Bohunt School *(Wokingham)*

Internet of Things (IoT) and Node-RED

This practical session should be a bit of fun for you. It involves creating a distance sensor node using the SRF05 ultrasonic device.

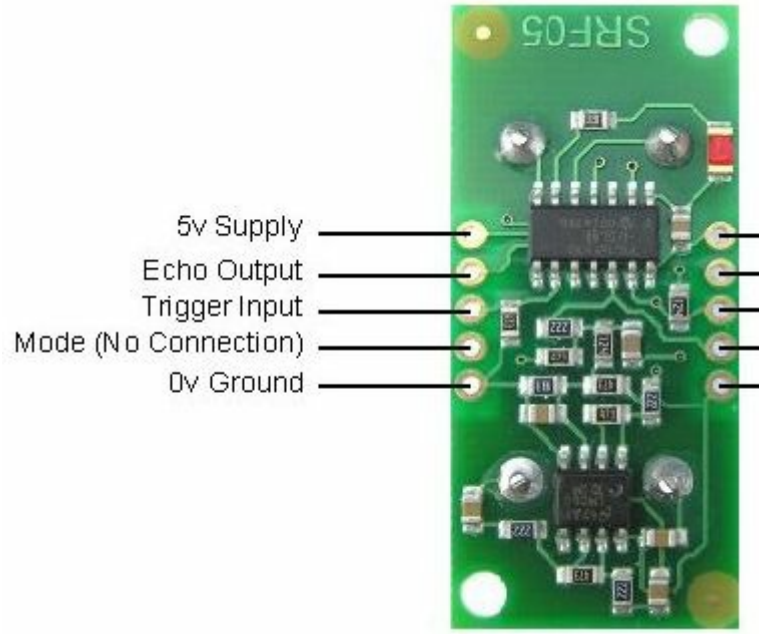
How the SRF05 works

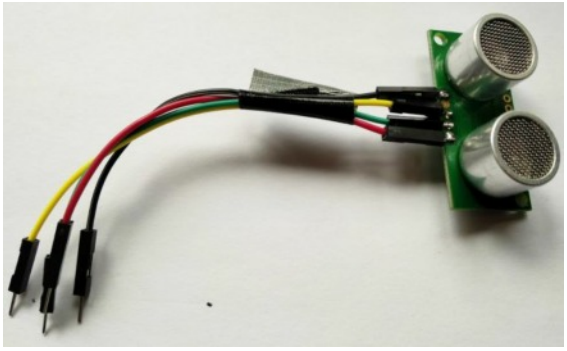
Here's a photo of the SRF05. The silver metal cans are part of the ultrasonic transmitter and receiver that sends and receives 40KHz sound waves.

Here's another photo showing the other side of the SRF05.

The unit requires a 5V supply and a Ground connection.

There are two important connections called 'Trigger Input' and 'Echo Output'.

