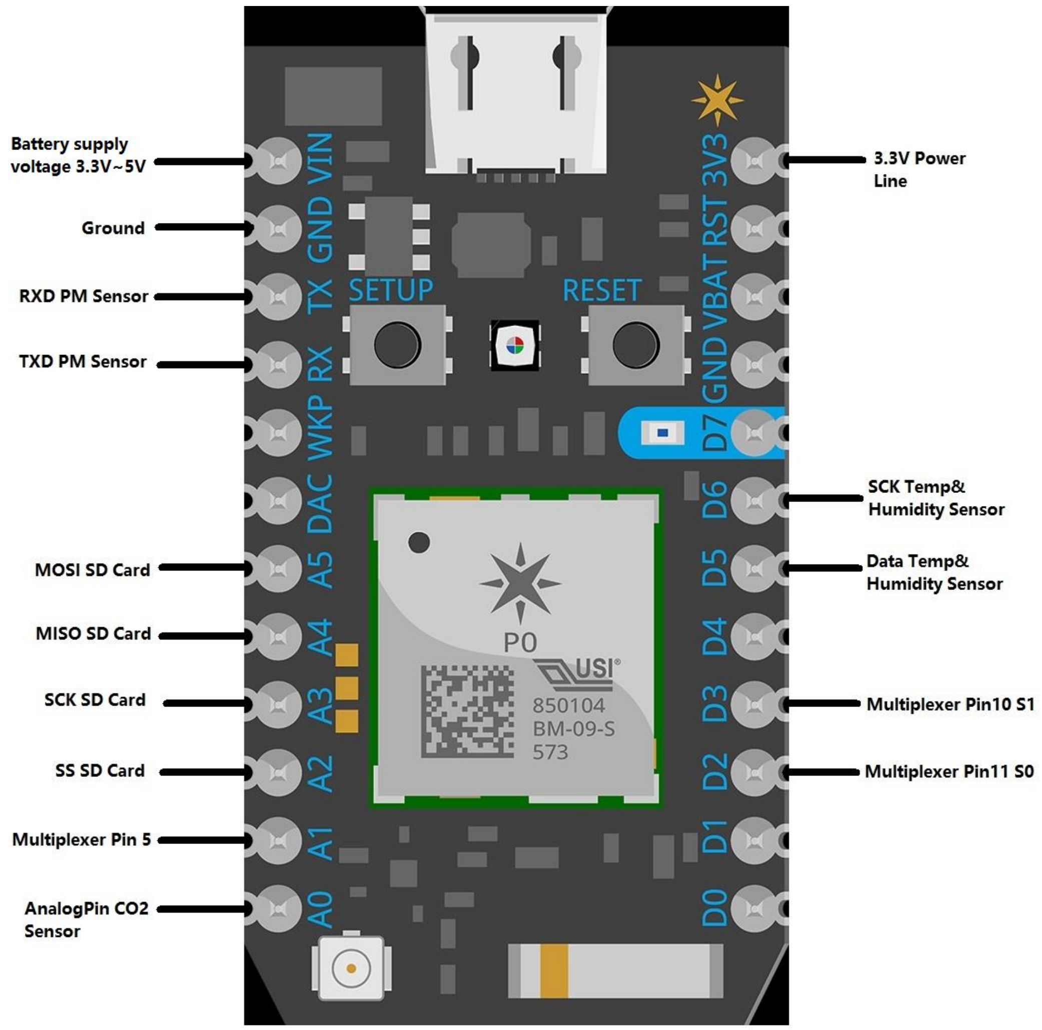
# Particle Photon Air Quality Sensor Package with Wifi

Vera Xu, 06/17/2016, Bergin's Sensor Team

**Introduction: why use Particle Photon?** The Particle Photon is a $19 Wifi development kit with detailed documentation online, straightforward interface and abundance of libraries similar to the Arduino libraries. Not only does Particle has an online dashboard that allows immediate data visualization, it also has a professional IDE called Particle Dev that works similar to the Arduino environment. The Wifi feature allows data login wirelessly without using a SD card or USB connection wire. In this Particle Photon Sensor Package, all sensors on the Teensy Air Quality Sensor Package (PM sensor, CO2 sensor, and Alphasensor, Temperature and humidity sensor) are incorporated, along with SD card to backup data for storage.

