Portable Wireless Weather Station

Author: Marco Branson Baca

Abstract:

In this project, the goal was to build a weather station, powered by a battery that would be charged by a solar panel, that could read the rain amount that's falling, wind direction, wind speed, temperature, and humidity of any location the station was at while also being able to do both: transfer that data wirelessly to some external source and track the motion of the sun. With a set of specific materials given to us along with any materials to our disposal, we could automate and build the weather station.

In the end we accomplished the main goal but weren't able to make the solar panel track the sun. Our weather station was able to read the input data and generate proper values for all sensors and transfer this data to an outside radio. Even though we had a code setup for the solar panel to track the sun, it wouldn't actually rotate to follow the sun's direction.

Introduction:

Any one person can build a weather station with a kit. An Argent Data Systems Weather Station or ADS-WS1 kit consist of sensors for wind direction, wind speed, and rain amount. These sensors are wired and can be connected to some computer-based circuit board that can be programmed to read the values of the sensors. The company also provides a weather station library to read their specific sensors using an Arduino circuit board. For temperature and humidity we used a DHT11 sensor.

An Arduino is a circuit board that can be programmed to build many electronic based designs. Using the Arduino IDE and the libraries within it, one can create and upload a code to the Arduino which will command it to run a certain algorithm. Since the ADS-WS1 kit sensors send out variating voltages or voltages at some rate, as they have built in resistors to output a

particular value when connected to a power supply, you can program an Arduino to read and forward these values.

A solar panel uses the energy it receives from the sun to power external objects. It does this by allowing photons to hit it which knock electrons off and generate a current and it also uses photovoltaic cells to convert solar energy into electricity. In this project we used a solar panel which outputted 75W when exposed to sunlight which charged a 12V battery.

We were given sensors and electronic materials to achieve this goal which included the anemometer, temperature and humidity sensor, rain gauge, and wind vane, radio, battery, solar panel, and Arduino. The wind vane sensor had resistors in it which would send out voltage values to the Arduino indicating its orientation, the rain gauge and anemometer had triggers that would interrupt its output voltage letting the Arduino know to increase the count.

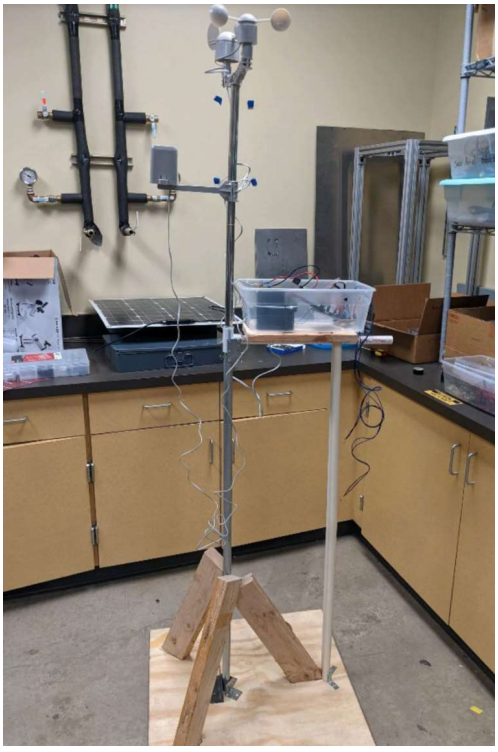
Since weather varies on location, and the weather we get on the news covers a whole city, we sometimes get weather information that isn't accurate to our direct location. By making a circuit that connected all the ADS-WS1 kit sensors and radio to an Arduino which connected to the battery and developing a code that would read the sensor values that were sent to the Arduino ports and to an external radio, we built a portable wireless weather station. With this we could test in real time the difference in weather at one spot compared to that of the general area (city).

Methods:

Structure: Our goal was to build a stable structure that would allow for the station to function properly without any damage to the devices. For the base we used a wooden platform with a tripod built on it that would be the basis for the support of the beam. Using metal an PVC

