SOUND AND THE SEAFLOOR

Determining Bathymetry Using Student-Built Acoustic Sensors

By Robert Levine, Sasha Seroy, and Daniel Grünbaum

SUPPLEMENT S1

Sensor Assembly Guide

The following instructions provide instructors and students with the necessary steps to assemble and operate the acoustic sensor used in the activity. Instructions in the Sensor Assembly and Initial Exploration section of the activity are summaries of sections 3, 4, and 5 below and assume students have microcontrollers that have been set up according to the instructions provided in sections 1 and 2 below. Instructors are encouraged to have students conduct as much of the sensor construction and assembly as possible, given the time and resources available.

Oceanography | https://doi.org/10.5670/oceanog.2020.305

1. MATERIALS LIST

Suggested materials are detailed in Table S1, including cost estimates. While we recommend the Feather Huzzah with ESP8266, the activity and materials provided are meant for use with any MicroPython-based microcontroller. Support for additional sensors and alternative microcontrollers is provided in the PublicSensors Microcontroller Kit GitHub repository. Other microcontrollers or programming languages (e.g., CircuitPython, Arduino) can be used, though instructions are not provided. We recommend the use of a breadboard and male/male jumper wires to connect the ultrasonic module to the microcontroller, though female/female wires directly connecting the module and microcontroller can be used in place of a breadboard.

Table S1. Required materials for sensor assembly, with cost per unit and total pricing for 12 kits (10 student groups + 2 spare). Additional materials for conducting the activity are included. Many of the Adafruit products listed are also available via Digi-Key (www.digikey.com) and other online retailers.

SENSOR MATERIALS		
COMPONENTS	**COST PER UNIT IN USD**	**COST PER 12 KITS**
Feather Huzzah with ESP8266	$16.95 (www.adafruit.com)	$203.40 (12)
Male/Male Jumper Wires	$1.95 (www.adafruit.com)	$7.80 (4 sets of 20)
JSN-SR04T (or JSN-SR04T2.0/AJ-SR04M) waterproof ultrasonic module	$10.95 (www.jameco.com)	$131.40 (12)
MicroUSB cable	$2.95 (www.adafruit.com)	$35.40 (12)
Half-size breadboard	$5.00 (www.adafruit.com)	$60.00 (12)
Total Cost		**$438.00**

ADDITIONAL MATERIALS	
Laptop	1 per group to communicate with sensors
Color Coding Electrical Tape	1 roll to label transducer cable
Tape Measure	1 to measure and mark sampling positions

2. ESP8266 SETUP

The following are instructions for getting started with an ESP8266-based microcontroller. Additional information and instructions can be found in the open-access textbook *Foundations of Environmental Sensing*, available via PublicSensors. The textbook and website also include additional activities and applications for MicroPython-based microcontrollers using other sensor modules.

2.1. Soldering

Microcontrollers can be purchased pre-soldered or with loose pin headers (Figure S1). Headers are required for use with both a breadboard and female jumper wires. Pin headers come in strips and can be cut to size using a sharp edge (wire cutters, knife, etc.). The pins can then be soldered in place. Using a breadboard to hold the pins and microcontroller will help stabilize the materials while soldering. Once soldered, the microcontroller should look like the photo below (Figure S1).