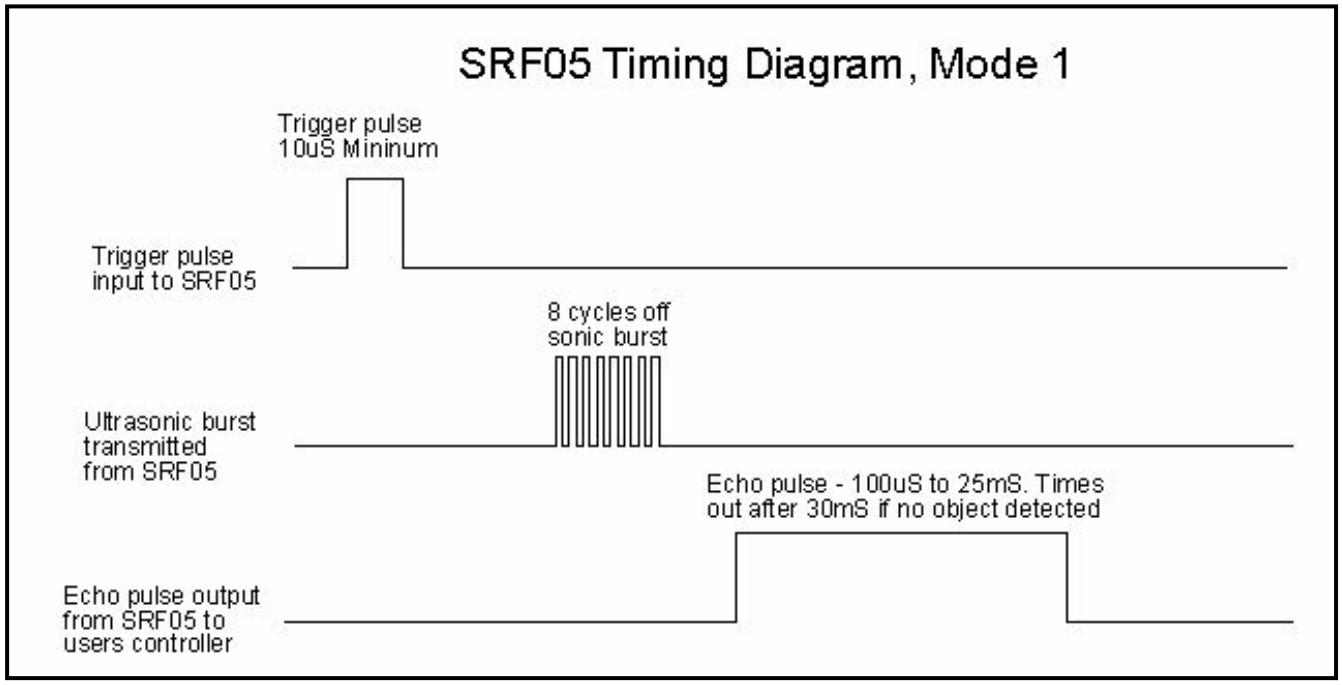




Here's a photo of an SRF05 with a set of flying leads soldered onto the connections.

*** Red lead is +5V**
*** Yellow lead is Trigger Input**
*** Green lead is Echo Output**
*** Black lead is Ground**

Here's the timing diagram for the SRF05



In this example the SRF05 is interfaced to a WeMos D1 Mini - 'Trigger Input' is connected to pin D7 and 'Echo Output' to pin D6.

In order to take a range/distance reading a short duration pulse is output from D7. This triggers the SRF05 device. After a short delay the SRF05 outputs a burst of 8 cycles of the ultrasonic frequency at 40 KHz.

At the same time a virtual counter is reset in the WeMos and a clock-pulse generator is started. Each clock pulse (which is the equivalent to the time it would take the ultrasonic wave to travel 20mm) is used to increment the counter. This value represents a distance of 10 mm as the ultrasonic wave has to travel to an object and back again. (I.e. This is a round trip of 20mm for a distance of 10mm.)

After a short delay the on-board microcontroller (on the SRF05) forces the 'Echo Output' signal to a logic 1 (i.e. +5V)

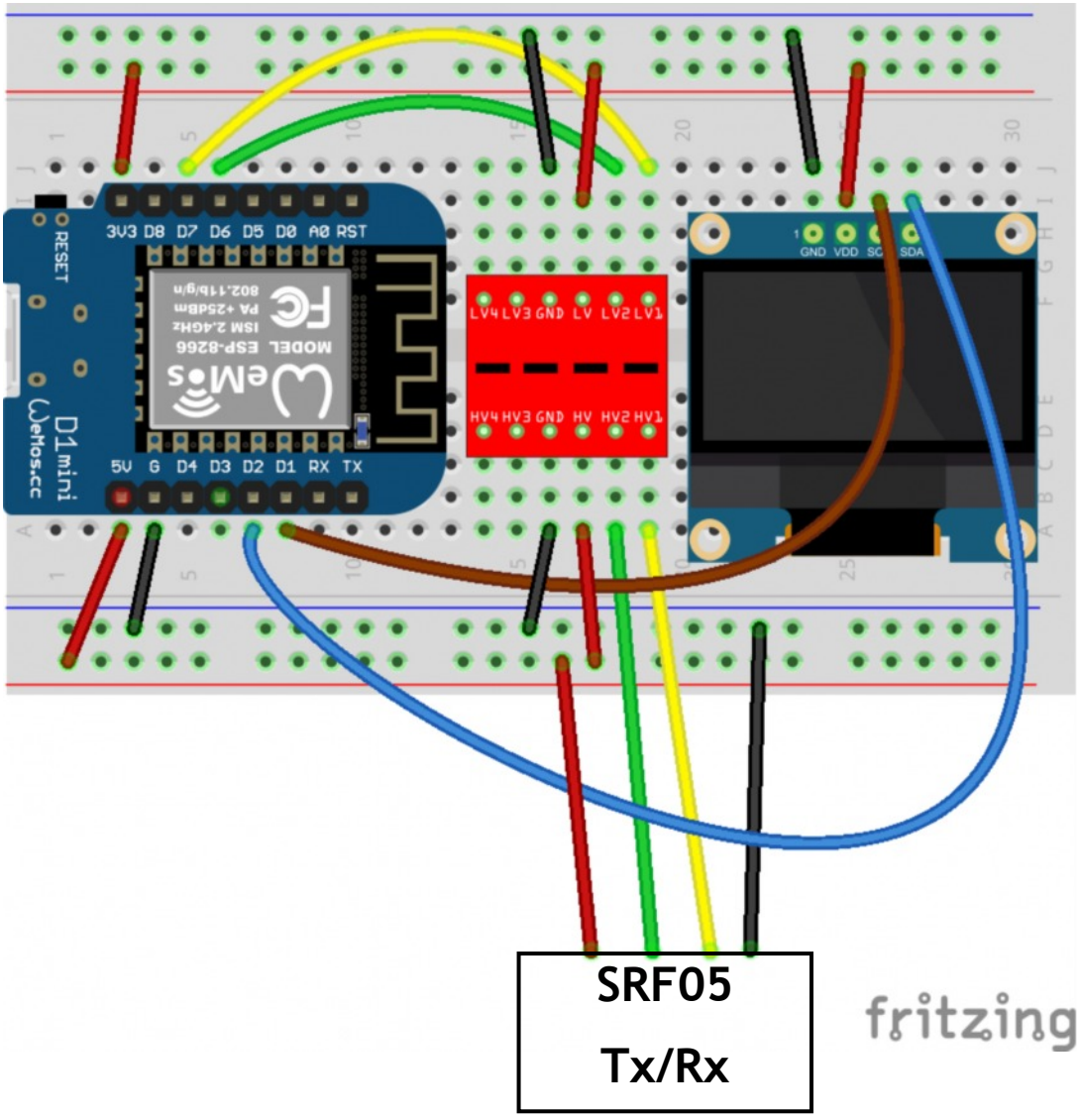
As can be seen from the timing diagram, the 'Echo Output' can remain at a logic 1 level for any time between 100 us to 25 ms. The actual time depends on the distance an object is from the SRF05.

