Title: VentMon: An Open Source Inline Ventilator Tester and Monitor

Authors: Robert L. Read, Lauria Clarke, Geoff Mulligan

Affiliations: Public Invention with support from the *Protocol Labs* and the *Mozilla Foundation*

Contact email: read.robert@gmail.com

Abstract:

Humanitarian engineers responded to the pandemic ventilator shortage of March, 2020 by beginning over 100 open source ventilator projects[10, 5]. By *ventilator*, we mean both an invasive ventilator (requiring intubation of the patient) and non-invasive ventilator (generally supporting spontaneously breathing). Inexpensive ventilator test equipment can facilitate projects forced to be geographically distributed by lockdowns. The VentMon is a modular, open source, IoT-enabled tester that plugs into a standard 22mm airway between a ventilator and a physical test lung to test any ventilator. The VentMon measures flow, pressure, fractional oxygen, humidity, and temperature. Data is stored and graphed at a data lake accessible to all devlopment team members, and, eventually, clinicians. The open source design of the VentMon, its firmware, and cloud-based software may allow it to be used as a component of modular ventilators to provide a clinical readout. The software system surrounding VentMon has been designed to be as modular and composable as possible. By combining new, openly published standards for data with composable and modifiable hardware, the VentMon forms the beginning of an open system or eco-system of ventilation devices and data. Thanks to grants, 20 VentMons have been given away free of charge to pandemic response teams building open source ventilators.

Keywords: COVID-19, open source medical device, ventilator, pandemic response, IoT, respiration waveform

Specifications table:

Hardware name	VentMon
Subject area	Educational Tools and Open Source Alternatives to Existing Infrastructure
Hardware type	Field measurements and sensors, Electrical engineering and computer science
Open source license	MIT License, CERN-OHL-S
Cost of hardware	$280
Source file repository	VentMon: https://doi.org/10.5281/zenodo.4079170 PIRDS (Public Invention Respiration Data Standard): http://doi.org/10.5281/zenodo.4079377 PIRDS logger: Zenodo. http://doi.org/10.5281/zenodo.4079382 SFM3X00: http://doi.org/10.5281/zenodo.4079355 VentDisplay: http://doi.org/10.5281/zenodo.4079374

1. Hardware in context

The Coronavirus pandemic has created a potential global shortage of ventilators[5], which has abated in the wealthy nations. Ventilators are expensive to manufacture and during uncertain times supply chains can be disrupted, increasing the cost and scarcity of these devices. Since the beginning of the pandemic in March, there has been a large movement to develop cheaper, more supply chain-resilient ventilators. The urgency and goals of this movement have shifted as the global understanding of the virus has evolved. Regardless of the current state of this movement, however, the use of open source medical devices in resource-limited emergency situations—common during a global pandemic—is a topic that has come to the forefront of may conversations about preparedness and treatment. Medical devices require rigorous testing before they can be used on the general public. Open source medical devices require the same level, if not more, scrutiny and documentation. If an open source movement is to succeed in this field, the tools to test and validate medical devices must be accessible. The goal of VentMon is to provide an equally open source, community based verification solution to increase the efficiency and accuracy of the development process for socially distanced, potentially international teams making open source ventilator, BPAP, and PAPR devices.

2. Hardware description

The VentMon plugs directly into a standard 22mm adult airway. As air or medical gases pass through the VentMon, it records flow and pressure, temperature, humidity, and fractional O_2 at rates of 25 Hz to once per minute depending on the measurement. When placed in the airway

between a ventilator under test and physical test lung, this data is sent to a public data lake [8] using a clearly defined respiration standard [7]. This data can be graphically rendered either live or statically to provide a display that is typical of the clinical display of advanced ventilators. This functionality is similar to that of commercially available test lung devices, which tend to start at USD$10,000.

Teams building open pandemic ventilators tend to be poorly funded and geographically distributed. VentMon's connectivity and overall cost address these two challenges by enabling team members on different continents to view waveforms in real time and communicate the necessary adjustments to the ventilator device under test. While reliance on the internet is a strength for a distributed and virtually connected engineering team, it would be an addressable disadvantage in a field hospital.

Being completely open, VentMon supports modification, extension, and has potential for integration into a complete ventilator. A team working to build a ventilator device with a graphical trace of pressure and flow—generally demanded by clinicians for all but emergency transport situations—could incorporate the open source VentMon into its complete design. VentMon has potential to be safe enough and reliable to be used as a monitor on human patients, however it requires much improvement before it is ready for that use.

The VentMon thus offers the following benefits to teams developing ventilation devices. It:

- is inexpensive,
- is completely open source and part of a modular hardware/software system that can be customized and extended,
- allows remote collaboration, and
- currently provides many of the clinically significant metrics found in commercially available ventilators.

3. Design files

3.1 Design Files Summary

Design filename	**File type**	**Open source license [check these]**	**Location of the file**
(1) Embedded Firmware (v0.3 PCB version)	C++ Source Code	MIT License	10.5281/zenodo.4079170
(2) Embedded Firmware (v0.2 COTS version)	C++ Source Code	MIT License	10.5281/zenodo.4291147
(3) VentDisplay PIRDS Data Viewer	HTML Source Code	MIT License	10.5281/zenodo.4079374
(4) PIRDS Data Logger	C Source Files	MIT License	10.5281/zenodo.4079382
(5) PIRDS Data Standard	Documented standard, and C-language binding	Creative Commons Zero (Universal) for documents, GPL 3.0 for code binding	10.5281/zenodo.4079355
(6) VentMon v0.3T PCB	PCB Design Files	MIT License	10.5281/zenodo.4079170 (see *design/pcb*)
(7) Pressure Sensor and Airway Adaptor	STL File	MIT License	10.5281/zenodo.4079170 (see *design/3dparts*)

We promote an open, modular, and composable respiration ecosystem. Therefore, we have more separate repositories and modules than would strictly be needed if we were only making the VentMon project. Our hope is that smaller modules may be reusable by others who do not need the VentMon specifically, but may have need for a particular component.

1. The main firmware is written in the Arduino environment. The firmware targets an ESP32, but could easily apply to other boards. This core firmware manages reading the various sensors, and outputs its data both on its small built-in OLED (used for functional testing), and via the PIRDS standard to a public data lake where a separate module renders it in a graphical display appropriate for a clinician.

2. Our current focus is on a printed circuit board version of the VentMon[1]. However, we also report here a system that can be made by an amateur using only commercial off-the-shelf (COTS) parts—version T0.2. Software that matches this version it is tagged as such.

3. VentMon has a tiny OLED screen which is used to graph pressure for quick "smoke" testing.

For serious performance testing and eventual clinical use, we use VentDisplay, a browser-delivered JavaScript code that presents a full clinical display.

4. The VentDisplay software can be pointed at any data logger that serves up data in the PIRDS format. This logger takes data from UDP packets and writes them to a file, which can then be served up by a webserver to any requester.

5. Integral to the VentDisplay and PIRDS Data Logger is the Public Invention Respiration Data Standard (PIRDS). This is a documented standard.

6. The VentMon v0.3T printed circuit board is designed with EAGLE CAD.

7. A small 3D printed housing epoxied over the ambient air quality sensor with a tiny pressure port allows us to share the air with the main pressure hose.

4. Bill of materials

The Bill of Materials specifies the parts needed to build the VentMon in a simple enclosure with drilled-out ports for cables. The parts are typical of hobby electronics.