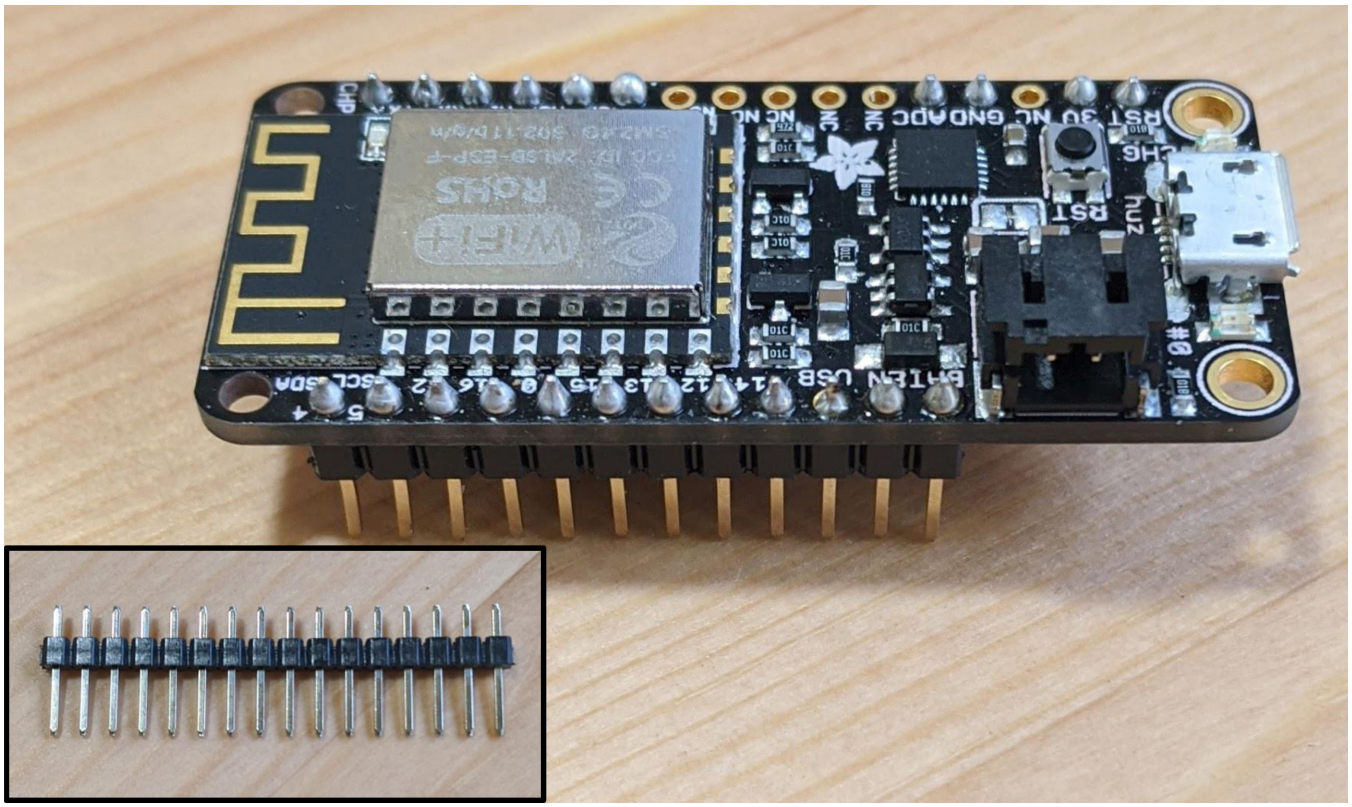


FIGURE S1. Feather Huzzah with ESP8266, with pin headers (shown prior to soldering in the inset, lower left) soldered for use in a breadboard.

2.2. MicroPython Firmware

MicroPython is a Python programming language interpreter for use on microcontrollers. It allows students to use Python, a programming language commonly used in geosciences, to interact with their sensors. New microcontrollers will require MicroPython firmware be installed. The most recent firmware available, as well as instructions for loading the firmware onto the microcontroller, can be found in the MicroPython documentation available on their website, as well as in the activity GitHub repository. To install the firmware:

1. Download the ESP8266 MicroPython firmware from MicroPython.org or the GitHub repository as linked above.

2. Install esptool (https://github.com/espressif/esptool/).

3. Connect the microcontroller to a computer using the microUSB cable and erase the flash by running esptool from the command line or terminal:

```
(base) C:\>esptool --port COM38 erase_flash
esptool.py v2.8
Serial port COM38
Connecting....
Detecting chip type... ESP8266
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: bc:dd:c2:14:74:c0
Uploading stub...
Running stub...
Stub running...
Erasing flash (this may take a while)...
Chip erase completed successfully in 6.8s
Hard resetting via RTS pin...

(base) C:\>
```

"COM38" is the device address of the microcontroller and will likely be different based on the computer and microcontroller. You can determine the device name using the Windows device manager or using the 'ls /dev/tty.*' command on Mac OS or Linux.

