HardwareX 8 (2020) e00112

Contents lists available at ScienceDirect

HardwareX

journal homepage: www.elsevier.com/locate/ohx

# Low-cost and precise inline pressure sensor housing and DAQ for use in laboratory experiments

Luke Goertzen [a], Nicole Mehr [b], Manuel Lopez [a,*], Chet Udell [a], John S. Selker [b]

[a] OPEnS Lab, Oregon State University, OR, United States
[b] Department of Biological & Ecological Engineering, Oregon State University, OR, United States

ARTICLE INFO

Article history:
Received 1 May 2019
Received in revised form 22 April 2020
Accepted 16 May 2020

Keywords:
Programmable sensor
Flow cell experiments
Arduino
Data logging
Customizable sensor
Temperature-compensated

ABSTRACT

There are many applications for inline pressure sensors, including fluid flow experiments, sensor field deployments, pumps, and Internet of Things systems. We developed a low-cost (~US$56), open-source, customizable inline pressure sensor system with operational flexibility and simple data logging. Most pressure sensors are expensive, not customizable, specific to a single tubing size, provide only analog readings, have poor stability and precision, or are incomplete without a data logger. These issues limit the usefulness of such hardware. Our system addresses all of these concerns. The customizability of both the hardware and firmware (via options or code modification) allows for the device to be tailored easily to each application. Tubing diameter, adapter dimensions, sensor used, logging behavior, and integration with other systems can be configured with ease. Much of the practicality and configurability of the software and hardware arise from the use of our Loom code and ecosystem. We present experimental data for the flow of a viscous fluid between two parallel plates that shows that sudden changes in fluid properties are not always discernible in static images, but are detectable as pressure signals with our inline pressure sensor.



Specifications table

| Hardware name | *InPres* |
|---|---|
| Subject area | Engineering and Material Science |
| Hardware type | Measuring physical properties and in-lab sensors |
| Open Source License | CERN Open Hardware License (hardware)GNU General Public License v3.0 (software) |
| Cost of Hardware | US$56 |
| Source File Repository | GitHub: https://github.com/OPEnSLab-OSU/Inline-Pressure-Sensor<br>Zenodo: https://zenodo.org/record/2459494<br>OSF: https://osf.io/2mx4a/ |

* Corresponding author.
*E-mail address:* lopezalj@oregonstate.edu (M. Lopez).

https://doi.org/10.1016/j.ohx.2020.e00112
2468-0672/© 2020 Published by Elsevier Ltd.

## 1. Hardware in context

Inline pressure sensors can be found in a variety of laboratory applications involving the flow of gases or liquids to detect changes in fluid properties. Bernoulli's equation states that an increase in fluid velocity occurs simultaneously with a decrease in pressure, which highlights the importance of having accurate, near real time pressure readings [1]. A sudden change in pressure can be used as a signal to detect simultaneous changes in velocity, and thus the flow regime or the properties of a fluid.

There is a need for an affordable, customizable, and reliable pressure sensor for use in laboratory flow chamber experiments. For a pressure sensor to be reliable, the avoidance of air bubbles and dead space is critical to the success of flow chamber experiments. Many other inline pressure sensors used today are restrictive because they often require a specific tubing size, are relatively expensive, are not inline, and/or only provide unconverted analog readings. Most importantly, they also generally lack a device to receive, parse, or log the pressure sensor data. For example, the SparkFun Pressure Sensor Breakout - MS5803-14BA (part number SEN-12909, $59.95) mounts the MS5803-14BA pressure sensor (0–14 bar range, accuracy ±20 mbar, precision of 0.2 mbar at highest oversampling rate) on a printed circuit board, but would require the addition of a waterproof physical interface and a component to receive, parse, or log the pressure sensor data [2,3]. The accuracy and precision of this pressure sensor were insufficient for the application described here, but it would provide a comparable cost alternate to the sensing part of the system for less precise applications. At the other extreme in terms of price, combining the ELVEFLOW MPS 3 Microfluidic High Precision Pressure Sensor (product SKU LVF-3532, 2 bar version, €435) [4], and Sensor Reader (product SKU LVF-3458, €1,200) [5] creates a system for 3/32″ ID tubing, unless a separate accessories kit is purchased, with more comparable accuracy (±0.2% of 2 bar or ± 40 mbar) to our system.

In order to expand upon the available options for inline pressure sensors, an open source, affordable inline pressure sensor has been designed by this study and can be configured or customized for a variety of experiments and fluids. This is also a study of a useful application of our Loom ecosystem, which provides easily customized open-source code to manage a wide variety of other sensing and data logging applications, including Internet of Things (IoT) systems. Loom abstracts the details of interfacing with a variety of hardware and platforms used in data acquisition and telemetry, providing a simplified and uniform way to program the behavior of the system. Loom was used rather than standalone code because of its versatility and numerous features, which simplify the process of configuring the system to various uses of the sensor [6]. There are other data acquisition systems, such as EnviroDIY's Mayfly [7] or the Northern Widget LLC ALog [8], that we could have used, but Loom was tailored to our needs, and one of the intents of this application was to test Loom's capabilities.

To demonstrate the usability of the pressure sensor and our Loom code and ecosystem, a viscous fluid was injected and withdrawn into saltwater between two parallel glass plates using a syringe pump. Imaging techniques are commonly used to visualize a flow system. Our results show that imaging techniques are not sufficient because sudden changes in velocity are not always discernible in static images, but are detectable as pressure signals.

## 2. Hardware description

The main components of our system are a pressure sensor, a development board with a microcontroller and SD logger add-on, a custom printed circuit board (PCB), and a 3D printed adapter [9]. We used the MS5803-02BA sensor [10], which is rated for operating pressures from 0.3 bar to 1.1 bar with accuracy of ± 1.5 mbar, and precision of 0.024 mbar (at the highest oversampling setting). This sensor also measures temperature, primarily for compensated pressure readings, but can eliminate the need for a separate temperature sensor. The Adafruit Feather M0 Proto development board with the Adalogger FeatherWing were selected to control the system and permit data logging to microSD cards, respectively. The boards share the same form factor and are designed to stack which simplifies wiring. The PCB also serves to simplify the wiring between the sensor and the development board. The 3D-printable sensor adapter, which is screwed directly onto the PCB, houses the sensor and provides the interface between the sensor and the tubing through which the fluid is flowing by making the sensor inline with the tubing. The fluid flows through the tubing into the adapter across the sensor and then out of the adapter through tubing on the other side of the adapter.