Designator	Component	Number	Cost/Unit	Total Cost	Source	Material Type
U1	PARTICLE ETHERNET FEATHERWING	1	$20.11	$20.11	Adafruit	Semiconductor/Other
U2	HUZZAH32 ESP32 FEATHER MALE HDR	1	$21.95	$21.95	Adafruit	Semiconductor/Other
U3	SPARKFUN QWIIC SHIELD FOR THING	1	$3.50	$3.50	Digikey	Semiconductor/Other
U4	ADS1115 16 Bit ADC	1	$14.95	$14.95	Adafruit	Semiconductor/Other
U5	OLED Feather Wing	1	$14.95	$14.95	Adafruit	Semiconductor/Other
C1, C2	QWIIC CABLE - 200MM	2	$1.56	$3.12	Digikey	Metal/Plastic
C3	USB Cable	1	$1.80	$1.80	Digikey	Metal/Plastic
C4	Oxygen Sensor Cable	1	$4.00	$4.00	Oxycheq	Metal/Plastic
H1	M2.5 Screws	12	$0.38	$4.56	Digikey	Metal
H2	M2.5 Standoffs - Female, Female	6	$0.45	$2.70	Digikey	Metal
OS1	R-17S Oxygen Sensor	1	$80.00	$80.00	Oxycheq	Metal/Plastic
FS1	**Flow Sensor**	1	**see note below**	$0.00		Metal/Plastic
PS1, PS2	BME680 SENSOR EVAL BOARD	2	$22.50	$45.00	Adafruit	Semiconductor/Other
P1	Processor Enclosure	1	$9.95	$9.95	Digikey	Plastic
P2	Bleed Adaptor	1	$5.50	$5.50	Oxygen Supply	Plastic
P3	Oxygen Sensor T	1	$9.00	$9.00	Oxycheq	Plastic
P4	15 mm OD to 22 mm OD adaptor	1	$0.54	$0.54		Plastic
P5	15 mm ID to 22 mm OD adaptor	1	$0.35	$0.35		Plastic
T1	Oxygen Tubing	1	$2.25	$2.25	Oxygen Supply	Plastic

Note: The flow sensor is one of the most expensive and critical parts. Throughout mid-2020 there was a global shortage of flow sensors, including the four electrically compatible with the Vent-

Mon as currently designed. The Sensirion SFM3400-AW ($180US) (a neonatal version with limited maximum flow sensing), SFM3200-33-D (disposable version $40USD), Sensirion SFM3200-250 ($145USD), and Sensirion SFM3200-AW ($180USD autoclavable version). We shipped VentMons with all of these depending on availability. The VentMon is not currently designed for use with humans, but if it were, the autoclavable version would have the advantage of sanitizability if a complete sanitization procedure were designed. This is important, as we hope the VentMon will be modified and integrated into a rapidly manufactured ventilator, for which sanitization is critical.

The link below is to a spreadsheet which is includes the PCB-version, which is very similar except for the PCB itself. By following this link: VentMon BOM, you can reach browsable URLs to most of the non-commodity parts. The specific BOM for the SMD parts needed to assemble the PCB can be found in our GitHub repo in the directory named *design/pcb* along with the Eagle files: [1].

5. Build instructions

Two versions of VentMon can be assembled depending on availability of parts and time. The most physically robust and complete version of VentMon requires the purchase and manufacture of a PCB as well as a number of 3D printed plastic parts. This version of the device requires the least amount of time to assemble and contains fewest discrete components. The designs of the PCB and 3D printed parts are fully open. Due to the relatively high single-unit cost of PCB manufacture and assembly it makes most sense to use the PCB if you are building 10 or more VentMons. VentMon can also be created using off the shelf components readily available from DIY electronics suppliers. This version requires significantly more assembly time. We believe this has been done twice by third parties with little to no assistance from the VentMon team. Both assembly procedures are outlined below.

5.1 COTS Based VentMon (v0.2T)

The VentMon v0.2T is not meant to be used on human patients. The COTS version uses firmware v0.2 tagged as such[9]. This version is somewhat incompatible with v0.3 (described below), which uses a PCB, and which is the focus of our main development.

1. Qwiic Shield Assembly

   See Figure 1 on page 8 for final assembly of Qwiic Shield.

   1.1 Trim two pieces of male header pins that are two positions long and one piece that is four positions long.

   1.2 With the connector side of the board facing up, solder one of the two position headers into the two holes labeled SDA and SCL.

   1.3 Solder the other two position header into the last two holes on the row along the opposite side.

   1.4 Solder the four position header into the last four holes along the right side. This should provide connections to the GND and 3V3 through holes.

   1.5 Connect both sides of the solder bridge to engage the pull-up resistors for the I^2C bus.

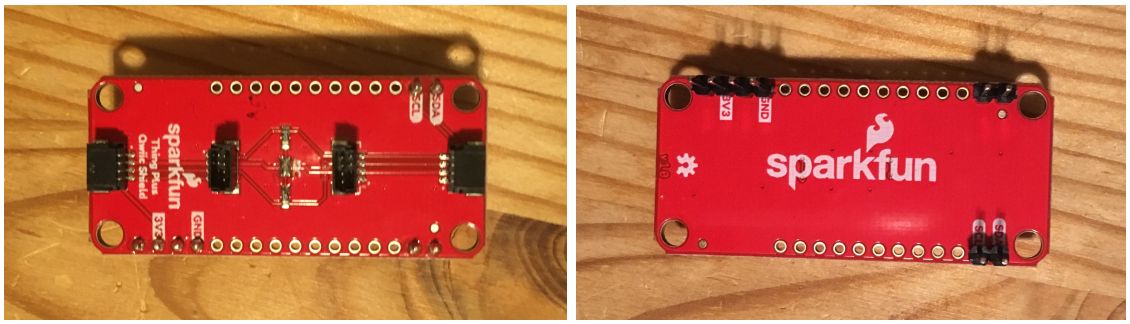

Figure 1: SparkFun Qwiic Shield with male headers and I^2C pull-ups connected

2. Enclosure

   Before starting this step download and print the drill template.

