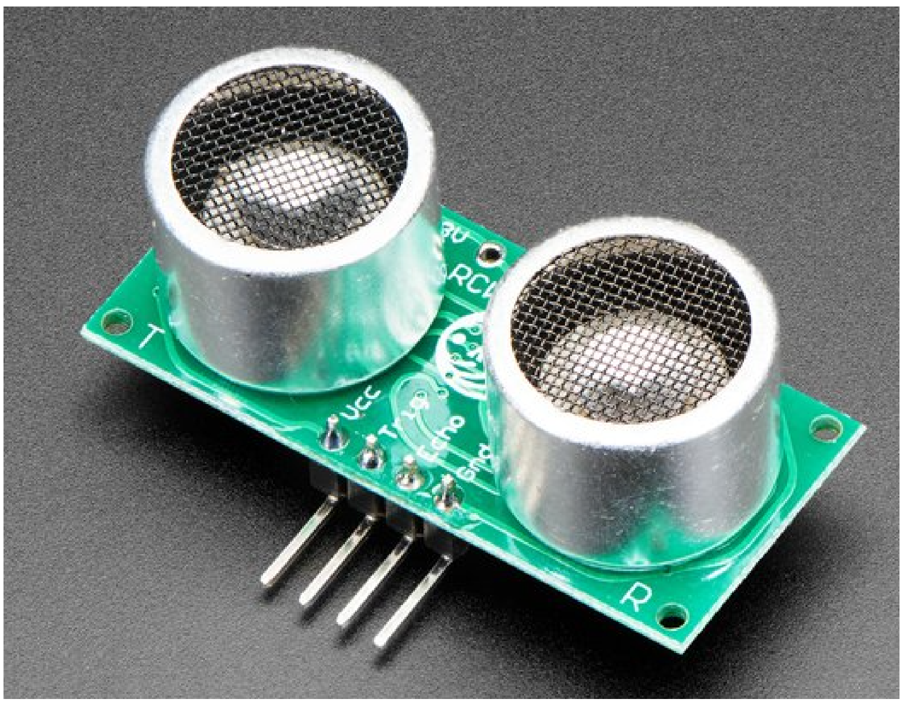
How-To Guide for Ultrasonic Distance Sensor

1. Introduction

This guide is for the Ultrasonic Distance Sensor - 3V or 5V - HC-SR04 compatible - RCWL-1601, found on adafruit. This sensor works for ~2cm to 450cm according to Adafruit, but we recommend it for distances between 2cm to 60 cm.

Important: This guide is for the **RCWL-1601** sensor and **NOT** the HC-SR04. The HC-SR04 requires 5V power. It is possible to use the HC-SR04, but be very careful never to send 5V to any of the GPIO pins as this could damage the board. You can use a voltage divider to lower the output voltage to a safe level below 3.3V (note: you do not need the voltage divider for the **RCWL-1601** when running using 3.3V power).



2. Setup

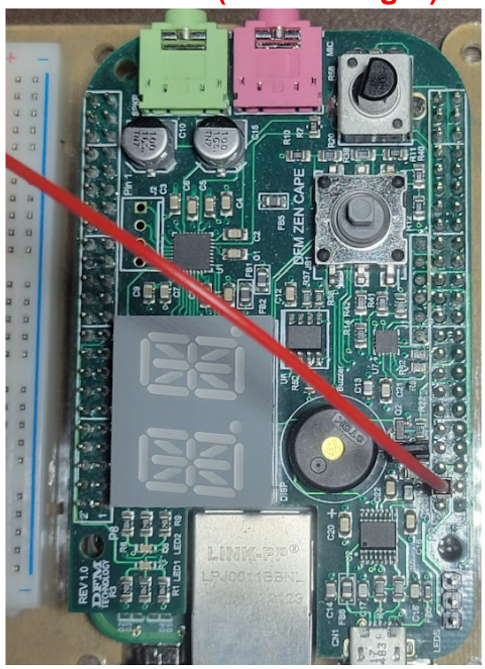
2.1. Wiring

As shown above, the sensor has 4 pins, VCC, Trigger, Echo, and Ground. Using the male/female jumper wires included in the provided beaglebone kit, follow the steps below for connecting the ultrasonic distance sensor pins, to the zen cape pins:

