurement. If the value of the second argument is *LOW*, then the function waits for the signal on the *ECHO_PIN* to change its state from *HIGH* to *LOW* to start the measurement. The return value is the *unsigned long* value, which represents the length of the pulse in microseconds.

The following line of code is used to measure the length of the pulse on the *ECHO_PIN*:

```
duration = pulseIn(ECHO_PIN, HIGH);
```

Where the return value from the *pulseln()* function is stored in the *duration* variable.

Next, the distance calculation is done, with the following lines of code:

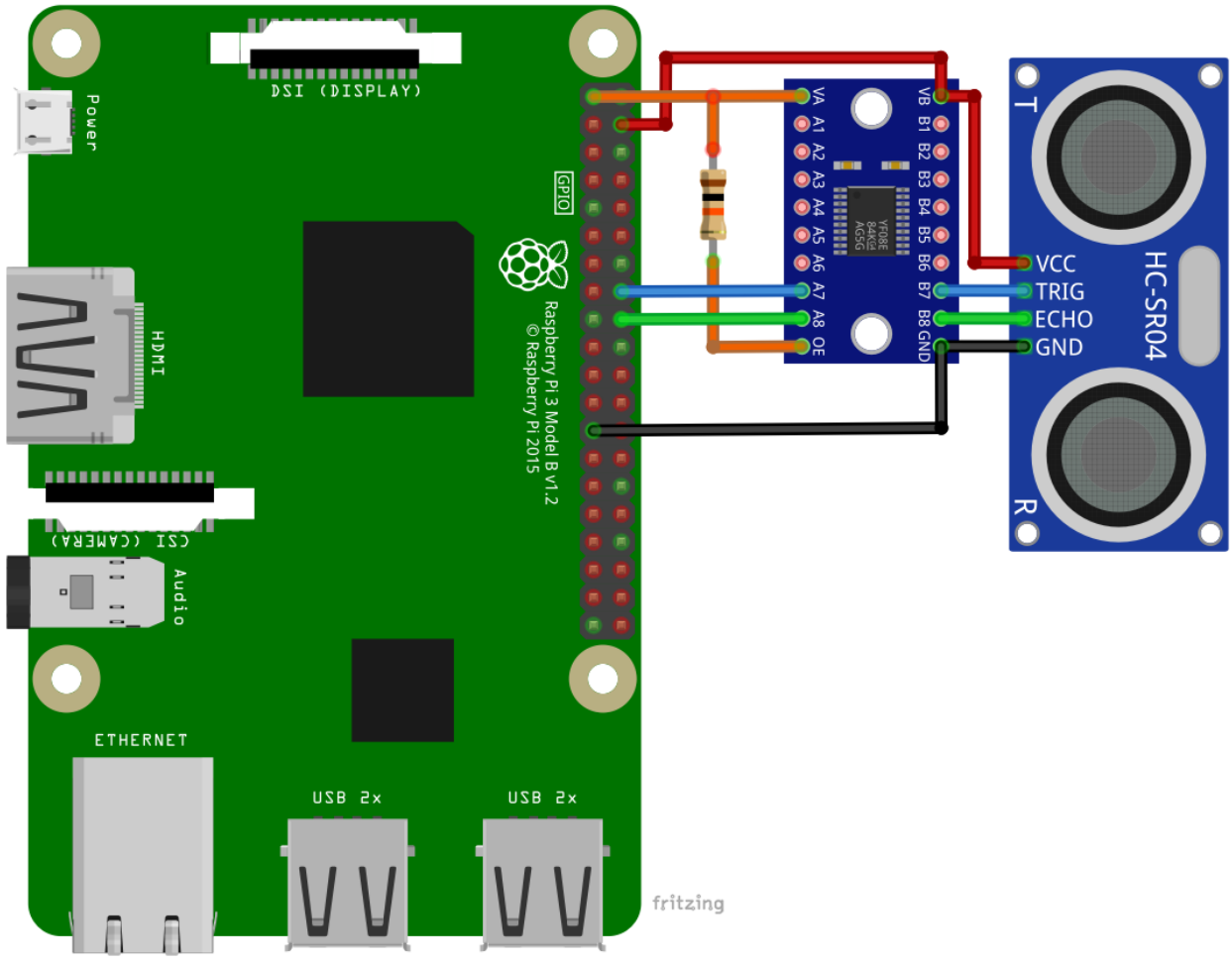
```
cm = (duration / 2) / 29.1;
inches = (duration / 2) / 74;
```

After this, the data is displayed in the Serial Monitor.