   2.1 Cut out the template and tape it to the enclosure.

   2.2 Make sure the lid of the enclosure is firmly screwed on before drilling

   2.3 Drill holes in the locations denoted on template using the recommended drill sizes for each hole.

3. Pressure Sensor Assembly

   3.1 Drill out mounting holes refer to figure 3.2 on page 8

      3.1.1 Using the 7/64" drill bit, carefully enlarge the mounting holes in each corner of the sensor

      3.1.2 You can do all four or just two - one on the left side of one board and one on the right side of the other

   3.2 Cut trace from SDO through hole to sensor. Refer to figure 3.2 on page 8.

      3.2.1 Carefully cut the trace coming from the SDO hole on the back side of one board

      3.2.2 This board will be used to measure the airway pressure

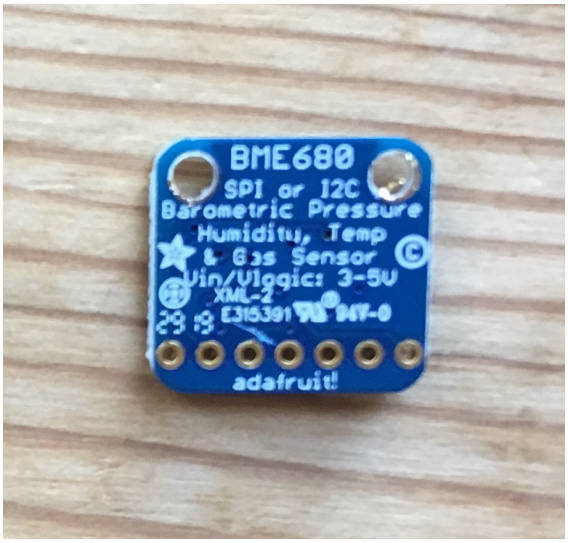

Figure 2: BME 280 Eval board with drilled out holes and trace cut

3.3 Assemble Cable. Refer to figure 3 on page 9.

3.3.1 Cut one connector off the Qwiic cable

3.3.2 Strip and tin the last 1/4" of the leads

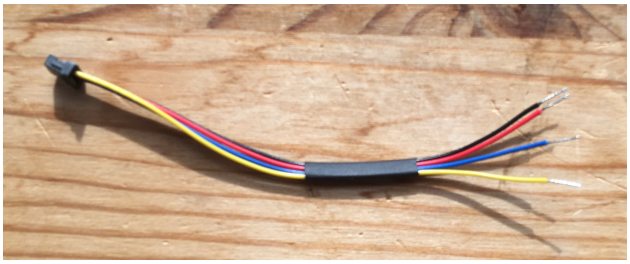

Figure 3: Prepared Qwiic cable for BME sensor assembly

3.4 Connect Sensors to Cable. Refer to figure 3.4 on page 10 and table 1 on page 9 for electrical connection.

3.4.1 Stack the two sensors on the breadboard and use two paper clips to anchor them

3.4.2 Be sure the airway board with the cut trace is facing down

3.4.3 Starting with Vin or GND insert the tinned lead through the appropriate hole and apply solder

3.4.4 Flip the sensor assembly over and make sure solder has flowed through both through holes, adding some if needed

3.4.5 Don't worry if the solder does not flow perfectly

3.4.6 Pin the assembly back down and move to another connection following the connection table below

3.4.7 Repeat the steps above until the two boards have been secured and all leads are connected

3.4.8 Re-flow all the joints to ensure a good connection

Signal	Qwiic Color
GND	black
VIN	red
SDA	blue
SCL	yellow

Table 1: pressure sensor wiring

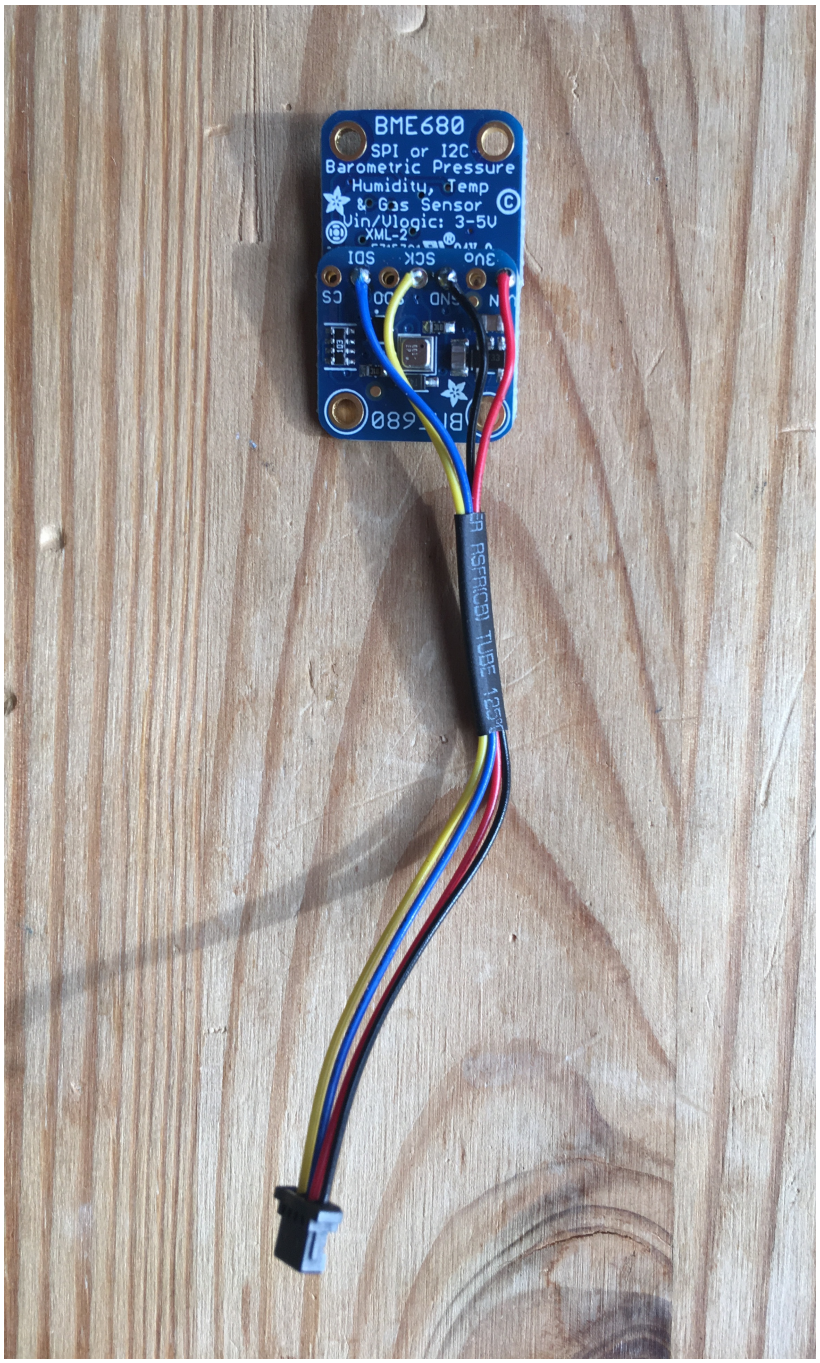

Figure 4: Qwiic cable soldered to BME sensor assembly

3.5 Add address jumper to ambient pressure sensor. Refer to figure 5 on page 11

3.5.1 Take a 1/2" piece of hookup wire and strip both ends

3.5.2 Flow the solder in the GND through hole and poke one end of the hook up wire into the hole

3.5.3 Put the other end of the wire into the SDO through hole and apply solder

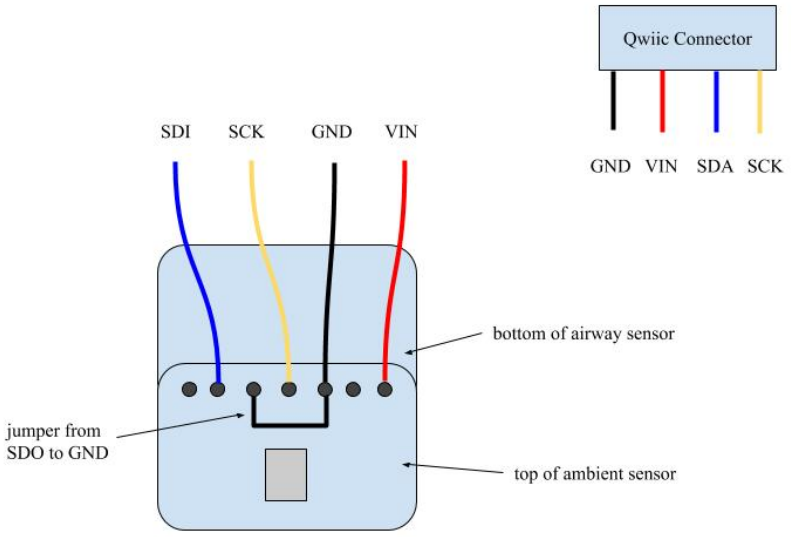

Figure 5: Qwiic cable soldered to BME sensor assembly

4. Flow Sensor Assembly

4.1 Assemble Cable. Refer to figure 6 on page 12

The flow sensor requires a 5 V input, however we will run the I^2C bus at 3.3 V necessitating this cable.

4.1.1 Cut the Qwiic cable in half

4.1.2 Cut off the red conductor near the base of the connector

4.1.3 Cut a 30 cm length from the USB cable

4.1.4 Strip the jacket back 1 1/2" from one end

4.1.5 Strip the first 1/4" of each conductor on one end and tin

4.1.6 Twist a small portion of the shield into the black GND conductor

4.1.7 Place a small piece of heat shrink on each conductor of the Qwicc cable and push them toward the connector end

4.1.8 Solder the ends of the Qwiic cable to the ends of the USB cable following Table 2.

USB	Qwiic	Signal
black	black	GND
green	blue	SDA
white	yellow	SCK
red	single 6" wire	5V