We designed the system to be flexible, so that different tubing or sensors can be used with some modification to the adapter, PCB, or code. The PCB and 3D adapter were designed primarily for the MS5803-XX family of pressure sensors, but we expect that with minimal modification the adapter would also work with the MS5837-XX family of pressure sensors [11]. This would also require modest modification of the PCB and code to properly interface with these sensors.

The adapter can easily be modified using Autodesk's Fusion 360 [12] to accommodate different inner and outer diameters of the adapter arms (among other parameters, detailed below) for suitability to various experimental setups. As such, only this adapter itself may need to be modified and switched out to use the system in an alternate context. Modification to accommodate a different sensor follows the same process. The adapter is intended to be printed on a Stereolithography (SLA) printer to avoid permeability and resolution issues more common on Fused Deposition Modeling (FDM) style 3D printers [13]. Various resins are available to obtain desired properties, such as translucency, chemical resistance, and mechanical durability [14].

To control the system, our selected development board needs to be programmed in C/C++. We did this using our existing, open-source, code base known as Loom which, in addition to support for our sensor, provides support for many other sensing, data logging, and Internet of Things systems with extensive user-friendly customizability [6,9]. With Loom, the behavior of our system or addition of more hardware can be configured via options without needing to modify the code. Further application of Loom can allow for easy integration of the pressure sensor into data-driven systems more elaborate than a simple data logger. While one can interface with the hardware through different code, the usage of Loom is central to providing a complete system in this study. The operational instructions below assume the usage of Loom via the Arduino IDE (version 1.8.5 for this application) [15].

Potential applications:

Millifluidic experiments
Hele-Shaw cells
Flow channels
Flow regulation feedback for liquid sampling devices

## 3. Design files

Design Files Summary

| Design file name | File type | Open source license | Location of the file |
|---|---|---|---|
| Sensor_Adapter.f3d | F3D | CERN Open Hardware License | https://osf.io/2mx4a/files/ |
| Sensor_Adapter.stl | STL | CERN Open Hardware License | https://osf.io/gztkm/ |
| MS5803-02BA.brd | BRD | CERN Open Hardware License | https://osf.io/ewbh5/ |
| MS5803-02BA.sch | SCH | CERN Open Hardware License | https://osf.io/fgqy6/ |
| Code | C++ | GNU General Public License v3.0 | https://osf.io/2mx4a/files/ |

**Sensor Adapter.f3d** is the Fusion 360 [12] design file of the sensor adapter that users can modify to fit their needs and export as a .stl file.

**Sensor Adapter.stl** is the file that gets loaded to the print preparation software to generate a file for a 3D printer for printing. The provided file is the version of the sensor adaptor with "standard" parameters as used in our experiments.

**MS5803-2BA.brd** is the file that the user can load into AutoDesk EAGLE [16] to get the layout of the board.

**MS5803-2BA.sch** complements the brd file that goes into EAGLE [16], and has the schematic of the electronics.

**Code** is the C++ code that is used to operate the device and sensor. Available via GitHub [9] under a GNU general public license.

## 4. Bill of materials

Bill of Materials

| Designator | Component | Number | Cost per unit (US$) | Total cost (US$) | Source of materials | Material type |
|---|---|---|---|---|---|---|
| Pressure Sensor | MS5803-02BA Pressure Sensor | 1 | $10.72 | $10.72 | DigiKey | Non-specific |
| Sensor Adapter | Custom Component | 1 | $0.54* | $0.54 | Printed on a Form-2 | Polymer |
| Development board | Adafruit Feather M0 Basic Proto - ATSAMD21 Cortex M0 | 1 | $19.95 | $19.95 | Adafruit | Non-specific |
| Data Logger | Adalogger FeatherWing - RTC + SD Add-on | 1 | $8.95 | $8.95 | Adafruit | Non-specific |
| Sensor PCB | Custom Component | 1 | $0.78 | $0.78* | OshPark | FR4 |
| Coin Cell Battery | CR1220 3 V Coin Cell Battery | 1 | $0.95 | $0.95 | Adafruit | Non-specific |
| MicroSD Card | 8 GB MicroSD Card | 1 | $3.99 | $3.99 | Amazon | Non-specific |
| Battery** | Lithium Ion Battery – 3.7v 400mAh | 1 | $6.95 | $6.95*** | Adafruit | Non-specific |
| Bolt | M2 bolt | 2 | $0.11 | $0.22 | McMaster-Carr | 18–8 Stainless Steel |

(continued on next page)

(continued)

| Designator | Component | Number | Cost per unit (US$) | Total cost (US$) | Source of materials | Material type |
|---|---|---|---|---|---|---|
| Threaded Inserts | M2 Threaded Inserts | 2 | $0.11 | $0.22 | McMaster-Carr | Brass |
| Stackable Headers | Stackable Headers | 1 | $1.25 | $1.25 | Adafruit | Non-specific |
| PCB Resistors | 10kΩ Resistor (2012/0805 case code) | 2 | $0.35 | $0.70 | Mouser | Non-specific |
| PCB Capacitor | 0.1μF capacitor (2012/0805 case code) | 1 | $0.28 | $0.28 | Mouser | Non-specific |
| PCB Header | Single Row, Male Headers 2.54 mm (5 pins) | 1 | $0.07 | $0.07 | Amazon | Non-specific |

Pricing Notes

* *price will vary with source*
** *optional*
*** *price will vary with capacity*

Additional equipment that may be needed:

SLA 3D printer, such as Form 2, to print the sensor adapter
Soldering station for soldering sensor and other parts to PCB