4. Navigate to the directory of the firmware file downloaded in step 1 and deploy the new firmware:

```
(base) C:\firmware_folder>esptool --port COM38 --baud 460800 write_flash --flash_size=detect 0 firmware.bin
esptool.py v2.8
Serial port COM38
Connecting....
Detecting chip type... ESP8266
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: bc:dd:c2:14:74:c0
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 460800
Changed.
Configuring flash size...
Auto-detected Flash size: 4MB
Flash params set to 0x0040
Compressed 601860 bytes to 393107...
Wrote 601860 bytes (393107 compressed) at 0x00000000 in 9.3 seconds (effective 520.4 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...

(base) C:\firmware_folder>
```

2.3. Transferring Files to the Microcontroller

The ESP8266 Feather can be communicated with via USB or WiFi. For simplicity, we recommend having students start using the USB connection. WiFi connectivity can be useful when working with more advanced environmental sensors that are likely to be deployed in waterproof or cableless housings. To communicate with ESP8266 microcontrollers running MicroPython via USB, we suggest using one of the following tools, both of which are available for all operating systems:

- mpfshell: a Python-based command line utility
- Beagle Term: an extension for Google Chrome

The key difference is that mpfshell allows for file transfer while Beagle Term does not. The following instructions are for transferring the necessary python files to the microcontrollers for the activity using mpfshell:

1. Install mpfshell by downloading the necessary files and following the instructions on the mpfshell GitHub repository.

2. Download "boot.py," "hcsr04.py," and "print_distance.py" from the activity GitHub repository.

3. With the microcontroller connected to the computer, run mpfshell in a command window or terminal by typing "mpfshell":

```
(base) C:\>mpfshell

** Micropython File Shell v0.9.1, sw@kaltpost.de **
-- Running on Python 3.7 using PySerial 3.4 --

mpfs [/]>
```

4. To connect to the microcontroller, type "open *SERIAL_DEVICE_ID*," where *SERIAL_DEVICE_ID* is the device label or com port of the microcontroller, as described in section 2.2 above:

```
mpfs [/]> open COM38
Connected to esp8266
mpfs [/]>
```

5. To add files to the microcontroller, use "put *FILENAME*.py," where *FILENAME*.py is the target file:

```
mpfs [/]> open COM38
Connected to esp8266
mpfs [/]> put hcsr04.py
mpfs [/]>
```

We recommend that instructors conduct this portion of the setup themselves and provide students with microcontrollers that have MicroPython firmware and necessary files already installed. Students can then use Beagle Term, which allows for serial communication within a Google Chrome browser window, to run commands and collect measurements. The instructions for conducting the activity using Beagle Term are presented in section 4 below. *Foundations of Environmental Sensing* provides additional instructions for using mpfshell (Chapter 1.2) or the ESP8266 WiFi connection (Chapter 1.3) for communication and file transfer using MicroPython's WebREPL client.

3. SENSOR ASSEMBLY

The following instructions are included in the Sensor Assembly and Initial Exploration section of the activity and are also contained in the student handouts in the associated GitHub repository. The microUSB cable provides 5V to the board, which means we can use the USB pin on the ESP8266 to give 5V to the JSN-SR04T. Newer versions of the sensor (JSN-SR04T-2.0) can operate at lower voltage, but still perform optimally at 5 volts. The JSN-SR04T sensor has four pins: GND, VCC, Trig, and Echo.

1. Connect the Trig pin on the JSN-SR04T to GPIO 12 on the ESP8266 Feather.
2. Connect the Echo pin on the JSN-SR04T to GPIO 14 on the ESP8266 Feather.
3. Connect the GND pin on the JSN-SR04T to the GND pin on the ESP8266 Feather.
4. Connect the VCC pin on the JSN-SR04T to the USB pin on the ESP8266 Feather.
   a. Using the breadboard, both the GND and VCC pins can be connected to the microcontroller via the breadboard rails (**Figure S2**).
5. If not already connected, connect the transducer to the JSN-SR04T module.