## PIN CONNECTIONS FOR PCB DESIGN



For sensors used in the teensy protocol already, the connection for each single sensor should be setup in similar fashion. The external clock is eliminated in the Photon protocol, since Particle Photon has an internal clock that’s pretty accurate and has command to sync time with designated time zone through Wifi. Connection for these sensors are listed in the table below for clarification:

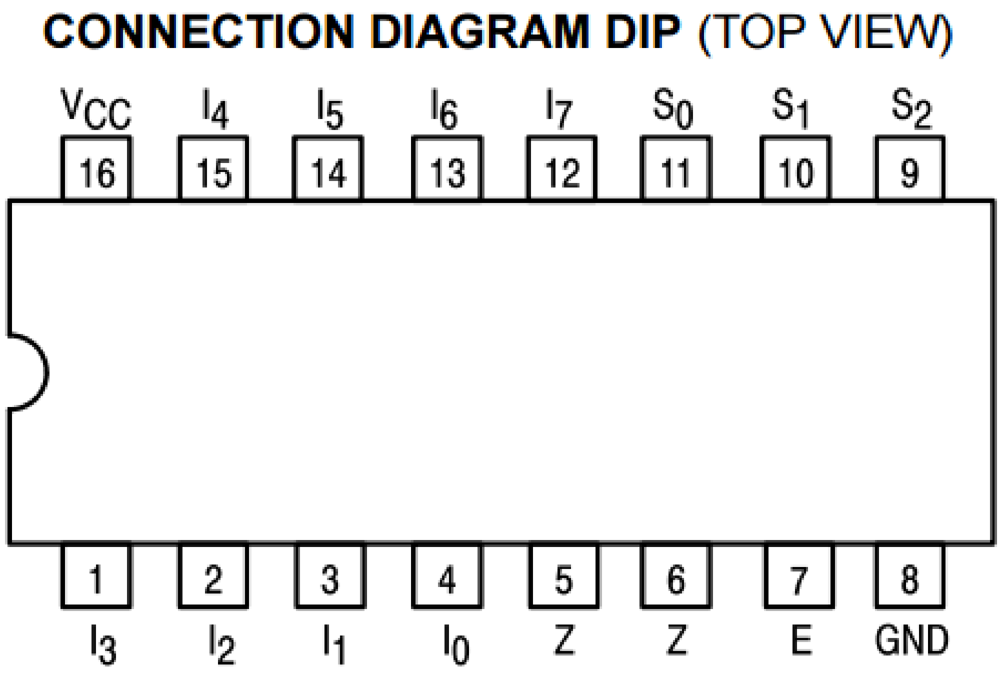
| Component | Sensor Pinout | Connection | Comments |
|---|---|---|---|
| SHT15 Temperature/ Humidity Sensor | 1, GND | GND | |
| | 2, VCC | 3.3V | Supplied by Photon |
| | 3, SDA | D5 | |
| | 4, SCK | D6 | |
| Plantower PM 2.5 Sensor | 1, VCC | 5V | |
| | 2, GND | GND | |
| | 3, SET | | |
| | 4, RXD | TX | Used for Serial1 |
| | 5, TXD | RX | Used for Serial1 |
| | 6, RESET | | |
| | 7/8, NC | | |
| COZIR CO2 Sensor | 1, GND | GND | |
| | 3, VCC | 3.3V | |
| | 5, RX | GND | Through 10k resistor to GND |
| | 7, TX | GND | Through 10k resistor to GND |
| | 9, Analog | A0 | Through 100k resistor to A0 |
| Alphasense NO2 O3 2-Sensor Package | SN1/GND | Pin3 MUX I1 | MUX stands for 8-input multiplexer |
| | SN1/6V | Pin4, MUX I0 | |
| | SN2/GND | Pin1, MUX I3 | |
| | SN2/6V | Pin2, MUX I2 | |
| | VCC | 5V | |
| | GND | GND | |
| SN74LS151 8-input Multiplexer | Pin1, I3 input | Alphasense SN2/GND | WorkO3 |
| | Pin2, I2 input | Alphasense SN2/6V | AuxO3 |
| | Pin3, I1 input | Alphasense SN1/GND | WorkNO2 |
| | Pin4, I0 input | Alphasense SN1/6V | AuxNO2 |
| | Pin5, Z output | A1 | |
| | Pin7, E enable | GND | |
| | Pin8, GND | GND | |
| | Pin 9, S2 select | GND | |
| | Pin10, S1 select | D3 | Digital Binary Select Bits for MUX |
| | Pin11, S0 select | D2 | |
| | Pin16, VCC | 5V | |
| SD Card Holder | GND | GND | |
| | SS | A2 | |
| | SLK | A3 | |
| | MOSI | A5 | |
| | MISO | A4 | |
| | VCC | 3.3V? | It worked with 3.3V but not 5V |