## 5. Build instructions

### *5.1. Modifying the sensor and tubing (optional)*

Using parameters in the Sensor_Adapter.f3d file, the arms (inlet and outlet) of the adapter for the pressure sensor can be modified for different inner and outer diameters to accommodate different tubing sizes. Additionally, the diameter of the space for inserting the pressure sensor can be modified for use with a different pressure sensor (support for which is not provided in the additional files or code). The process for modifying the .f3d file is as follows:

Open the Sensor_Adapter.f3d file in AutoDesk Fusion 360 [12]
Ensure that you are in the 'Model' workspace
From the 'Modify' drop-down menu, select 'Change Parameters'
Change any of the following parameters (which are further documented in the 'Change Parameters' window):
- **Outer_diameter_arm_end** – Outer diameter of the end of adapter arms
- **Outer_diameter_arm_start** – Outer diameter of the adapter arms closest to sensor (keep slightly larger than 'Outer_diameter_arm_end' for taper of arms for more secure connection of tubing)
- **Inner_diameter_arms** – Inner diameter of adapter arms
- **Diameter_pressure_sensor** – Diameter of the hole for pressure sensor
- **Diameter_threaded_insert** – Diameter of holes for screws to attach to PCB

### *5.2. 3D printing*

The first requirement for constructing the inline pressure sensor is to print the adapter into which the sensor is inserted (Fig. 1). The adapter should be printed with an SLA printer to ensure that the part is impermeable using a resin that will cure rigid and is compatible with the fluid being used in the system [14]. The provided STL file can be used to print the part with the settings utilized for this application, or an STL file can be exported from Fusion 360 [12] after modifying the part's parameters. Using the highest resolution of the printer will yield the best fit of the adapter to the sensor.

### *5.3. PCB assembly*

This assembly is meant to guide hand soldering of surface-mounted devices.

Clean the pads on the front and back of the PCB with isopropyl alcohol to ensure the solder will have a secure connection to the pads.

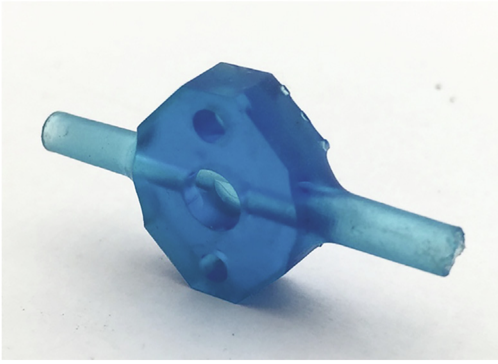

**Fig. 1.** The sensor adapter as printed on a Form 2 using a translucent tough resin (formlabs RS-F2-TOTL-05) for this application of the system. Approximately 4.3 cm in length, 2.0 cm in width.

Cover the pads with a thin layer of solder (Fig. 2A: sensor pads; Fig. 2B: capacitor and resistor pads).
Clean the pads again with isopropyl alcohol and place the capacitor and resistors on their respective locations on the back side of the PCB. The board should look like Fig. 3A.
Using a hot air gun, blow the heat at ~350 °C and at the lowest possible blow-speed. This is to ensure that the components do not fly off while still providing sufficient heat. The finished board should look like Fig. 3B. Allow the board to cool before continuing with the sensor itself.
As before, place the pressure sensor on the pads on the front side of the PCB, and then proceed with blowing the hot air to solder it to the pads.
Finally, trim about 1.5 mm off of the shorter ends of the pins of the male headers and solder that side into the board. The final assembly should resemble Fig. 4.

### 5.4. Attaching PCB and sensor to adaptor

Using a soldering iron, carefully heat the M2 threaded insert to approximately 330 °C and press it into the corresponding hole. Then, use M2 bolts to secure the PCB to the casing, enclosing the attached sensor (Fig. 5, Fig. 6).

### 5.5. Wiring Adalogger and sensor PCB to Adafruit Feather

The Adalogger FeatherWing allows writing to microSD cards and is attached directly above or below the Feather M0 Proto by soldering stackable headers to the Adalogger. Wires from the pressure sensor PCB are attached to either the development board or the Adalogger.

There are four wires that need to be connected to establish an $I^2C$ connection from the Feather to the pressure sensor: power, ground, data, and clock. Additionally, the CSB pin needs to be pulled high (default) or low to set the sensor's $I^2C$ address. These wires should be connected according to the provided schematic (Fig. 7, Supplement A) and table (Table 1):

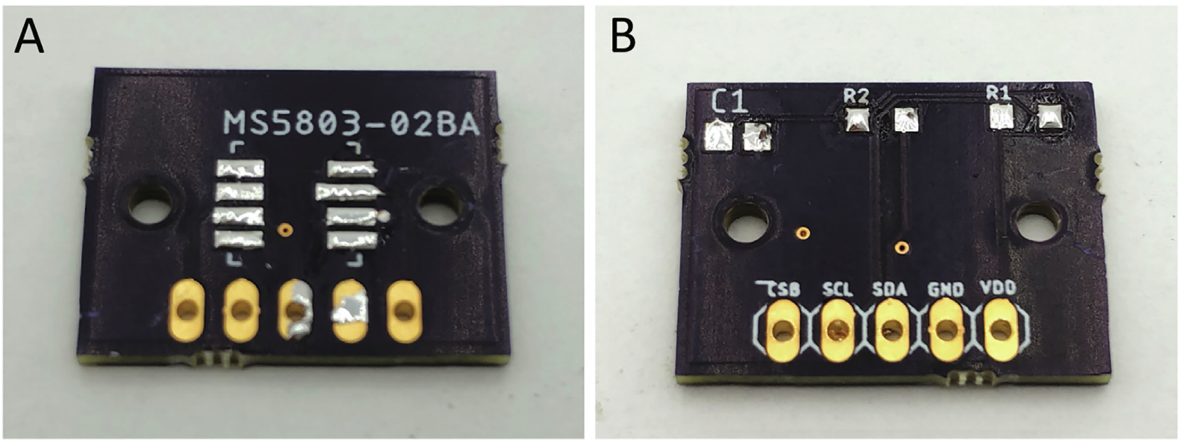


**Fig. 2.** The PCB with a small layer of solder on the pads that will have SMD components. A) Front side: sensor pads. B) Back side: capacitor (C1) and resistor (R1, R2) pads.

