ESCUELA DE INGENIERIA

INGENIERIA CIVIL ELECTRICA

DISEÑO, CONSTRUCCION E IMPLEMENTACION DE UNA ESTACION METEOROLOGICA DE BAJO COSTO BASADA EN ARDUINO

ANIBAL IGNACIO CERDA OLEA

PROFESOR GUIA: RAUL VALENZUELA RUIZ

MEMORIA PARA OPTAR AL TITULO DE INGENIERO CIVIL ELECTRICO

RANCAGUA, CHILE

FEBRERO 2022

Dedicatoria

A mis padres, que se empeñaron en entregarme todas las herramientas para que estudiara una carrera universitaria. Los cuales me guiaron durante este camino y de los cuales aprendí que con esfuerzo, perseverancia y dedicación se pueden lograr grandes metas.

A mi pareja Nicole, una gran mujer, la cual me ha acompañado y motivado durante todo este proceso. A quien admiro mucho y quien me ha ayudado a ser una mejor persona, como también, a perseguir y anhelar nuestros sueños.

A mi hermana, la cual pasamos excelentes momentos de ocio y compañía. Espero que esto sea un ejemplo y un punta pie inicial para que persigas y puedas lograr tus metas y objetivos.

A mi familia por el enorme apoyo durante todo este proceso.

Y a los que partieron, que desde su lugar nos acompañan y guían.

Agradecimientos

Quiero agradecer a:

A mi profesor guía, Raúl Valenzuela por proponerme este proyecto, además por el apoyo y la ayuda recibida durante este proceso.

A la escuela de ingeniería de la Universidad de O'Higgins y a la Fábrica Digital O'Higgins por financiar este proyecto.

Al profesor Miguel Torres, que me facilito un proyecto de estación meteorológica y de la cual se pudieron obtener valiosos componentes.

A Carlos Vera y al personal de mantención de la Universidad de O'Higgins, los cuales facilitaron e instalaron el gabinete que alberga la estación meteorológica junto con la estructura que lo sostiene.

Al equipo de Dirección de tecnologías de información (DTI) de la Universidad de O'Higgins, los cuales habilitaron un punto de red en la azotea del edificio A, el cual permite tener una conexión estable a internet para poder realizar un monitoreo de las variables medidas por la estación meteorológica Arduino.

Al equipo y encargados de la Fábrica Digital O'Higgins por facilitar el uso de sus dependencias, herramientas y prestar ayuda técnica cuando fue requerida.

A los miembros de la comisión evaluativa, los profesores Miguel Torres y Daniel Casagrande por resolver dudas e inquietudes durante este proceso.

A mis amigos y compañeros, por el apoyo y buenos momentos vividos durante todo el proceso universitario.

Finalmente, quiero agradecer a todos los profesores y profesionales que me entregaron valiosas herramientas y enseñanzas durante este largo camino.

Índice

Resumen

El proyecto busca diseñar, construir e implementar una estación meteorológica de bajo costo basada en la plataforma Arduino Mega, que permita realizar mediciones y obtener datos de las siguientes variables meteorológicas: temperatura, humedad del aire, presión atmosférica, precipitaciones, intensidad y dirección del viento de un determinado lugar. La estación meteorológica cuenta con acceso a internet, lo que permite él envió de las mediciones hacia un servidor web en donde se almacenan los datos y también, la visualización de estos de forma remota. Además, la estación meteorológica cuenta con un sistema de alimentación autónomo compuesto por paneles solares y baterías.

El trabajo realizado consistió en un diseño de la estación, en donde se definió el hardware y software a utilizar. Se implemento el hardware seleccionado en primera instancia por separado, sensor a sensor, lo que permitió que la implementación final se ejecute de manera sencilla. Se realizaron cuatro pruebas de funcionamiento, las cuales consisten en verificar que las mediciones de las variables meteorológicas se realizarán de forma correcta, además de la monitorización de las mediciones correspondientes al sistema de alimentación. Como resultado de estas pruebas, se decidió agregar un panel solar adicional, debido a que el sistema de alimentación en las primeras pruebas no lograba realizar la carga de la batería, adicionalmente se agrega otra batería con el fin de aumentar la capacidad de almacenamiento de energía. Durante el transcurso de las pruebas, se evidencian efectos que disminuyen la producción de energía por parte de los paneles solares, los cuales son la presencia de nubosidad y de incendios forestales en las cercanías del emplazamiento, los cuales provocaron efectos en la carga de las baterías. Finalmente se agrega un modo de ahorro de energía en Arduino, el cual permitió encontrar una configuración definitiva para la estación meteorológica, la cual se mantuvo funcionando de forma correcta durante todo el periodo de prueba.

