



Hardware Article

eGreenhouse: Robotically positioned, low-cost, open-source CO₂ analyzer and sensor device for greenhouse applications

Elad Levintal^{a,*,1}, Kenneth Lee Kang^{b,c,1}, Lars Larson^b, Eli Winkelman^b, Lloyd Nackley^d, Noam Weisbrod^a, John S. Selker^{b,e}, Chester J. Udell^b

^a Environmental Hydrology and Microbiology, The Zuckerberg Institute for Water Research, The Jacob Blaustein Institutes for Desert Research, Ben-Gurion University of the Negev, Israel

^b Openly Published Environmental Sensing (OPeNS) Lab, Oregon State University, OR, United States

^c Department of Electrical Engineering and Computer Science, Oregon State University, OR, United States

^d Department of Horticulture, North Willamette Research and Extension Center, Oregon State University, OR, United States

^e Department of Biological & Ecological Engineering, Oregon State University, OR, United States

ARTICLE INFO

Article history:

Received 13 January 2021

Received in revised form 7 March 2021

Accepted 18 March 2021

Keywords:

CO₂ sensing

Agriculture

Open source hardware

Environmental sensing

ABSTRACT

Advances in gas sensors and open-source hardware are enabling new options for low-cost and light-weight gas sampling devices that are also robust and easy to use and construct. Although the number of studies investigating these sensors has been increasing in the last few years, they are still scarce with respect to agricultural applications. Here, we present a complete system for high-accuracy measurements of temperature, relative humidity, luminosity, and CO₂ concentrations. The sensors suite is integrated on the previously developed HyperRail device (Lopez Alcala et al., 2019) – a reliable, accurate, and affordable linear motion control system. All measurements are logged with a location and time-stamp. The system was assembled from only off-the-shelf or 3D printable products. We deployed the system in an agricultural greenhouse to demonstrate the system capabilities.

© 2021 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

Specifications table

Hardware name	eGreenhouse
Subject area	Environmental, Planetary and Agricultural Sciences
Hardware type	• Field measurements and sensors • Electrical engineering and computer science
Open Source License	GPL General Public License v3.0 CERN Open Hardware License
Cost of Hardware	eGreenhouse Sensor Package: \$277.49 Hub: \$108.39
Source File Repository	GitHub: https://github.com/OPeNSLab-OSU/eGreenHouse/tree/1D Zenodo: https://doi.org/10.5281/zenodo.4113959

* Corresponding author.

E-mail address: eladlevi@post.bgu.ac.il (E. Levintal).

¹ Contributed equally.

1. Hardware in context

The spatial and temporal distribution of CO₂ at the earth-atmosphere interface and at the boundary layer above it is of great importance for soil, agricultural and atmospheric sciences. In general, the source of elevated CO₂ concentrations at this boundary layer depends mostly upon root respiration and soil microbial activity [2–4] and on atmospheric transport (diffusion, dispersion, and advection) [5–8].

Improvement of CO₂ gas concentration detection could lead to better understanding of agricultural productivity and the transport mechanisms at this critical interface. Accurate monitoring of CO₂ can improve modeling and decision-making to enhance agricultural productivity [9]. Until recently, detecting CO₂ only employed stationary sensors providing low spatial resolution [10].

The growing use of open-source hardware for research purposes creates new opportunities to bridge the gap between CO₂ detection at high resolution and affordability. In agriculture, the use of open-source hardware and sensors for CO₂ gas sensing is scarce. However, recent research has revealed the possible applications of operating such gas sensing systems for agricultural purposes. For example, the use of an open-source gas sensing device to detect CO₂ concentrations in a rectangular greenhouse (106 m × 47 m) was proven by Roldán et al. [11]. Using a low-cost electrochemical CO₂ sensor, which is a relatively low-accuracy analog sensor, they provided a CO₂ concentration map inside a greenhouse. Their device also had temperature, relative humidity and luminosity sensors. Data from these sensors can be used for better greenhouse operation decision-making.

The aim of this work was to develop the eGreenhouse, a suite of sensors to evaluate CO₂ concentrations and dynamics using an advanced non-dispersive infrared (NDIR) CO₂ sensor inside a greenhouse. NDIR sensors are based on absorption spectroscopy and consist of a light source, a sample cavity and a detector [12]. We integrated this gas sensor together with temperature, relative humidity, and luminosity sensors, on a single logging device to gain high spatial and temporal resolution of the greenhouse environment. In addition, the device is small and transportable to allow deployment on a drone or a rail system to get location-tagged data throughout the greenhouse. The latter option was used to validate system performance.

2. Hardware description

The sensors used in this work are detailed in the next sections. First, the sections describe the sensors, then their integration on a single PCB, and finally operational instructions (software and data access). All sensors were chosen to: (1) provide accuracy similar to that of a laboratory sensor; (2) employ only off-the-shelf, open source sensors; (3) be low-cost compared to existing sensors; and (4) result in a lightweight system. All enclosures and fixtures were designed as 3D printable to allow broadly accessible replication.

A linear-motion actuator (HyperRail) positioned the sensor system [1]. The eGreenhouse includes two main components, the sensor package which is mounted on the HyperRail and the static hub from which the user can control the HyperRail and upload the data online (e.g., GoogleSheets)

The eGreenhouse can also be used in the following settings:

- Mounted on a drone for large area measurements (not necessarily within greenhouses).
- Static measurements within caves, boreholes or underground cavities in which CO₂ concentration regimes are of interest, e.g. [13–16].
- Adjacent to an open-source wind anemometer for a simple but efficient meteorological station with CO₂ readings

We note that the validation data presented in section 7 was obtained using an earlier model due to specific site requirements at which the validation was done (see more information in section 7). Below we present the hardware design that is similarly configured (e.g., K30 for CO₂ and SHT31-D for temperature and relative humidity), but with some minor improvements (re-designed data logger board).

3. Design files

Design Files Summary

Design file name	File type	Open source license	Location of the file
eGH	BRD	CERN Open Hardware License	https://doi.org/10.5281/zenodo.4113959
eGH	SCH	CERN Open Hardware License	https://doi.org/10.5281/zenodo.4113959
Code	C++	GNU General Public License v3.0	https://doi.org/10.5281/zenodo.4113959

eGH.brd: BRD file that can be loaded into AutoDesk EAGLE to get the layout of the eGreenhouse PCB.

eGH.sch: SCH file that can be loaded into AutoDesk ENGLE to get the schematic of the circuit for the eGreenhouse PCB.

4. Bill of materials

4.1. Materials for eGH_Sensor_Package

Designator	Component	Number	Cost per unit \$	Total cost\$	Source of materials	Material type
DevelopmentBoard	Adafruit Feather M0 with RFM95 LoRa	1	\$34.95	\$34.95	Adafruit	Non-specific
Antenna Kit	Radio – 900 MHz - RadioFruit 900Mhz Antenna Kit - For LoPy, LoRa, etc	1	\$12.75	\$12.75	Adafruit	Non-specific
Antenna Connector	uFL SMT Antenna Connector	1	\$0.75	\$0.75	Adafruit	Non-specific
Data Loggerwith RTC Board	Adalogger FeatherWing - RTC + SD Add-on For All Feather Boards	1	\$8.95****	\$8.95****	Adafruit	Non-specific
OPeN Data Logger with RTC Board	Hypnos	1	\$33.04****	\$33.04****	OPeN Lab	Non-specific
Micro SD Card	16 GB MicroSD Card	1	\$6.19	\$6.19	Amazon	Non-specific
CO ₂ Sensor	K30 10,000 ppm CO ₂ Sensor	1	\$85.00 ***	\$85.00	CO2Meter	Non-specific
Lux Sensor	Adafruit TSL2591 High Dynamic Range Digital Light Sensor	1	\$6.95	\$6.95	Adafruit	Non-specific
Temperatureand HumiditySensor	Adafruit Sensirion SHT31-D - Temperature & Humidity Sensor	1	\$13.95	\$13.95	Adafruit	Non-specific
Power Convertor	PowerBoost 1000 Charger - Rechargeable 5 V Lipo USB Boost @ 1A – 1000C	1	\$19.95	\$19.95	Adafruit	Non-specific
Sensor PCB	Custom Component	1	\$1.00*	\$23.00*** including shipping & minimum order	PCBWay	FR-4
Short Male Headers	Short Feather Male Headers – 12-pin and 16-pin Male Header Set	2	\$0.50	\$1.00	Adafruit	Non-specific
Short Female Headers	Short Headers Kit for Feather – 12-pin + 16-pin Female Headers	1	\$1.50	\$1.50	Adafruit	Non-specific
Battery	Lithium Ion Battery Pack – 3.7 V 6600mAh	1	\$29.50	\$29.50	Adafruit	Ion

Pricing Notes

* price will vary with source

** optional

*** price will vary with capacity

**** must have either one of them

Additional equipment that may be needed:

Soldering station for soldering sensor and other parts to PCB

4.2. Materials for Hub

Designator	Component	Number	Cost per unit \$	Total cost\$	Source of materials	Material type
DevelopmentBoard	Adafruit Feather M0 with RFM95 LoRa Radio – 900 MHz - RadioFruit	1	\$34.95	\$34.95	Adafruit	Non-specific
Antenna Kit	900Mhz Antenna Kit - For LoPy, LoRa, etc	1	\$12.75	\$12.75	Adafruit	Non-specific
Antenna Connector	uFL SMT Antenna Connector	1	\$0.75	\$0.75	Adafruit	Non-specific
Ethernet Connector	Adafruit Ethernet FeatherWing	1	\$19.95	\$19.95	Adafruit	Non-specific
Battery	Lithium Ion Battery Pack – 3.7 V 6600mAh	1	\$29.50	\$29.50	Adafruit	Ion
Short Female Headers	Short Headers Kit for Feather – 12-pin + 16-pin Female Headers	1	\$1.50	\$1.50	Adafruit	Non-specific
Ethernet Cable	6ft Ethernet Cable	1	\$8.99	\$8.99*	Amazon	Non-specific

Pricing Notes

* optional

Additional equipment that may be needed:

Soldering station for soldering development board and Ethernet Connector

5. Build instructions

Here we provide building instructions for each individual part and then demonstrate prototype wiring using the Sensor PCB. Additional information, instructions and updates, such as the PCB design and connections can be found at <https://github.com/OPeNslab-OSU/eGreenHouse/wiki> (eGreenhouse tab)

5.1. eGreenhouse sensor package individual hardware components setup

All steps in this section require a soldering iron and solder. Follow typical safety protocols for soldering to prevent burns and inhalation of lead solder or flux/rosin solder: eye protection, heat resistant work surface, clamps to hold materials to be soldered, ventilation, etc.

5.1.1. Development board

In this step, you need the Antenna Connector, Antenna Kit, development board, and 12-pin + 16-pin male headers that came with the development board (Fig. 1).

First, solder the Antenna Connector in the designated location on the board. Fig. 2 with a yellow arrow will tell where to solder the Antenna Connector and Fig. 3 shows the soldered Antenna Connector.

Once that step is complete, solder 12-pin + 16-pin male headers (Fig. 4). Make sure that the long side of the male headers are facing down while the short side of the male headers go through the board towards the top.

Connect the Antenna Kit, except the actual plastic antenna, on the Antenna Connector. Because it is easy for the Antenna Kit to be disconnected from the Antenna Connector, secure (strain relief) the wire for the Antenna Kit to the Development Board with either electrical tape or hot glue so that it will stay in position (Fig. 5).

5.1.2. Data logger with RTC board

As shown from the bill of materials, there is an option to use either a RTC Adalogger from Adafruit or the OPeNslab Data Logger with RTC Board that was created from OPeNslab Lab. We will be using the Hypnos board rather than Adalogger. If you are going to use the Hypnos board, then reference here.

For this step, you will need the Hypnos board, 12-pin + 16-pin male headers, 12-pin + 16-pin female headers, coin cell battery, and Micro SD card (Fig. 6). If you are planning to use Adalogger Adafruit as Data Logger with RTC Board, then you need Adalogger board, 12-pin + 16-pin stacking headers, coin cell battery, and Micro SD Card.

Solder the female headers on the top of the board (side with microSD slot) where it has a label of “feather” (Fig. 7).

Once female headers are soldered, then solder the male headers such that the long male headers are facing out on the bottom (side with coin cell battery holder) (Figs. 8 and 9).

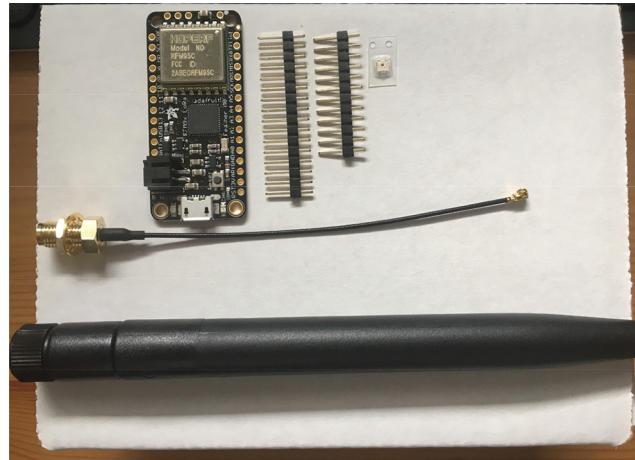


Fig. 1. The materials that are required for the Development Board setup.

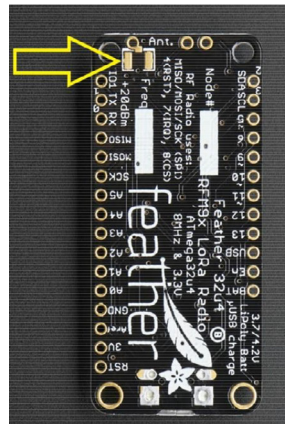


Fig. 2. Bottom side of the Development Board as received from vendor. The Yellow arrow indicates where to solder the Antenna Connector.