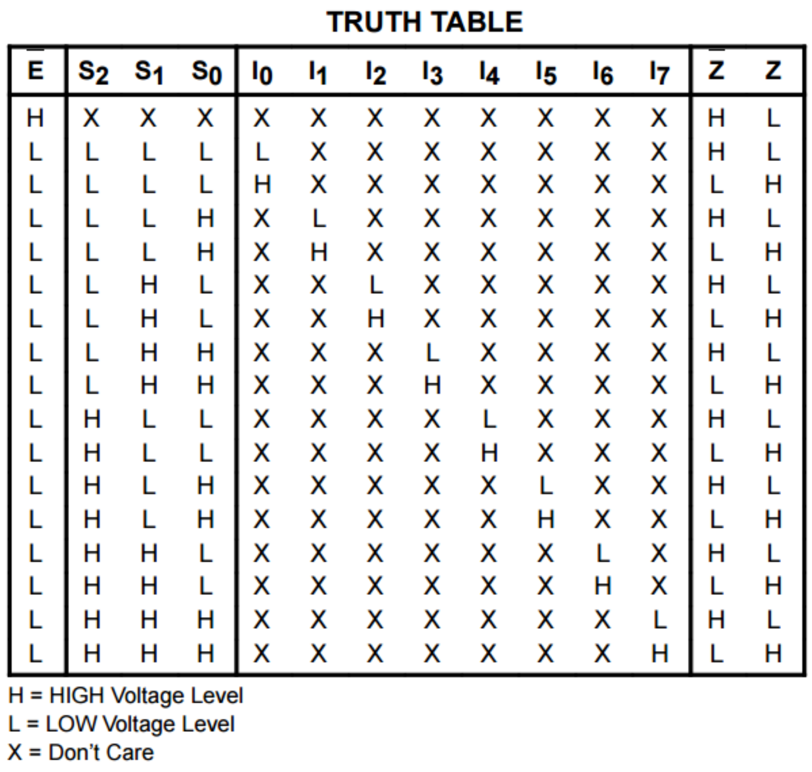
**NEW COMPONENT: 8:1 MULTIPLEXER**

A new component that's incorporated into the Photon design is the SN74LS151 8-to-1 Multiplexer (also called MUX). (Datasheet: http://ecelabs.njit.edu/student_resources/datas/54LS151.pdf). It's a $0.86 digital component used to select one bit of data from up to eight sources. Pin9, 10 and 11 are binary select inputs that can either be HIGH or LOW. Since we only have four Alphasense output to select from, we are not using select pin S2 and it's always set to LOW. Using the two select pins S0, S1, we are able to select from inputs I0, I1, I2, I3, each of which contains data to generate valid NO2, O3 values. The code essentially takes one of these four data each time (loop is fast so they are taken essentially the same time) and thus greatly reduces the analog pin required on the Photon.

The Connection diagram for the 8:1 MUX is included as below. The truth table for selecting inputs is also included that can be used to debug selection. To understand the truth table, "H" means voltage high, "L" means voltage low, "X" means don't-care. Take the second line for example, select pins are all supplied low voltages, which in binary is "000" and select I0 as output. Therefore, Z output is always consistent with I0 when the select pins are "LLL". Similarly, when select pins are "LLH", the output Z will be consistent with I1 pin.

A further step with the multiplexer and Alphasense is to check validity of the NO2, O3 output. Before calibration, it doesn't look like the data output we currently have makes sense. It would be beneficial to look at the data and see if there's a pattern. I'm also a little worried that multiplexer might corrupt the data accuracy. Definitely something we'll need to consider in the fall.





TRUTH TABLE

| $\overline{E}$ | $S_2$ | $S_1$ | $S_0$ | $I_0$ | $I_1$ | $I_2$ | $I_3$ | $I_4$ | $I_5$ | $I_6$ | $I_7$ | $\overline{Z}$ | $Z$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| H | X | X | X | X | X | X | X | X | X | X | X | H | L |
| L | L | L | L | L | X | X | X | X | X | X | X | H | L |
| L | L | L | L | H | X | X | X | X | X | X | X | L | H |
| L | L | L | H | X | L | X | X | X | X | X | X | H | L |
| L | L | L | H | X | H | X | X | X | X | X | X | L | H |
| L | L | H | L | X | X | L | X | X | X | X | X | H | L |
| L | L | H | L | X | X | H | X | X | X | X | X | L | H |
| L | L | H | H | X | X | X | L | X | X | X | X | H | L |
| L | L | H | H | X | X | X | H | X | X | X | X | L | H |
| L | H | L | L | X | X | X | X | L | X | X | X | H | L |
| L | H | L | L | X | X | X | X | H | X | X | X | L | H |
| L | H | L | H | X | X | X | X | X | L | X | X | H | L |
| L | H | L | H | X | X | X | X | X | H | X | X | L | H |
| L | H | H | L | X | X | X | X | X | X | L | X | H | L |
| L | H | H | L | X | X | X | X | X | X | H | X | L | H |
| L | H | H | H | X | X | X | X | X | X | X | L | H | L |
| L | H | H | H | X | X | X | X | X | X | X | H | L | H |