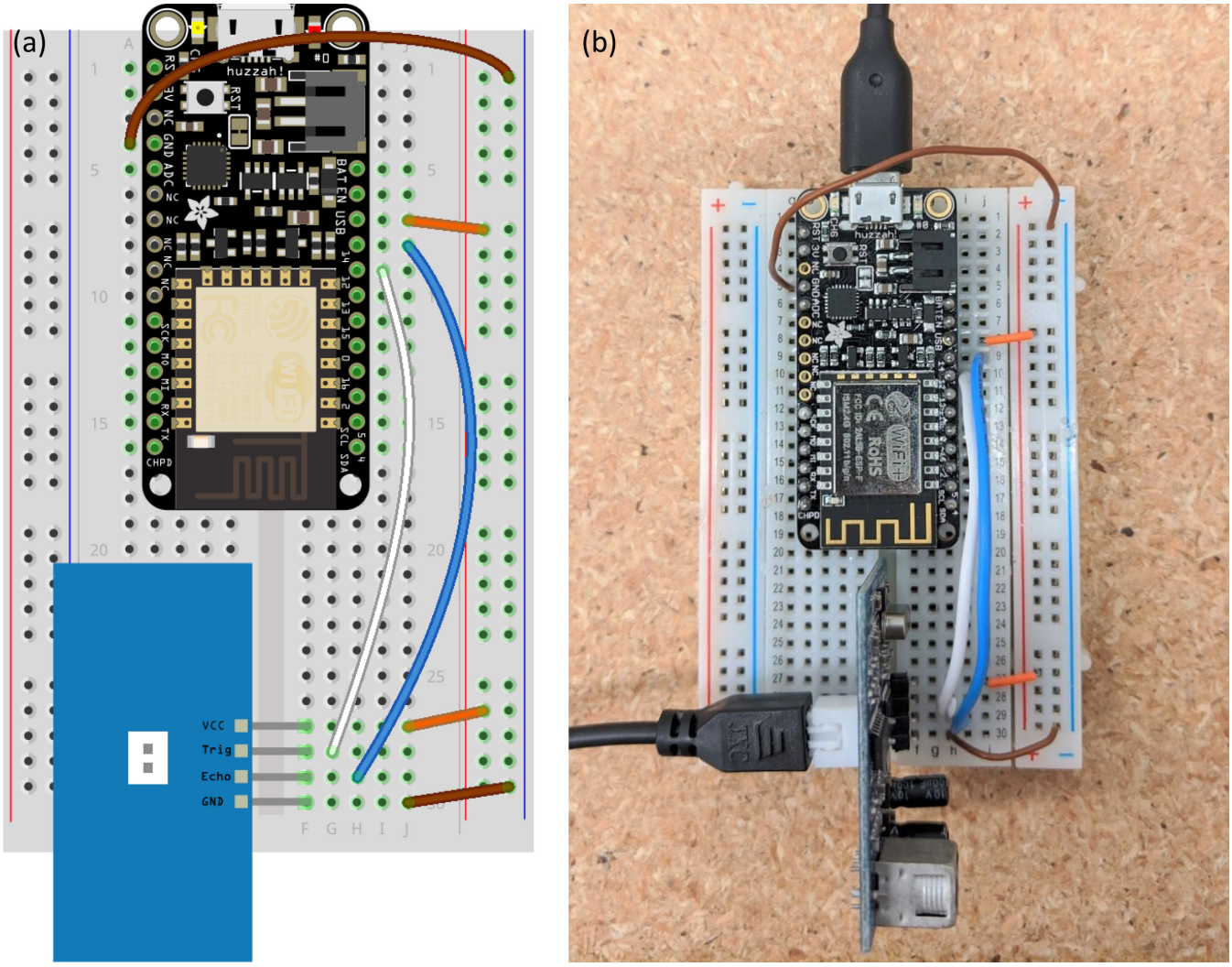
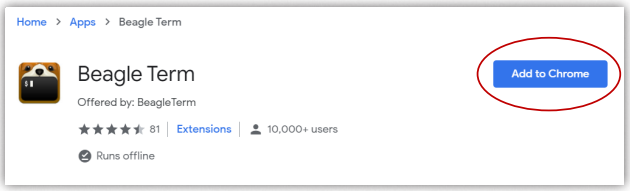


FIGURE S2. (a) Fritzing (Fritzing.org) diagram showing the wiring required for the JSN-SR04T using a breadboard. (b) Photo of assembled sensor.

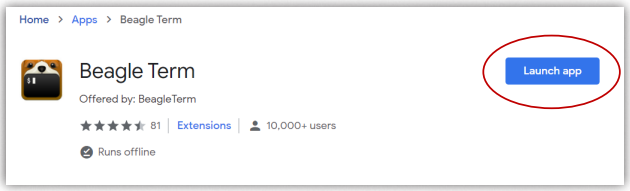
4. CONNECTING TO THE MICROCONTROLLER

The instructions below should be provided to students for communication via Beagle Term. Additional materials and troubleshooting are available on the PublicSensors resources page.