5.1.3. CO₂ sensor

For this step, you need the K30 CO₂ Sensor and 4-pin male headers (Fig. 10). The 4-pin male headers can be created by breaking the 12-pin male header into a 4-pin male header.

Solder the 4-pin male headers on the four black dots shown by the red arrows in Fig. 11. The soldered CO₂ Sensor is shown in Figs. 12 and 13.

5.1.4. PowerBoost hardware

For this step, you need the PowerBoost without the USB Type-A adapter and 8-pin male headers (Fig. 14). The USB Type-A adapter will not be used in this project. The 8-pin male headers are the remaining male-pins after using the 4-pin male headers from the CO₂ Sensor.

Solder the 8-pin male headers such that the long pin will be projecting out of the bottom of the board (Fig. 15).

5.1.5. Lux sensor

For this step, you need Lux Sensor and 6-pin male headers (Fig. 16). You can create the 6-pin male headers from the 16-pin male headers by breaking it to 6-pin male headers.

Solder the 6-pin male headers onto the PowerBoost such that the long side of the headers are on the bottom of the board (Fig. 17).

5.1.6. Temperature and relative humidity sensor

For this step, you need the Temperature and Relative Humidity Sensor and 7-pin male headers (Fig. 18). You can create the 7-pin male headers from the 16-pin male headers that was used for setting up the Lux Sensor by breaking it to 7-pin male headers.

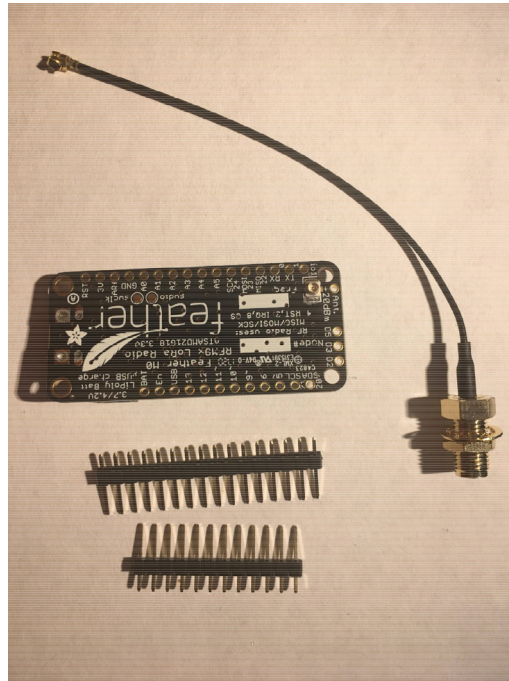


Fig. 3. Development board with accessories after soldering the Antenna Connector on the board (see the yellow arrow at Fig. 2 for the soldering location).

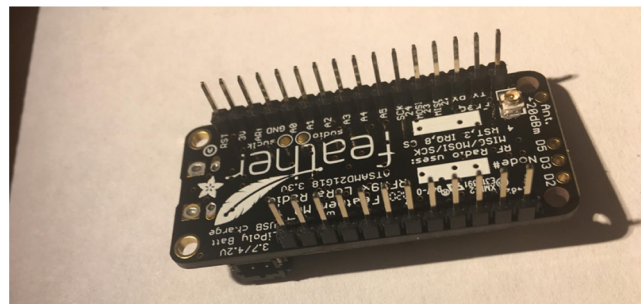


Fig. 4. Bottom of the Development board with both male headers soldered on.

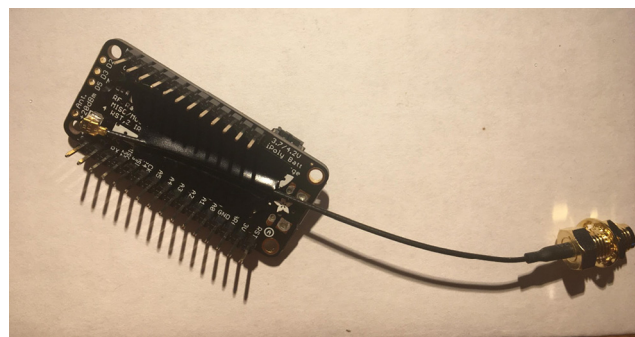


Fig. 5. The finished development board for the eGreenhouse sensor package.

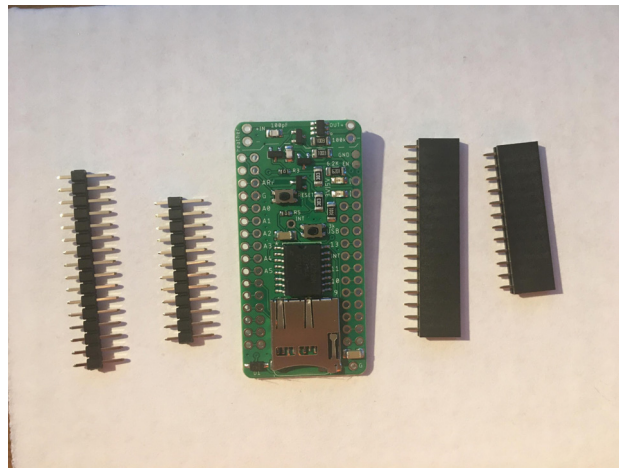


Fig. 6. The materials for setting up the Data Logger with the RTC Board (cell battery and Micro SD card not shown).

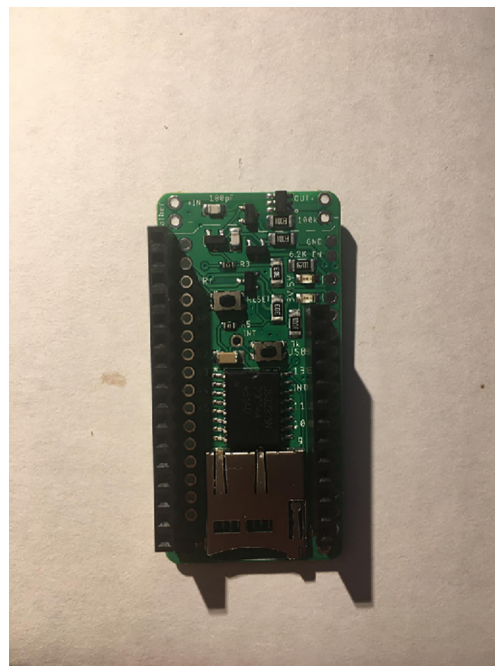


Fig. 7. Top view of the finished Data Logger with RTC Board.

Solder the 7-pin male headers onto the Temperature and Relative Humidity Sensor such that the long side of the headers are on the bottom of the board (Fig. 19).

Once this step is complete, then you are done with eGreenhouse sensor package individual hardware setup step. The next step is the assembly of the sub-systems.

5.2. eGreenhouse Sensor package connection setup

Steps in this section require a soldering iron and solder.

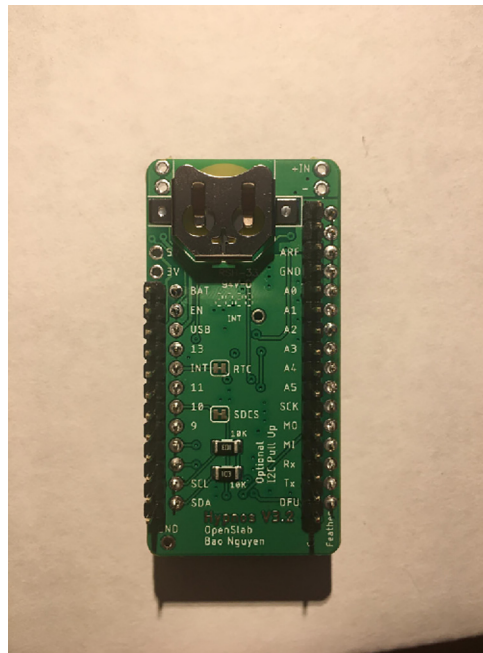


Fig. 8. The top view of the male headers soldered into the bottom of the Data Logger with the RTC Board.

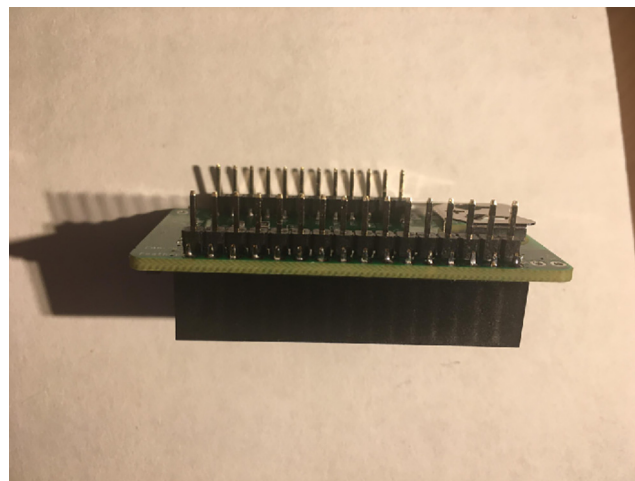


Fig. 9. Side view of the Data Logger with the RTC Board with both male and female headers soldered on. Insert the Micro SD card into the Micro SD card holder and the coin cell battery.

5.2.1. Connect Development Board to the Data Logger with RTC Board

For this step, you need the Development Board and the Data Logger with RTC Board. Connect the Development Board on top of the Data Logger with RTC Board using the headers (Figs. 20 and 21).

Once they are connected, then move to the next step. However, for the next few steps, you can disconnect them for convenience.

5.2.2. Connect Data Logger with RTC Board to the PCB

For this step, you need the PCB and Data Logger with RTC Board and the PCB. Fig. 22 shows the PCB front view. Make sure the PCB is facing forward.

Solder the Data Logger to RTC Board where it says “**Adafruit Feather**” on the PCB such that the long male headers are facing downward. Make sure the 12-pin male headers follows the 12-pin holes, the 16-pin male headers, and 16-pin holes. Figs. 23 and 24 show both top and bottom view after soldering the Data Logger with RTC Board on the PCB. Alternatively,



Fig. 10. The CO₂ Sensor and 4-pin male headers.

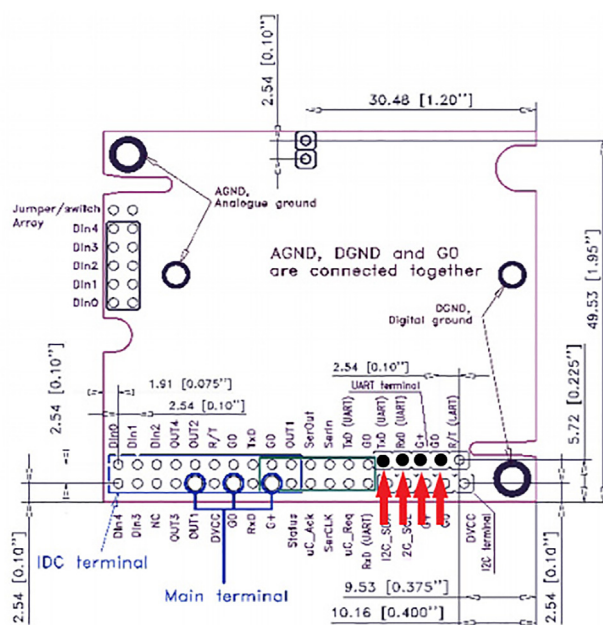


Fig. 11. The CO₂ Sensor layout with red arrows indicating where to solder the 4-pin male headers.

12-pin and 16-pin female headers can be soldered onto the Sensor PCB to allow future removal of the Data Logger with RTC Board unless there is a height restriction.

5.2.3. Connect Lux Sensor to the Sensor PCB

For this step, you need the Sensor PCB and Lux Sensor. Solder the Lux Sensor where it is labeled as “**TSL2591**”. Make sure that the label of each pin follows the correct pin hole. For example, the GND on the Lux Sensor is connected to GND on the PCB. Figs. 25 and 26 show the top and bottom view after soldering Lux Sensor to the PCB.

5.2.4. Connect Temperature and relative Humidity Sensor to the Sensor PCB

For this step, you need the Sensor PCB and Temperature and Relative Humidity Sensor. Solder the Temperature and Relative Humidity Sensor where it is labeled as “**SHT31-D**”. Make sure that the label of each pin follows the correct pin hole. Figs. 27 and 28 show the top and bottom view after soldering the Temperature and Relative Humidity Sensor to the PCB.



Fig. 12. Top view of the CO₂ Sensor after soldering 4-pin male headers.

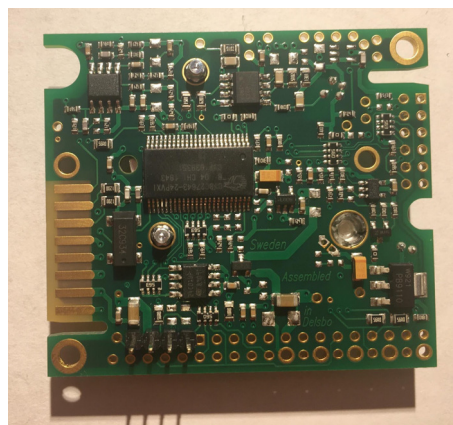


Fig. 13. Bottom view of the CO₂ sensor after soldering the 4-pin male headers.

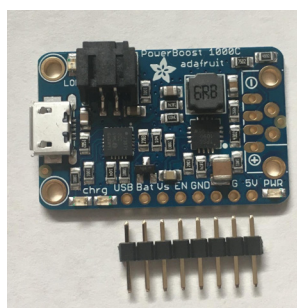


Fig. 14. The PowerBoost.

