Purpose

The purpose of this project is to create a system to be able to monitor wind turbines without needing to physically be at the turbine. This will be done by installing a system at the turbine which is capable of measuring various fields, such as power output and usage. This device would then transmit this information to a website which can be viewed from anywhere. This enables us to have more information about a turbine and allows us to know when to perform maintenance without relying on the accounts of the turbine owner. Our purpose in continuing this project is to experiment with the viability of using a Particle Electron as the base device instead of both an Arduino and a Raspberry Pi.

A Particle Electron is basically an arduino that is packaged in a way to streamline the process of transmitting data from devices over the internet. It has the same pins as an arduino and is coded in basically the same way and we will be using the same sensors that were used in the previous design that used an arduino. The difference is that the Particle Electron has a built in sim card. There is also serverside software that is available for use with the Particle Electron to help streamline the process of pushing data to a website.

Materials

All particle electron models can be found here: https://store.particle.io/collections/electron

We bought and used the asset tracker for our experimentation. The asset tracker is a 2G particle electron with the removable addition of a gps. This was because we had originally considered transmitting coordinates along with the rest of our data. We now think that this is redundant and unnecessary and for future monitors we think it may be advantageous to buy one of the other, cheaper models. As stated above we were using a 2G Particle Electron. It may be best to simply use the 2G (global) model because of the fact that it is cheapest. However if it is found that there are connectivity problems at the installation site then it would be a worthwhile investment to get the 3G model.

We also purchased an anemometer for measuring the wind speed at the location of the turbine from https://www.adafruit.com/products/1733 as well as a current sensor http://www.robotshop.com/en/pololu-30a-acs714-current-sensor.html. In order to power the current sensor we need a constant 5volts while the Electron can only supply 3.3. The simplest solution would be to use this: https://www.sparkfun.com/products/12009, but we did not have one on hand so we instead created a voltage regulator and will be supplying the voltage from the battery instead of the Electron.

Setup

Login information:

https://www.particle.io:

Email: volunteer@windaid.org

Password: Wind365!

https://thingspeak.com/:

Email: volunteer@windaid.org

Password: Wind365!

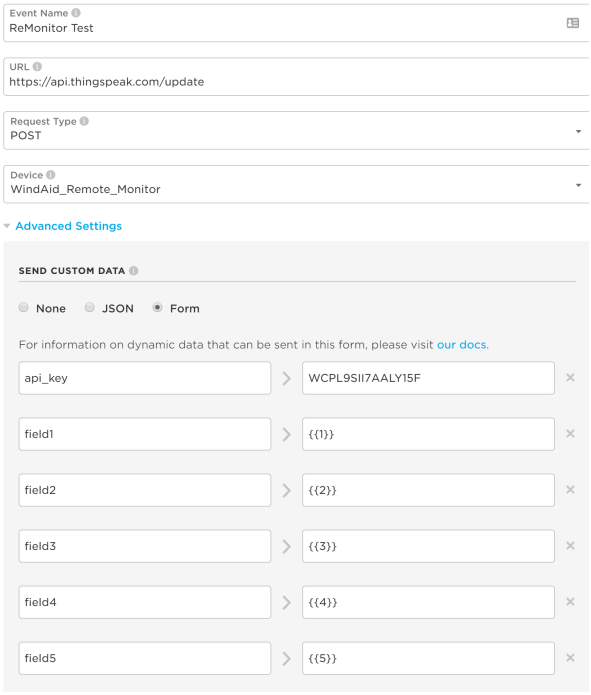
Most information about the particle can be found here: https://docs.particle.io/guide/getting-started/intro/electron/