Table 2: flow sensor wiring

4.1.9 Connect the red wire to the red conductor in the USB cable and solder the extra long male header pin to the end

4.1.10 Cover the finished joints with a short 1/4” piece of heat shrink

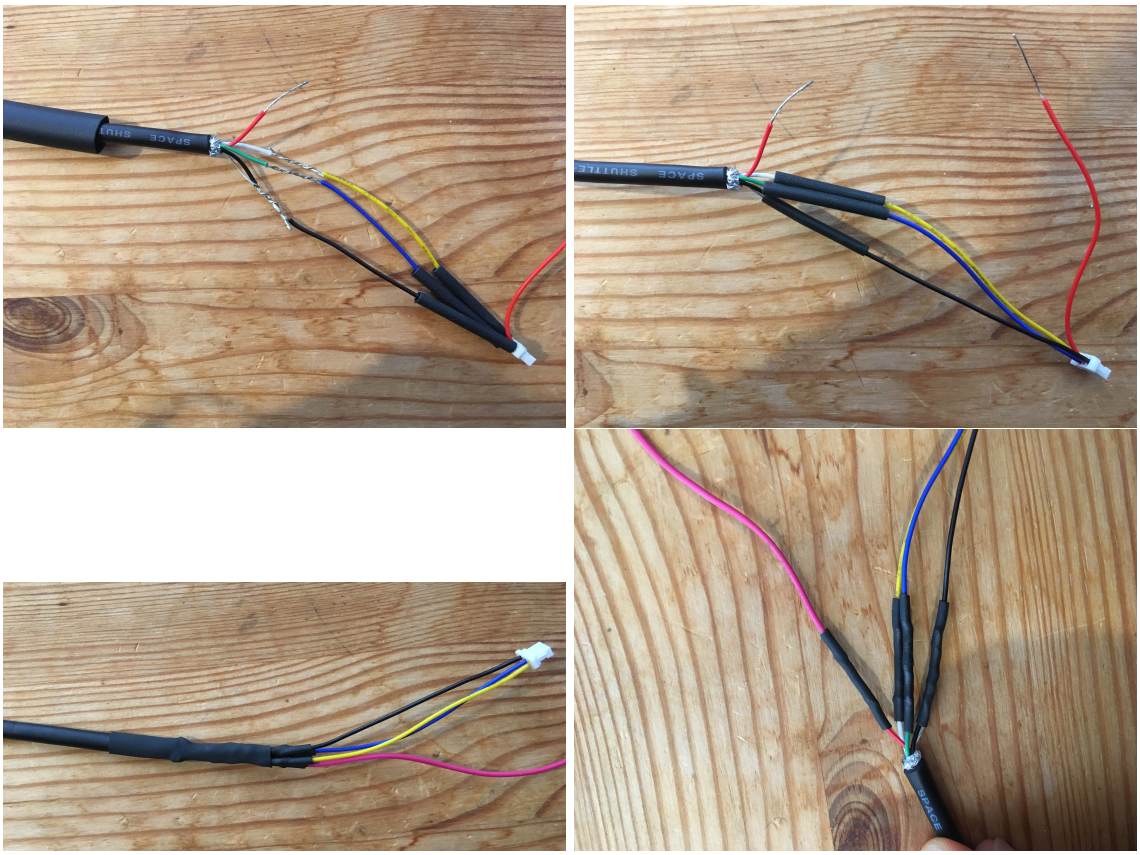

Figure 6: Clockwise progression of sensor cable assembly.

4.2 Connect cable to sensor. Refer to figure 4.2 on page 13.

4.2.1 Apply a small quantity of solder to the four center pads on the SFM3400 flow sensor

4.2.2 Remove 5/8” of the jacket on the bare end of the USB cable

4.2.3 Strip back 1/8” from each conductor and tin the ends

4.2.4 Thread the cable through the hole in the sensor as shown in the photo and make sure the tinned leads can reach the sensor pads

4.2.5 Connect the end of each lead to the pads you have prepared by touching the tip of the soldering iron to the pad and then placing the tinned lead onto it.

4.2.6 Refer to the SFM3400 datasheet to be sure you are connecting them in the proper position.

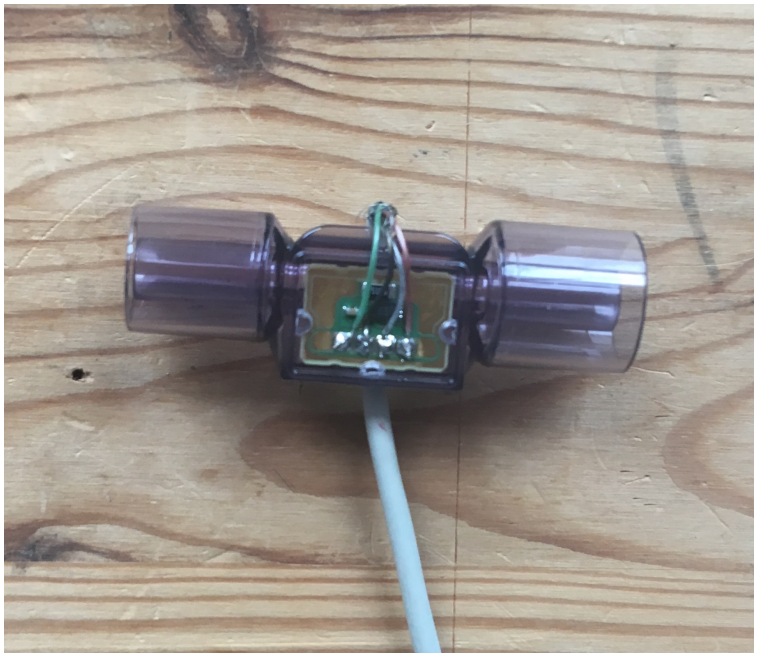

Figure 7: Sensor connected to cable assembly.

4.3 Assemble sensor adaptors. Refer to figure 8 on page 14. *NOTE: These photos and the 15mm adaptor show and refer to the neonatal version of the Sensirion flow sensor (SFM3400), which we used only because the adult-sized versions were impossible to find (temporarily). However, they physical assembly is exactly the same for all versions. You will not need the 15mm to 22mm adaptor with an adult-sized flow sensor.*

4.3.1 Cut a 1 1/2" section of petex tubing

4.3.2 Insert the tubing into the wide end of the flow sensor

4.3.3 Insert the frosted 15 mm to 22 mm adaptor over the petex

4.3.4 Add the clear 15 mm to 22 mm adaptor over the small end of the flow sensor

4.3.5 Place the bleed adaptor over the clear adaptor

4.3.6 Use electrical tape to secure any loose connections

Figure 8: Adaptors assembled inline with sensor.

5. Final Assembly

5.1 Glue oxygen hose to pressure sensor assembly. Refer to figure 9 on page 15.

5.1.1 Cut a section of oxygen hose approximately the same length as the flow sensor cable

5.1.2 Make sure your cut is flat on the end

5.1.3 Place one end of the hose over the airway pressure sensor and apply glue around the edges

5.1.4 Be very careful not to get glue on the sensor and make sure that the hose is securely attached to the pcb

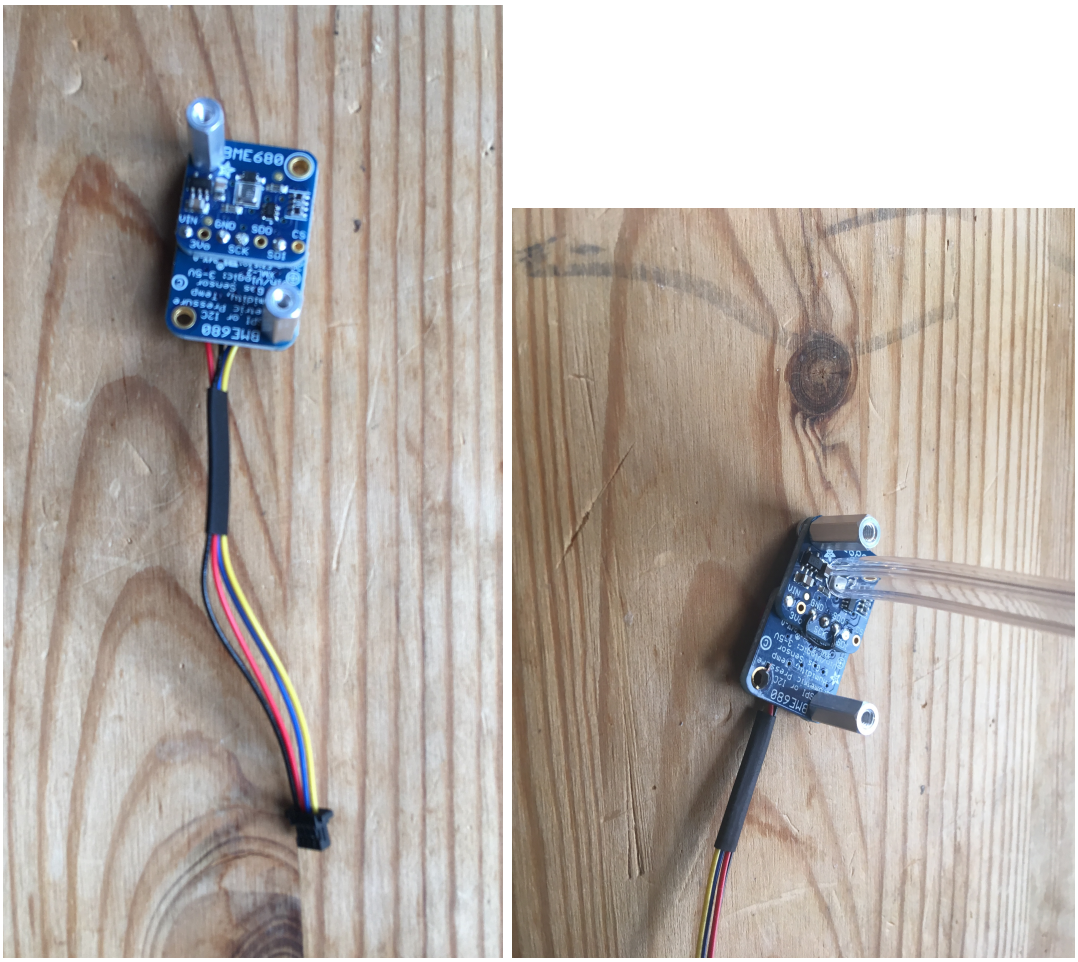

Figure 9: Standoffs installed on pressure sensor assembly and oxygen hose prepared for gluing.