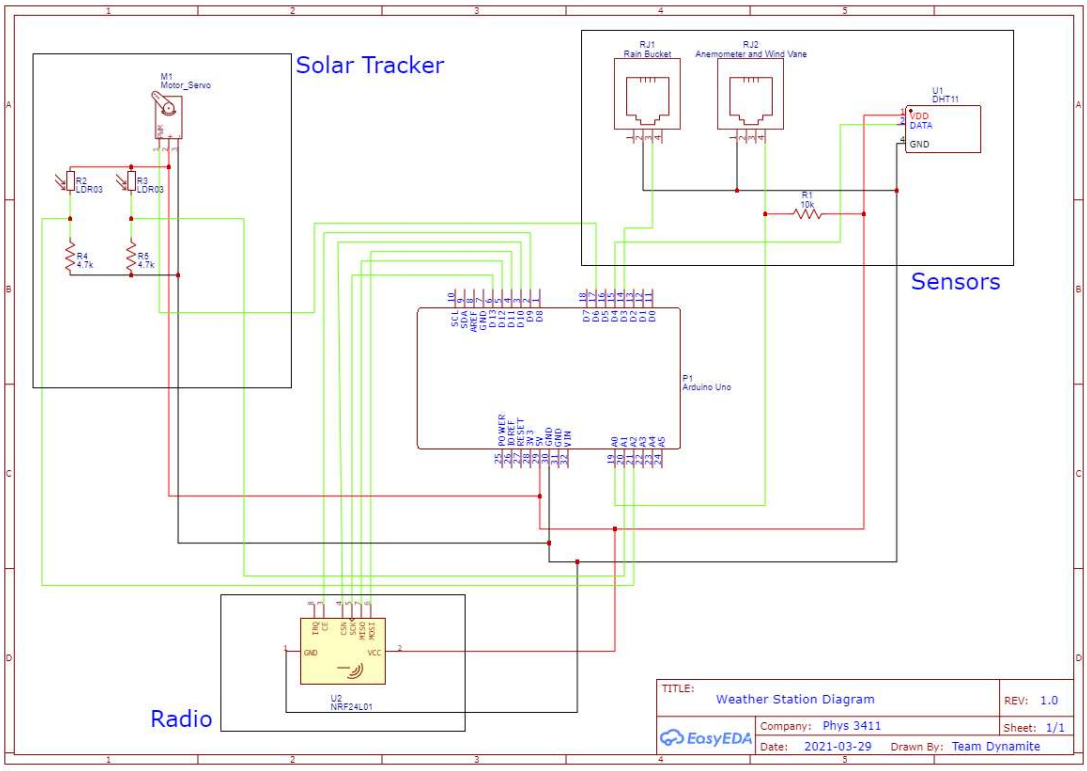
pipes we could lengthen our build and have all the materials at the top including the Arduino, sensors, battery, and solar panel. We made a small platform extended out so we could place our whole circuit. In the end by using screws, wood, and metal and PVC pipes we made a stable structure that would allow for the station to properly function.

Here we have the design of how we setup the structure of our weather station.

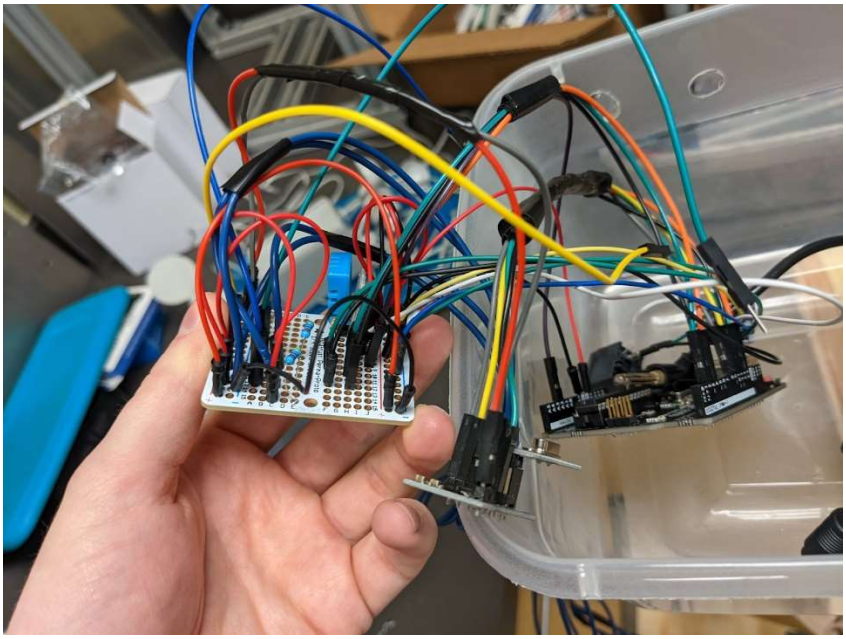


This is an image of how we assembled everything together. The solar Panel would go directly over the plastic box at an angle.

Circuit: With a breadboard and Arduino, we were able to combine smaller circuits for different parts of the station into one big circuit. Using a soldered breadboard, we were able to connect all the wires and solder them so they would be stable and no wire would fall out of place.



This is a circuit design that specifies what ports each individual sensor connected to in the Arduino.



This is the soldered breadboard which has all the wires in the picture soldered onto it.