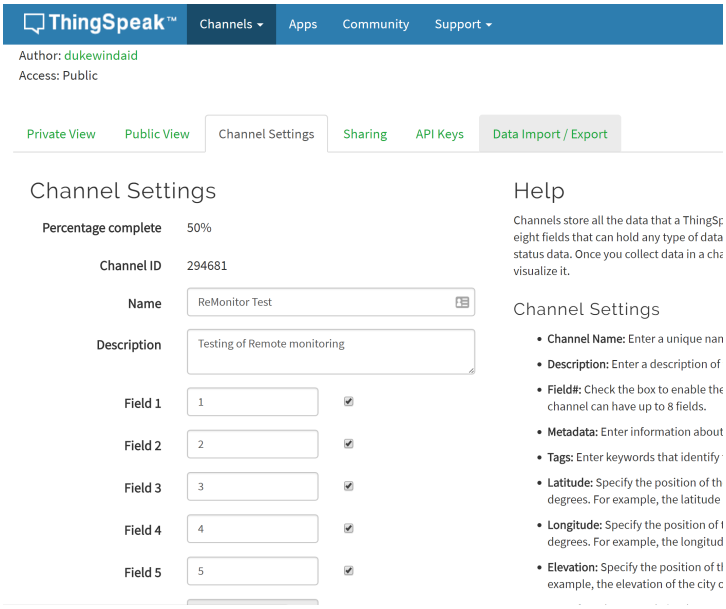
First thing you have to do is set up your device. This can be done at https://setup.particle.io/ and requires making/using a WindAid account. All devices can be registered to the same Windaid account which makes deploying code to all of them much easier. There are many ways to get code to the Particle. The method we have been using is writing the code in this online IDE(Integrated Development Environment): https://build.particle.io/build/new. From there you have the capability to push the code over the air to the Electron by clicking on the lightning bolt icon in the top left. This uses cellular data so while this may be useful for when the device is deployed at the turbine, we have been using another method for experimentation in the shop. We have using the command line interface as outlined here:

https://docs.particle.io/guide/tools-and-features/cli/core/

This involves installing the CLI, downloading the firmware binary, then pushing it by opening the command line and running `particle flash --serial firmware.bin`.

We are using https://thingspeak.com/ to collect and display our data. The process of setting this up is outlined here: https://docs.particle.io/guide/tools-and-features/webhooks/. This requires going to the particle console (https://console.particle.io/integrations) and creating a new webhook. Our webhook and thingspeak are pictured below:





Then to push the data you have to writing the code: Particle.publish("WEBHOOKNAME", data, PRIVATE);

In this case WEBHOOKNAME is ReMonitor Test and data is a string declared as

String data = String::format(

```
"{ \"1\": %d, \"2\": %d, \"3\": %d, \"4\": %d, \"5\": %d }",
wind, current, voltage, load_current, load_voltage);
```

We have done it in this manner where are variables are named 1,2,3,4,5 as to minimize data usage when pushing this data over cellular data.

Our current code (not finished) can be found here: https://pastebin.com/0sSLvkei

Hardware

Beyond the anemometer, the two main sensors that need to be constructed are a voltmeter and a current sensor. There are a variety of guides online for how to build simple, yet effective voltmeters using only a breadboard, an arduino, and a couple resistors. Here's how we did it.

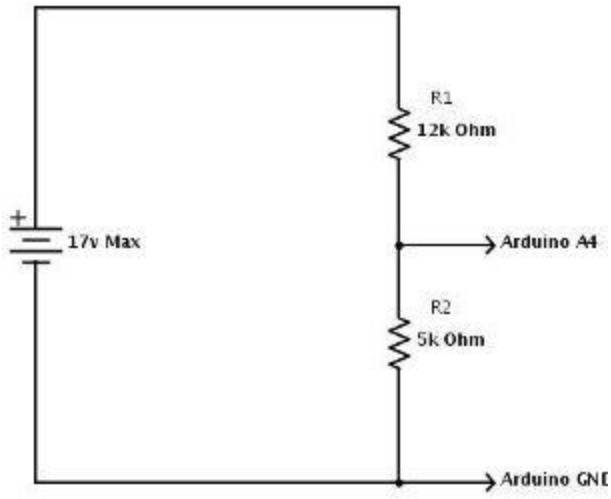
Voltmeter

There is a nice guide online which we followed, which can be found at: https://hacktronics.com/Tutorials/arduino-current-sensor.html

Essentially, the circuit used is a voltage divider. A voltage divider takes an input voltage and outputs a fraction of it to be read. Using a ratio calculated from the circuit itself, we can use code in the Electron to calculate the input voltage coming in from the battery.