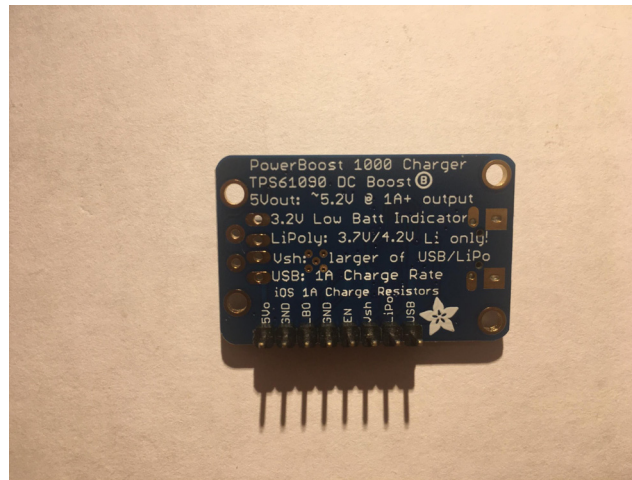


Fig. 15. The bottom view after soldering the 8-pin male headers on the PowerBoost.

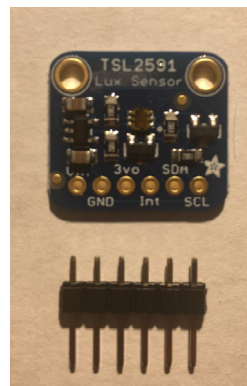


Fig. 16. Top side of Lux Sensor and 6-pin male headers.

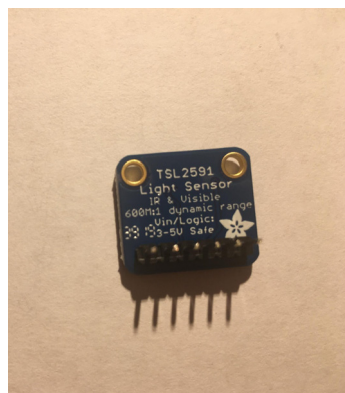


Fig. 17. Finished bottom side of the Lux Sensor with 6-pin male headers soldered on.

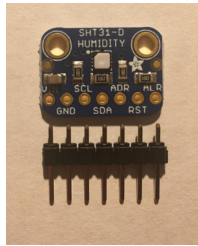


Fig. 18. Temperature and Relative Humidity Sensor and 7-pin male headers.

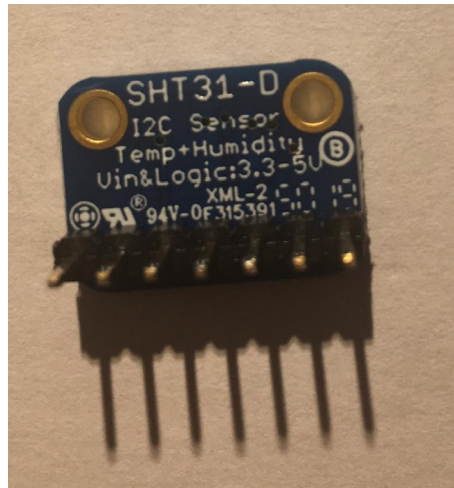


Fig. 19. Bottom view of the finished Temperature and Relative Humidity Sensor.

5.2.5. Connect the PowerBoost to the Sensor PCB

For this step, you need PCB and PowerBoost. Solder the PowerBoost where it is labeled as “**PowerBoost 1000C.**” Make sure that the label of each pin follows the correct pin hole. [Figs. 29 and 30](#) show the top and bottom view after PowerBoost to the PCB.

5.2.6. Connect CO₂ Sensor to the Sensor PCB

For this step, you need the Sensor PCB and CO₂ Sensor. Solder the CO₂ Sensor where it is labeled as “**K30.**” Make sure to match the pin labels between the CO₂ Sensor and Sensor PCB: G0 pin will be connected to GND on the PCB, G+ to 5 V, RXD to RXD, and TXD to TXD. [Figs. 31 and 32](#) show the top and bottom view after CO₂ Sensor to the Sensor PCB.

Once that step is complete, you are done with the eGreenhouse Sensor Package Setup step. For verification, connect the battery to the Development Board as shown in [Fig. 33](#). If it lights up for both the Data Logger with the RTC Board and the Development Board, and flashing for the CO₂ Sensor, then everything is working perfectly.

5.3. Hub hardware setup

All steps in this section require a soldering iron and solder.

5.3.1. Development Board

In this step, you need the Antenna Connector, Antenna Kit, Development Board, and 12-pin + 16-pin female headers ([Fig. 34](#)).

First solder 12-pin + 16-pin female headers to the Development Board such that the socket side is on the top of the board ([Figs. 35 and 36](#)).

Once that step is complete, then solder the Antenna Connector to the bottom side of the Development Board ([Fig. 37](#)).

Once that step is complete, then connect the Antenna Kit, except the actual plastic antenna, on the Antenna Connector. Because it is easy to be disconnect, either add electrical tape or hot glue so that it will be in position ([Fig. 38](#)).

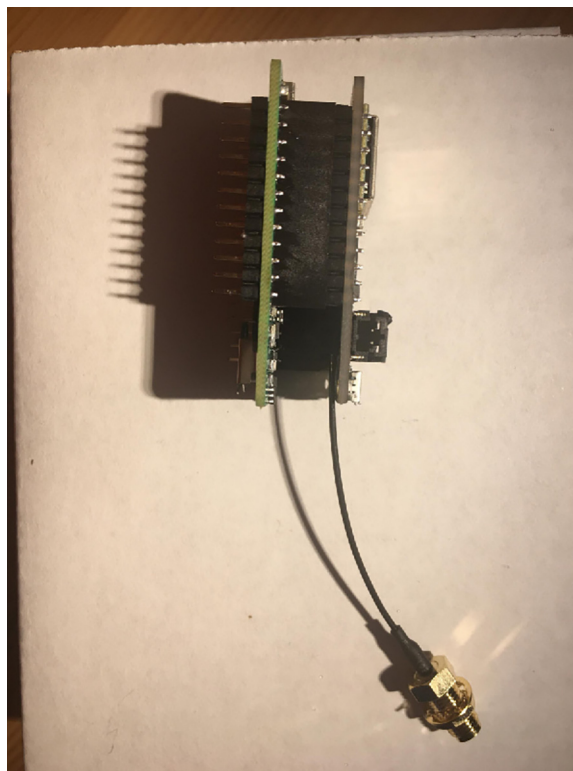


Fig. 20. Connected Development Board and Data Logger with RTC Board on the one side.

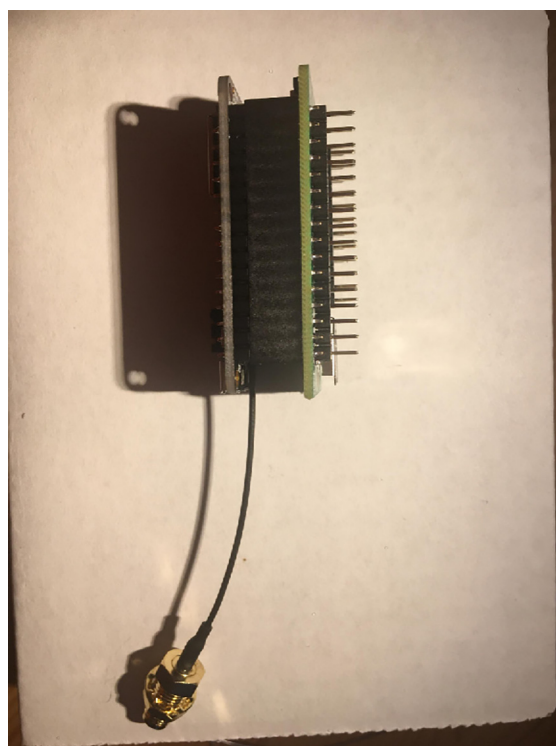


Fig. 21. Connected Development Board and Data Logger with RTC Board on the other side.

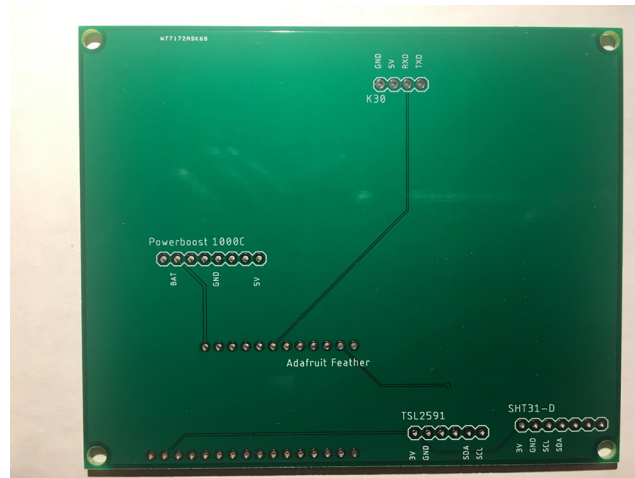


Fig. 22. Top view of the eGreenhouse sensor PCB.

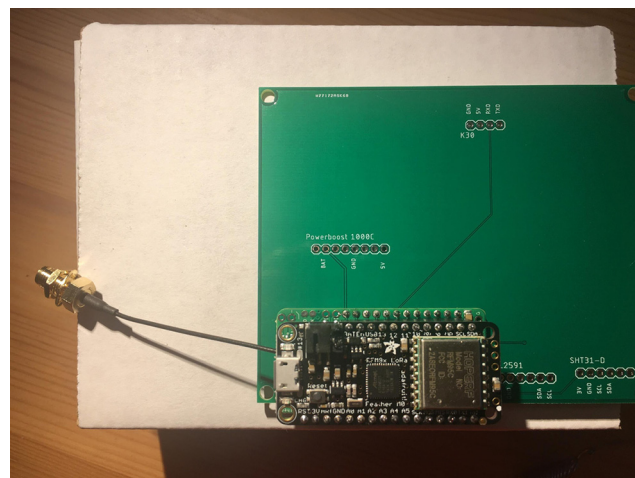


Fig. 23. The top view of the PCB after soldering the Data Logger with the RTC Board and stacked Development Board.

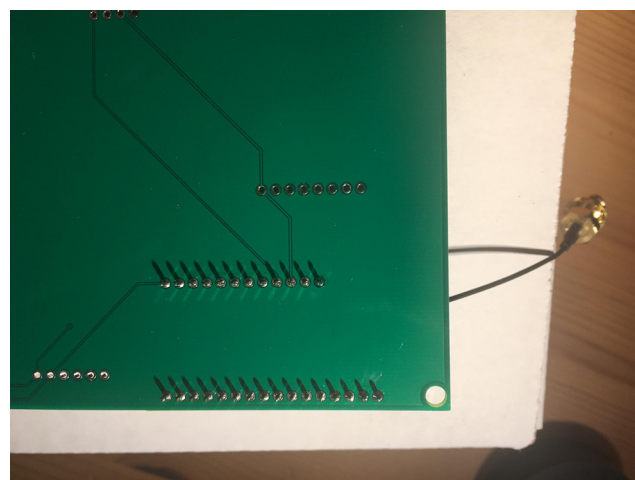


Fig. 24. The bottom view of the PCB after soldering the Data Logger with the RTC Board.

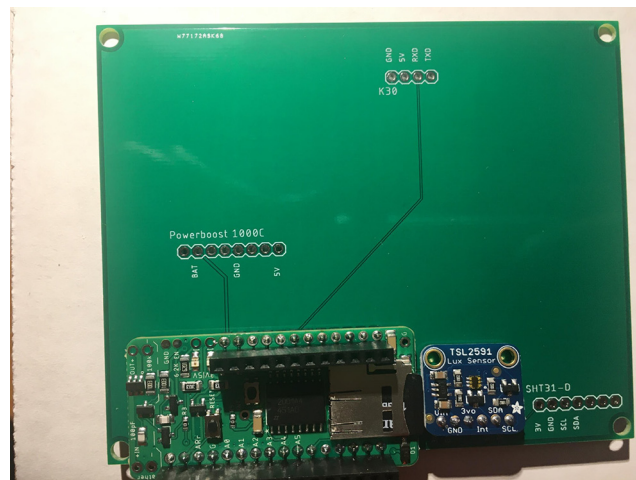


Fig. 25. Top view after soldering the Lux Sensor to the PCB.

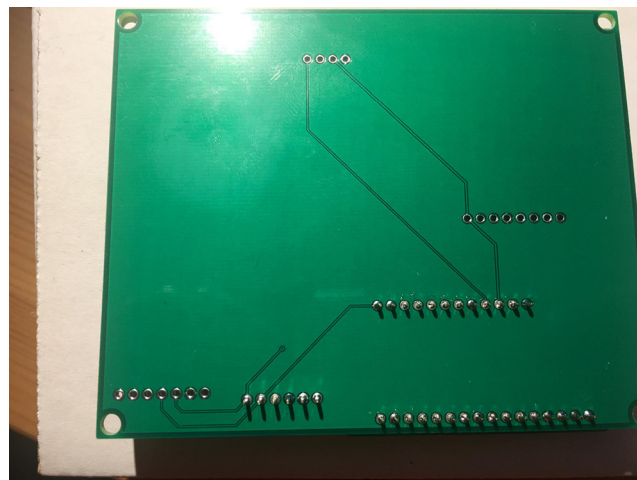


Fig. 26. Bottom view after soldering the Lux Sensor to the PCB.

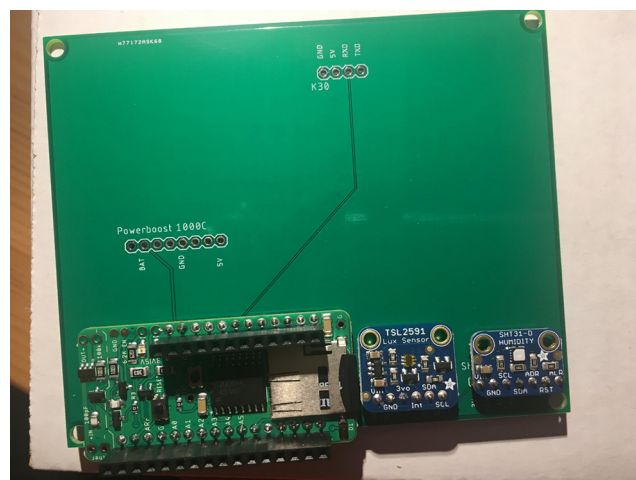


Fig. 27. Top view after soldering the Temperature and Relative Humidity Sensor to the PCB.