H = HIGH Voltage Level
L = LOW Voltage Level
X = Don't Care

## HOW TO SETUP PHOTON UNDER WIFI CREDENTIAL

1. With mobile app Particle (easiest method):
   a. Download the Particle App from Apple Store
   b. Press SETUP button on the photon until it starts blinking blue (listening mode)
   c. Open mobile app, press on the "+" sign on the top right to add device
   d. Select Photon
   e. Go to Wifi setting on your phone, connect to network Photon-XXXX and go back to app
   f. The app will go through a set of auto setup processes with light changing color on Photon
   g. Make sure the photon is breathing cyan (connected to Wifi) in the end

   Good things to know about Wifi network: while setting up Wifi, the setup might ask about the network you want to connect to.

   h. If you are on campus at Duke, use the DUKE public Wifi. Its security level is "unsecured". Do not use DUKEBLUE. It's a new Wifi extra secured network and requires registration that Photon can't do. To use DUKE Wifi on Photon, you need to register on https://dukereg.duke.edu/ with the MAC address of the device. To obtain MAC address of the Photon, put the device into listening mode while connected to computer with USB, press "m" in the terminal or command line portal and the address will show up. The computer and the Photon don't need to be connected under the same Wifi credential. (more information about connecting to Wifi on the Particle website https://docs.particle.io/support/troubleshooting/troubleshooting-tools/photon/#display-mac-address)

   i. If you are using a home Wifi, it's easier to setup than any of the Duke Wifi. Make sure you have your Wifi username and password at hand, use the mobile app and follow the prompt. It's pretty self-explanatory.

2. With Particle CLI command line interface: well documented here on the Particle website https://docs.particle.io/guide/getting-started/connect/photon/ https://docs.particle.io/reference/cli/#particle-setup-wifi
3. Documentation about what different colors of light on photon means is documented here. https://docs.particle.io/guide/getting-started/modes/photon/

## HOW TO CONNECT PHOTON TO WIFI ONCE CREDENTIALS HAVE BEEN SET UP

1. Power Photon with either USB cord or power supply of 3.6V-5.5VDC
2. Check to make sure the Photon is breathing cyan
3. Debug tricks if not connected: due to the complexity of Duke's network, Photon shows unpredictable behaviors under the Duke network especially. It was suggested by the Colab that it might work better under simpler network. For some reason, Duke's network might be rejecting Photon's connection and that's probably why sometimes Photon has such difficulty connecting.
   a. Reconnect power supply
   b. Press on the SETUP for 3 seconds repeatedly until Photon starts blinking green (connecting to Wifi) rapidly. Give it a few sec to react.
   c. Keep it connect via USB and wait.
   d. Hold down both SETUP and RESET for 3 seconds and release RESET button until the Photon starts blinking magenta, then release the SETUP button. This put Photon into safe mode. Then re-upload the code and see if the Photon will automatically connect to Wifi. (More info about safety mode here https://docs.particle.io/guide/getting-started/modes/photon/)
   e. Enter DFU mode and do DFU factory reset. (DFU means firmware upgrade. This method is not recommended due to its complexity, and I recommend use it as the last resort. I got some help from the Colab, but a Photon is truly broken, DFU won't even fix it. Explore it here https://docs.particle.io/reference/firmware/photon/#dfu- )