5.2 Install pressure sensor assembly. Refer to figure 10 on page 16.

5.2.1 Add two standoffs to opposite corners of the pressure sensor assembly

5.2.2 These should correspond to the mounting holes you drilled in the enclosure

5.2.3 Thread the bare end of the oxygen hose through its hole in the enclose from the inside until the sensor assembly is pressed against the wall of the enclosure

5.2.4 Attach the standoffs into the enclosure from the outside using two machine screws

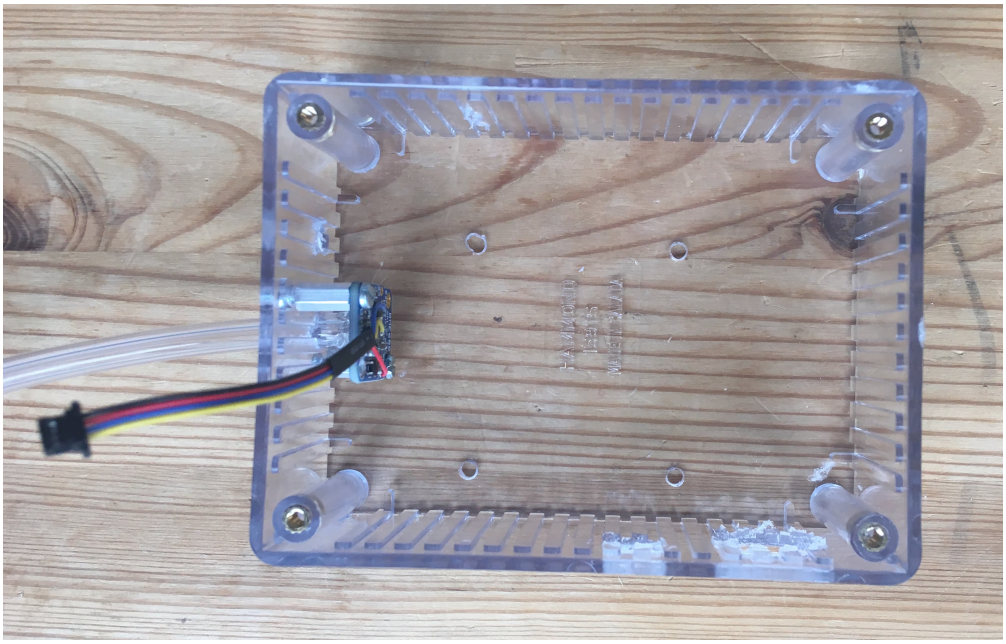

Figure 10: Pressure sensor assembly installed in enclosure.

5.3 Zip tie sensor cable to enclosure. Refer to figure 11 on page 16.

5.3.1 Thread the small zip tie through the two mounting holes in the enclosure

5.3.2 Push the qwiic connector end of the flow sensor cable through its hole in the enclosure. It's a tight fit.

5.3.3 Zip tie the cable to the side of the enclosure making sure that the jacket of the USB cable is engaged by the zip tie

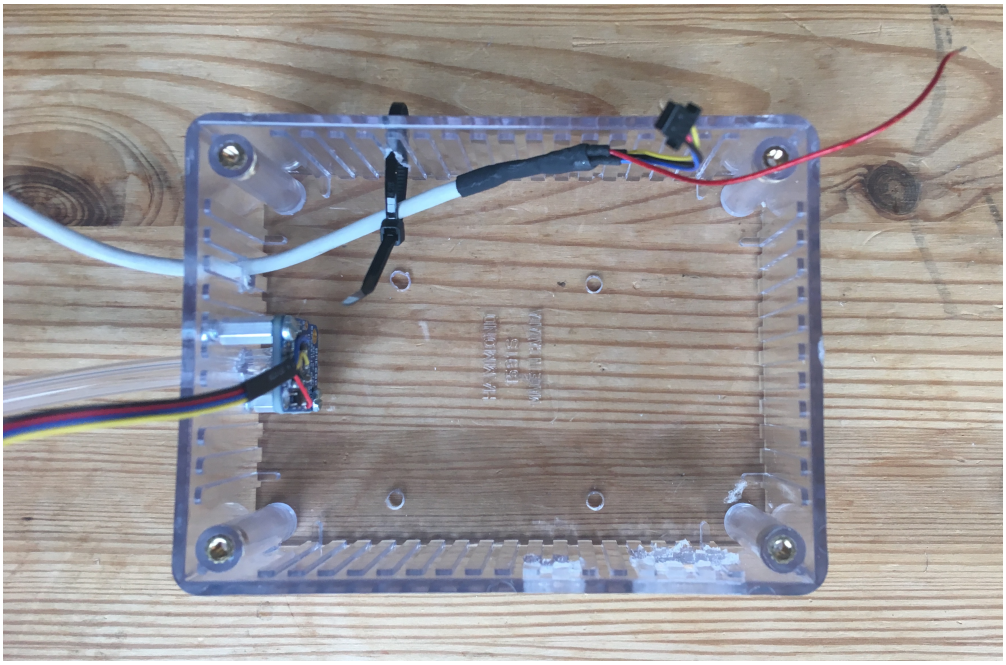

Figure 11: Flow sensor cable with strain relief.