Code: In the code, we had to compile an algorithm which could run the following functions at the same time, read all the sensor values, convert the sensor values to the actual real-world values, output the new values, use radio communication to send these values, and setup functions to have the solar panel track the sun. The company that made the kit for the weather station also developed a library (ADSWeather) that had embedded functions that would read and convert these values. By using their functions along with functions we made and included, we were able to interpret the sensor values. For the temperature and humidity sensor we used the DHT library. Like the ADS library, this had built in functions to read the DHT11 sensor. The radio code consisted of three different libraries (SPI.h, nRF24L01.h, RF24.h). Since we could only send data of one type between radios, we stuck with the character type (“char”). This was optimal since we wanted to not only send values, but a worded message along with it. We developed functions that would convert our “string” and “int” type variables into “char” types which would allow for the data to transfer. To make the solar panel track the sun we made a

function that would read a two photoresistor on opposite ends of the solar panel, if one read a greater value it would make the motor move towards that direction. All these functions that were used were properly arranged to compile cleanly and nested in different loops including void, if, and while or set in global or void setup.

Finished_WeatherStation_Code

```
#include <ADSWeather.h>
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <dht.h>
#include <Servo.h>

#define ANEMOMETER_PIN 2
#define VANE_PIN A0
#define RAIN_PIN 3
#define DHT11_PIN 4
#define CALC_INTERVAL 2000
#define CE_PIN   9
#define CSN_PIN 10
```

This image is part of the code. Here we include our libraries and declare which pin each sensor connects to by defining them.

Results:

We took our data on Monday March 29 at around 3:30 pm and according to timeanddate.com their recorded values for the average weather data at 3 pm in San Marcos Texas was the following:

Wind Speed (mph)	Wind Direction	Rain Amount (milli inches)	Temperature (Celsius)	Humidity (%)
9	North	0	24	30

Our data updated once every two seconds and for multiple recordings, we got the following:

Wind Speed (mph)	Wind Direction	Rain Amount (milli inches)	Temperature (Celsius)	Humidity (%)
1.4	North	0	25	20
2.9	South West	0	25	21
4.4	South West	0	25	21
5.9	West	0	25	21
7.4	South West	0	25	20
4.4	West	0	25	20
4.4	South	0	25	20
2.9	North	0	26	21
2.9	North West	0	26	21
5.9	West	0	26	21
4.4	North	0	26	21
4.4	North West	0	26	21
4.4	North West	0	26	21
2.9	South West	0	26	20
2.9	South West	0	26	20

For our averages over the period that the station was recording data we had the following:

Wind Speed (mph)	Wind Direction	Rain Amount (milli inches)	Temperature (Celsius)	Humidity (%)
4.1	North West - West	0	25.5333	20.533

Discussion:

According to timeanddate.com, at 3pm the average wind speed in all of San Marcos came out to 9 mph due north. In our data we get an average wind speed of 4.1 mph usually in the direction of North West to West. On this day, it wasn't raining, so there was no rain amount for both our averages. Our average temperature was slightly over the temperature that timeanddate.com had. Our average value for percent humidity was around 10% lower than the average for timeanddate.com.

There could be many reasons for all these discrepancies. To begin with, timeanddate.com has an average of the whole city while we measured our values on the Texas State University campus outside of the RFM building. Because we were at one spot, we could expect to see different fluctuations in values.

We can't determine the degree to the error in our weather station, whether there was a lot or barely any, because we have no exact value that we can look at to compare it too. Our weather station outputted values as a function of the environment it was around so being in different locations around San Marcos would result in outputs of different values.

Besides the values, our weather station was properly transmitting the data it received from the sensors to an external radio. The structure seemed to hold up fine for the time that we had it outside except for the wind occasionally blowing the solar panel out of place. Sometimes some of our wires would get detached which would cause us to have to reattach them or it would lead to a short in our circuit. Because our photoresistors were parallel they would both receive the same amount of sunlight for any given direction of the sun, this results in not having the solar panel rotate to follow the sun.

Conclusion:

In the end, we were able to build a portable weather station that communicated wirelessly and read the humidity, wind speed, wind direction, rain amount, and temperature of the outside environment. By using many components of the Arduino IDE we successfully programed our Arduino board to read and output the sensor values it received. We were unsuccessful in making the solar panel track the sun, but this wasn't one of the required fields we had to do. It was hard to determine the error in our experiment since we had no certain values for our measurements in the location we took them at. For next time, we could do the following: first, we could attach the wires more firmly so they don't fall out of place and second, instead of taking values for a short period of time, we can leave the station outside and register the daily average of values and compare those to the daily average of an online source.

Sources:

Weather in March 2018 in San Marcos, Texas, USA (timeanddate.com)

2.4 GHz Wireless Communication Between Two Arduinos - Circuit Basics

nRF24L01 - How It Works, Arduino Interface, Code, Schematic (howtomechatronics.com)

Arduino Wireless Communication Using NRF24L01 : 4 Steps - Instructables

In-Depth: How nRF24L01 Wireless Module Works & Interface with Arduino (lastminuteengineers.com)

How to Set Up the DHT11 Humidity Sensor on an Arduino (circuitbasics.com)

GitHub - argent-cape/ADSWeather: Arduino library to interface with the Argent Data Systems weather station sensor assembly

ALLPOWERS 50W /100W 18V 12V Monocrystalline Flexible Solar Panel – AllPowers (iallpowers.com)