We use a voltage divider because the turbine powers a 12 V battery, but the max voltage that the Particle can read is 3.3 V, so we need to lower the voltage to a level that can be measured and recorded.

Here is the circuit for sensing the voltage:



Some calculations must be performed to determine the best resistor values to use. Even though the voltage will come from a 12 V battery, we used 15 V in our calculations to account for voltage spikes

that may occur. We also used 3 V output voltage for the Particle to further minimize the chances of damaging the Particle.

Calculations for resistor values:

Using Ohm's Law, it is easy to calculate the resistor values needed. We choose 100 μA arbitrarily as the current value to streamline the calculations.

$$V = IR_{total}$$

$$R_{total} = \frac{V}{I}$$

$$R_{total} = \frac{15\,V}{100\,\mu A}$$

$$R_{total} = 150\ \text{k}\Omega$$

Now that we found the total resistance necessary for the circuit, we need to calculate the individual values of R1 and R2. Substitute the value for R_{total} and solve for R1. We also know that the voltage out can be calculated by this equation $V_{out} = \frac{R2}{(R1 + R2)} * V_{in}$

$$R_{total} = R1 + R2$$

Rearranging, we get:

$$R2 = \frac{V_{out}}{V_{in}} * R_{total}$$

$$R2 = \frac{3\,V}{15\,V} * 150\,k\Omega$$

$$\text{R2} = 30\ \text{k}\Omega$$

$$\text{R1} = 150\ \text{k}\Omega - 30\ \text{k}\Omega$$

$$R1 = 120\ k\Omega$$

Now we have two resistor values of R1 = 120 kΩ and R2 = 30kΩ.

When you construct the circuit, use a multimeter to measure the input and output voltages to determine the proper ratio to use in the code to calculate the actual input voltage. The ratio is found by this formula:

$$\text{Ratio} = \frac{V_m}{V_{out}}$$

On our circuit, we measured V_{in} to be 12.32 V from the DC power supply and V_{out} to be 2.43 V going into the particle.

$$\text{Ratio} = \frac{12.32\ V}{2.43\ V}$$

$$\text{Ratio} = 5.06995$$

Anemometer

We used this anemometer (https://www.adafruit.com/product/1733) and followed ths guide (http://www.geeky-gadgets.com/arduino-wind-speed-meter-anemometer-project-30032016/) for the anemometer. The anemometer is relatively easy to set up. It simply needs to be set up in a high place where it can accurately measure wind speed. Then a three phase cable can be used to bring the three output down to the controller. The anemometer requires 7-24VDC to operate so we connected the brown and black wires to the battery/controller for power. Brown wire to power and black to ground. The blue outputs a voltage from .4v to 2v corresponding to wind at 0m/s and 32.4m/s. You simply have to use a

