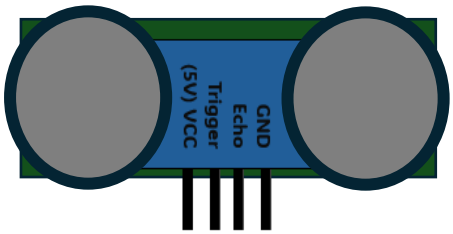
How to use the SRO4v2 Ultrasonic Distance Sensor with the Pico Microcontroller

This sensor is very cheap.

It can sense distance, sending the data to the Pico.



Contents

The ultrasonic distance sensor SR04v2

The **HC-SR04v2** ultrasonic sensor is a cheap easy to use distance sensor.

I have also seen this sensor sold as the **RCWL – 1601**.

There is a waterproof version of this ultrasonic sensor, called **JSN-SR04T** sensor.

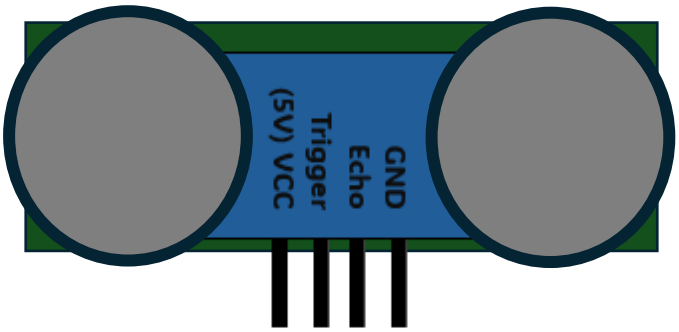
We can treat these two devices the same way.

The **HC-SR04v2** sensor can read objects as close as 2cm or as far away as 4 meters.

There is a waterproof ultrasonic sensor is sold as "**AJ-SR04M**".

Note that the minimum distance is about 20cm for this one.

We wire up all the variants of this ultrasonic sensor in the same way.



All these sensors have 4 pins that work the same way.

- Two of the pins are for power (**VCC** and **GND**).
- The 3rd pin is a **trigger** pin which instructs the sensor to send out an ultrasonic sound.
- The 4th pin is the **echo** pin. It inputs a signal to the Pico if a reflected sound wave is detected.

Hardware Setup:

Connect the bottom right Pin on the Pico to the positive rail on the breadboard.

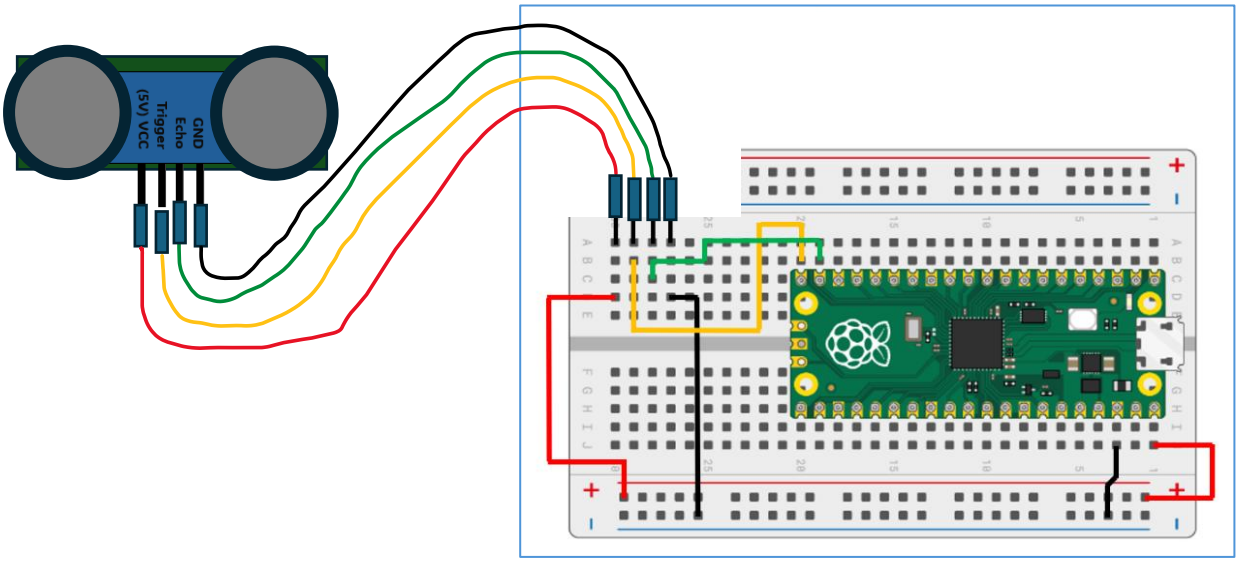
Wire the negative breadboard rail to the GND pin (3rd in from bottom right).