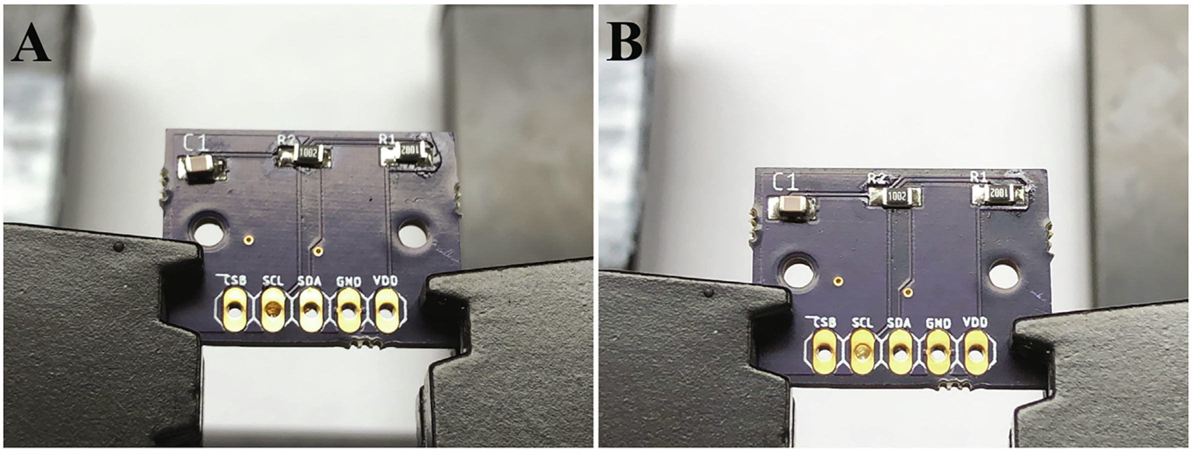


**Fig. 3.** A) The PCB with the capacitor (C1, left) and resistors (middle, R1, and right, R2) placed on their respective pads before soldering. The middle and right resistors are interchangeable as they are the same value. B) The finished part after soldering the components on.

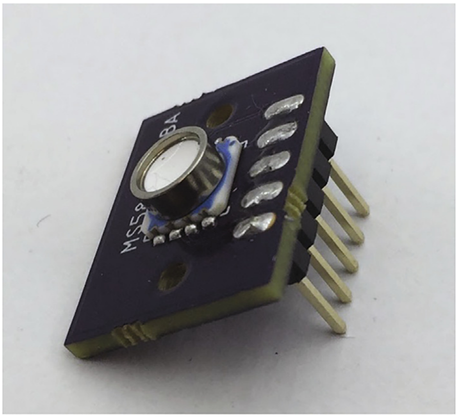

**Fig. 4.** Final assembly of the PCB with the sensor.

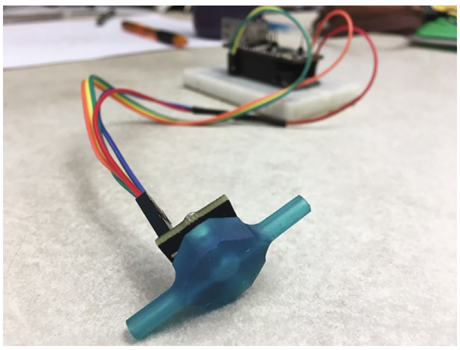

**Fig. 5.** Image of adapter attached to PCB with enclosed sensor.

### 5.6. Attaching tubing

The tubing is attached simply by sliding it over both arms of the adapter (Fig. 8), which can be slightly tapered for a secure connection. Use the customizable Fusion 360 file to ensure that the dimensions are correct for the tubing being used, or to adjust the taper of the adapter arms Table 2.

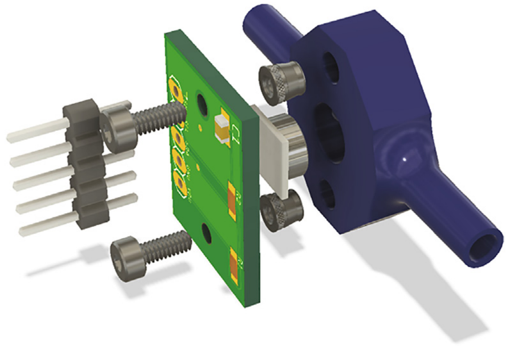

**Fig. 6.** Exploded render of PCB parts, sensor, and sensor adapter using Fusion 360.

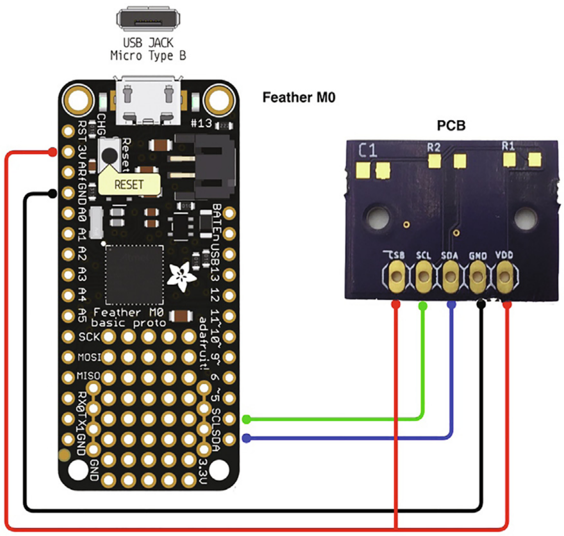


**Fig. 7.** Wiring schematic from Feather M0 Proto development board to sensor PCB.

**Table 1**
Sensor performances [9].

| | |
|---|---|
| **Pressure** | |
| Accuracy | ±1.5 mbar |
| Precision [(1)] | 0.13/0.084/0.054/0.036/0.024 mbar |
| Operating Range | 0.3 to 1.1 bar |
| Maximum Pressure | 10 bar |
| Response Time[(1)] | 0.5/1.1/2.1/4.1/8.22 ms |
| **Temperature** | |
| Accuracy | ±0.8 °C |
| Resolution | <0.01 °C |
| Operational Range | −40 °C – 85 °C |

[(1)] Oversampling Ratio: 256/512/1024/2048/4096.