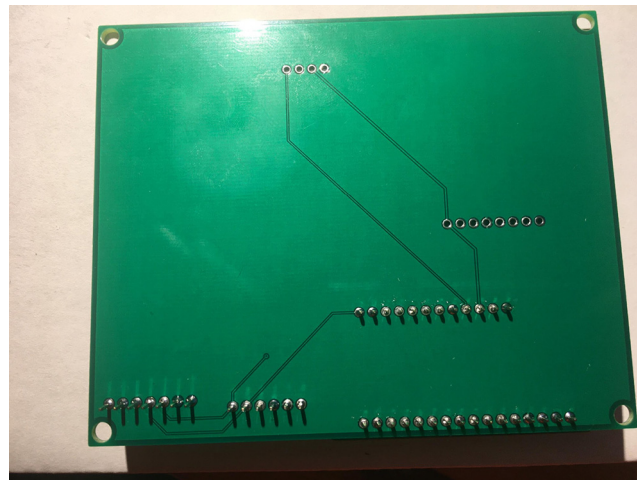


Fig. 28. Bottom view after soldering the Temperature and Relative Humidity Sensor to the PCB.

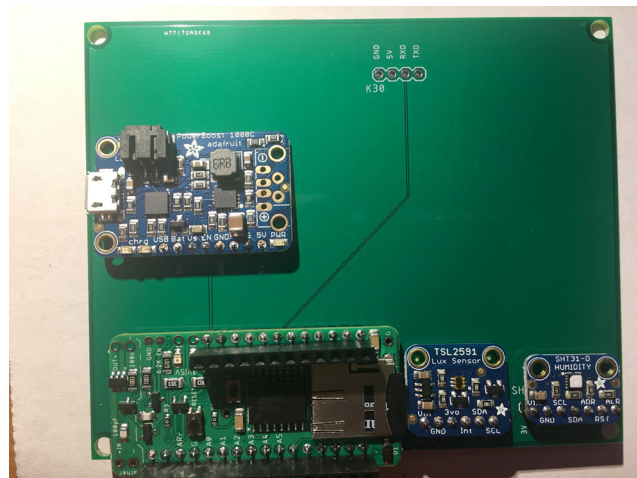


Fig. 29. Top view after soldering the PowerBoost to the PCB.

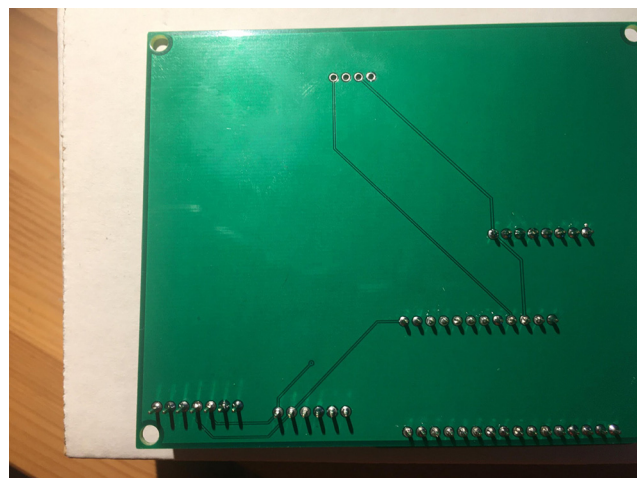


Fig. 30. Bottom view after soldering the PowerBoost to the PCB.

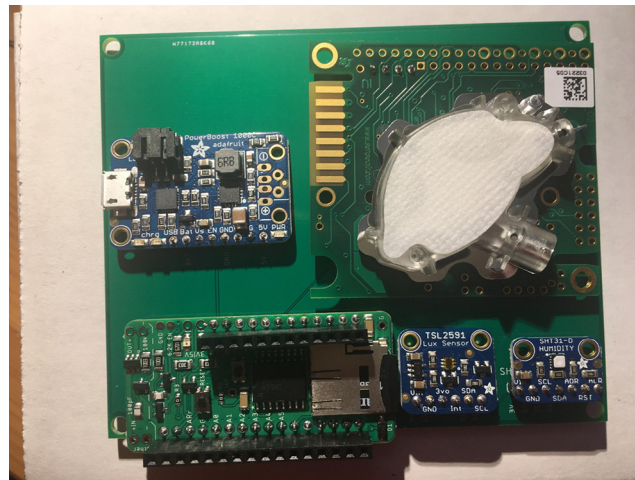


Fig. 31. Top view after soldering the CO₂ Sensor to the PCB.

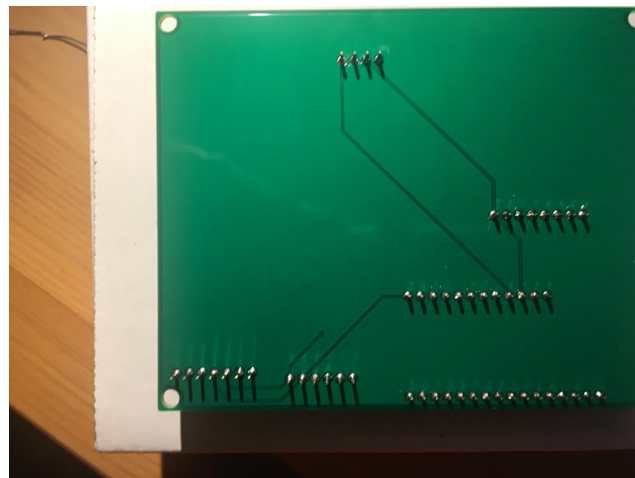


Fig. 32. Bottom view after soldering the CO₂ Sensor to the PCB.

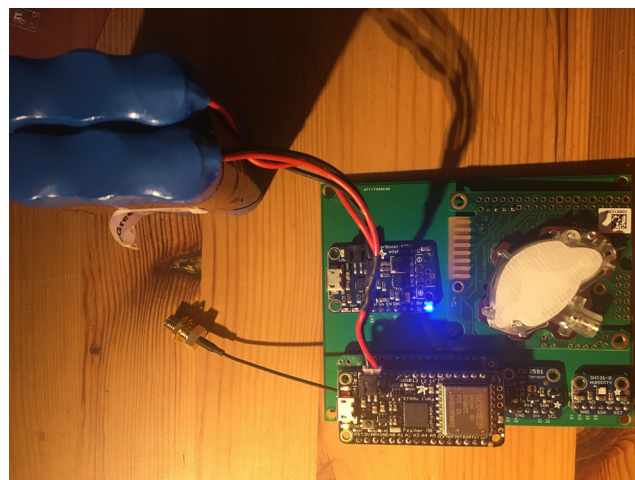


Fig. 33. The eGreenhouse Sensor Package with power on from the battery.

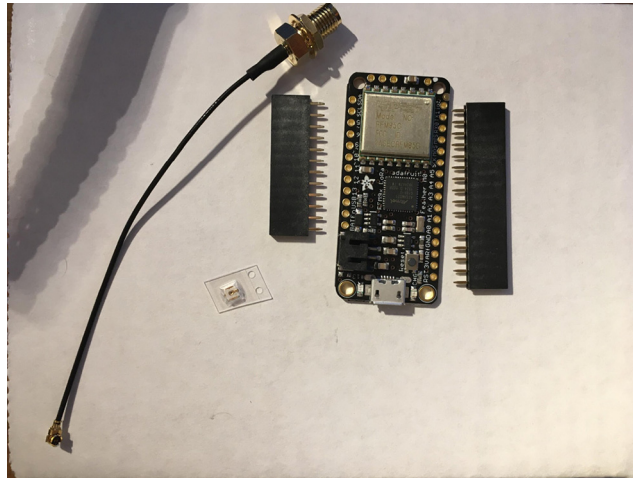


Fig. 34. Antenna Connector, Antenna Kit, Development Board, and 12-pin + 16-pin female headers for hub setup.



Fig. 35. Top view after soldering female headers on the Development Board.

5.3.2. Ethernet Connector

For this step, you need Ethernet Connector and 12-pin + 16-pin male headers that came with the Development Board package. Solder the male headers facing down (Figs. 39 and 40).



Fig. 36. Bottom view after soldering female headers on the Development Board.

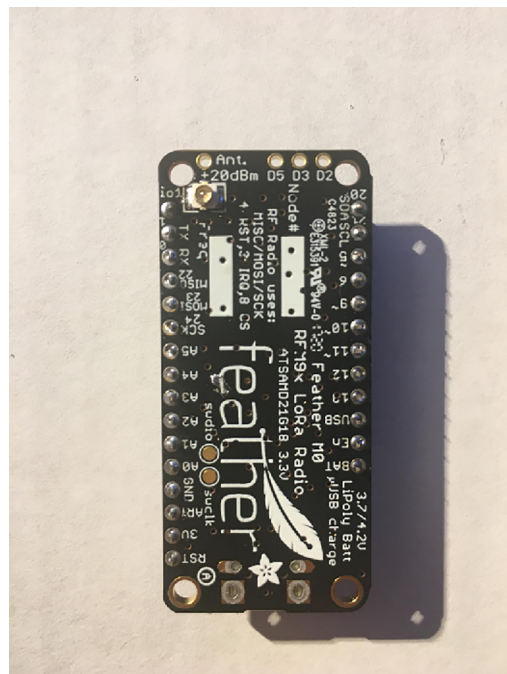


Fig. 37. Bottom side of the Development Board for the hub after soldering on the Antenna Connector.

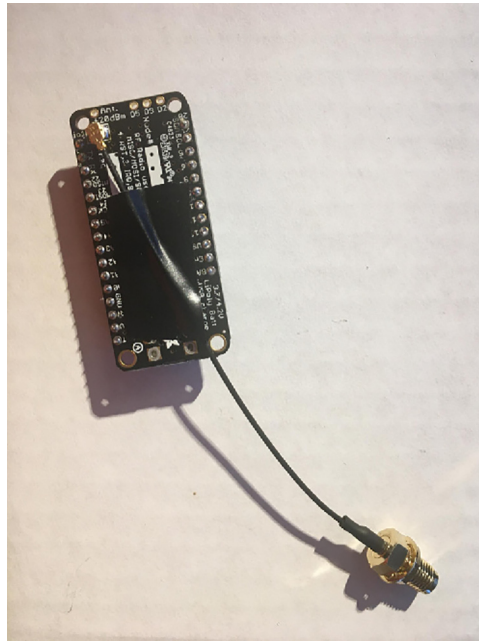


Fig. 38. The Development Board for the Hub.



Fig. 39. Top view after soldering the male headers on the Ethernet Connector.

5.3.3. Connect Ethernet Connector to Development Board

For this step, you need the Development Board and Ethernet Connector. Connect the Ethernet Connector on top of the Development Board ([Fig. 41](#)).

Once that is done, you are completely done with the Build Instructions.

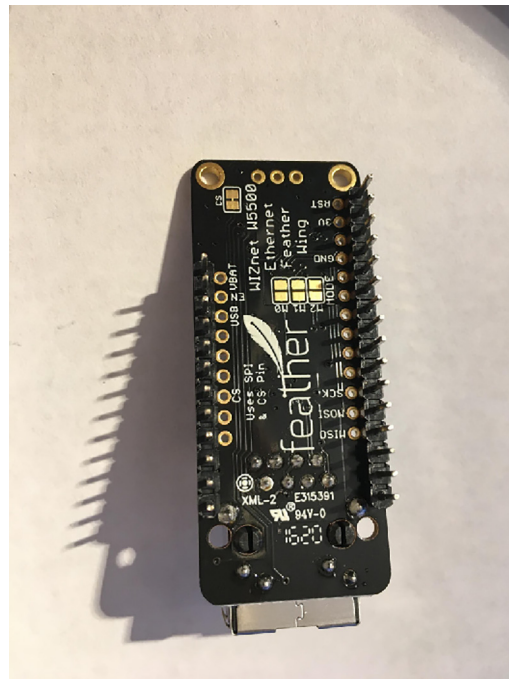


Fig. 40. Bottom view after soldering the male headers on the Ethernet Connector.

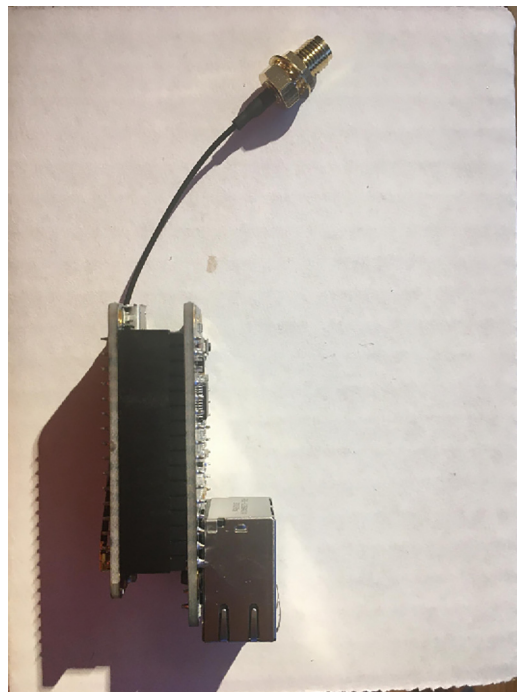


Fig. 41. The connected Development Board and Ethernet Connector.