Step	Sensor Pin	P9/P8 Header	Pin #

<table>
<tr><td>1</td><td>VCC</td><td>P9</td><td>4 (VDD_3V3)</td></tr>
<tr><td colspan="4">

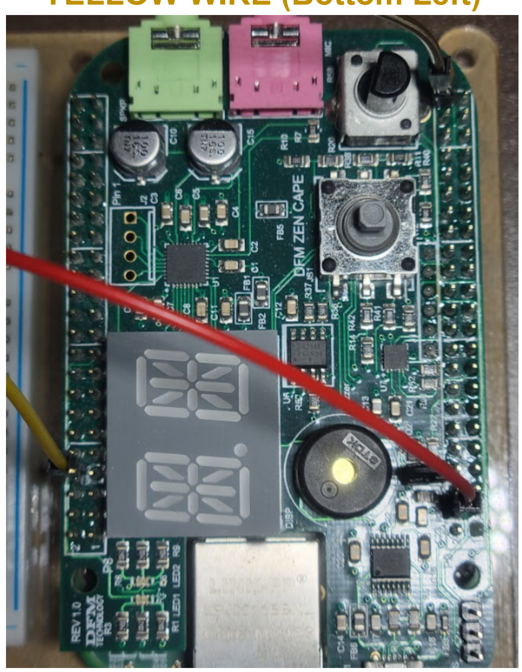
RED WIRE (Bottom Right)



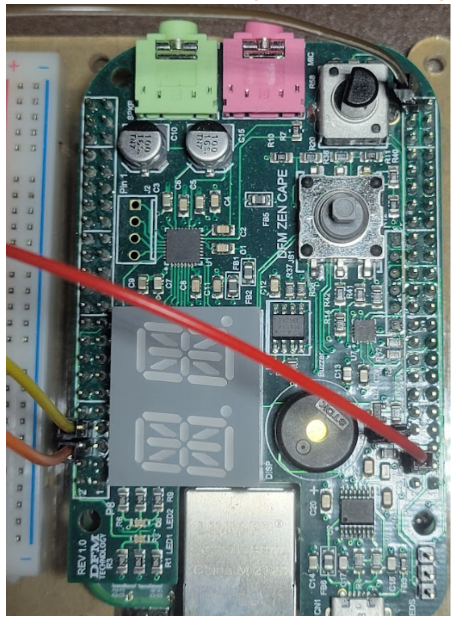
</td></tr>
</table>

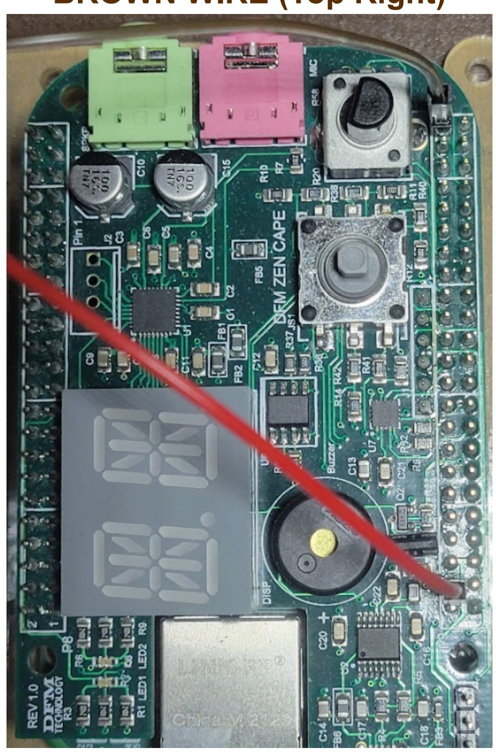
<table>
<tr><td>2</td><td>Trigger</td><td>P8</td><td>8 (GPIO 67)</td></tr>
<tr><td colspan="4">