5.4 Install ethernet shield and processor and add final touches. Refer to figure 11 on page 16.

5.4.1 Attach standoffs to the four corners of the ethernet shield

5.4.2 Screw the standoffs into the enclosure from the bottom

5.4.3 Place the ESP32, OLED wing, and Qwiic shield on top of the ethernet shield as depicted in the diagram

5.4.4 Plug the flow sensor and pressure sensor assembly into the qwiic shield

5.4.5 Add a small piece of electrical tape every 5 inches attaching the oxygen tubing and flow sensor cable

5.4.6 Wrap a piece of electrical tape around the flow sensor covering the exposed pads of the flow sensor

5.4.7 Add identification card on the next page card to enclosure.

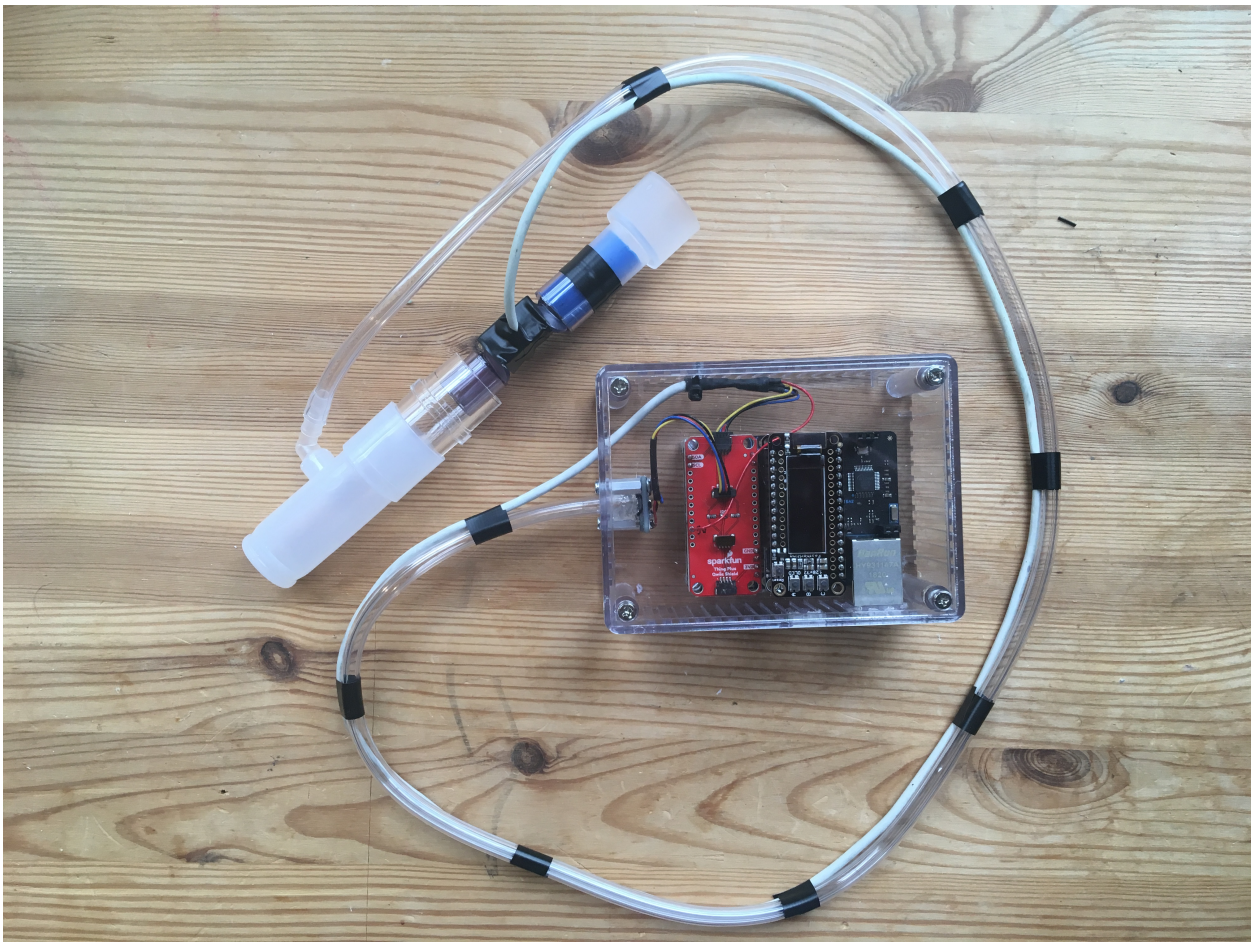

Figure 12: Final assembly.

5.2 PCB Based VentMon (v0.3T)

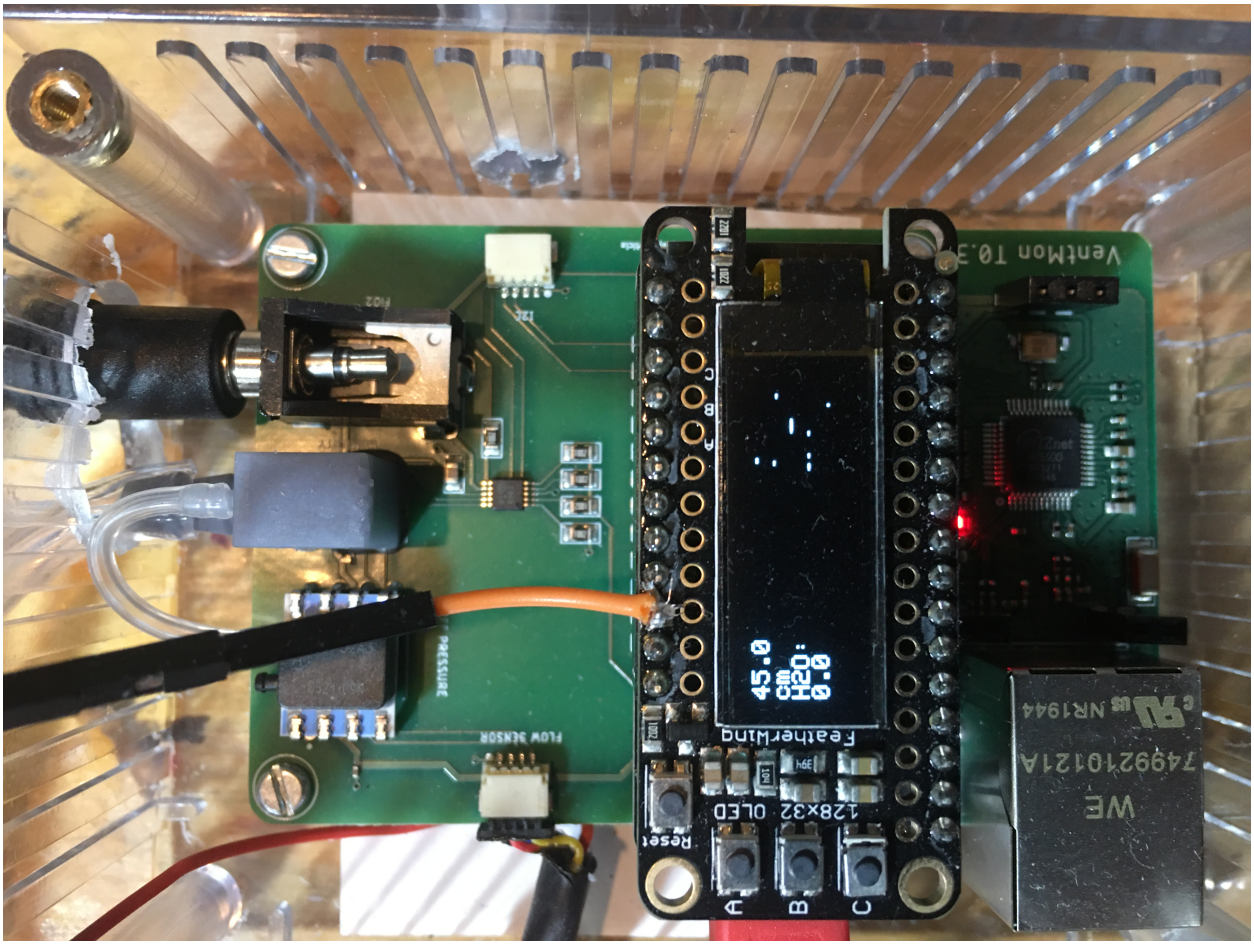

Figure 13: Final assembly of the PCB version

This PCB-based v0.3T version of VentMon (see Figure 13) requires two 3D printed parts—one encapsulated an on-board pressure sensor and one is an airway adaptor—as well as a PCB assembly. Before beginning the assembly process make sure that you have manufactured those parts.

1. PCB Assembly

    1.1 Add standoffs to PCB

        1.1.1 take hardware and put it in the holes

    1.2 Mount sensor enclosure for BME280 pressure sensor

        1.2.1 Insert plastic part into mounting holes on PCB to check fit and alignment. The barb should face toward the outer edge of the PCB.

        1.2.2 Apply epoxy glue to bottom edge and mounting pegs of plastic.

        1.2.3 Insert plastic back into holes being careful not to get any glue on the sensor.

        1.2.4 Allow 24 hours for epoxy glue to cure before attaching a hose to the barb.

2. Enclosure

Follow assembly instructions in section 2.

3. Flow Sensor Assembly

   Follow assembly instructions in section 4.

4. Oxygen Sensor Assembly

   4.1 Screw the airflow flange on to the oxygen sensor
   4.2 Plug sensor into the blue airway adaptor piece
   4.3 Plug one end of mono audio cable into bottom of oxygen sensor
   4.4 Plug the other end into the audio connector on the PCB

5. Final Assembly

   4.1 Connect Qwiic connector.
   4.2 Screw lid on.
   4.3 Optionally apply hot-melt glue to Qwiic cable connector where it enters the port.