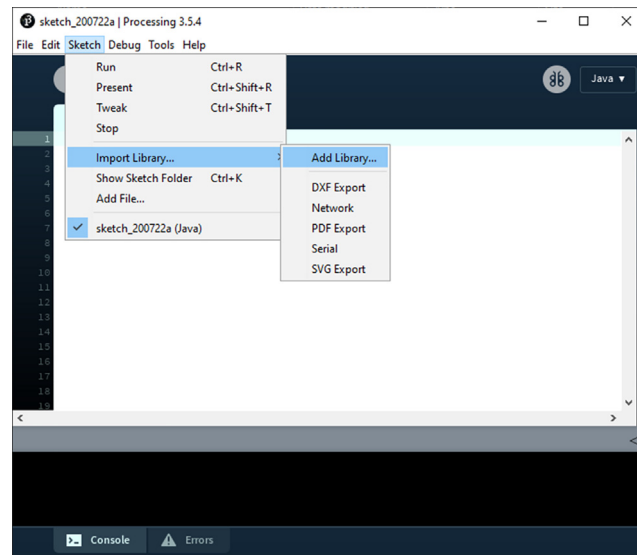


Fig. 42. Processing Application: where to select to add libraries to the board.

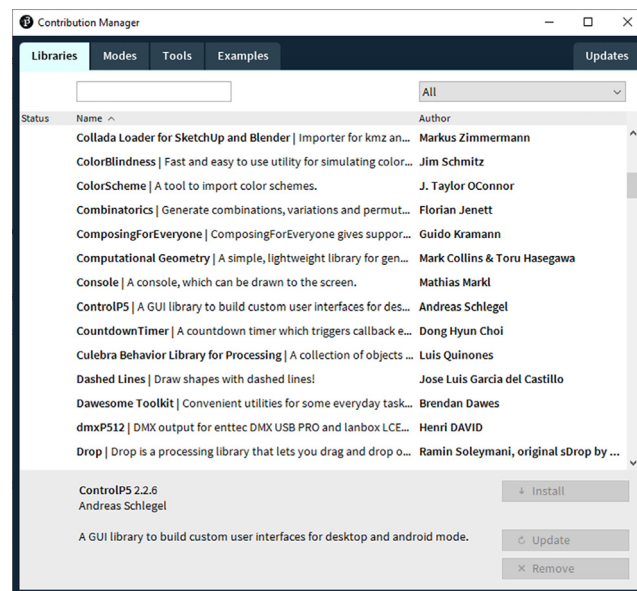


Fig. 43. The Processing Contribution Manager for installing new Libraries for the Processing.

6. Operation instructions

6.1. Software download

If you are using the Windows Operating System, download Git Bash Shell, Arduino IDE, and Processing.

If you are using the Mac Operating System, download Arduino IDE and Processing. You do not need Git Bash Shell.

Open Git Bash for Windows and Terminal for Mac. Enter the following command. This should install all the dependencies and libraries to operate properly.

```
curl -fsSL https://raw.githubusercontent.com/OPEnSLab-OSU/Loom/master/setup/setup-windows.sh | sh
```

Upon completion, open Processing. Then go to Sketch -> Import Library... -> Add Library... See Fig. 42 for location of the option.

Once you select Add Library, you will get a pop shown in Fig. 43. From the menus on the top left of the window, search both ControlP5 and Arduino (Firmate) and install both.

In the Code folder, which is under the Design folder, you can find a folder named Dependencies. In that folder, copy the AccelStepper folder and paste it in Document -> Arduino -> libraries such as Fig. 44. Each system setup might vary the location of the folder; therefore, please check where the Arduino -> libraries is located.

Once you complete this setup, then you are set up with the Arduino IDE software.

Note: when you open the Arduino IDE, you will get the following message in Fig. 45. You can simply ignore that.

6.2. eGreenhouse Sensor package

There are two different versions of eGH_Sensor_Package code to use.

If you are using Adalogger from Adafruit as the Data Logger with the RTC Board, then open the folder called **Non Hypnos eGH_Sensor_Package**.

If you follow the build instructions using the OPeNS Data Logger with RTC Board, then open the folder called **Hypnos eGH_Sensor_Package**.

In the code folder, go to **Hypnos eGH_Sensor_Package** folder. Then inside that folder there will be another folder called eGH_Sensor_Package. In that folder, open eGH_Sensor_Package.ino file, and you will see code as in Fig. 46.

In that screen, select Tools -> Board -> Loom SAMD Boards -> Loomified Feather M0 (Fig. 47).

Once that is complete, connect the eGreenhouse Sensor Package to the computer with Micro USB Cable. The Micro USB Cable should be connected to the Development Board, not the PowerBoost. It should recognize the board. Then go to Tools -> Port and see if there is an option to select. If there is no option, try either a different cable or reconnect it. It should be as in Fig. 48.

Once the computer recognizes the board, then go to the config.h. In that file, you will find either

```
{'name':'DS3231','params':[10,false]}
```

or

```
{'name':'PCF8523','params':[10,false]}
```

In the config, you can see it has the same as [10,false]. The 10 is indicating the time zone, which it is currently set up as PDT. You can change the time zone by referring here. For example, if you are in Eastern Standard Time, it will be 5 rather than 10, which will be [5,false].

Once you change the config.h file, then click upload using the arrow pointing to the right (Fig. 49).

Wait until it gets the message shown in Fig. 50. Once the message appears as "CPU Reset", then it has uploaded properly, and is ready to use.

6.3. Hub

6.3.1. Before for setting up the Hub

For this step, you will need Ethernet cable that must be connected to an open Ethernet port. At the same time, you will need your own Mac Address. Make sure the Mac Address is in decimal (not hexadecimal).

6.3.2. GoogleSheets setup

First, create a new GoogleSheets in either your personal drive or an organization drive. Once it is created, go to Tools -> Script editor (Fig. 51). Once you select Script editor, then you will get a new tab (Fig. 52).

Inside the Code file, go to the folder called **GoogleSheets**. Inside, you will have Spreadsheet.gs. Open that file with any text editor or simply notepad (Fig. 53)

In this file, copy everything and paste that in the Script editor. Before you paste it to the Script editor, make sure it is empty (Fig. 54).

Once you save it, you will get this pop up (Fig. 55).

You can name the project as you prefer. Once that is complete, then go to Publish -> Deploy as web app... (Fig. 56).

In there, change the option Who has access to the app from Only myself to Anyone, even anonymous. (Fig. 57).

Once you change the option, then select the blue button Deploy. There will be some permission review when you select that option. Make sure you authorize with your google account.

Then you will get a pop up that is like Fig. 58. Copy the URL that is provided in the pop up and save on notepad.

Once you save it in a separate location, then you can close the Script editor.

Going back to your GoogleSheets, there is an URL for that GoogleSheets. Copy that and save it in the same location where you save your other URL. But make sure which one is which. These URLs will be used for the next step. For example, this will be what will look in your notepad after saving both URLs. (Fig. 59).

6.3.3. Upload to Hub code on device

Within the main project folder, open the Hub folder. In that folder, only open Hub.ino. Once you open it, you will get the screen in Fig. 60.

If you have previously selected the board, it should set as default. If not, then change the board to Loomified Feather M0 like Fig. 47.

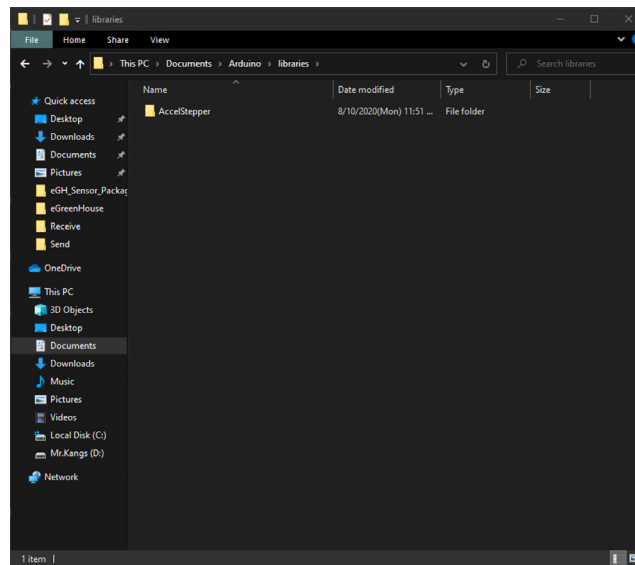


Fig. 44. Paste the AccelStepper folder in Document -> Arduino -> libraries. Here, the libraries folder is located This PC -> Document -> Arduino -> libraries.

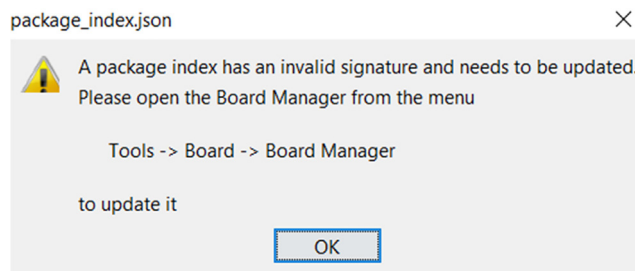


Fig. 45. Warning message after installing the Arduino and the Command line. This can be ignored.

```

eGH_Sensor_Package [Arduino 1.8.13]
File Edit Sketch Tools Help
eGH_Sensor_Package config.h
1 //////////////////////////////////////////////////
2 //
3 // This is the eGreenHouse Sensor Package.
4 // This program will get the coordinates from HyperDrive Code
5 // then measure values of the following sensors
6 // then log that data values into SD, and send the data value
7 // to the Hub over LoRa(Radio Communicator).
8 //
9 // This version of code is designed for the Hynnos Board.
10 // If you are not using a Hynnos Board, then please use the other code.
11 //
12 // CO2 Sensor: K20
13 // Luminosity Sensor(Light Sensor): TSL2591
14 // Temperature & Relative Humidity Sensor: SHT31-D
15 //
16 // In that file, the order will be the following:
17 // A. Device Name
18 // B. Device Number
19 // C. Date from RTC for UTC
20 // D. Time from RTC for UTC
21 // E. Package Number
22 // F. Temperature in celsius from SHT-01(there is +/-0.3 celsius error)
23 // G. Humidity in % from SHT-01(there is +/- 2% error)
24 // H. Visible Spectrum Light in nm from TSL2591
25 // I. Infrared Light in nm from TSL2591
26 // J. Full Spectrum Light in nm from TSL2591
27 // K. CO2 value in ppm
28 // L. Location in nm
29 //
30 // Author: Kenneth Kang
31 //
32 //
33 //////////////////////////////////////////////////

```

Fig. 46. The eGH_Sensor_Package.ino code from the Arduino IDE. Read the description for version type and additional information.

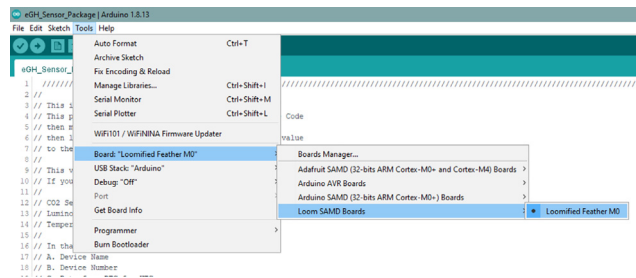


Fig. 47. Selecting the Board in the Arduino IDE.

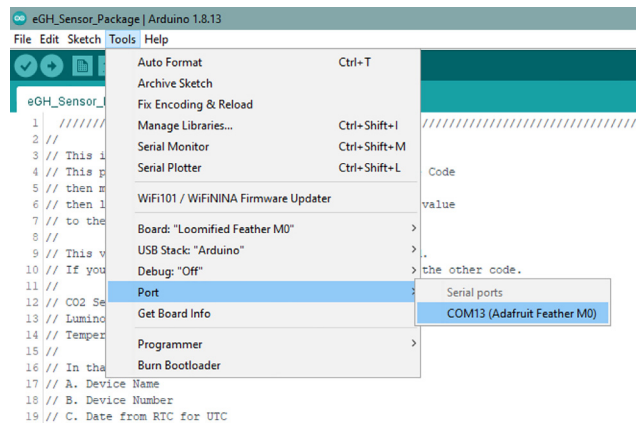


Fig. 48. The proper look when you connect the eGreenhouse Sensor Package. Note that the port number may be different.

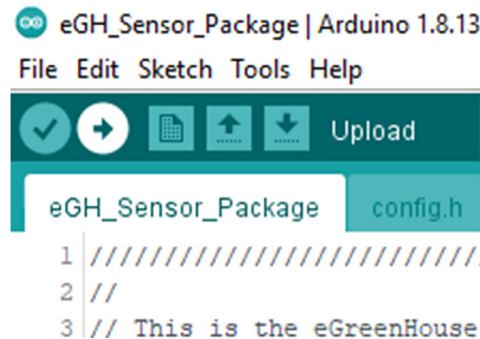


Fig. 49. Upload button on the Arduino IDE.

Once that is complete, then connect the Hub to the computer with Micro USB Cable. It should recognize the board. Then go to Tools -> Port and see if there an option to select. If there is no option, try either a different cable or reconnect it. It should be as in Fig. 61. Also, writing down the port number just for the Hub will be useful for later.

Once the computer recognizes the board, go to config.h file (Fig. 62).

You can see that there are some parts that are commented or in <> (e.g., Fig. 62, line 7). For the Mac address, enter them in decimal. For example, it will be formatted as [1,542,241,65,23] (Fig. 63, line 7). Note that this Mac address is a fake one.

For the < your-script-id>, it will be replaced with the URL from the Script editor. In this example, it will be AKfycbyRTgyVAms4SBZaVxhs-CpaaegEvKzoXVe5f25BmC24TOi22NGj

```

Arduino : CAN_CHECKSUM_MEMORY_BUFFER
Erase flash
done in 0.829 seconds

Write 157008 bytes to flash (2454 pages)
[=====] 100% (2454/2454 pages)
done in 1.183 seconds

Verify 157008 bytes of flash with checksum.
Verify successful
done in 0.102 seconds
CPU reset.