When the 'Echo Input' returns to a logic 0 level (i.e. 0V) the clock-pulse generator is stopped. The number held in the counter indicates the number of 10mm gaps between the SRF05 and the object. For example, if the counter held the value 5, then this would mean the object is 50mm away and if it held the value 38, then the object is 380mm away.

At the end of this document are details about calculating the speed of sound.

Wiring the breadboard

Here's a view of the suggested wiring layout for your breadboard.



As can be seen from the above diagram, the yellow wire connects D7 (on the WeMos) to LV1 (on the level shifter). HV1 (on the level shifter) connects to the 'Trigger Input' on the SRF05.

Likewise, the green wire connects the 'Echo Output' (on the SRF05) to HV2 (on the level shifter). LV2 (on the level shifter) connects to D6 (on the WeMos).

The black and red wires connect 0V/GND and +5V from the WeMos to the SRF05 unit.

The connection to/from the miniature OLED panel will be covered later.

Congratulations you have now completed putting the hardware together.

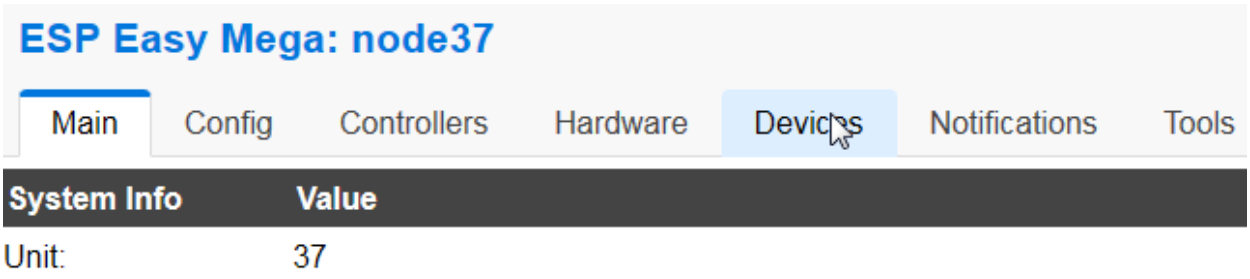
Configuring your WeMos D1 Mini

The first thing you need to do is set-up the device driver for the SRF05.

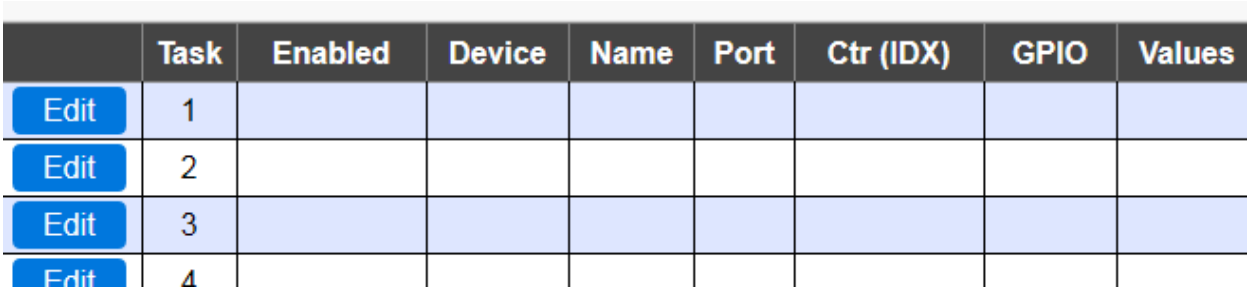
The WeMos node I'm using is node37. Your node number may be different.