## 6. Operating instructions

The pressure sensor is simple to operate with various options for how the data may be measured and recorded. The only setup steps (described below) are: ensuring the sensor is connected, synchronizing the time (optional), configuring the code, and uploading the code to the development board.

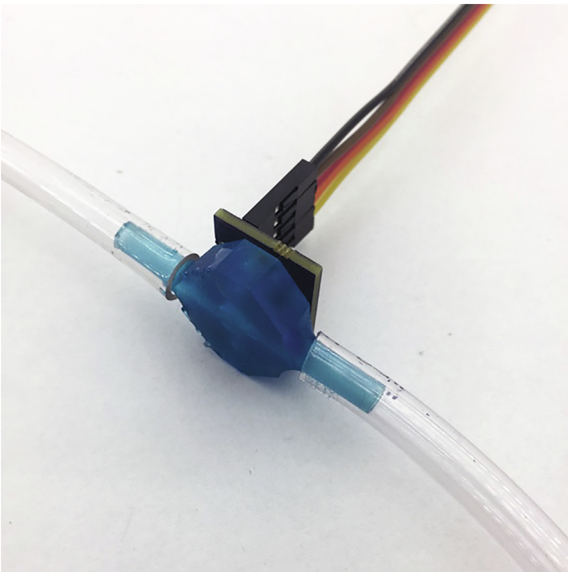

**Fig. 8.** Sensor adapter with connected tubing.

**Table 2**
PCB wiring to Feather development board.

| Pin connections | |
|---|---|
| PCB | Feather |
| VDD | 3 V |
| GND | GND |
| SDA | SDA |
| SCL | SCL |
| CSB | 3 V |

Notes: An $I^2C$ multiplexer can be used to run additional pressure (or other $I^2C$) sensors. The details of doing so are beyond the scope of this paper, but are documented in the Loom repository [6].
The MS5803 sensor also supports the SPI interface, but our provided PCB and code do not.

### 6.1. Connecting the sensor

See Sections 5.5 and 5.6 of this document for instructions on wiring the sensor PCB to the Feather development board and connecting the tubing to the adapter, respectively.

### 6.2. Time synchronization (optional)

If time synchronization with a computer is needed, an additional procedure needs to be followed to synchronize the RTC clock on the Adalogger FeatherWing. This entails uploading separate code to the Feather development board and running a basic command or script on the computer you wish to synchronize the time with.

- Open the Serial_RTC_Time_Sync.ino file in the Arduino IDE
- Modify lines 17–18 to select the correct RTC being used (the part listed in the bill of materials uses a PCF8523)
- Modify line 20 to specify a second adjustment to account for time zone difference from UTC
- Upload the firmware to the Feather M0 (and do not open the Serial Monitor)
- Send the computer time to the Feather:
  - Mac/Linux
    - Check the COM port of the Arduino via the Tools > Port menu of the IDE (Or Run ls /dev/cu.usbmodem* in the Terminal and note the output)
    - Open Terminal/command line

Run date + T%s > /dev/cu.usbmodem# where # matches the number at the end of the output of the previous command

Windows

The process is the same as above, but instead of running date + T%s > /dev/cu.usbmodem#, run the GetTimeT.bat script with the command getTimeT > COM#, where # is the COM port the Feather is on

Verify that the time was received and applied by opening the Arduino IDE Serial Monitor – it should print out the time of the RTC clock

The complete documentation of this process can be found in the associated GitHub repository [9].

Additionally, the Loom code supports synchronizing the RTC to time obtained from the internet if using an Adafruit Feather M0 Wifi or an Ethernet FeatherWing.

### 6.3. Flashing the development board

The firmware that runs on the Feather M0 is an instance of the Loom library, with any necessary modifications or configuration being done in two files, the primary.ino Arduino sketch, and a config.h file defining the device's hardware and behavior. The process of customizing, compiling, and uploading the code to the Feather is as follows:

Install the Arduino IDE:

Follow Adafruit's instructions on how to:

- Add the Adafruit boards index
- Install Adafruit SAMD Boards
- Install drivers if on Windows 7

Connect the Feather development board to a computer with the USB micro cable

From the Arduino IDE menu Tools > Port, select the connected Feather development board

Open the config.h file

- There are many options in this file, but the relevant options here are largely related to the Serial output, SD card, and RTC. Each option in the file is explained with a comment.
- The most relevant options are:
  - LOOM_DEBUG – controls whether or not to display information to the Serial Monitor if connected via USB (set to 0/1 for false/true)
  - Dynamic_serial_output – whether or not to wait for the Serial Monitor to open to begin operation (set to 0/1 for false/true)
  - Use_utc_time – whether to use UTC or local time (set to 0/1 for false/true)
  - SD_save_filter – whether or not to enforce a minimum time between saving to the microSD (set to 0/1 for false/true)
    - SD_save_min_delay – the associated delay (time in seconds)

Open the main.ino file of the program in the Arduino IDE

- An example.ino loop may look like the following, which reads the pressure sensor and saves the data to the SD card every second:

```
void loop()
{
OSCBundle bndl;
// Read sensors, store data in sensor state struct
measure_sensors();
// Copy sensor data from state to provided bundle
package_data(&bndl);
// Print data
print_bundle(&bndl);
// Save bundled data to SD card
log_bundle(&bndl, SDCARD, ‘‘data.csv”);
// Delay between saves
delay(1000);
// Miscellaneous checks
additional_loop_checks();
}
```

Ensure that the code compiles by pressing the checkbox of the Arduino IDE

- As the provided code compiles, any errors here are most likely caused by modifications to the .ino file

Once the code compiles, it can be uploaded to the Feather with the upload (right arrow symbol) button

Further details on modifying the code and additional features of the system beyond the pressure sensor can be found in documentation of the Loom GitHub repository [6].