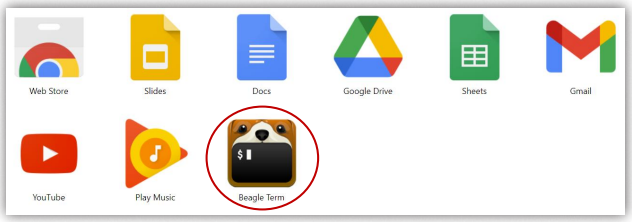
1. Install the Google Chrome app Beagle Term, which emulates a serial terminal inside of a browser window. From the Chrome Web Store, add the Beagle Term app to the browser. If you have difficulty finding it, searching "Beagle Term"' on Google will direct you to the app page in the Chrome Web Store.



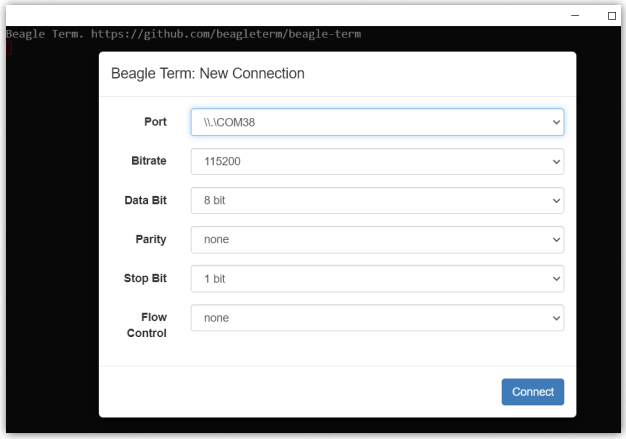
2. Once you add the app, you will be able to launch it from the same page:



Or access it via the Chrome Apps page:

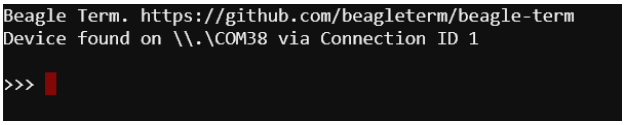


3. Plug in the ESP8266 Feather using the microUSB cable, then launch the Beagle Term app. It will automatically detect the necessary settings if the ESP8266 Feather is plugged in. Click "Connect":



Some computers will require an additional USB to UART driver, available for Mac and Windows. Occasionally Beagle Term will not detect the correct COM port. If you are not brought to the correct screen, see if there are other COM port options in the dropdown menu. If so, repeat this process using a different COM port. Often, it will be the highest number COM port. For Macs, try a COM port that contains the address: tty/USBmodem/.

4. Once you connect, click inside the Beagle Term window and press "Enter." You should see a screen like the one below, which indicates you are connected to a MicroPython device:



If '>>>' does not appear as shown, try selecting the terminal window with the cursor and pressing "Enter".

5. SENSOR OPERATION

The assembled sensor uses the *hcsr04* library to communicate between the microcontroller and the distance module. The library and the associated python script, which can be used to collect measurements, can be found in the microcontroller code folder of the associated GitHub repository.

The *hcsr04.HCSR04* class requires three input variables:

1. *trigger_pin*, the GPIO pin on the ESP8266 Feather corresponding to the Trig pin on the JSN-SR04T (Pin 12 following the instructions in section 4 above).
2. *echo_pin*, the GPIO pin on the ESP8266 Feather corresponding to the Echo pin on the JSN-SR04T (Pin 14 following the instructions in section 4 above).
3. *c*, the speed of sound to use to calculate distance.

Because this activity is done in water, have students choose a speed of sound appropriate for the body of water (i.e., 1,480 m s^{-1} for freshwater, 1,500 m s^{-1} for salt water). Values specific to the body of water being measured can be determined using calculations as described in the Activity Extensions supplementary material S2 or instrumentation such as that for conductivity, temperature, and depth (CTD) to better assess the speed of sound for the activity.

To collect a measurement, first import the *hcsr04* library so that you can initialize the sensor with the appropriate pin designations and speed of sound. You can then use the *distance* function to take a measurement (reported in centimeters):

```
Beagle Term. https://github.com/beagleterm/beagle-term
Device found on \\.\COM38 via Connection ID 3

>>> import hcsr04
>>> sensor = hcsr04.HCSR04(trigger_pin=12,echo_pin=14,c=1500)
>>> sensor.distance()
1001.0
>>>
```