At the end of the *loop()* function, a delay pause of a half second is set. This pause represents the pause between two measurements.

Connecting the module with Raspberry Pi

Connect the module with the Raspberry Pi as shown on the following image:



Module pin	Logic Level Converter (LLC) pin	Wire color
VCC	VB	**Red wire**
TRIG	B7	**Blue wire**
ECHO	B8	**Green wire**
GND	GND	**Black wire**

LLC pin	Raspberry Pi pin	Physical pin	Wire color
VB	5V	4	**Red wire**
VA	3V3	1	**Orange wire**
A7	GPIO23	16	**Blue wire**
A8	GPIO24	18	**Green wire**
GND	GND	25	**Black wire**
OE	3V3 via 10kΩ *	1	**Orange wire**

* Connect the OE pin of the LLC with 3V3 via 10kΩ resistor (pull-up resistor) to enable the LLC operation.

Libraries and tools for Python

To use the module with the Raspberry Pi, the library RPi.GPIO has to be installed. If the library is not installed, open the terminal and run the following commands, one by one:

```
sudo apt-get update && sudo apt-get upgrade -y
sudo apt-get install python3-rpi.gpio
```

The example of the original script made by *Matt Hawkins* can be found on the following *link*.

Python script

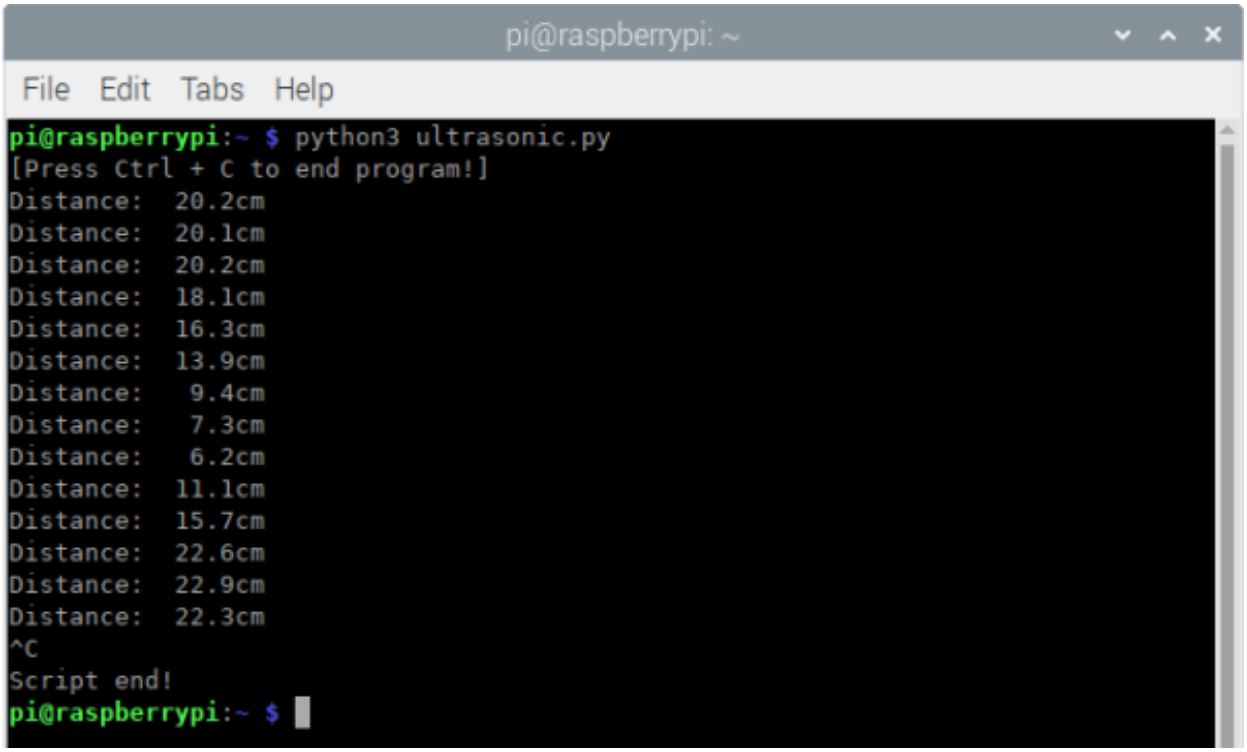
```
import time
import RPi.GPIO as GPIO

def measure():
  speed_of_sound = 34300
  GPIO.output(TRIG_PIN, True)
  time.sleep(0.00001)
  GPIO.output(TRIG_PIN, False)
  start = time.time()
  while GPIO.input(ECHO_PIN) == 0:
    start = time.time()

  while GPIO.input(ECHO_PIN) == 1:
    stop = time.time()

  elapsed = stop - start
  distance = (elapsed * speed_of_sound) / 2
  return distance

def measure_average():
  distance1 = measure()
  time.sleep(0.1)
  distance2 = measure()
  time.sleep(0.1)
  distance3 = measure()
  distance = distance1 + distance2 + distance3
  distance = distance / 3
  return distance
```

```
GPIO.setmode(GPIO.BCM)

TRIG_PIN = 23
ECHO_PIN = 24
GPIO.setup(TRIG_PIN, GPIO.OUT)
GPIO.setup(ECHO_PIN, GPIO.IN)

print('[Press Ctrl + C to end program!]')
try:
  GPIO.output(TRIG_PIN, False)
  time.sleep(0.5)

  while True:
    distance = measure_average()
    print('Distance: {:5.1f}cm'.format(distance))
    time.sleep(1)

except KeyboardInterrupt:
  print('\nScript end!')

finally:
  GPIO.cleanup()
```

Save the script by the name *ultrasonic.py*. To run the script, open the terminal in the directory where the script is saved and run the following command:

python3 ultrasonic.py

The result should look like as on the following image:



To stop the script press ‘CTRL + C’ on the keyboard.

The script starts with importing two libraries: *time* and *RPi.GPIO*.

Next, two functions are created: *measure()* and *measure_average()*. Both functions have no arguments and return the *double* value. The *measure()* function is used to do the distance measurement and calculation. The algorithm for measuring and calculating the distance is in this function. The *measure_average()* function is used to do three measurements and make an average value out of three measurements. The return value of these functions represents the calculated distance value.

Then, the GPIO pin naming is set to BCM, and the GPIO pin modes for *TRIG_PIN* and *ECHO_PIN* are set to output and input, respectively.

After this, the *try-except-finally* block of code is created. In the *try* block of code the state of *TRIG_PIN* is set to LOW, after which the indefinite loop block of code is created (*while True:*).

In the indefinite loop block of code, first, the *measure_average()* function is executed and the return value is stored to the distance variable. Next, the value in the distance variable is displayed in the terminal, with the following line of code:

```
print('Distance: {:5.1f}cm'.format(distance))
```

Where “5.1f”: “5” means the output contains 5 decimal places; “.1” means that there is one digit after the decimal point; and “f” means that the value is of type *float*.

To stop the script press ‘CTRL + C’ on the keyboard. This is called the keyboard interrupt. When the keyboard interrupt happens, the *except* block of code is executed, displaying message *Script end!* in the terminal.

The *finally* block of code is executed at the script end. When the *finally* block of code is executed, the function called *cleanup()* is executed. The *cleanup()* function disables all used GPIO interfaces and GPIO pin modes.

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