The code that directly interfaces with the pressure sensor is the MS5803_02 library developed by Luke Miller [17].

### 6.4. Basic pressure sensor usage

The Feather will begin operation once the code has finished uploading, unless the Serial Monitor is being used, in which case the device can optionally be set to wait until the Arduino IDE Serial Monitor is opened: Dynamic_serial_output setting set to True. The usage of the Serial Monitor is optional, permitting the display of pressure sensor readings in near real time, but requires that the device remain connected to the computer via USB during operation. If the Serial Monitor is not needed or if maintaining a USB connection during operation is impractical, a 3.7 V Lithium Polymer (LiPo) battery can be used to power the device instead, saving pressure measurements to a microSD card. It is recommended that the sensor first be tested with the Serial Monitor enabled to ensure correct operation before performing an experiment.

The device will begin setup upon the completion of code upload, or upon connecting to power (via USB or battery) if already programmed. The setup routine should take a few seconds and the Feather will rapidly flash the built-in LED to indicate the transition from setup to normal operation. Once the device is running, it will take measurements at the interval specified in the config.h file and log the data to the Serial Monitor and/or microSD card. If the Adalogger FeatherWing is being used, ensure that the coin cell battery and SD card are inserted prior to starting the device.

Once the device is confirmed to be operational and is returning readings that are reasonable, one can perform desired experiments and tests. Note that once the device is flashed with the code, it does not need to be re-flashed for subsequent operation unless the device behavior needs to be modified.

There are no envisioned safety hazards involved with the operation of this device. However, if the diameter of the sensor adapter arms is not correct for the tubing being used, it is possible that leaks or incidental disconnection of the tubing could occur, resulting in the release of the fluid being measured. The maximum safe pressure for operating this system will be a function of the size, elasticity, and clamping of the tubing. In addition, the system should be operated at less than the over-pressure threshold of the pressure sensor, 10 bar in this application, to avoid damaging the device. Depending on the proximity of the sensor to the Feather development board, release of fluid from the sensor adapter arm to tubing connection could damage the electronics.

## 7. Validation and characterization

Using a syringe pump, a viscous solution of 0.3 wt% xanthan gum, 1.5 wt% sodium chloride, and 0.025 wt% dye was injected between two radial parallel-plates, which was filled with a resident solution of 1.5 wt% saltwater [18]. A Canon EOS 50D camera was situated approximately a meter overhead to take images of the flow system once every second.

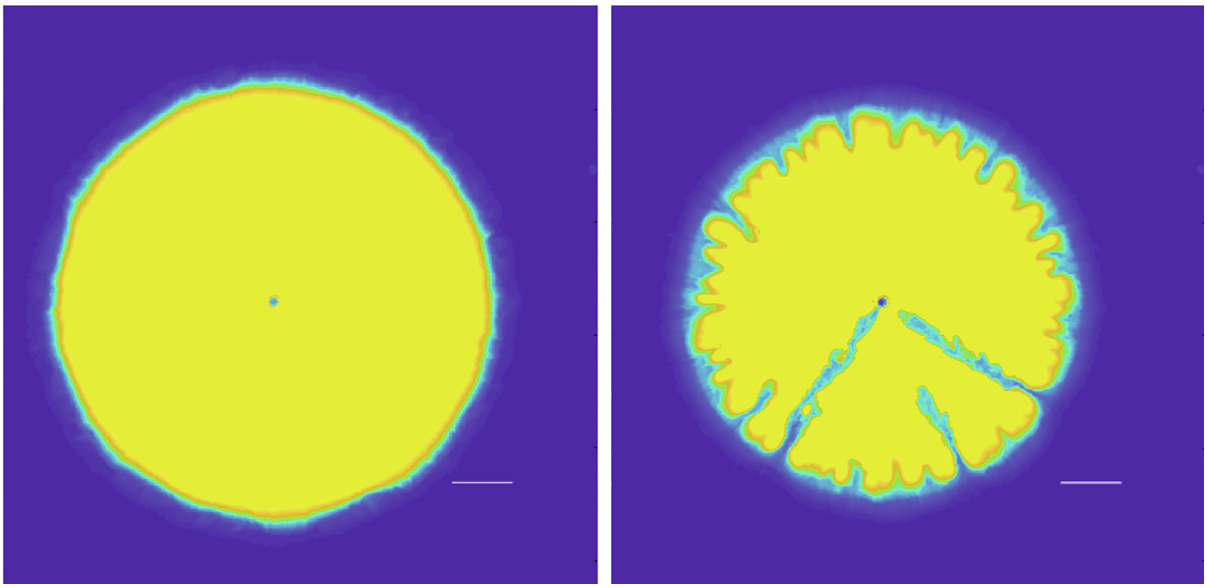

**Fig. 9.** Images taken from the experiment with an injection rate of 1 ml/min. Colors of the images have been modified for visualization. The white line in both images represents a 20 mm length scale. A) the end of the injection phase. B) image taken when the first instability reached the withdrawal site during the withdrawal phase.

Pressure data was also collected every second, which is much slower than the pressure sensor's response time of 1.1 ms. The inline pressure sensor was installed in the tubing between the syringe pump and the injection site of the parallel plates. After the viscous solution was injected into the cell, the fluid was withdrawn by reversing the direction of the syringe pump. During withdrawal of the fluid, the saltwater pushed the viscous fluid in the same direction as the syringe pump withdrew fluid toward the center of the injection site. Instabilities formed along the now unstable fluid interface, which occurs when a less viscous fluid (saltwater) displaces a fluid of higher viscosity (xanthan gum) [19].