```

Fig. 50. The console log showing successful upload.

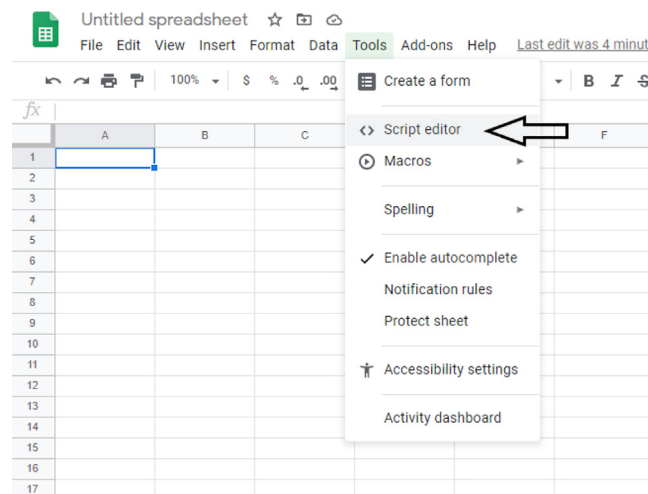


Fig. 51. The drop-down menu location where Script editor is found.

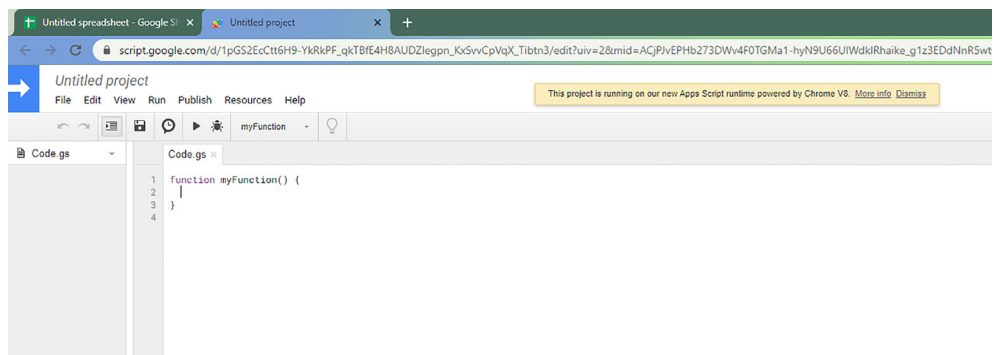


Fig. 52. The Script editor tab for your GoogleSheets.

```

Spreadsheet.gs - Notepad
File Edit Format View Help
/*This needs to be applied using the script editor tool within google spreadsheets in order to function.
Specifics on how to set this up can be found here. https://www.hackster.io/detox/transmit-esp8266-data-to-google-sheets-8fc617
Code:*/
//-----
// Originally published by Mogsdad@Stackoverflow
// Modified for The OPEnS Lab
// by Thomas DeBell, Luke Goertzen
//-----

// Automatically called whenever the linked PushingBox Scenario
// has data sent to it in the form of key-value pairs.
function doGet(e) {

    var result = 'Ok'; // assume success

    if (e.parameter == undefined) {
        result = 'No Parameters';
    }
    else {
        // Turn the Get Request arguments into a dictionary using the key-value pairs from the request.
        // Adds Date and Time fields to start of dictionary
        var data_dict = getDataDict(e.parameter);

        // The keys from data_dict is the set of columns in the spreadsheet.
        var ignore_keys = ["sheetID", "tabID", "null"];
        var column_set = difference(Object.keys(data_dict), ignore_keys);

        // The name of the sheet for the data to go to is the IDtag field that is sent.
        var sheet_id = stripQuotes(data_dict['tabID']);
        var spreadsheet = SpreadsheetApp.openById(data_dict['sheetID']);
        var sheet_list = spreadsheet.getSheets();
        var sheet = spreadsheet.getSheetByName(sheet_id);

        // If the tabID is now, then create a sheet for it
    }
}
Ln 1, Col 1 100% Windows (CRLF) UTF-8

```

Fig. 53. The Spreadsheet.gs file opened with Notepad.

```

Untitled spreadsheet - Google S Kenneth Kang
script.google.com/d/1pGS2eCtt6H9-YkrKPF_qkT8FE4H8AUDZlegpn_KxSvCpVqX_Tibtn3/edit?uiw=2&

Kenneth Kang
File Edit View Run Publish Resources Help

Code.gs Code.gs
151
152
153 // Returns a list of column names
154 function getColumnList(col_string) {
155     return col_string.split(',');
156 }
157
158
159 // Sets the set of columns that a sheet is using
160 function setSheetColumnList(sheet_name, col_list) {
161     var doc_properties = PropertiesService.getDocumentProperties();
162     doc_properties.setProperty(sheet_name, getColumnString(col_list));
163 }
164
165
166 // Gets the set of columns that a sheet is using
167 function getSheetColumnList(sheet_name) {
168     var doc_properties = PropertiesService.getDocumentProperties();
169     return getColumnList(doc_properties.getProperty(sheet_name));
170 }
171
172
173 // Remove leading and trailing single or double quotes
174 function stripQuotes( value ) {
175     return value.replace(/^[\'"]|[\']$/g, "");
176 }
177
178
179 // Checks if two arrays are equal to each other.
180 function arraysEqual(a, b) {
181     if (a === b) return true;
182     if (a == null || b == null) return false;
183     if (a.length != b.length) return false;
184
185     for (var i = 0; i < a.length; ++i) {
186         if (a[i] != b[i]) return false;
187     }
188     return true;
189 }
190
191
192 // Finds the difference of two arrays
193 // Used to remove various columns from being printed
194 function difference(a1, a2) {
195     var result = [];
196     for (var i = 0; i < a1.length; i++)
197         if ((a2.indexOf(a1[i]) == -1) && (a1[i][0] != '$'))
198             result.push(a1[i]);
199     return result;
200 }
201
202

```

Fig. 54. The Spreadsheet.gs code on your Script editor.

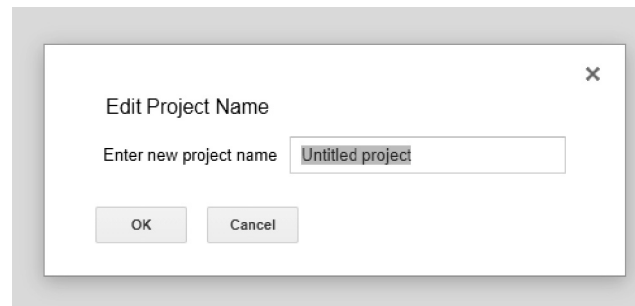


Fig. 55. Pop up for saving the Script Editor.

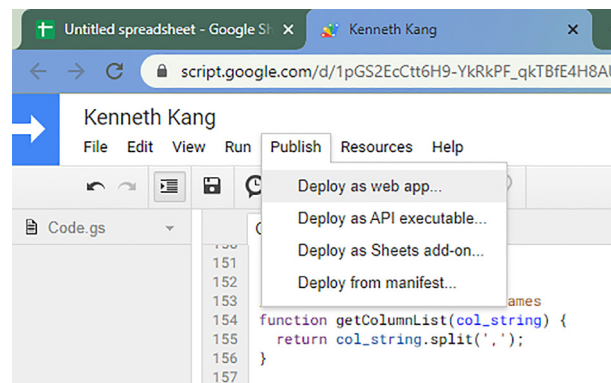


Fig. 56. The option to select Deploy as web app...

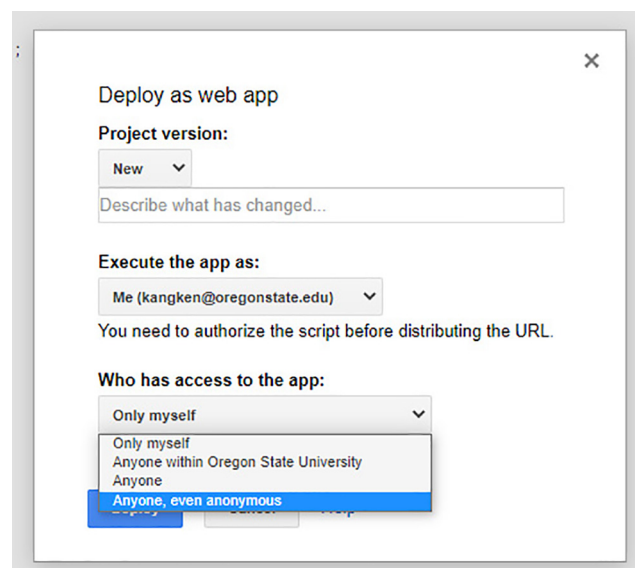


Fig. 57. The option to select Anyone, even anonymous rather than Only myself.

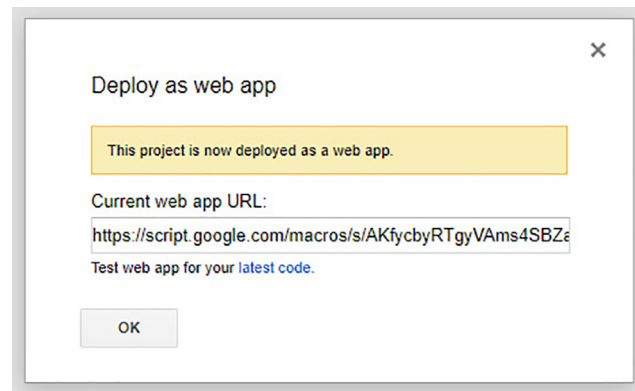


Fig. 58. The URL once you deploy and review your account permissions.

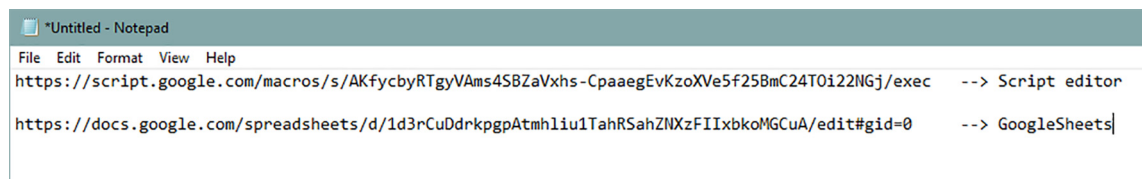


Fig. 59. Both URLs used for the next step.



Fig. 60. The Hub Code Screen from the Arduino IDE. Make sure to read the description if you are in the correct code.

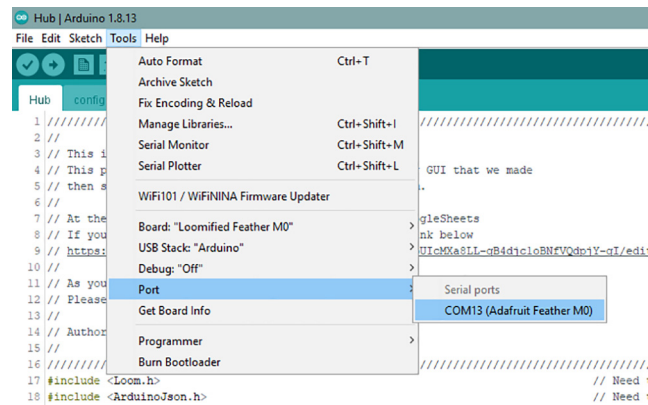


Fig. 61. The proper look when you connect the Hub. Note that the port number may be different.



Fig. 62. The config.h file for the Hub.ino file.

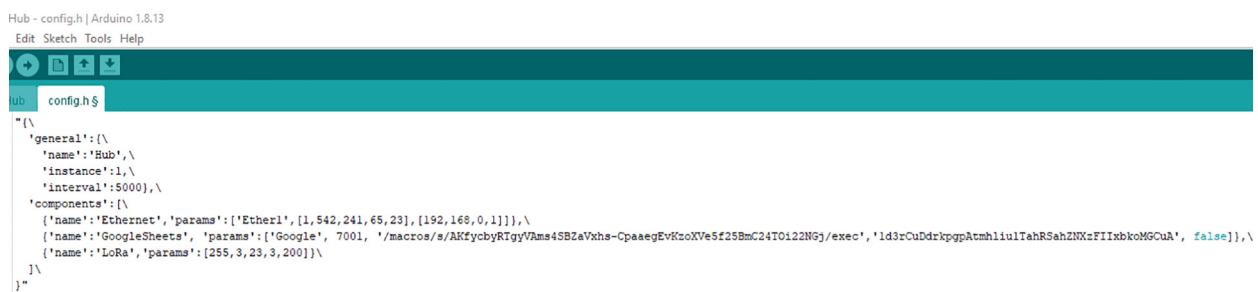


Fig. 63. The updated config.h file after entering Mac Address, Script ID, and Sheet ID.

Last, <your-sheet-id> will be from the second URL, which it is from the Sheet. In this example, it will be 1d3rCuDdrkpgpAtmhliu1TahRSahZNXzFI1xbkoMGCuA (Fig. 63, line 8).

Once you make the changes in the config.h file, save it and click upload using the arrow pointing to the right (Fig. 64).

Wait until it gets the message shown in Fig. 65. Once you get this message as "CPU Reset", then it uploaded properly and is ready to use.

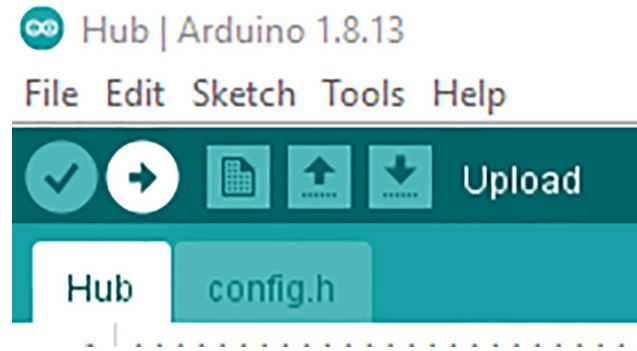


Fig. 64. Upload button on the Arduino IDE.