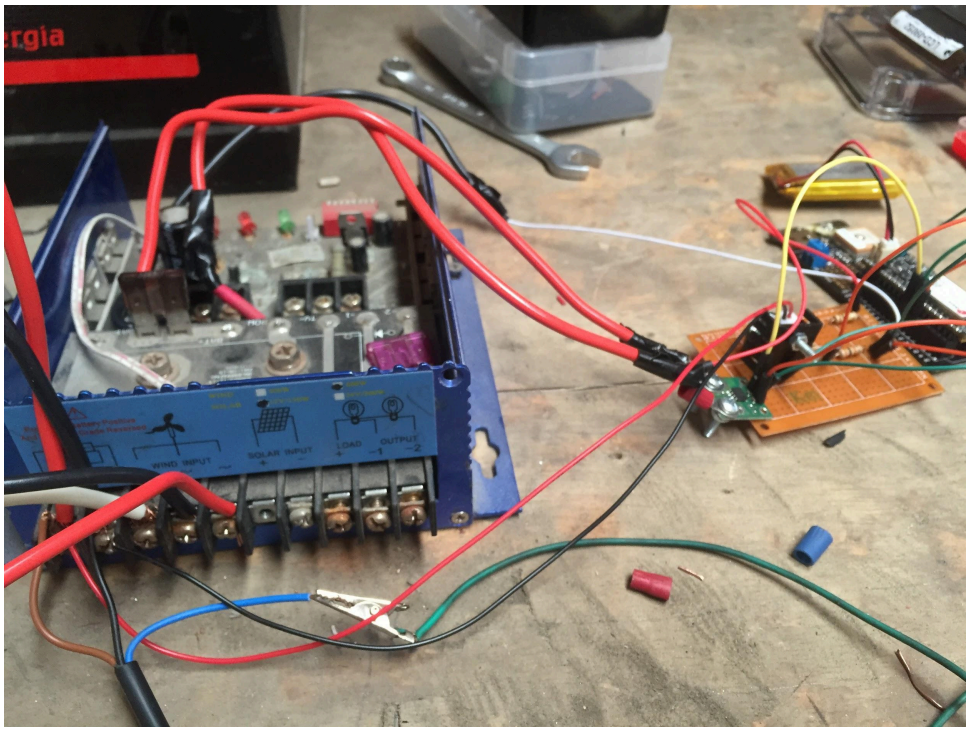
linear regression to estimate the wind speed for any voltage within that range and assume no wind if the voltage is below that range.

Current Sensor

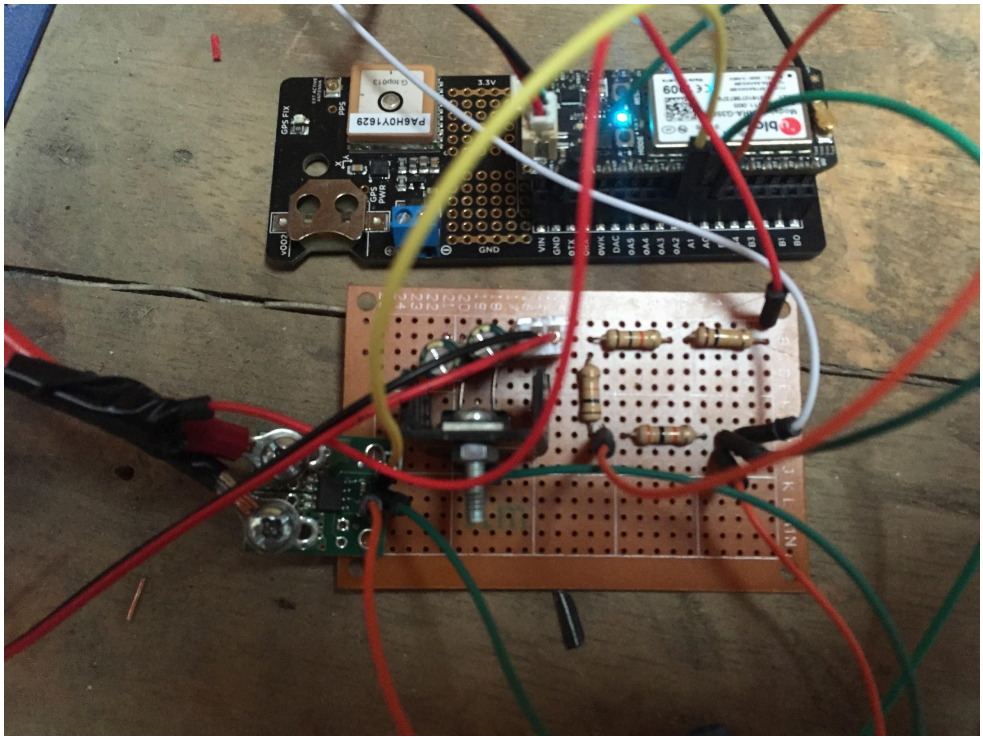
We are using this (http://www.robotshop.com/en/pololu-30a-acs714-current-sensor.html) current sensor and used (https://www.pololu.com/product/1187) for more information. This is an invasive current sensor that must be spliced into a current carrying wire. In addition to the current that it is measuring the sensor also requires a constant 5VDC to operate. This can be accomplished with power from the electron with https://www.sparkfun.com/products/12009. Unfortunately we could not find this part in peru so we are building a voltage regulator and getting out power straight from the battery/controller. The sensor also has a pin for an output voltage which is connected to the Particle. This voltage is 2.5v when there is no current passing through the sensor and changes by .066V/A as the current changes. So since the sensor is rated at ±30A that means that the output voltage will range from .5v to 4.5v. Because the electron can only measure up to 3.3V, we chose to connect the current carrying wires backwards. So the positive current that we are measuring would normally make the voltage range from 2.5V to 4.5V but by reversing the direction we will have voltages of 2.5v to .5v which is more easily measured by the electron.