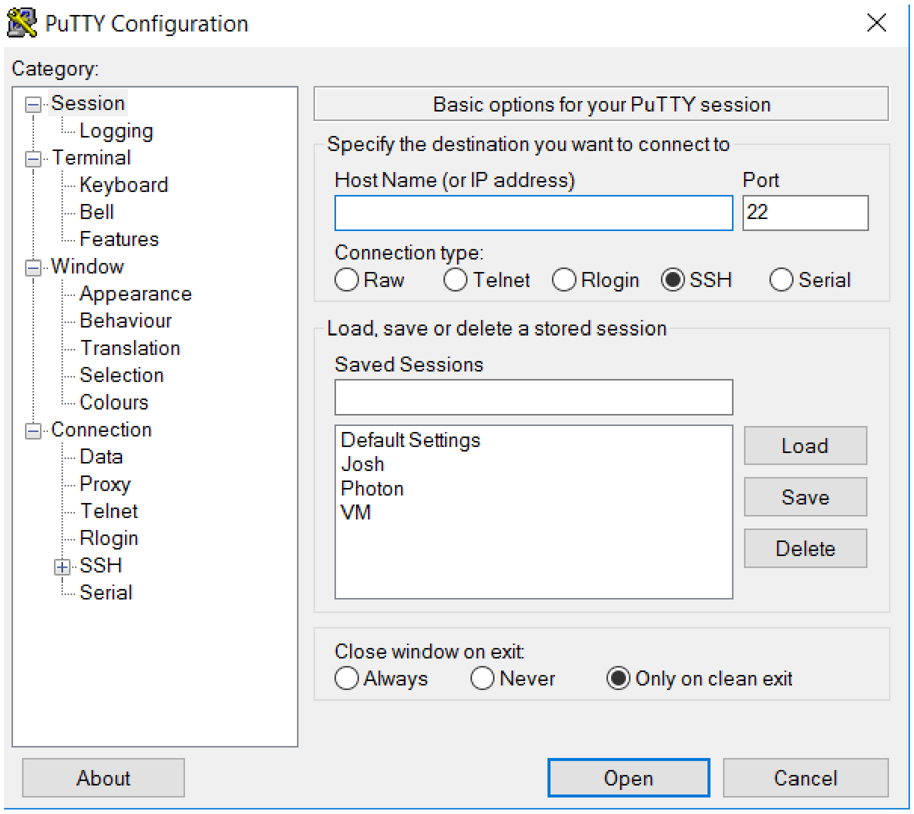
## HOW TO UPLOAD CODE TO PHOTON

1. Open Particle Build website (https://build.particle.io/build/new) and here is the IDE to write code and flash code to the device.
2. All codes I've written are in Lucy's account. "PHOTON" is the main code with collecting data, logging into SD card and FTP. "FTP" code is to read SD and FTP files in SD card to Colab's server. "SDTEST" is a file to read data from a certain file in SD card and print content to serial monitor. The other files are just my test files, which can be ignored.
   **Username: spothorse9.lucy@gmail.com    Password: BerginLabs**
3. Make sure you log in with the same account that you logged in on your mobile app when you connected your Photon to Wifi
4. Make sure your Photon is working under the same Wifi credential that you registered with, or you will have to setup the Photon to the new Wifi credentials with the mobile app again.

5. Go to the device manager on the left column. Your device is connected to the IDE if there is a small cyan circle on the side of your device name.
6. When multiple devices are present, click on the left of the device you want to use. A yellow star will show up, which mark the device that you want to upload the code to.
7. Write your code. Verify code using the checkmark button on the top left corner and flash code to Photon using the flash button on top left.
8. The Photon will start blinking magenta and then go back to breathing cyan. On the bottom of the Particle IDE, a prompt will tell you if the flash is successful.

## HOW TO MONITOR SERIAL OUTPUT FOR DEBUGGING

1. Download Putty.exe
2. Open Device Manager on your computer, press the "scan for hardware changes" button (most of the time this process is automatic)
3. Under Ports (COM & LPT), there will be something named Photon (COM…). If it's not showing Photon, right click on the device that shows up and click "update driver software". Take note of the correct COM number.
4. Open PuTTY. Select **Serial** and write "**COM...**" under the host name. COM should be followed by the exact same number you took note of in the last step, for example, "COM4". The COM number differs from device to device, make sure you change the COM number if you change to another Photon device.
5. You can save a login session by typing a name in the "Saved Session" box and press Save. You can load an earlier session by double-click on the session name you saved.
6. This works the same way as the automatic serial monitor in the Arduino IDE.


PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions:

Default Settings
Josh
Photon
VM

Close window on exit: Always Never [✓ Only on clean exit]

## USEFUL TOOLS TO WORK WITH PHOTON

1. **Particle dev:** a professional IDE developed for Particle based on Github's Atom project. It incorporates the Serial monitor features and also allows user a summary of all the functions and variables created. It would be a great interface to use if it's all setup. I'm not using this interface because I didn't want to get through all the trouble of setting up libraries. I use the web IDE, since all the community developed libraries are there. To use the libraries locally, you will have to download them from GitHub and put them all in the correct folder.
2. **Tera term / Putty:** used for serial monitoring, command line control. They both works and works in similar fashion. Read the section "How to monitor serial outputs" above for details.
3. **Particle website:** particle build, particle dashboard. The dashboard allows you to check data published real-time, anything with the command "Particle. publish ("name", "data")". However, the dashboard is a little slow and takes a couple seconds to react after uploading the code. I switched from dashboard to serial monitor to improve efficiency of my time.
4. **IFTTT:** useful tool to check real-time published data and transfer it to Google doc or Google spreadsheet. (Explained in later session). IFTTT transfer data from one interface to the other. The received end can be anything like Gmail, GitHub, Google Drive, etc.

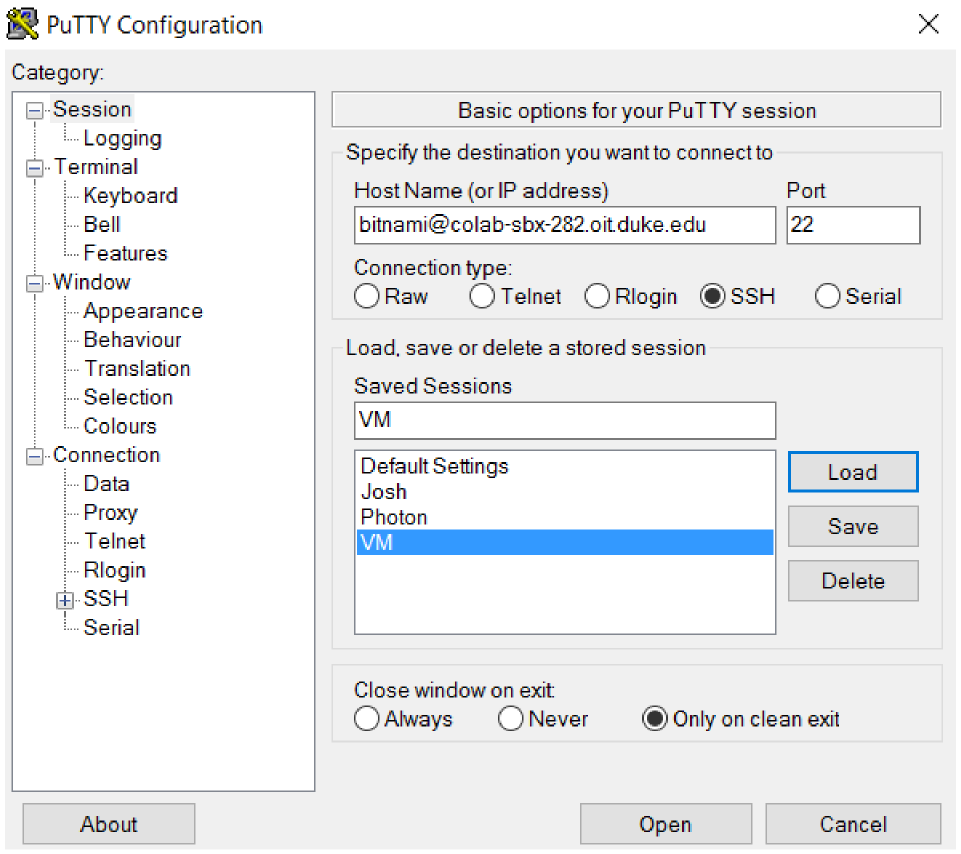
## HOW TO RECEIVE REAL-TIME PHOTON DATA

1. Use Particle.publish ("Event name", "Data") for the data you want to publish. Note that it can only publish up to 255 bytes of data once, in form of ASCII characters. "Data" need to be a string. String concatenation can be used to form a string that stores information of the data.
2. An external software, like Matlab or R studio can be used to translate data back into human-readable strings. However, I haven't figured out the way to do it in Particle Photon IDE.
3. Go to IFTTT website (https://ifttt.com/recipes), register with any Google account and click My Recipes and then create new recipe.
4. Choose from Particle to Google spreadsheet.
5. Set up the trigger to an event name of your choice which is consistent with the event name in your Particle. publish ("name", "data"). Leave the event contents empty. That's to say whatever data you publish, as long as it's called the same thing, it will be recorded in your Google spreadsheet.
6. Formatted row: only with Event Contents.
7. Now upload your Photon codes once more. On the dashboard, you will get notification that an external software is getting the data from you.
8. Check your data in Google spreadsheet.