As instabilities develop along the interface, they grow and advance toward the center, until one instability breaks through to the injection site (Fig. 9). In order to be able to associate images with pressure data, both the camera and our system were synchronized to the clock of the same computer, with images and pressure data being timestamped. The inline pressure sensor was able to detect a pressure signal for the injection phase, withdrawal phase, and the breakthrough phase for the first

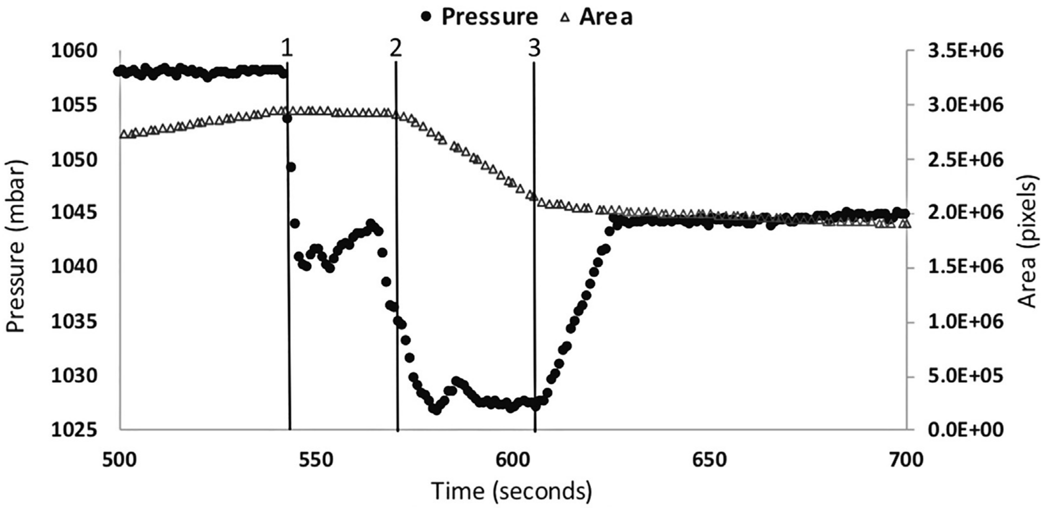


**Fig. 10.** The figure above shows the pressure data measured by the inline sensor and the regional area of the viscous fluid, which was calculated using image processing techniques in Matlab. Area of the viscous fluid region increases during the injection phase and decreases during the withdrawal phase. A pressure signal and area signal were detected as the viscous fluid dynamically moved in the flow cell. Changes in pressure and area data can be seen clearly as the syringe pump switches from the injection phase to the withdrawal phase. Line 1 marks the end of the injection phase. Line 2 marks the start of the withdrawal phase. Line 3 marks the time when the first instability reached the withdrawal site. The times of lines 2 and 3 correspond to the times of the images A and B in Fig. 9, respectively.

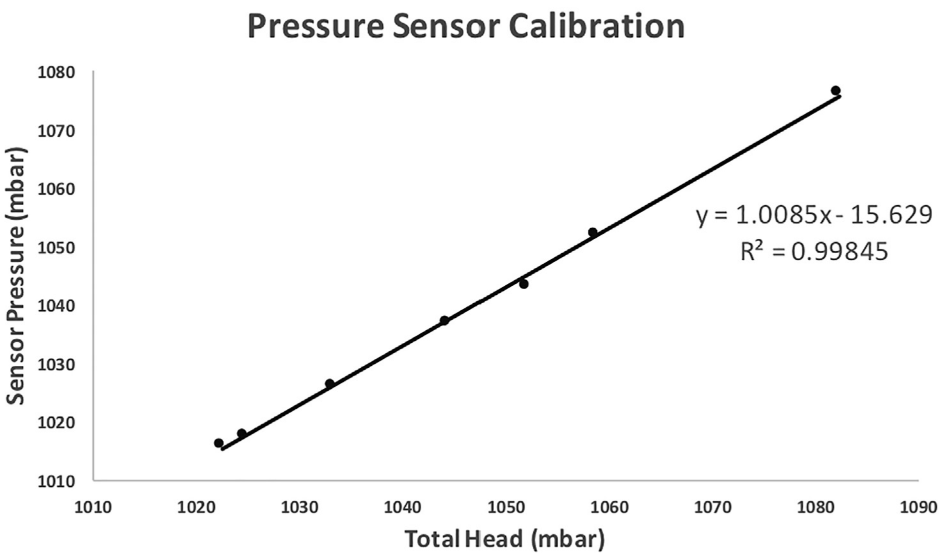


**Fig. 11.** The pressure sensor readings and the corresponding total head of the water column during the calibration.

instability. The beginning of the injection phase and the start of the withdrawal phase are not discernible in static images, yet a pressure signal was detected for both of these image times.

To check the timing of the pressure signals, the pressure data was plotted alongside area of the viscous fluid from the image (Fig. 10). This allows for interpretation of the pressure signals, as we can pair the signal with imaging data, which confirms the identification of flow behaviors including injection, withdrawal, and breakthrough (Figs. 9, 10). Area of the viscous fluid region increased during the injection phase and decreased during the withdrawal phase. Changes in pressure data correspond to similarly timed signals in the area data and can be seen clearly as the syringe pump switches from the injection phase to the withdrawal phase (Fig. 10). There is a lag time between the injection and withdrawal phases due to hysteresis of the syringe pump. This hysteresis of the syringe pump was not discernable in static images, but the pressure sensor was capable of detecting a signal change to indicate this hysteresis. Under constant and ramping pressure, the pressure sensor was able to discretize 0.05 mbar (Supplement B). Further analysis will be needed to fully interpret the pressure data, but for the scope of this paper, proof of concept has been met. Our experimental results demonstrate that the inline sensor is capable of accurately measuring pressure changes in a complex flow experiment with a viscous fluid, which can be used to detect changes in fluid properties and flow behavior.