Login to the WeMos by entering: 192.168.1.37 in your browser.

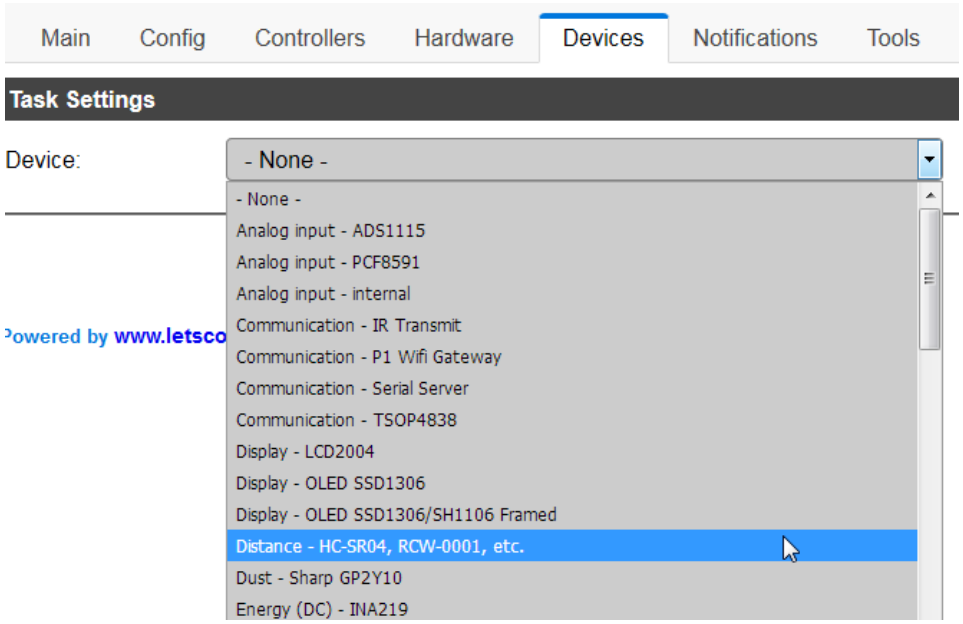
The following web management screen should appear.



Click the tab labelled 'Devices' to reveal the following management screen.