## HOW TO SET UP COLAB'S UBUNTU SERVER WITH FTP

We are currently running the FTP server on the Duke CoLab Virtual Machines (VM). Due to the fact that these VMs are not intended to host large scale production service, it doesn't have enough storage space for both the FTP server and Shiny to be installed. Right now, the CoLab's service is simply used to receive FTP files and store them. In the future, we could consider contacting OIT to see if they have similar server service available.

1. The Server is setup on the lab's computer already. This tutorial is just in case if server needs to be setup again. The server can be accessed on multiple computers. To access the server that is setup already, go to section "How to access server and check files"
2. Go to Duke's VM Manager online tool (https://oit.duke.edu/comp-print/labs/vcl/). Log in with NetID and password. Click "Get your own VM". This create a virtual machine that can be used on your computer.
3. Only one VM is granted for each student. Create a new VM if you do not have one already.
4. Choose Ubuntu 14 with nothing else installed to it. We will install FTP server by ourselves.
5. Click "How to access your VM" and copy the address after ssh.
6. In your PuTTY, choose SSH and paste the address you just copied. Enter the password provided on the VM website. The cursor will not move when you type in password.
7. The following command will helps you setup the server:
   - ifconfig (Getting IP address and update the one in the Photon FTP code)
   - sudo apt install vsftpd (Install FTP server on the Ubuntu)
   - sudo nano /etc/vsftpd.conf (This will open up a text editor. Uncomment the WRITE_ENABLE and use ^X to exit the editor. Make sure you save the document after making changes).
   - sudo adduser particle (Add another user called "particle" and type in "photon" when the prompt asks for the new password)
   - sudo chmod 777 /home/particle (This grant the new user "particle" access to the root)
   - sudo restart vsftpd (Make sure the FTP server is running again)



Specify the destination you want to connect to

Host Name (or IP address): bitnami@colab-sbx-282.oit.duke.edu

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

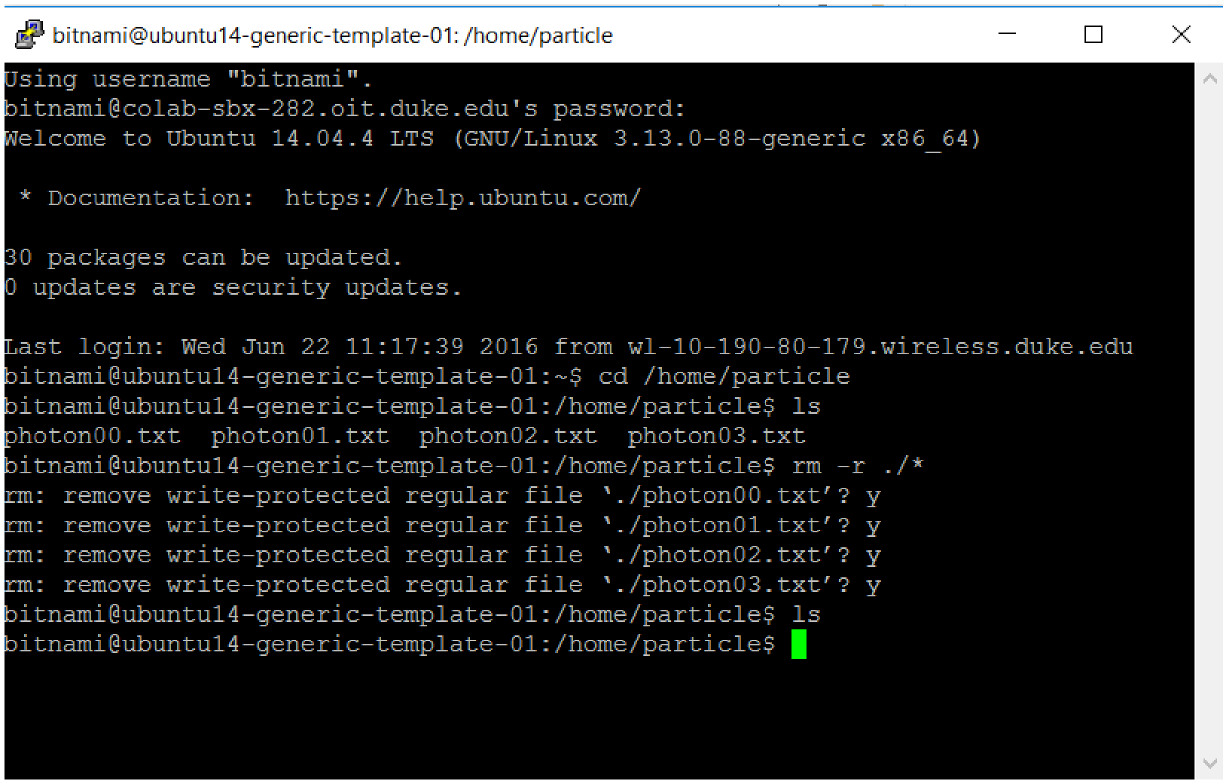
Saved Sessions: VM

Default Settings
Josh
Photon
VM

Close window on exit: Always Never [✓ Only on clean exit]