A pressure sensor calibration was performed to test the validity of the inline pressure sensor using the static pressure term from Bernoulli's equation (Fig. 11). A column of water was suspended above the pressure sensor and the water level was periodically increased or decreased and the corresponding pressure sensor readings were recorded. Using the static pressure term from Bernoulli's equation, the pressure was theoretically calculated where,

$$p = \rho g h$$

Where p is the water pressure in $N/m^2$ and $\rho$ is the density of water in $kg/m^3$, g is the gravitational constant in $m/s^2$ and h is the height of water in the column measured in m [20]. Based on this relationship, we would expect pressure to have a direct linear relationship with water column height. The well-defined linear relationship with an $R^2$ value of 0.998 between the sensor pressure readings and the calculated pressure values verifies the validity of the pressure sensor. We would expect an $R^2$ of 1, but small deviations from this value result from experimental error in the water column readings. This calibration can be used to verify sensor pressure readings and can easily be reproduced for a larger range of pressures.

## 8. Conclusions

We have presented an open-source, inline pressure sensor system that is extensively customizable for a variety of conditions and applications. The sensor has been shown to work effectively in a complex radial parallel plate experiment using a viscous fluid. This application in itself will provide researchers with valuable information needed to detect changes in fluid properties and flow behavior. We have included all relevant design and code files, with associated documentation, necessary to customize the device. The system is complete, requiring no further hardware, and remains low-cost (~US$56 for the configuration used in our experiments). Additionally, further usage of the Loom code and ecosystem, which the presented system is an application of, permits easy integration of the device into complex wireless sensor and Internet of Things systems. Our device will be useful for researchers by enabling the integration of customized pressure sensing and data logging solutions at a comparatively low cost.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgements

Funding:

This work was supported by the USDA National Institute of Food and Agriculture, Hatch project [grant number: NI18HFPXXXXXG055], and by the National Science Foundation [grant number:1446915].

## Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ohx.2020.e00112.

## References

[1] C.W. Fetter, Applied Hydrogeology, fourth ed., Prentice Hall, Upper Saddle River, New Jersey, 2011.
[2] Sparkfun. SparkFun Pressure Sensor Breakout - MS5803-14BA, https://www.sparkfun.com/products/12909 (accessed 03.02.20)
[3] Measurement Specialties, MS5803-14BA Miniature 14 bar Module Datasheet. https://www.te.com/commerce/DocumentDelivery/DDEController?Action=showdoc&DocId=Data+Sheet%7FMS5803-14BA%7FB3%7Fpdf%7FEnglish%7FENG_DS_MS5803-14BA_B3.pdf%7FCAT-BLPS0013, 2017 (accessed 03.02.20)

[4] Darwin MicroFluidics, MPS Microfluidic High Precision Pressure Sensor- Darwin Microfluidics, https://darwin-microfluidics.com/products/microfluidic-high-precision-pressure-sensor (accessed 04.10.20)
[5] Darwin MicroFluidics, Sensor Reader – Darwin Microfluidics, https://darwin-microfluidics.com/collections/microfluidic-pressure-sensor/products/microfluidic-sensor-data-acquisition-module (accessed 04.10.20)
[6] L.Goertzen, W. Selbie, swopet, noblek, leedongj, debellt, . . . C. Udell, OPEnSLab-OSU/InternetOfAg: Loom 1.1 Release for Zenodo (Version v1.1). Zenodo. 2018, December 21. http://doi.org/10.5281/zenodo.2483255
[7] Shannon Hicks, Anthony Aufdenkampe, Sara Damiano, & Dave Arscott. (2019, February 18). EnviroDIY Mayfly Logger: v0.5b (Version v0.5b). Zenodo. http://doi.org/10.5281/zenodo.2572006
[8] A.D. Wickert, The alog: inexpensive, open-source, automated data collection in the field, Bull. Ecol. Soc. Am. 95 (2014) 166–176, https://doi.org/10.1890/0012-9623-95.2.68.
[9] L. Goertzen, N. Mehr, M. Lopez, OPEnSLab-OSU/Inline-Pressure-Sensor: V1.1 (Version v1.1), Zenodo. 2019, January 24. http://doi.org/10.5281/zenodo.2548837.
[10] Measurement Specialties, MS5803-02BA Miniature Altimeter Module Datasheet. https://www.te.com/commerce/DocumentDelivery/DDEController?Action=srchrtrv&DocNm=MS5803-02BA&DocType=Data+Sheet&DocLang=English, 2017 (accessed 1.30.19)
[11] Measurement Specialties, MS5837-03BA Ultra Small Gel Filled Pressure Sensor Datasheet. https://www.te.com/commerce/DocumentDelivery/DDEController?Action=srchrtrv&DocNm=MS5837-30BA&DocType=DS&DocLang=English, 2015 (accessed 1.30.19)
[12] Autodesk Inc., A360 – Home. https://a360.autodesk.com/, 2018 (access 01.30.19)
[13] E.G. Gordeev, A.S. Galushko, V.P. Ananikov, Improvement of quality of 3D printed objects by elimination of microscopic structural defects in fused deposition modeling, PLoS One 13 (6) (2018), https://doi.org/10.1371/journal.pone.0198370 e0198370.
[14] formlabs, Choosing the right material. https://support.formlabs.com/s/article/Choosing-the-Right-Material?language=en_US (accessed 4.10.20)
[15] Arduino – OldSoftwareReleases. https://www.arduino.cc/en/Main/OldSoftwareReleases#previous, (access 04.16.20).
[16] Autodesk Inc., Eagle | PCB Design Software | Autodesk. https://www.autodesk.com/products/eagle/overview. (access 04.16.20)
[17] L. Miller, MS5803_02. https://github.com/millerlp/MS5803_02, 2014 (accessed 01.30.19)
[18] N. Mehr, C. Roques, Y. Méheust, S. Rochetfort, J.S. Selker, Mixing and finger morphologies in miscible non-Newtonian solution displacement, Exp Fluids 61 (2020) 96, https://doi.org/10.1007/s00348-020-2932-x.
[19] P.G. Saffman, G.I. Taylor, The penetration of a fluid into a porous medium or Hele-Shaw cell containing a more viscous liquid, Proc. R. London Ser. A 245 (1958) 312–329.
[20] B.R. Munson, D.F. Young, T.H. Okiishi, Fundamentals of Fluid Mechanics, 5th ed., John Wiley & Sons Inc., 2006.