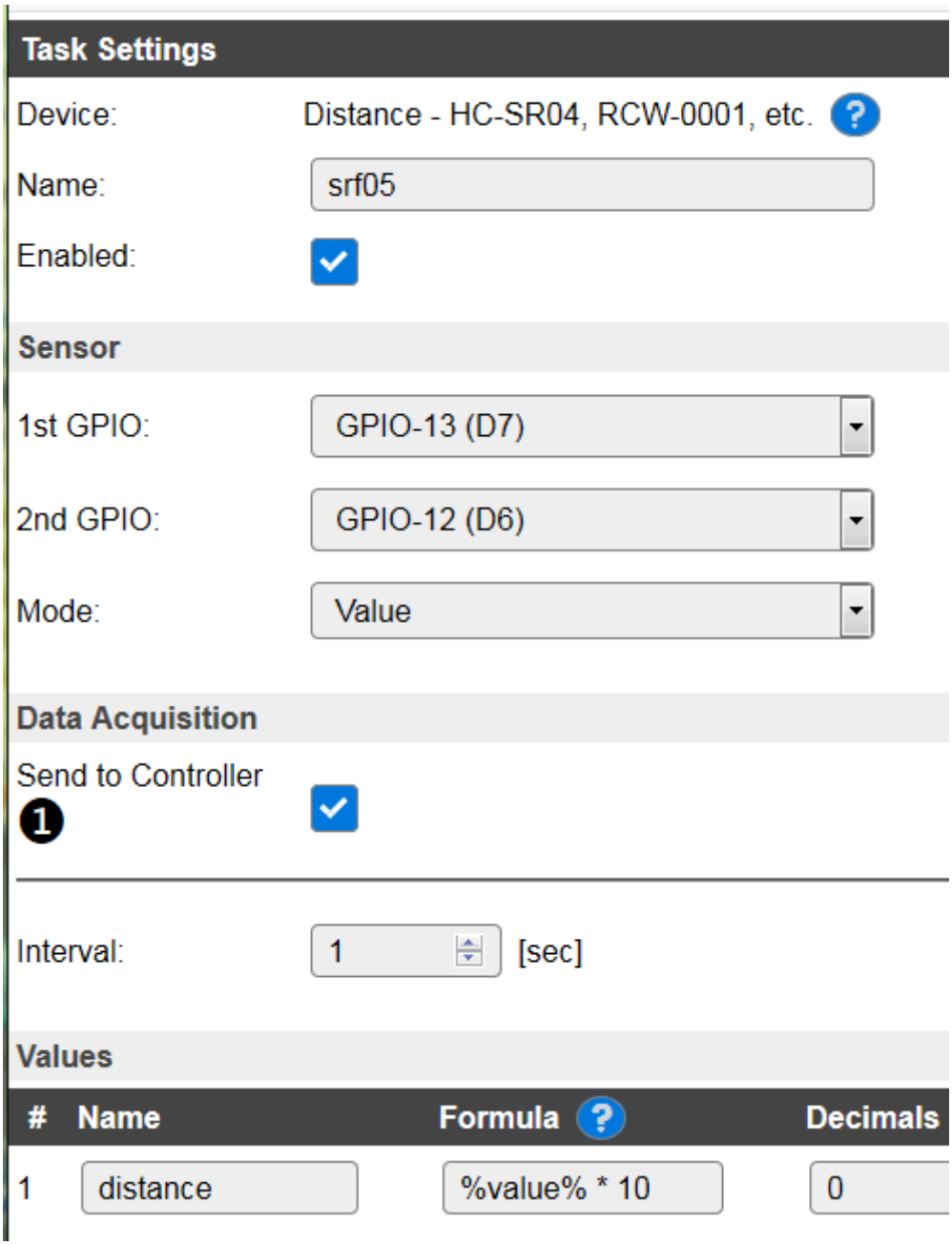


Click the blue [Edit] box on the first row to reveal the following screen.



Scroll down the list of available device drivers until you find HC-SR04, then click the 'Submit' button.

You now need to fill-in the details for your ultrasonic device.



I've called named my device 'srf05' (because I can't think outside the box).

The 1st GPIO entry is for the 'Trigger Input' - I'm using GPIO-13 (D7).

The 2nd GPIO entry is for 'Echo Output' - I'm using GPIO-12 (D6).

These entries must match the WeMos wiring you did on page 3.

I've also set the sampling 'Interval' to 1 second and the name of the variable as 'distance' (again because I can't think outside the box). The device driver gives the distance in centimetres, so I multiplied this value by 10 (in the Formula box) to obtain a result that will be shown in millimetres.

If you click back on the 'Devices' tab you should see this summary.

	Task	Enabled	Device	Name	Port	Ctr (IDX)	GPIO	Values
Edit	1	✓	Distance - HC-SR04, RCW-0001, etc.	srf05		1	GPIO-13 GPIO-12	distance: 0
Edit	2							

You have now setup the device driver, the next item to configure is MQTT. Click on the 'Controllers' tab to reveal this management screen.

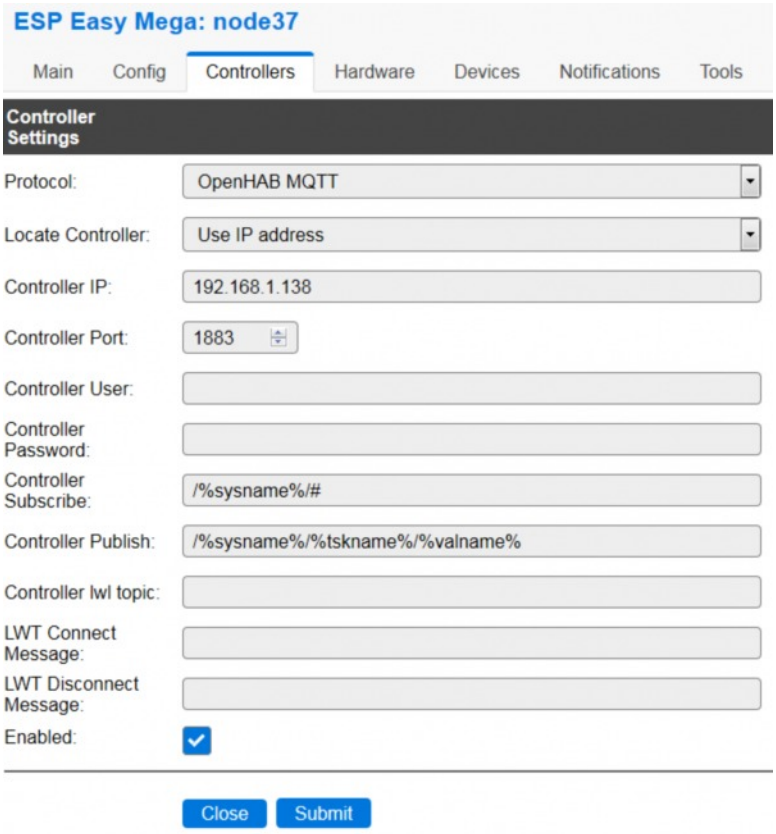
ESP Easy Mega: node37

Main | Config | Controllers | Hardware | Devices | Notifications | Tools

	Nr	Enabled	Protocol	Host	Port
Edit	1	✓	OpenHAB MQTT	192.168.1.138	1883
Edit	2				
Edit	3				

I'm using a Raspberry Pi with an IP address of 192.168.1.138 as you can see from the above screen shot. Your IP address may be different, so check it.