Palabras claves: Estación meteorológica Arduino, Estación meteorológica de bajo costo, estación Arduino, Estación meteorológica autónoma, Estación meteorológica Arduino Mega

1. Introducción

1.1. Descripción del problema

Este proyecto, nace bajo la necesidad de construir una estación meteorológica de bajo costo que posea características similares a una estación meteorológica automática, con el fin de poder realizar mediciones y llevar un registro de estas, para la generación de datos meteorológicos de un determinado lugar o zona. Con esta idea en mente, se buscan distintas formas de realizar el proyecto utilizando diferentes plataformas de desarrollo. Tras una revisión bibliográfica, se escoge la plataforma Arduino debido a las prestaciones que posee, el nivel de desarrollo que permite realizar un proyecto como este sin inconvenientes, el bajo costo de la placa junto a los accesorios y lo más importante es que es una plataforma de código abierto, permitiendo en un futuro que este proyecto sea una base para que cualquier persona pueda seguir perfeccionándolo.

1.2. Alcances y usos de una estación meteorológica basada en Arduino

Una estación meteorológica basada en la tecnología Arduino, que sea capaz de medir variables meteorológicas, cuenta con un gran abanico de posibles usos. Por ejemplo, para el área agronómica, se puede enfocar en la monitorización del tiempo atmosférico en donde se tengan diferentes cultivos, sobre todo para cultivos que requieran cuidados extras y que se encuentren en zonas en que las temperaturas son muy bajas [1] . Se puede utilizar en viveros para controlar las condiciones óptimas que requieren las plantas que cultivan [2].

Logrando mediciones de varios años, se puede utilizar para medir los efectos provocados por el cambio climático, con la ventaja de que al ser de bajo costo y autosustentable energéticamente, se pueden instalar diversas estaciones meteorológicas en

distintas zonas o puntos de interés, entregando datos mucho más significativos y en tiempo real.

Además del área agronómica, una estación meteorológica basada en Arduino puede facilitar la toma y validación de datos meteorológicos con el fin de ayudar en la realización de estudios sobre temas contingentes como la sequía, cambio climático o el avance del desierto de atacama hacía el sur de nuestro país.

Otra área importante en donde una estación meteorológica de bajo costo puede resultar de gran utilidad es en el área energética, más específicamente en el estudio, implementación y análisis de rendimiento de parques eólicos. En esta área, las estaciones meteorológicas cuentan con un gran aporte debido a los análisis que se pueden realizar con la obtención de datos del comportamiento del viento en las zonas que se pretenden implementar nuevos parques o como se mencionó anteriormente, evaluar el rendimiento de plantas que se encuentren en funcionamiento. Además de la generación eólica, una estación meteorológica de bajo costo puede ser utilizada para medir el rendimiento de parques fotovoltaicos o de arreglos solares, a través de la medición del voltaje y corriente producidos por los paneles, además de monitorear temperatura, humedad e intensidad de luz [3].

El alcancé de este proyecto es poder medir, realizar un registro y analizar las siguientes variables meteorológicas: temperatura, humedad del aire, presión atmosférica, precipitación, intensidad y dirección del viento, además de variables relacionadas al sistema de alimentación autónomo que posee la estación meteorológica Arduino, las cuales son voltaje, corriente producidos por el panel solar y el voltaje de la batería.

2. Objetivos

2.1. Objetivo general

Diseñar, construir e implementar una estación meteorológica de bajo costo basada en Arduino que permita medir y registrar valores de temperatura, humedad, presión atmosférica, precipitaciones, intensidad y dirección del viento de un lugar en específico.

2.2. Objetivos específicos

- Describir las variables meteorológicas que la estación medirá.
- Realizar un diseño definiendo el hardware y software a utilizar.
- Diseñar e implementar un sistema de alimentación autónomo que le entregue a la estación meteorológica la energía necesaria para su correcto funcionamiento.
- Definir un emplazamiento para la estación meteorológica Arduino.
- Construir e implementar la estructura que albergará al hardware y sistema de alimentación seleccionado.
- Implementar y validar el correcto funcionamiento del hardware seleccionado.
- Analizar el funcionamiento del sistema de alimentación seleccionado.
- Analizar los valores de las mediciones realizadas por la estación meteorológica Arduino.