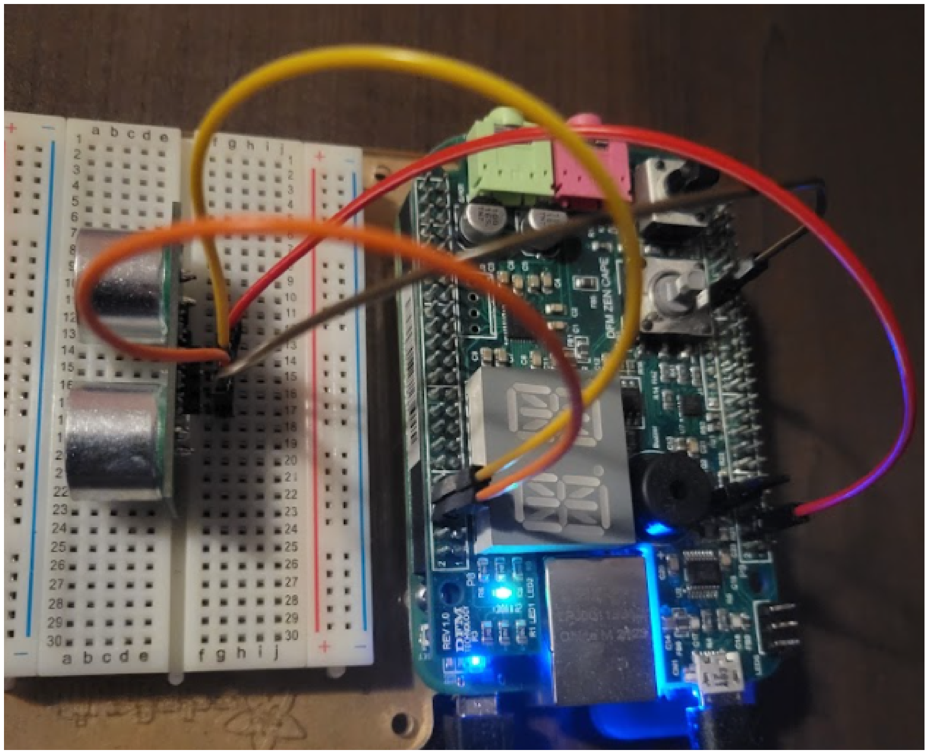
YELLOW WIRE (Bottom Left)



</td></tr>
</table>

<table>
<tr><td>3</td><td>Echo</td><td>P8</td><td>6 (GPIO 35)</td></tr>
<tr><td colspan="4">ORANGE WIRE (Bottom Left)

</td></tr>
</table>

<table>
<tr><td>4</td><td>Ground</td><td>P9</td><td>46 (DGND)</td></tr>
<tr><td colspan="4">BROWN WIRE (Top Right)

</td></tr>
</table>

Completed Wiring


Refer to the [Appendix 3.1](#) for all digital I/Os on the BeagleBone

2.2. Programming (C++)

A sample C++ project is provided in the support files. The GPIO trigger must be configured as an output, and the GPIO echo must be an input. When the trigger is activated, the sensor will send an ultrasonic wave, which bounces back and hits the echo. We must time how long this takes, and do some basic math using the speed of sound to figure out the distance. Sometimes an echo is not received, so it's useful to keep track of a maximum wait time, at which the code can be set up to return an infinite distance.

Some Python pseudo code for a higher level understanding of how the trigger/echo works (from here. Note: the wiring for this guide is for the 5V version, not the 3.3V one, and the RPI library will not work on the Beaglebone):

```python
def distance():
    # set Trigger to HIGH
    GPIO.output(GPIO_TRIGGER, True)

    # set Trigger after 0.01ms to LOW
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER, False)

    StartTime = time.time()
    StopTime = time.time()

    # save StartTime
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()

    # save time of arrival
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()

    # time difference between start and arrival
    TimeElapsed = StopTime - StartTime
    # multiply with the sonic speed (34300 cm/s)
    # and divide by 2, because there and back
    distance = (TimeElapsed * 34300) / 2

    return distance
```

3. Troubleshooting

- If wiring is correct and you are unable to read from the sensor, try different GPIO pins for both Echo and Trigger.
- The sensor may not reliably detect soft objects. It relies on echoing sound, so if the object is too soft it may absorb too much sound and not produce a proper reading. It could also happen if the object is too small or has an irregular shape.
- There may be a lot of noise when measuring readings from the sensor. Consider using methods such as hysteresis, exponential averaging, and outlier removal when working with readings.

- Sometimes the wires can come loose; especially on the side of the beaglebone. If after attaching the wire to the beaglebone pin it still feels very loose, try pushing down just from the wire (just above the black plastic casing). This can help make a stronger connection.

4. Appendix

4.1. All possible I/O

65 possible digital I/Os

P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71