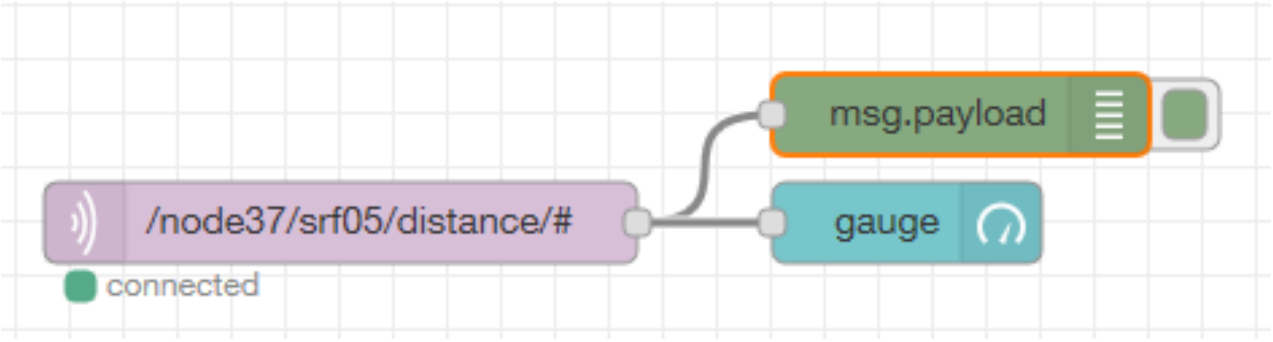
If you need to make adjustments then click the blue [Edit] box.



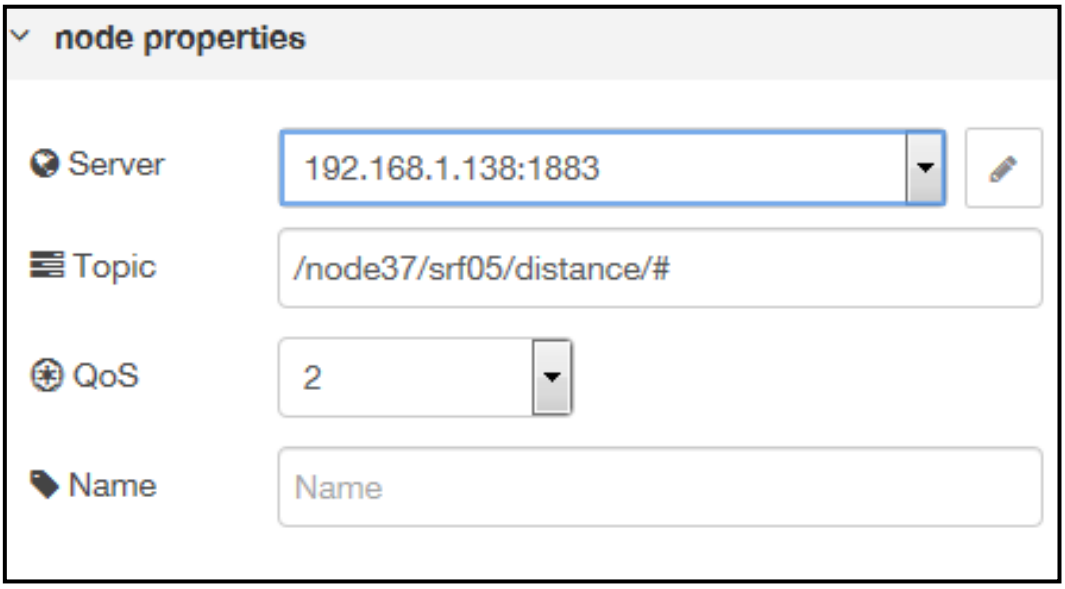
That should be all your need to do to the WeMos, you can now write your flow.