Wire up the ultrasonic board as shown here.

Yellow trigger wire =GP15 (top left).

Echo plugs into GP14 (one in from top left).

You can in theory plug the sensor directly into the breadboard. But the connection is poor and it bends the pins. You will get a more stable setup using jumper wires.



Test the Pico's connection to the PC.

We always check our setup in stages, decomposing the problem into sub tasks.

First, we check whether the USB connection from the Pico to the PC is working. Note that this is not testing the sensor. It is just checking the Pico is connected to the PC.

Copy/paste this test code and run it.

If it is working the on-board LED on the Pico will light up for 3 seconds

Test Code: checking Pico to PC connection

```
from machine import Pin
import time
led = Pin("LED", Pin.OUT)
led.value(1)
```

```
time.sleep(3)
led.value(0)
```

Hopefully the green onboard LED did light up.

Now we can test the ultrasonic distance sensor.

Test Code for Ultrasonic Sensor:

The test code below will check to see if your ultra-sonic sensor is sending data back to the Pico.

I included an IF statement making the on-board LED turn on if the measured distance is **less than** 30cm.

You can copy/paste test code from my GitHub page (this maintains the formatting)

https://github.com/pythonninja-code/Sensor-Example-Code

I have also included it here, in case the school firewall blocks GitHub.

```
from machine import Pin, time_pulse_us
import time
led = Pin("LED", Pin.OUT)

SOUND_SPEED=340 # Speed of sound through the air
TRIG_PULSE_DURATION_US=10

trig_pin = Pin(15, Pin.OUT) # Trigger Pin
echo_pin = Pin(14, Pin.IN)  # Echo Pin

while True:
    # Prepare the signal
    trig_pin.value(0)
    time.sleep_us(5)
    # Create a 10 µs pulse
    trig_pin.value(1)
    time.sleep_us(TRIG_PULSE_DURATION_US)
    trig_pin.value(0)


    ultrasound_duration = time_pulse_us(echo_pin, 1, 30000) # Returns the wave propagation time (in µs)
    distance_cm = SOUND_SPEED * ultrasound_duration / 20000

    print("Distance : ", distance_cm,  "cm")
    time.sleep_ms(500)

    #On board LED to light up if closer than 30cm
    if distance_cm <=30:
        led.value(1)
    else:
        led.value(0)
```

Put your hand in front of the sensor to see whether the output changes:

```
Distance : 152.711 cm
Distance : 152.371 cm
Distance : 154.445 cm
Distance : 153.646 cm
Distance : 152.83 cm
Distance : 151.963 cm
Distance : 20.298 cm
Distance : 20.298 cm
Distance : 20.366 cm
```

Be aware that minimum range for some versions of this US sensor is about 20cm so if you are closer than that it will not output a reading.

Save your program into your own OneDrive area as "ultrasonic v1.py".

Understanding the Code:

```
#Test Code checking US Sensor is connected to Pico
from machine import Pin, time_pulse_us
import time
led = Pin("LED", Pin.OUT)

SOUND_SPEED=340 # Speed of sound through the air
TRIG_PULSE_DURATION_US=10

trig_pin = Pin(15, Pin.OUT) # Trigger Pin
echo_pin = Pin(14, Pin.IN)  # Echo Pin

while True:
    # Prepare the signal
    trig_pin.value(0)
    time.sleep_us(5)
    # Create a 10 µs pulse
    trig_pin.value(1)
    time.sleep_us(TRIG_PULSE_DURATION_US)
    trig_pin.value(0)


    ultrasound_duration = time_pulse_us(echo_pin, 1, 30000)
    distance_cm = SOUND_SPEED * ultrasound_duration / 20000

    print("Distance : ", distance_cm,  "cm")
    time.sleep_ms(500)

    #On board LED to light up if closer than 30cm
    if distance_cm <=30:
        led.value(1)
    else:
        led.value(0)
```

The modules needed

The GP Pins here are 15 and 14. These are in the top left corner. See diagram below.

While loop, so the sensor reading is checked repeatedly

Sends out a pulse

Reads the return value

Prints the value in the shell

If the valueless is less than 30cm, the on-board green LED lights up.

If you are getting readings like this in the Thonny Shell we can now move onto adding a LED strip to the breadboard.

```
Distance : 152.711 cm
Distance : 152.371 cm
Distance : 154.445 cm
Distance : 153.646 cm
Distance : 152.83 cm
Distance : 151.963 cm
Distance : 20.298 cm
Distance : 20.298 cm
Distance : 20.366 cm
```

If you are not getting this output:

- Check your wires are connected to the correct sensor pins.
- Check the wires connecting to the Pico are connecting to the correct pins.

Adding a LED strip.