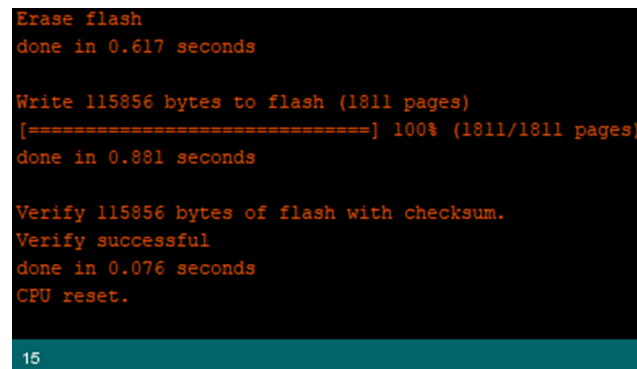


Fig. 65. The console log stating that the upload was successful.

6.4. HyperRail

We note that additional description regarding the HyperRail can be found in Lopez et al. [1]. Within the main project folder, open the Hyper folder. In that folder, only open Hyper.ino. Once you open it, you will get the screen shown in Fig. 66.

If you have previously selected the board, it should set as default. If not, then change the board to Loomified Feather M0.

Once that is complete, then connect the Hyper to the computer with Micro USB Cable. It should recognize the board. Then go to Tools -> Port and see if there is an option to select (Fig. 67). If there is no option, try either a different cable or reconnect it.

Once the computer recognizes the board, then click upload using the arrow pointing to the right (Fig. 68).

Wait until it gets the message shown in Fig. 69. Once you get this message as “CPU Reset”, then it has uploaded properly and is ready to use.

6.5. Loading firmware on microcontroller

In the design file, open the GUI folder. Inside that folder, only open the GUI.pde file (Fig. 70).

In the code, change the port number for the Hub Board where it is written “String port = “COM5””. You just need to change the number (Fig. 71, line 38). We note that Mac ports show up as USBmodem#### instead of COM# (Windows).

Once that is set, connect the Hub board to the computer and click the green arrow button on the top left (Fig. 72).

Then you will get this interface as shown Fig. 73.

In the application, you can move the position of the HyperRail, the velocity, the radius (not relevant in the eGreenhouse application), and the duration or number of loops. All the text inputs only take integers. As shown in the GUI, there is a timer for the CO₂ sensor warm up time. This timer is included because the CO₂ sensor needs at least 6 min warmup time to get accurate measurements. If the eGreenhouse sensor package was running for more than 6 min, this time can be ignored. On the bottom right, you can see the console log. If the user selects one of the four buttons, it will print out if it transmits

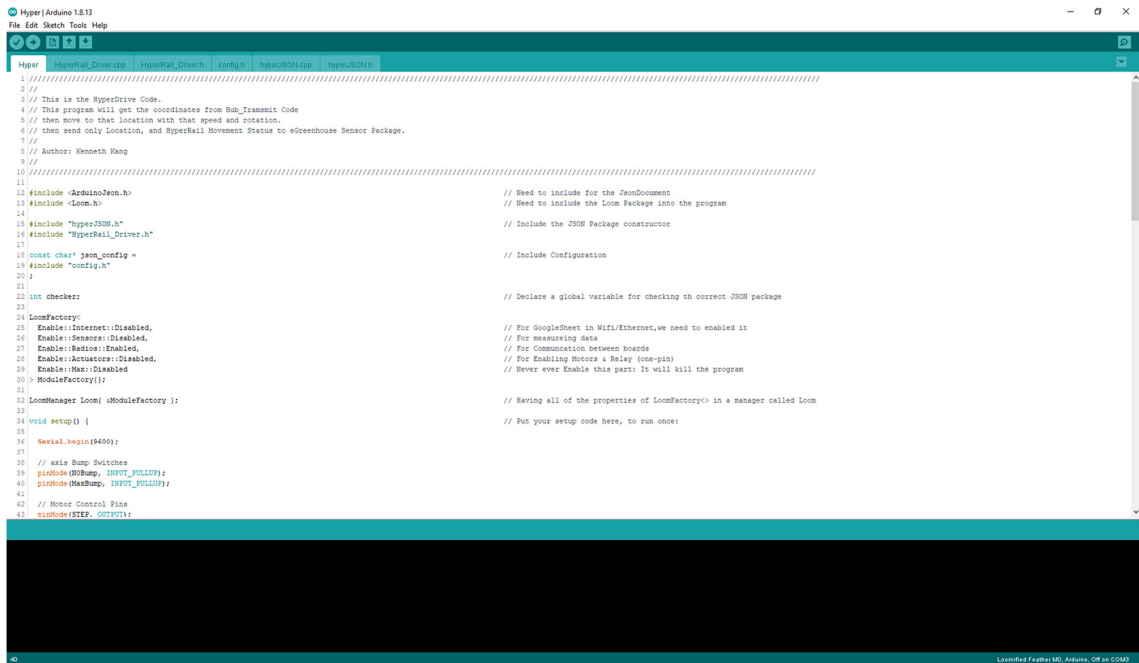


Fig. 66. The Hyper Code Screen from the Arduino IDE.

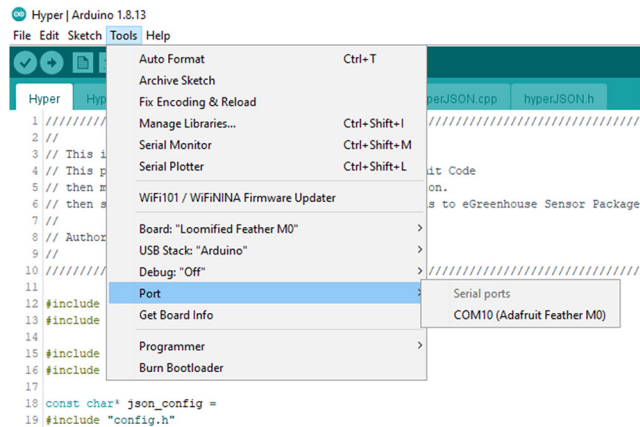


Fig. 67. The proper look when you connect the Hyper. Note that the port number may be different.

to the HyperRail board or not. If not, the failed transmission will be displayed. Once you have successful transmission, you are set to control the HyperRail and measure the eGreenhouse sensor package if both boards are on.

7. Validation and characterization

System validation was performed in a full-scale deployment of the eGreenhouse on a 25-meter HyperRail at the North Willamette Research and Extension Center, OR, USA (Fig. 74). The project was conducted in a 30 m (L) × 3 m (W) × 2.5 m (H) naturally lit, steel framed hoop house (also known as a cold frame). The house was covered with polyethylene plastic, which was rated for 90% light transmissivity. The house was naturally ventilated by raising the plastic on the sidewalls 1 m on both sides of the 30 m length. Due to this specific site requirements, in this eGreenhouse setup we used

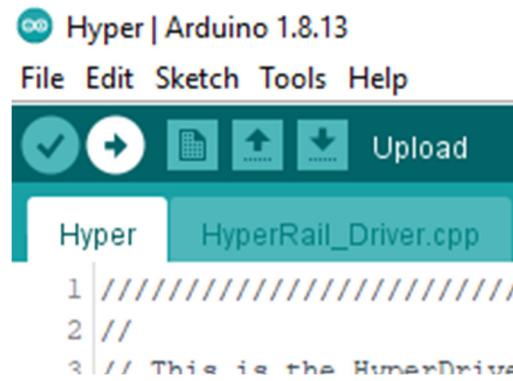


Fig. 68. Upload button on the Arduino IDE.

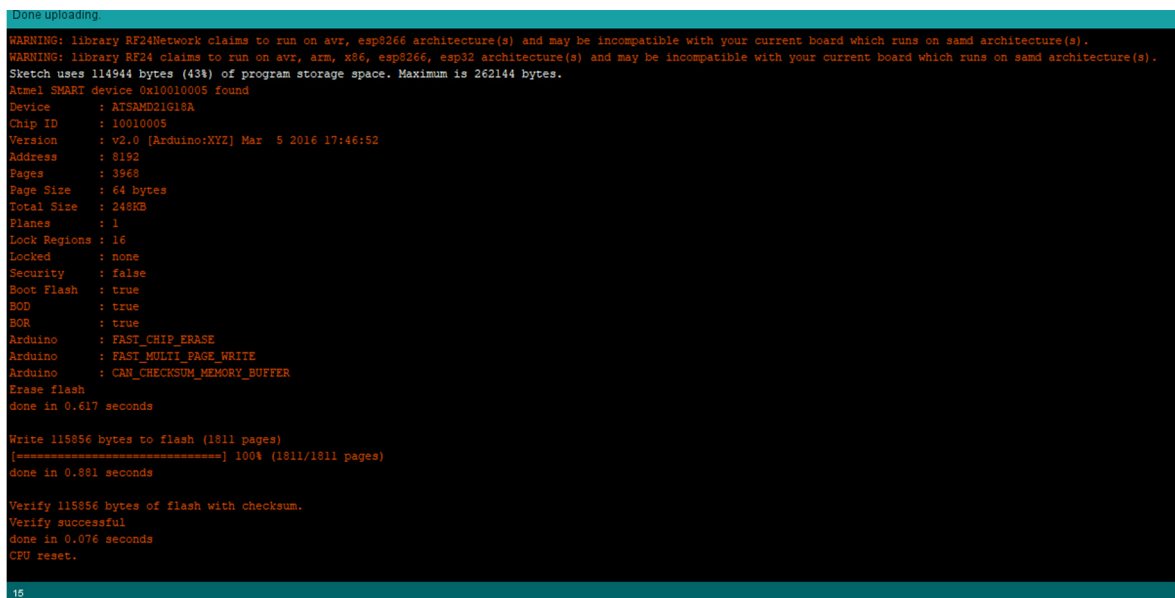


Fig. 69. The console log stating that the upload was successful.

a nRF radio instead of the LoRa radio with all other key sensors being the same (K30 for CO₂ and SHT31-D for temperature and relative humidity). Data was sent via nRF across three nodes spanning over 25 m (0, 12.5, and 25 m). The HyperDrive hub [1] drove the sensor package along the rail and sent requests for specified sensors at any spatial interval. Upon receipt of these data, the hub sent the data bundles to a third node at the extension office (~100 m from the greenhouse) with ethernet connectivity. The data was uploaded to Google Sheets, in addition to the onboard SD card logging. Data from the eGreenhouse was compared to data from a nearby meteorological station located ~200 m from the greenhouse.

Data from six consecutive days are presented in Fig. 75. Temperature and relative humidity measured by the eGreenhouse followed the expected trend of higher temperature and lower relative humidity during daytime compared to nighttime (Fig. 75-a and b). Within the greenhouse, temperatures were higher by ~20 °C than the outside temperature during daytime due to the greenhouse effect. No significant spatial differences were found between the three nodes of 0, 12, and 25 m. During nighttime, temperature and relative humidity within the greenhouse were the same as atmospheric values.

CO₂ values ranged between ~400 (atmospheric concentration) and ~550 ppm, with higher values during nighttime (Fig. 75-c). This is expected because throughout the night the plants respire (CO₂ is emitted to the greenhouse atmosphere) and photosynthesis, the process by which plants use sunlight and CO₂ to synthesize food, is suppressed.

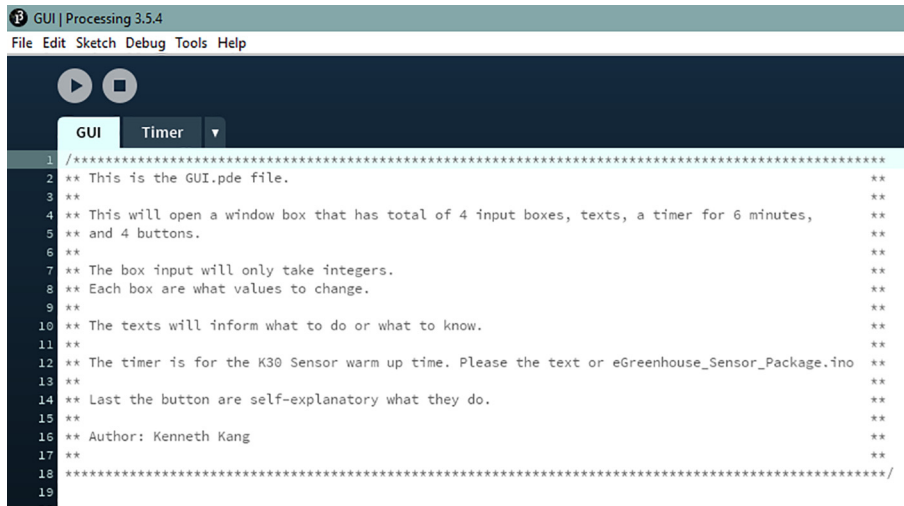


Fig. 70. Processing Application for the GUI.

```

33 Arduino arduino;
34
35 /*****
36 ****Declare variable for functions****
37 *****/
38 String port = "COM5";
39
40
41 String GoTo = "0";
42 int Axis_Length;

```

Fig. 71. The location where the port number can be changed.

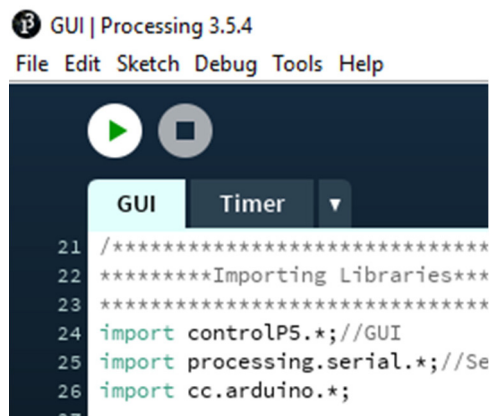


Fig. 72. The location to run the Processing Application.