Again, all the code for all of this can be found here https://pastebin.com/0sSLvkei

Understanding our design



In this photo, you can see our remote monitor system connected to the controller for the turbine in the shop. To monitor the voltage and current in the controller, all that needed to be done is extend the positive red wire found inside the controller to our system, and connect it back to ground. You can see a bit of that here. The red wire is coming out of the controller into our monitor, and a white pin connected to a black cable grounds the monitor to the negative side of the controller.



As you can see, our remote monitor is still in the testing/prototype phase. In the picture, you can see the Particle Electron right above our breadboard. On the right side of the breadboard, we installed our voltage divider. You can see that it is powered by the red wire that connects to the same wire that is connected to the positive end of the wind turbine controller. The current goes through a 100 kΩ resistor and two 10 kΩ resistors before reaching the orange wire connector. This orange wire goes to the A0 pin on the Electron. This is the pin that measures the voltage. Following that, there is another 30kΩ resistor before reaching a green wire, which is connected to ground on the Electron, and a white wire, which is connected to the negative terminal in the turbine controller.

Installing the current sensor was a bit tricker. We needed to install a voltage regulator to supply 5V to the current sensor. This was a simple circuit consisting of a diode, two 100 µF capacitors, and a 7805 linear regulator. The regulator is attached to a heat sink to dissipate any heat that is produced. Using the black and red connector wires, this circuit takes the 12V from the battery and converts it to a 5V that the sensor can use. As described previously, the current sensor is an invasive current sensor, so it needed

to be connected in series with the positive wire from the turbine controller. On the sensor itself, the orange wire is connected to ground, the green wire is connected to 5 V from the linear regulator, and the yellow wire is connected to pin A2 on the Electron.

Moving Forward

Our initial plan was to measure 5 variables remotely but we only had time to measure 3. We were going to measure wind speed, current, voltage, load voltage, and load current and we only have sensors for the first three. Fortunately, it is very simply to measure the load voltage and load current. It only requires duplicating our current sensors for current and voltage and placing them on the battery and on the cable to the load. Our sensors are currently located in the controller to measure the voltage and current actively being generated by the turbine.

If it is found that the Electron is using too much data to push data every 10 minutes, it is possible to implement a system of continuously monitoring data and bulk updating a ThingSpeak channel. The bulk update APIfor thingspeak can be found here (https://www.mathworks.com/help/thingspeak/bulk-update-a-channel-feed-1.html). The code for the Electron would have to be reworked a decent amount to collect and store the data before pushing. Currently there is no code for storing the data on the device as it just pushed it immediately after collecting it.

We are currently powering the electron through a small LiPo battery. To install the Particle in a remote location, another source of power would have to be used. The Particle is able to accept up to 12v of power on its Vin pin. This would allow the particle to be powered by the battery on site.

We also ran out of time to get our current sensor working. We haven't been able to do enough testing to determine the problem but we suspect that we just have a faulty sensor and should use another. There is another model of current sensor in the electronics room that most likely can be used. This is just an area that needs experimentation and testing.