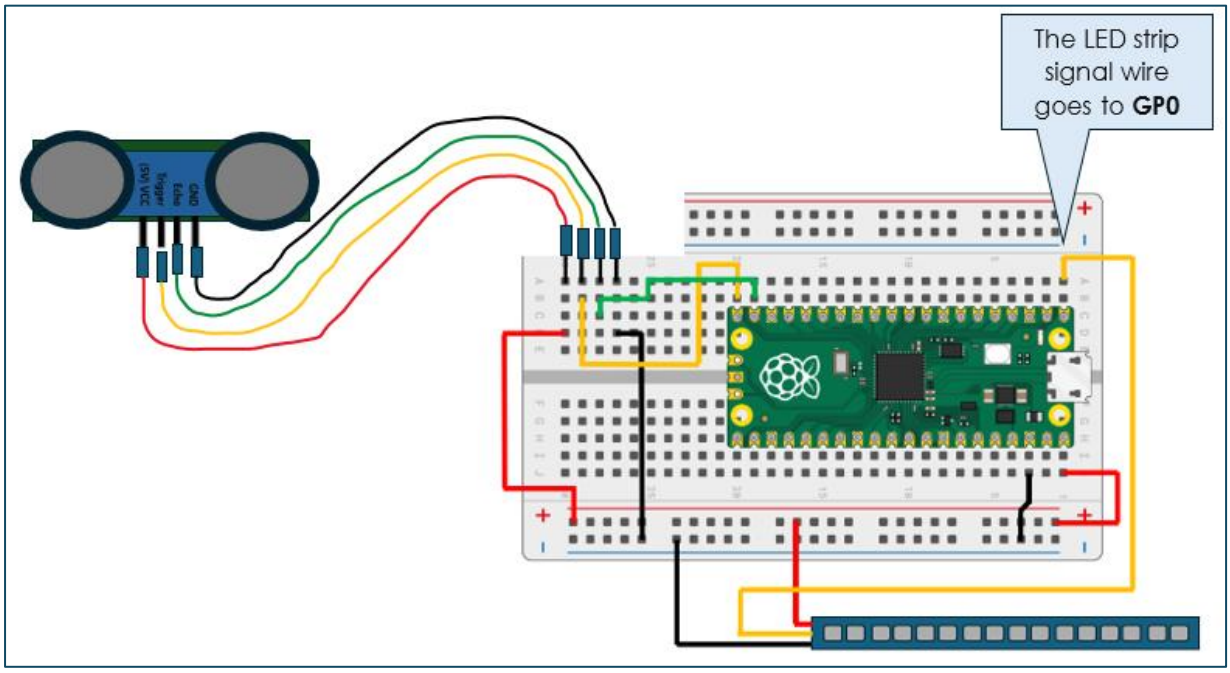
You have used LED strips before. If you have not, see the 2.1 guide.

Hopefully you have a short length of LEDs in your kit.

The diagram below shows you how to wire the LED strip to your breadboard and Pico.

Note the yellow jumper wire is carrying the control signal from the Pico GP0 (top right) pin, to the LED strip.

Wiring Diagram including LED Strip

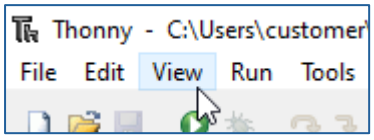


The NeoPixel Library

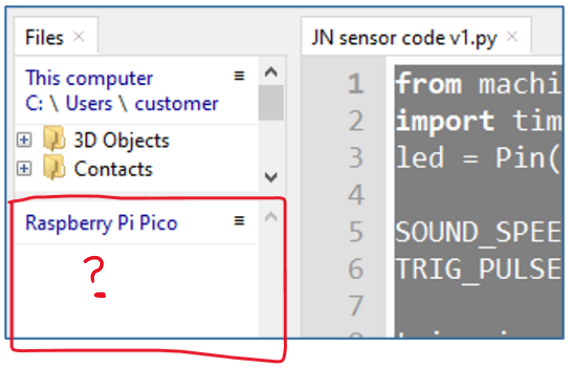
When we use LED strips, we need to install the neopixel library onto the Pico device.

Check whether your Pico already has the neopixel library installed. Make sure the Pico is connected to the PC with the micro usb cable. Now we can check if the neopixel.py file is on the pico device.

In Thonny, go to VIEW/ FILES



You should see this window.



There is no neopixel.py file visib

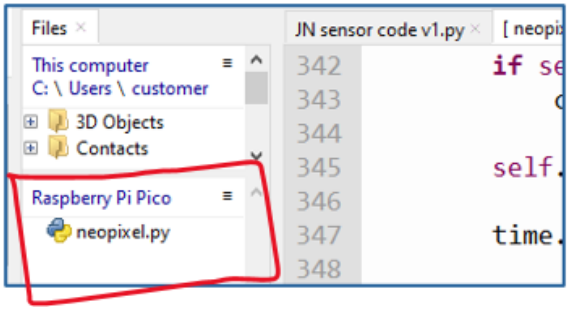
Skip onto the "Testing the LED strip" sub heading if you *can* see this file.