6. Operation instructions

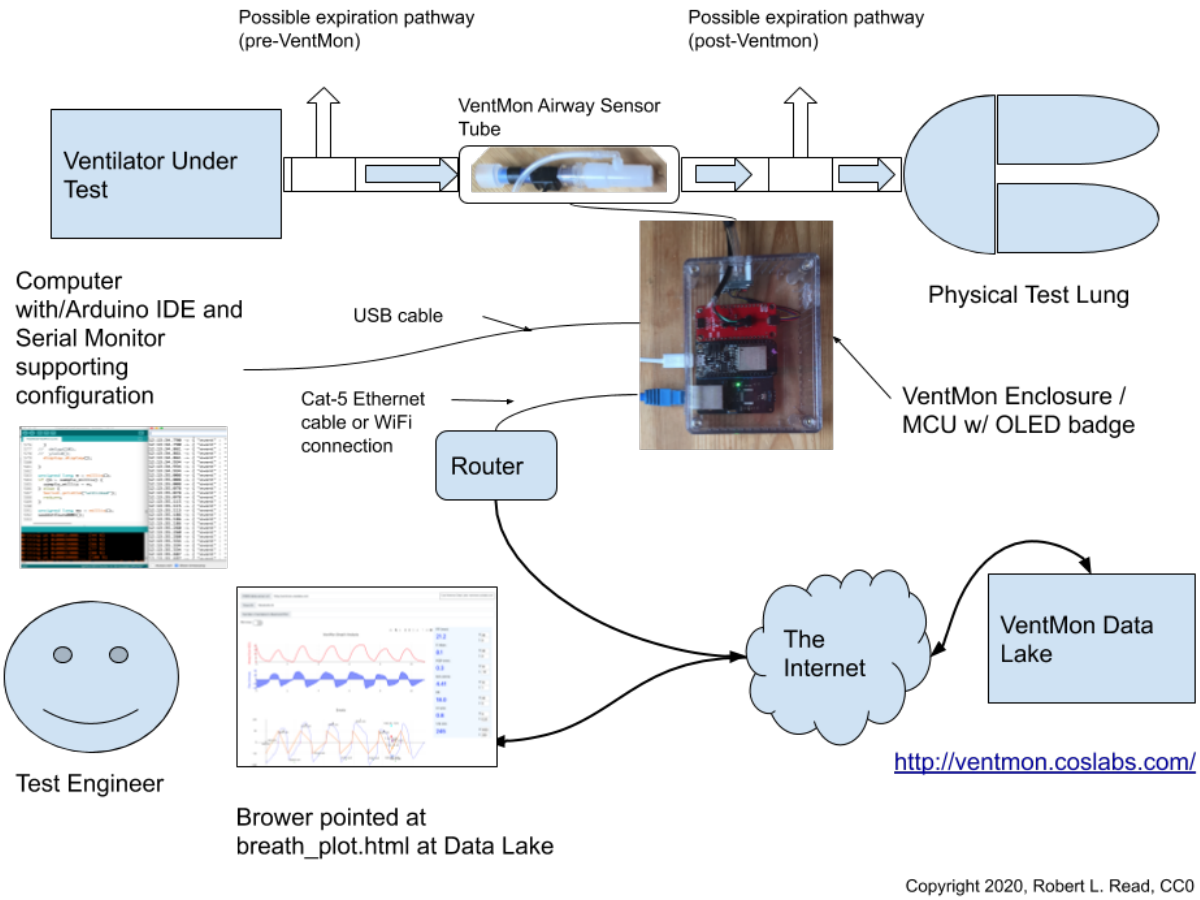

Figure 14: Ventilator Test Usage

6.1 VentMon v0.3T

The VentMon v0.3T is not meant to be used on human patients. The purpose of the VentMon is to measure, log, validate and verify the performance of rapidly manufactured ventilators, PAPRs, BPAP, and CPAP machines. The basic usage is to plug the VentMon 22mm airway in between a respiration assistance device being tested (such as a pandemic ventilator or CPAP machine) and a simple physical test lung which models compliance and resistance of the human lung (see Figure 14). The VentMon then provides approximately the functionality of a *breathing simulator*, without being able to model the patient triggering a breath spontaneously.

There are four ways to get data from the VentMon of varying quality and convenience with the firmware at the time of this writing.

1. After startup, the VentMon will begin displaying a real-time pressure graph on its small OLED screen, with a scale of 40 cmH_2O. This display is too small for testing performance, but is a nice check that the breathing circuit is in place and functioning. You can "smoke test" this by breathing into the VentMon and plugging one end with your hand.

2. The VentMon spews a LOT of data out on the serial port in the PIRDS format. Using the serial monitor built into the Arduino IDE, you can visually inspect this data. The volume of the data makes this a cumbersome. You are free to write a serial-port reader of your own for programmatic interpretation.

3. At startup, by default configuration (it can be enabled or disabled), the VentMon tries to see if their is a cable connected to the Ethernet RJ45 jack. Plug the other end of the cable into your router. If found and enabled, the VentMon will begin sending PIRDS[7] data to the logger at ventmon.coslabs.com[6]. Your data can be viewed in real time by pointing a browser at the URL. Your IP address will appear near the top of the list with other recent or current users; by clicking on "Breath Plot" you will be able to see a complete clinical display.

4. Similarly at startup, the VentMon will attempt to establish a WiFi connection. Using the WiFi connection will transmit exactly the same PIRDS data the logger at ventmon.coslabs.com and can be used in precisely the same way. However, the VentMon has to know a valid SSID and password to make the WiFi connection. This is configured via the Arduino serial monitor by sending a "c" character (for configure) to the VentMon. It will then pause operation for 10 seconds and give instructions for entering the SSID and password. In this same configuration you can enable or disable the WiFi and the Ethernet connection if you choose. These settings are stored in the EEPROM, and survive a loss of power.

Pressing the "C" button on the VentMon OLED badge sends a special signal to the PIRDS logger to "rotate the log" from this IP address. At the public data lake the current file will be copied to a file which a date and timestamp.

6.2 VentMon COTS Version

A practitioner who makes their own version of the VentMon from commercial off-the-shelf parts without ordering a PCB will likely understand how to install and modify the firmware. The VentMon v0.2T[9] version is tagged specifically for this hardware release. The usage will likely be the same, though at the time of this writing some features may not be available in the software they are using, such as WiFi configuration.

7. Validation and characterization

The primary use of the VentMon is to develop to and verify that rapidly manufactured ventilator meets requires specification, such as those published by the UK MHRA[3].

It is easy to have confidence in the basic dynamic function of the VentMon by simply breathing through it into a plastic test lung. Figure 15 is a screenshot of VentDisplay rendering data from the VentMon with a human (RLR) breathing normally into a plastic test lung for 10 seconds. Although the VentDisplay and VentMon can be used independently, VentDisplay demonstrates what VentMon collects. The top trace is a pressure trace. The next shows flow. These curves are generated from samples published into the public data lake. VentMon produces pressure and flow graphs at about 25 samples per seconds which is fast compared to the physiological process of breathing. One can easily see that flow and pressure dynamically match the breath or changes in the mechanical ventilation, or even changes induces by changes in the resistance of compliance of the test lung.

The bottom trace on the display is the "event" curve which shows a number of computed values and events published from the VentMon, such as the humidity, FiO_2, and temperature. Additionally, the beginning and ending of each breath is detected, and total volume of the breath computed by integrating the flow.