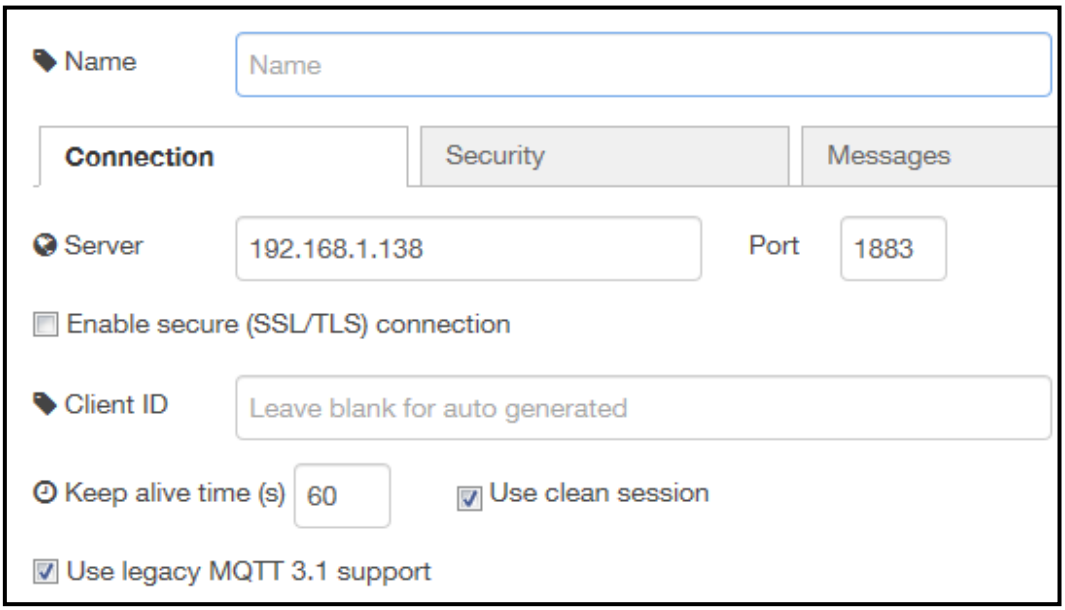
Writing your Node-RED flow

Here's a very simple flow to see if the SRF05 is working correctly.



You'll need to configure the MQTT node, so double-click it and check details.

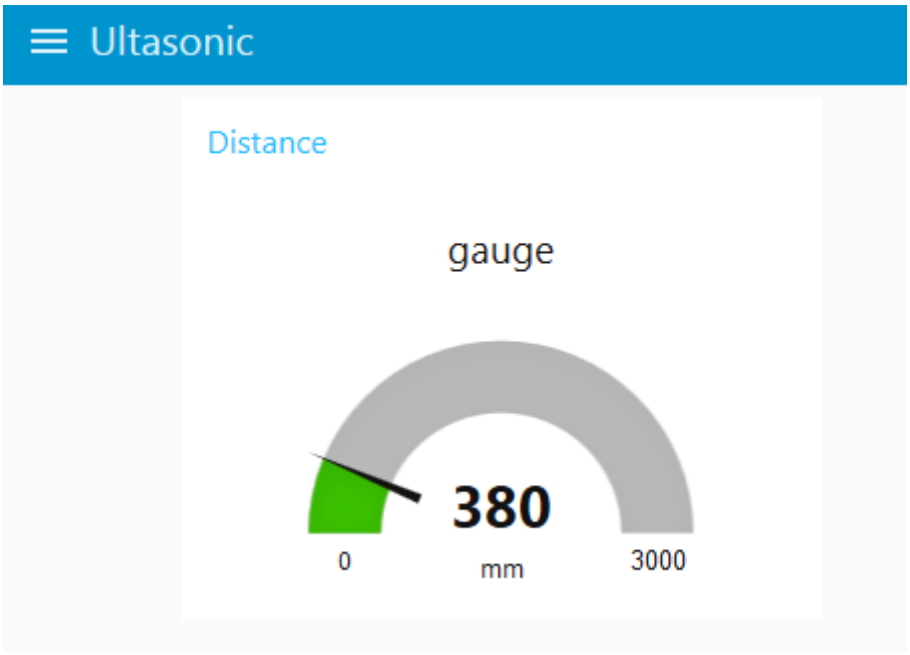




That completes the simple flow for Node-RED. Next item is the dashboard.

Designing a dashboard

Here's a screen shot of a simple dashboard. I'm sure you could do a better job.



Calculating the speed of sound waves

Sound waves travel at a speed of 343 metres per second.

That's 343,000 mm per 1,000,000 microseconds.

Dividing both sides of the equation by 1,000 gives 343 mm per 1,000 us

Dividing both sides of the equation by 17.15 gives 20 mm per 58.31 us

This means an ultrasonic signal will travel 20 mm in 58.31 microseconds.

20 mm is the round trip for an ultrasonic signal to travel 10 mm to an object, bounce off of it and return 10 mm to its starting point.

Converting 58.31 us into a frequency gives 17.15 KHz, this means the virtual counter (that was mentioned on page 2) needs to be clocked at this frequency.

So if a counter is clocked every 58.31 us (i.e. 17.15 KHz) whilst the 'Echo Output' signal is a logic 1, then it will record how many sets of 10 mm of travel the ultrasonic signal has made.

Please talk to Mr D if you need further details about the 'maths' involved.

Congratulations you have built an ultrasonic distance sensor.