## HOW TO ACCESS SERVER AND CHECK FILES

1. Go to PuTTY and type in the host name, port number and connection type exactly as above.
2. Again, you can save sessions using the save button below.
3. Click open or double click on a saved session that you would like to open.
4. The command line interface will pop up, as shown below. This window is how you will interact with the server (with Ubuntu operating system).
5. Type in password: tUlabreer5 (The cursor will not move. Just keep typing!)
6. To change current directory to path /home/particle: cd /home/particle
7. List all items in the current directory: ls
8. As shown below, there are originally 4 txt files in the directory. We can use the following command to delete all the files in the directory: rm –r ./*
9. To remove a selected file from the directory: rm photon00.txt
10. Now type in ls again. There are no files in the directory anymore.
11. The FTP server will put all the files it receives in the folder /home/particle. Every hour it will put in a new file named "Photonxx".



```
Using username "bitnami".
bitnami@colab-sbx-282.oit.duke.edu's password:
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 3.13.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

30 packages can be updated.
0 updates are security updates.

Last login: Wed Jun 22 11:17:39 2016 from wl-10-190-80-179.wireless.duke.edu
bitnami@ubuntu14-generic-template-01:~$ cd /home/particle
bitnami@ubuntu14-generic-template-01:/home/particle$ ls
photon00.txt  photon01.txt  photon02.txt  photon03.txt
bitnami@ubuntu14-generic-template-01:/home/particle$ rm -r ./*
rm: remove write-protected regular file './photon00.txt'? y
rm: remove write-protected regular file './photon01.txt'? y
rm: remove write-protected regular file './photon02.txt'? y
rm: remove write-protected regular file './photon03.txt'? y
bitnami@ubuntu14-generic-template-01:/home/particle$ ls
bitnami@ubuntu14-generic-template-01:/home/particle$ 
```

## FTP FILE TRANSFER PROTOCOL EXPLAINED

The photon development community incorporated the FTP library that can be easily used to transfer files. A discussion about best way to transfer large amount of data with Photon is quoted here https://community.particle.io/t/best-way-to-transfer-large-data-from-photon/15442

The FTP.ino in the Particle Photon website is the modified version of this FTP libraries that can be used to quickly transfer text files over Wifi. To test the FTP.ino file and understand more about FTP, several things need to be changed:

- IP address has to match the IP address of the CoLab server (which can be found using ifconfig in the Ubuntu command line)
- filename can be changed to any file name of your choices, as long as the file is present in the SD card (if not, the command line will prompt an error message)
- client.println("USER particle") and client.println("PASS photon") are the username and password that the Photon uses to access the CoLab's server. No permission to access files will be granted if the username and password don't match up. In the FTP.ino code here, it's better not to change anything here.
- doFTP() function is used to do the actual file transfer.
- readSD() function can be used to print contents from the SD card on the serial monitor.

## STEPS MOVING FORWARD

1. To figure out a way to transfer files from colab's server to our server/our computer.
2. There's a hardware problem with SD card that I didn't have enough time to figure out. SD card functions properly when the 5V power supply is not plugged in, but when I tried to plug in the power supply, it shuts down and can't be read by the Photon. I would recommend going through the hardware and see if I wired anything wrong. It's a weird problem because the 5V power supply is not even wired to either the Photon or the SD card holder. It's only used to supply 5V voltage for certain sensors, such as Plantower PM Sensor and Alphasense.
3. Make sure the multiplexer is not corrupting any data for the Alphasense.
4. Implement checksum for PM values, etc.
5. Debug and make sure all the sensor outputs make sense.
6. Make some sample PCB board for the Photon.
7. Calibrations and come up with easier calibration algorithms as the ones we wrote for the Teensy.
8. The Colab's server is always running. If time allows, we can try to integrate the Shiny server with the Colab's server to generate a real-time website, which allows us to check the data anywhere, anytime.