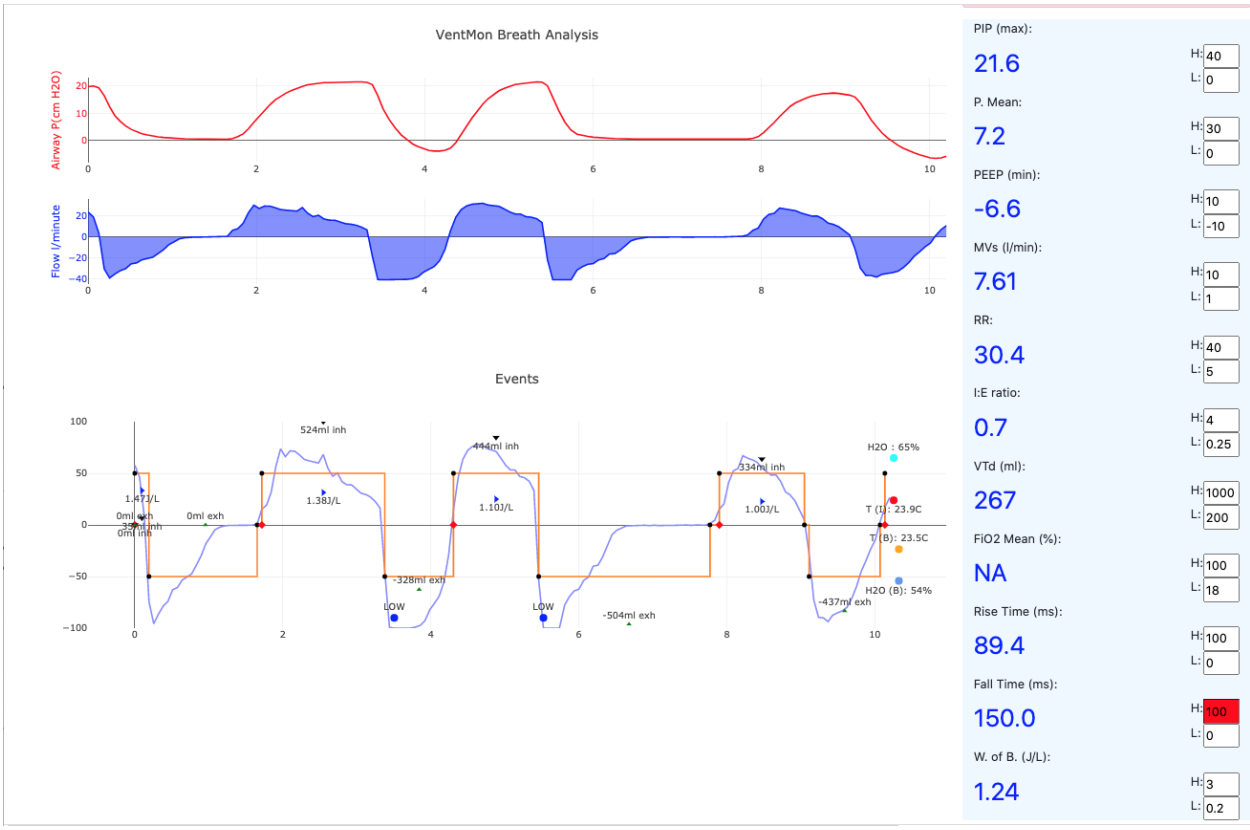

Figure 15: VentDisplay Data Display

These calculations allows us to present the numbers on the right side of the display, which are typical metric found on a commercial ventilator. Computation of these values allows a would-be ventilator's performance to be verified and characterized against a repeatable and relevant standard.

Features such as rise and fall time (at the bottom on the right) are particularly important. However, the most basic function is to dynamically report the respiration rate, peak pressure, minimum pressure, tidal volume and minute volume, and I:E ratio. In addition to the standard metrics, our software also measures pressure rise time which we have argued is an under-appreciated performance characteristic of rapidly manufactured emergency ventilators[13].

This software has been shown to be valuable, although it is dependent on accurate software decisions about the beginning and the ending of a breath, which can be subtle in some situations. The reliability of the data is inherently tied to the accuracy and precision of the device producing data for VentDisplay. VentMon is unique in this regard as the flow, differential pressure, and oxygen sensors used in the design require no calibration. After independent testing each component the VentMon team found that the flow and oxygen measurements are suitable for accurately measuring FiO_2 mixing in mechanical ventilation.

The Public Invention Respiration Data Standard format [7], is the underlying method by which data is communicated from the VentMon device to VentDisplay. The standard was created to provide a repeatable way for multiple teams to cooperate. Using this standard and cloud-based software enables VentMon to support a distributed engineering team—often required by the COVID-19 social distancing measures. The VentDisplay software allows export of the JSON binding of the PIRDS data to allow other programs to process if it if desired. We believe these tools could someday allow us to support telemedicine as suggested by others [12].

In fact we hope the VentMon will evolve into an open-source component that can be freely integrated into a pandemic ventilator or simply "bolted on" to be used to control ventilators or to monitor human patients. Although it can effectively test ventilation equipment today, before it can be recommended for use on patients, it will require regulatory approval, such as FDA approval. Before that can happens, the VentMon must progress beyond a number of limitations:

1. The current device is not autoclavable or easy to sanitize. Use on a human patient would therefore require it to be disposed of immediately.

2. Our software has not be subject to rigorous Quality Assurance (QA) and Risk Analysis (RA) as the FDA would request. We have argued[11] for an ecosystem in which design teams such as Public Invention produce extensive documentation including QA/RA documentation to be reused by firms that become medical device manufactures incorporating open-source designs.

3. The mechanical design of the VentMon will require additional alteration to make it robust enough for use in a clinical setting.

4. The current data lake and software allow data logging over days or weeks. To enable longitudinal study of ventilator reliability we would like to create software that allows this data to be analyzed or accessed across long periods of time.

8. Acknowledgements

Robert L. Read: Conceptualization, Methodology, Software, Writing - Original Draft, Supervision. Lauria Clarke: Software, Hardware, Investigation, Writing - Review and Editing. Geoff Mulligan: Conecptualiztion, Software.

Robert L. Read conceived of and wrote most of the firmware for the VentMon. Geoff Mulligan created the initial cloud-based IoT approach which became the PIRDS-logger. Lauria Clarke did most of the hardware design, including the assembly instructions and the printed circuit board based on an MIT-licensed Ethernet Feather Wing design from Particle IoT[2], and the SFM3X00

software repo for encapsulating Sensirion flow sensors in the Arduino environment. This paper was written by Robert L. Read and Lauria Clarke.

The Mozilla Open Source Software foundation gave Public Invention a grant that allowed us to develop the hardware and provide 20 VentMons to open source teams around the world free-of-charge, including paying shipping costs. Protocol Labs provided Public Invention a grant that has been used for web communication and conference costs.

We would like to thank early adopters of the VentMon on pandemic ventilator teams such as the ARMEE[4], PolyVent[14] and DIY-Beatmungsgerät.de[15] who gave us valuable bug reports and feedback.

Adafruit graciously continued to supply pandemic engineers and researchers even when their normal business was disrupted.

Student volunteers on the COVID19-vent-list project[10] helped to make the need for the VentMon clear by assessing over 100 pandemic ventilator teams.

9. Declaration of interest

None.

References

[1] Lauria Clarke. *VentMon v0.3T PCB*. Version v0.3T. Aug. 2020. URL: https://github.com/PubInv/ventmon-ventilator-inline-test-monitor/tree/master/design/pcb.

[2] *Hardware design files for the Particle Ethernet Wing*. Nov. 2020. URL: hhttps://github.com/particle-iot/ethernet-wing.

[3] UK MHRA. *Specification for ventilators to be used in UK hospitals during the coronavirus (COVID-19) outbreak*. 2020.

[4] *Millions of Ventilators*. Oct. 2020. URL: https://armeevent.com/.

[5] Joshua M Pearce. "A review of open source ventilators for COVID-19 and future pandemics". In: *F1000Research* 9 (2020).

[6] Robert L. Read. *PIRDS Data Logger*. Version v0.1. Aug. 2020. DOI: 10.5281/zenodo.4079382. URL: https://doi.org/10.5281/zenodo.4079382.

[7] Robert L. Read. *Public Invention Respiration Data Standard*. Version v0.1. Aug. 2020. DOI: 10.5281/zenodo.4079355. URL: http://doi.org/10.5281/zenodo.4079355.

[8] Robert L. Read. *VentDisplay v0.3T Release*. Version v0.3T. Aug. 2020. DOI: 10.5281/zenodo.4079374. URL: http://doi.org/10.5281/zenodo.4079355.

[9] Robert L. Read. *VentMon T0.2 Firmware Release*. Version T0.2. Aug. 2020. DOI: 10.5281/zenodo.4291147. URL: https://doi.org/10.5281/zenodo.4291147.

[10] Robert L. Read et al. *COVID-19 Vent List*. Oct. 2020. URL: https://docs.google.com/spreadsheets/d/1inYw5H4RiL0AC_J9vPWzJxXCdlkMLPBRdPgEVKF8DZw/edit#gid=0.

[11] Robert L. Read et al. *The Pandemic-inspired Case for an Open-Source Medical Hardware EcosystemVentDisplay v0.3T Release*. Sept. 2020. URL: https://makezine.com/2020/09/21/the-pandemic-inspired-case-for-an-open-source-medical-hardware-ecosystem/.

[12] Gregory B. Rehm et al. "Development of a research-oriented system for collecting mechanical ventilator waveform data". In: *Journal of the American Medical Informatics Association* 25.3 (2018), pp. 295–299.

[13] Erich B. Schulz and Robert L. Read. “The importance of characterising dynamic response and inertia in potential rapidly manufactured ventilator systems”. In: *Anaesthesia* (July 2020). DOI: 10.1111/anae.15190.

[14] *Welcome To Project PolyVent*. Oct. 2020. URL: https://www.polyvent.org/.

[15] *WILLKOMMEN BEIM PROJEKT DIY-BEATMUNGSGERÄT*. Oct. 2020. URL: https://www.xn--diy-beatmungsgert-5qb.de/.