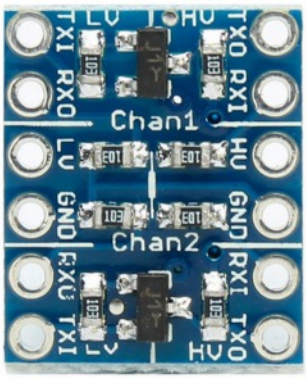
Examples of Level Shifters that shift a voltage 'UP'

This project needs a level shifter to convert:

*** WeMos D1 Mini logic levels (3.3v) to those used by the SRF05 (5v)**

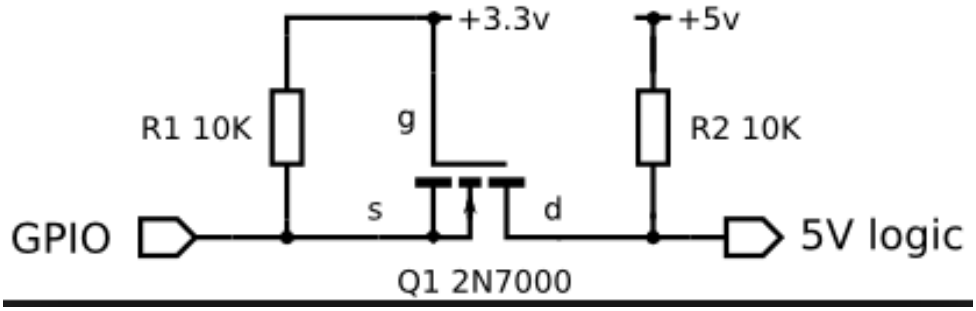
*** SRF05 output signals (5v) to WeMos levels (max 3.3v)**

You can use a level shifter like the one shown here.

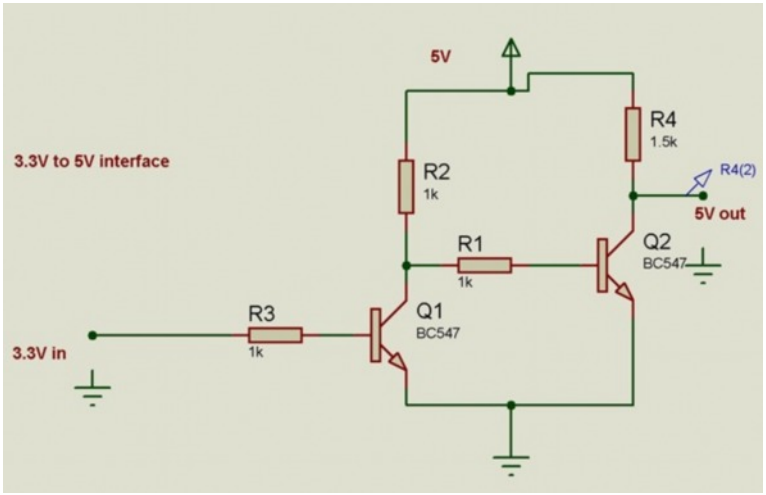


Or you could construct your own with discreet components.

This version uses an N-channel MOSFET - 2N7000 and a couple of resistors.



This version uses two NPN BJTs (Bipolar junction transistors) and two resistors.



The level shifters, shown on the previous page, will convert a +3.3v signal to a +5v signal, and a 0v signal to a 0v signal.

The output from the WeMos D1 Mini on pin D7 connects to the 'trigger input' on the SRF05. You may find you don't need the level shifter as the logic 'high' output from the WeMos (i.e. +3.3v) is above threshold level for 5v logic and will probably operate the SRF05 correctly.

The problem you are faced with is shifting an output voltage down.

For example, the 'Echo Output' from the SRF05 will be at +5v when a logic 1 is output during the time the distance is being measured.

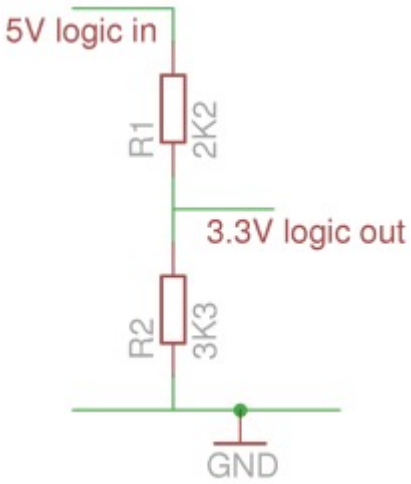
Please refer back to page-2 and study the timing diagram.

If you connect a +5v logic level to an input pin on the WeMos then you will almost certainly damage the device - so don't ever do that.

Example of a Level Shifter that will shift a voltage 'DOWN'

Although you can use transistor circuits as level shifters - similar to the ones on the previous page, in this situation a couple resistors will work just fine.

The circuit shown opposite is a simple voltage divider.



Using the 2K2 and 3K3 resistors gives a division ratio of 0.6, so if you multiply 5v by 0.6 you get 3v.

This circuit has reduced the voltage from 5v to 3v (which is acceptable to the WeMos).

The voltage divider circuit is based on applying Ohms Law to resistors connected in series.

Mr D will explain Ohms Law to you in later session
or
you could look it up on Google or Wikipedia.