3. Marco teórico y estado del arte

3.1. Conceptos

Para poder comprender de mejor forma las mediciones que se realizarán en la estación meteorológica de bajo costo, en el siguiente apartado se definirán las siguientes variables meteorológicas: temperatura, humedad, precipitaciones, viento y presión atmosférica. También se hablará sobre la forma en que se miden comúnmente, finalmente se abordará sobre la manera en que una estación meteorológica automática puede realizar las mediciones de estas variables.

3.1.1. Temperatura.

La temperatura es una propiedad física de un sistema. El término se utiliza para identificar la medición de energía cinética que compone un sistema. Existen diferentes escalas de temperaturas, algunas son la escala Celsius (centígrados, utilizados en gran parte de los países), la escala Fahrenheit (usada en Estados Unidos) y la escala absoluta o Kelvin. El sistema internacional de unidades (SI) reconoce al Kelvin como unidad de temperatura y a la escala Celsius como unidad derivada y relativa a $273{,}16\,K$. Por lo que existe una similitud entre la escala Kelvin y la escala Celsius, esta es, que una diferencia de temperatura o variación es correspondiente entre una escala y la otra [4].

En la escala Kelvin, existe un valor inferior y uno superior en los cuales se fijan referencias. El valor de referencia inferior se le conoce como cero absoluto el cual corresponde a la temperatura a la cual las moléculas de un sistema no poseen energía cinética, mientras que el superior corresponde al punto triple de agua o $273{,}16°K$ que es donde los tres estados del agua (solido, líquido, gaseoso) conviven.

En nuestro país, la escala que se utiliza asociada a la temperatura es el Celsius. En esta escala, el cero corresponde al punto en donde el agua pura se congela (estado sólido) a nivel

del mar, mientras que el cien corresponde al punto en donde el agua entra en su punto de ebullición (estado gaseoso) [4].

Existe una correspondencia entre las tres escalas mencionadas. A continuación se podrá observar dicha correspondencia en base a la escala Celsius:

$$T[°K] = T[°C] + 273{,}16 \quad (1)$$

$$T[°F] = T[°C] * \frac{9}{5} + 32 \quad (2)$$

La forma tradicional de medir la temperatura es a través de un termómetro de líquido. Este termómetro consta de un tubo de vidrio graduado en una de las escalas mencionadas anteriormente, en un extremo posee un bulbo que contiene un líquido el cual a medida que aumenta la temperatura se expande pudiendo registrar la temperatura. Se utilizan dos tipos de líquidos los cuales pueden ser mercurio o alcohol. El primero permite medir temperaturas desde los -39°C mientras que el segundo puede medir temperatura desde los -62°C.

Para poder registrar la temperatura mínima y máxima se utilizan dos termómetros, comúnmente llamados termómetro de mínima y termómetro de máxima. El termómetro de mínima utiliza alcohol debido a que su punto de congelación es menor en comparación que el de mercurio. Para realizar la medición, este se instala de forma horizontal. En su interior posee un vástago con forma de mancuerna, al aumentar la temperatura el alcohol fluye por alrededor de la mancuerna dejándola en una posición fija. Cuando desciende la temperatura, el menisco[1] del alcohol arrastra a la mancuerna hacia la parte de abajo del termómetro lo que permite registrar la temperatura mínima. El termómetro de máxima utiliza mercurio. Este líquido al aumentar la temperatura se expande desde el bulbo pasando por una zona de menor tamaño que la columna por donde se expande, al momento de disminuir la temperatura, el líquido se contrae desde la zona de menor tamaño hasta el bulbo quedando marcada la expansión

[1] Se conoce como menisco a la curvatura de la superficie de un líquido que se produce en respuesta a la superficie de su recipiente. Esta curvatura puede ser cóncava o convexa, dependiendo de la interacción de las moléculas del liquido con las del recipiente, estas se pueden atraer (agua y vidrio) o repeler (mercurio y vidrio).

máxima del líquido y permitiendo registrar la temperatura máxima alcanzada. Este termómetro se restablece mediante agitación mecánica [5].