Installing the NeoPixel library

For more detailed instructions see the 'LED Strip Intro' guide. In this guide, I am assuming you know how to save the neopixel.py file onto the Pico device.

You can get the neopixel code from here: https://github.com/blaz-r/pi_pico_neopixel

Once you have saved the neopixel.py file onto the pico you should be able to see the file in Thonny.



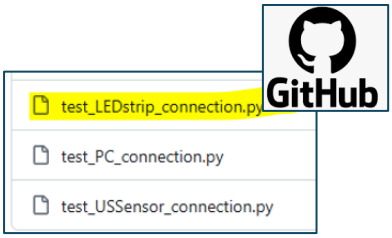
Test Code: Checking the LED strip

This text code will check your LED strip is wired up correctly, and that the neopixel library is installed.

You can copy/paste test code from my GitHub page (this maintains the formatting)

https://github.com/pythonninja-code/Sensor-Example-Code

You want this one:



I have also included the test code here, in case the school firewall blocks GitHub.

```
from machine import Pin
import time
from neopixel import Neopixel

#variables for the LED strip
numpix = 20
strip = Neopixel(numpix, 0, 0, "GRB")

#colour variables
red = (255, 0, 0)
blank = (0, 0, 0)
#commands
strip.fill(red)
strip.show()
time.sleep(2)
strip.fill(blank)
strip.show()
```

Open a new Thonny page.

Paste in this test code.

Check you have entered the correct value for numpix.

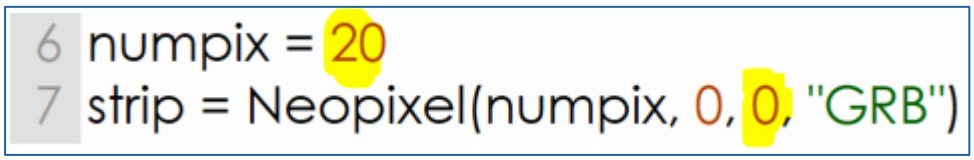
Check you are using the correct Pin value for the yellow LED strip signal wire.

Run the code.

Hopefully your LED strip did light up.

If it did not, check the number of pixels is correct.

Also check the code it talking to Pin 0



The next step is to merge the code we have just used to test the LED strip, with the code we used to test the ultrasonic sensor.

Challenge 1 – Make the LED strip light up if <=50cm

Add a new IF statement at the bottom of your ultrasonic v1.py file.

```
    #On board LED to light up if closer than 30cm
    if distance_cm <=30:
        led.value(1)
    else:
        led.value(0)

    #LED strip code
    if distance_cm <=50:
       #add code that sets the LED strip to a colour
    else:
       #add code that sets the LED strip to blank
```

Go back to the LED Strip code.

Copy all of the code up to the **#commands** sub heading.

Go back to ultrasonic v1.py.

Paste it into the top of the page.

```
from machine import Pin
import time
from neopixel import Neopixel

#variables for the LED strip
numpix = 20
strip = Neopixel(numpix, 0, 0, "GRB")

#colour variables
red = (255, 0, 0)
blank = (0, 0, 0)
```

```
from machine import Pin
import time
from neopixel import Neopixel

#variables for the LED strip
numpix = 20
strip = Neopixel(numpix, 0, 0, "GRB")

#colour variables
red = (255, 0, 0)
blank = (0, 0, 0)

from machine import Pin, time_pulse_us
import time
led = Pin("LED", Pin.OUT)
```

Scroll down to the IF statement.

Type in the commands you want to happen when the sensor reads less than 50cm.

```
    #On board LED to light up if closer than 30cm
    if distance_cm <=30:
        led.value(1)
    else:
        led.value(0)

    #LED strip code
    if distance_cm <=50:
       strip.fill(red)
       strip.show()
    else:
       strip.fill(blank)
       strip.show()
```

Challenge 2 – Adding a delay.

Sometimes the sensors readings jump around a bit.

This makes the LED strip flash.

How could you edit the code so, once triggered, the LED strip lights up for 10 seconds.

10 seconds is just for testing purposes. When I put this into the cloakroom I will make it stay on for 2 minutes.

Challenge 3 – Change the brightness.

Do you remember how to adjust the brightness of the LED strip?

Challenge 4 – LED indicator for Distance.

Make LEDs light up one after the other as the distance increases.

So, if the distance was less than 50cm you could make 1 LED light up.

If the distance was between 50 and 60cm two LEDS could light up.

70cm = 3 LEDs etc.

If you need to remind yourself how to program a LED strip look at the LED strip guide in PythonNinja.co.uk