The light values within the greenhouse followed the daily solar cycle (Fig. 75-d). Lux values increased with sunrise until reaching sensor saturation each day around noontime and then decreasing after sunset to 0 lux. Light values showed higher spatial distribution than other tested parameters. This is demonstrated by the differences of lux values within the green-

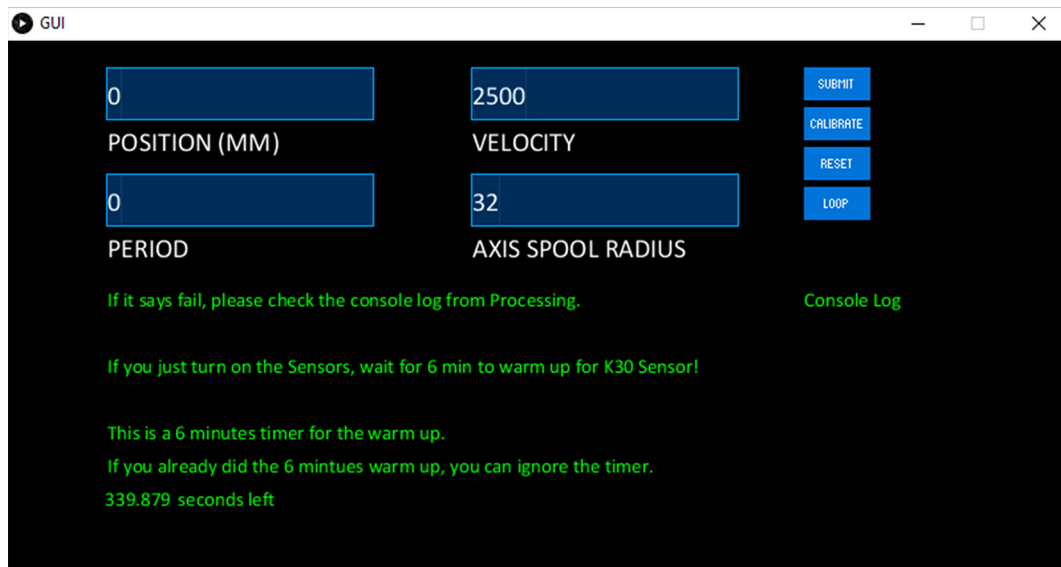


Fig. 73. The Graphical User Interface (GUI).

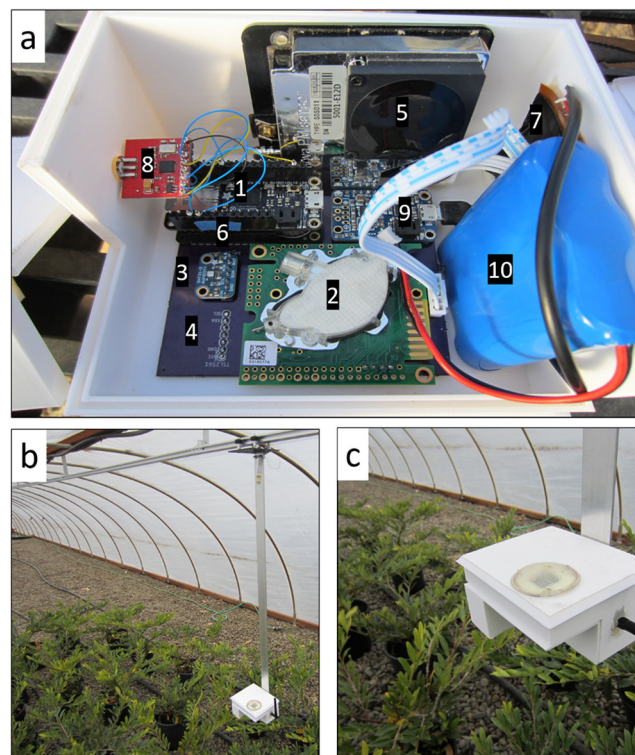


Fig. 74. Experimental setting. (a) eGreenhouse hardware – 1. Adafruit Feather Adalogger, 2. CO₂ sensor Senseair K30, 3. Temperature and relative humidity sensor Adafruit SHT31-D, 4. Light sensor Adafruit TSL2561, 5. Particulate matter sensor Nova PM SDS011 (not operational in this experiment), 6. RTC Adafruit DS3231 Featherwing, 7. Wireless charging Adafruit Universal Qi, 8. Transmission Nordic Semiconductor 2.4 GHz Nrf, 9. Power management Adafruit PowerBoost 1000C, 10. Battery Adafruit 6600 mAh lithium ion battery pack. (b) and (c) are examples of the final configuration mounted on the HyperRail within the greenhouse.

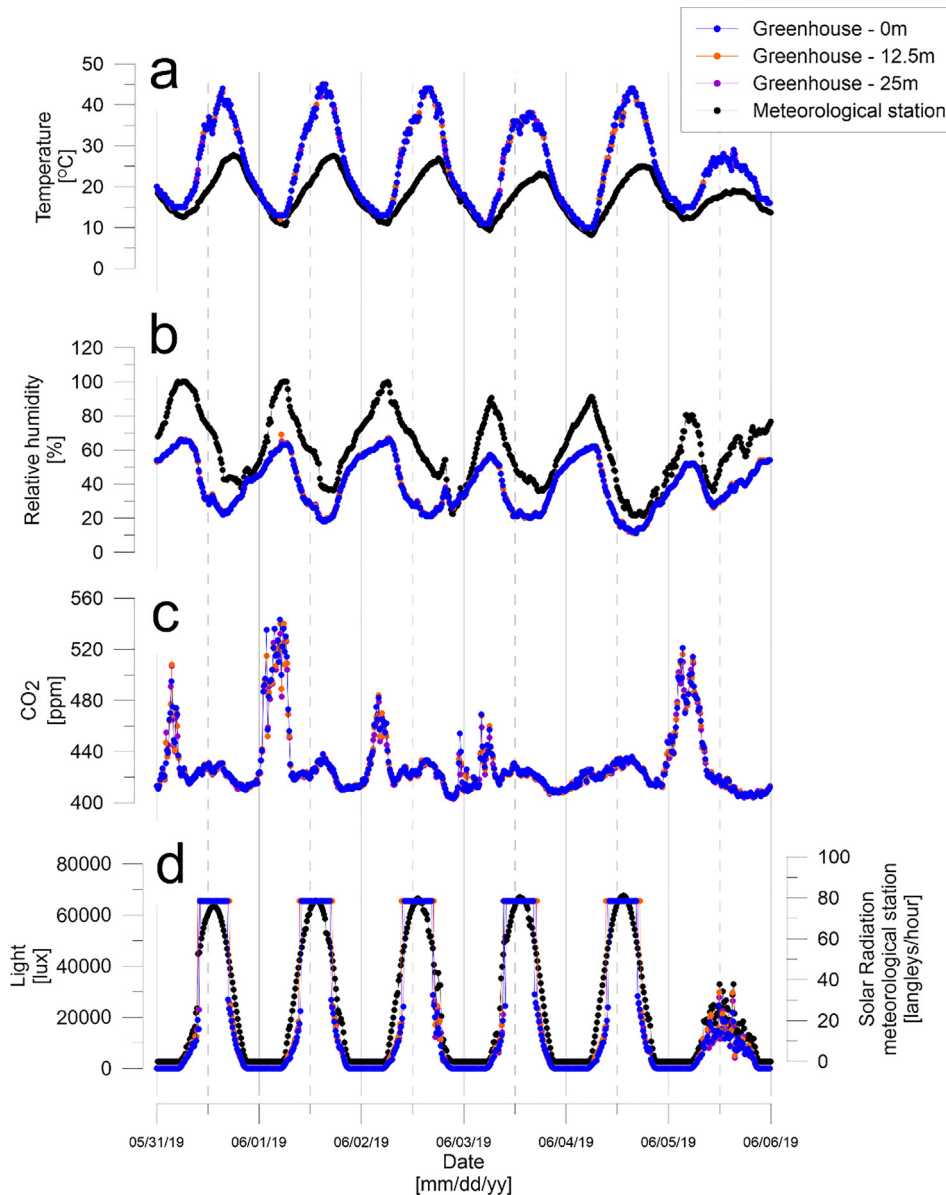


Fig. 75. Time series results from six days in the greenhouse and from the meteorological station situated 200 m from the greenhouse.

house between 0, 12.5, and 25 m (Fig. 76). This is most likely due to changes in local shading at each node or as a result of the dynamic cloud movements. The only exception to the daily trend was in the last day (06/05/2019) where light values fluctuated throughout the day without reaching saturation. This day was the cloudiest day of the week, causing lower radiation as measured also by the temperature sensor (Fig. 75-a). We note that additional settings and versions of the eGreenhouse are described in the OPEnS lab website (<http://www.open-sensing.org/>) and GitHub page (<https://github.com/OPEnSLab-OSU/eGreenhouse>).

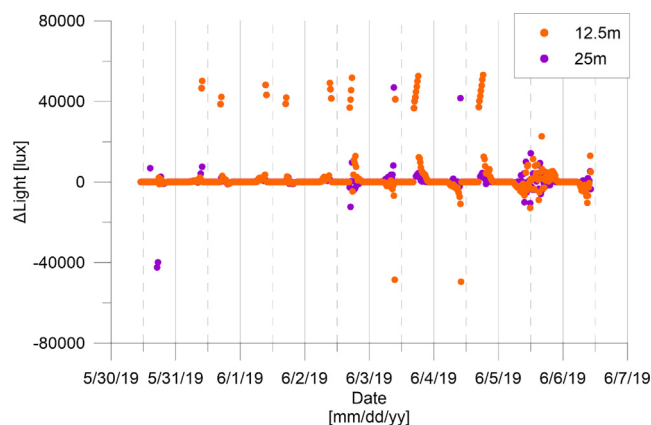


Fig. 76. Lux differences between measurements at 12.5 m (orange) or 25 m (purple) and 0 m.

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

This research was supported by the USDA National Institute of Food and Agriculture, Hatch Project N118HFPXXXXXG055, by the United States – Israel Binational Agricultural Research and Development Fund (BARD), Graduate Student Fellow Award No. GS-37-2017 (granted to EL), and by the Oregon Dept. of Agriculture, Nursery Research Program. We also thank the Oregon Association of Nurseries research committee for providing aspects of technical oversight.

References

- [1] J. M. Lopez Alcalá, M. Haagsma, C. J. Udell, and J. S. Selker, "HyperRail: Modular, 3D printed, 1–100 m, programmable, and low-cost linear motion control system for imaging and sensor suites," *HardwareX*, vol. 6, p. e00081, Oct. 2019.
- [2] T. Nakadai, M. Yokozawa, H. Ikeda, H. Koizumi, Diurnal changes of carbon dioxide flux from bare soil in agricultural field in Japan, *Appl. Soil Ecol.* 19 (2) (2002) 161–171.
- [3] S. Blagodatsky, P. Smith, Soil physics meets soil biology: Towards better mechanistic prediction of greenhouse gas emissions from soil, *Soil Biol. Biochem.* 47 (2012) 78–92.
- [4] J.S. Buchner et al, Evaluation of CO₂ fluxes from an agricultural field using a process-based numerical model, *J. Hydrol.* 361 (1–2) (2008) 131–143.
- [5] X. Kuang, J.J. Jiao, H. Li, Review on airflow in unsaturated zones induced by natural forcings, *Water Resour. Res.* 49 (10) (2013) 6137–6165.
- [6] E. Levintal, M.I. Dragila, N. Weisbrod, Impact of wind speed and soil permeability on aeration time in the upper vadose zone, *Agric. For. Meteorol.* 269–270 (May) (2019) 294–304.
- [7] E. Levintal, M.I. Dragila, T. Kamai, N. Weisbrod, Free and forced gas convection in highly permeable, dry porous media, *Agric. For. Meteorol.* 232 (2017) 469–478.
- [8] E. Levintal, M.I. Dragila, T. Kamai, N. Weisbrod, Measurement of Gas Diffusion Coefficient in Highly Permeable Porous Media, *Vadose Zo. J.* 18 (1) (2019) 1–9.
- [9] S.L. Brown, C.S. Goulsbra, M.G. Evans, T. Heath, E. Shuttleworth, Low cost CO₂ sensing: A simple microcontroller approach with calibration and field use, *HardwareX* 8 (Oct. 2020) e00136.
- [10] C. Toth, G. Józków, Remote sensing platforms and sensors: A survey, *ISPRS J. Photogramm. Remote Sens.* 115 (May 2016) 22–36.
- [11] J. Roldán, G. Joossen, D. Sanz, J. del Cerro, A. Barrientos, Mini-UAV Based Sensory System for Measuring Environmental Variables in Greenhouses, *Sensors* 15 (2) (2015) 3334–3350.
- [12] C. Hummelgard et al, Low-cost NDIR based sensor platform for sub-ppm gas detection, *Urban Clim.* 14 (2015) 342–350.
- [13] E. Levintal, M.I. Dragila, H. Zafir, N. Weisbrod, The role of atmospheric conditions in CO₂ and radon emissions from an abandoned water well, *Sci. Total Environ.* 722 (2020) 137857.
- [14] E. Levintal, M.I. Dragila, N.G. Lensky, N. Weisbrod, Mechanisms Controlling Air Stratification Within a Large Diameter Borehole and Atmospheric Exchange, *J. Geophys. Res. Earth Surf.* 123 (12) (2018) 3251–3268.
- [15] S. Guillon, P. Agrinier, É. Pili, Monitoring CO₂ concentration and δ¹³C in an underground cavity using a commercial isotope ratio infrared spectrometer, *Appl. Phys. B* 119 (1) (2015) 165–175.
- [16] C. Pla et al, Changes in the CO₂ dynamics in near-surface cavities under a future warming scenario: Factors and evidence from the field and experimental findings, *Sci. Total Environ.* 565 (2016) 1151–1164.