La forma en que una estación meteorológica automática mide la temperatura del aire es a través de sensores electrónicos. Los sensores de temperatura que se utilizan en estaciones meteorológicas se clasifican según el principio físico que utilizan, algunos de estos son:

- Expansión térmica
- Termoeléctricos
- Resistencia eléctrica
- Capacitancia eléctrica

Estos sensores convierten la salida (valor medido) en una señal de voltaje, luego se trabaja con ese voltaje y se convierte a un valor deseable (valor en °C). [5]

3.1.2. Humedad.

La humedad es la cantidad de vapor de agua que contiene el aire [6]. Para expresar el contenido de humedad en la atmósfera se utilizan diferentes parámetros, los cuales son: presión de vapor, humedad absoluta, razón de mezcla, humedad específica, humedad relativa y temperatura del punto de rocío. Se ahondará en la humedad relativa debido a que la mayoría de los sensores miden este tipo de humedad.

La humedad relativa (HR) corresponde al resultado de la división (cociente) entre la masa del vapor de agua contenido en un volumen dado de aire y la masa de vapor de agua que contendría ese mismo volumen si estuviera saturado. Este cociente se expresa en porcentaje en donde 0% corresponde a aire seco y 100% corresponde a aire saturado.

La forma convencional de medir humedad es mediante el uso del higrómetro. Existen dos tipos de higrómetros, los mecánicos y los basados en electrónica. Los mecánicos se basan en la propiedad de algunos materiales de expandirse o contraerse dependiendo de la humedad que presenta el aire. Se utilizan cabellos humanos, sedas, algodones, etc. Se elige uno de los

materiales mencionados anteriormente y se fijan en la parte superior mientras la parte inferior sujeta un peso, este peso se conectada a un sistema que se mueve en una escala e indica la humedad relativa. Los higrómetros basados en electrónica absorben moléculas de vapor de agua que modifican las propiedades eléctricas de los materiales que componen el sensor, lo que se traduce en una señal eléctrica proporcional a la humedad [5].

Las estaciones meteorológicas automáticas utilizan higrómetros basados en electrónica, específicamente utilizan sensores que miden la humedad relativa del ambiente. Estos sensores generalmente además de realizar la medición de humedad pueden medir temperatura.

3.1.3. Precipitaciones.

La precipitación corresponde a la cantidad de agua caída en forma de lluvia o nieve en un plano horizontal en un intervalo de tiempo. Generalmente se mide en las unidades $\left[\frac{mm}{h}\right]$ o $\left[\frac{mm}{dia}\right]$. Existen instrumentos que miden solo lluvia mientras que otros pueden medir la cantidad de lluvia caída o nieve [5]. Una acumulación de X mm de precipitación significa que cayó un volumen de X litros de agua por metro cuadrado de superficie.

La forma tradicional para medir la precipitación o cantidad de agua caída es a través de un pluviómetro. Un pluviómetro se compone de una probeta graduada la cual recibe el agua que proviene de un embudo, el cual capta el agua caída [4]. Una vez transcurrido el periodo de tiempo seleccionado, se puede visualizar directamente la cantidad de agua caída gracias a la graduación de la probeta. Una de las desventajas de este tipo de pluviómetro es que la cantidad de precipitación caída puede superar la cantidad límite de agua admitida en la probeta lo que generaría perdida de líquido y una medición inexacta.

Considerando lo anterior, es que uno de los instrumentos que proporciona mayor precisión al momento de medir precipitaciones es el pluviómetro de balancín o “tipping bucket”. Este pluviómetro posee una cubeta con un embudo en su parte superior que capta el agua. Dentro de esta cubeta se encuentran dos baldes que pivotean sobre un eje horizontal

similar a un balancín. El agua cae en uno de los dos baldes colectores, una vez que el balde se llena, por medio de la fuerza de gravedad vuelca dejando caer el agua colectada hacia un desagüe, posicionando el otro balde para seguir colectando agua. Por cada vuelco se acciona un switch que permite contabilizar la cantidad de vuelcos y de esta forma la cantidad de precipitación caída. Cada balde posee una capacidad fijada en mm [5].

3.1.4. Viento.

El viento o movimiento de masas de aire que se producen en la atmósfera son causados por diferencias de presión atmosféricas [4]. La velocidad del viento posee dirección y magnitud, por lo tanto es considerado como cantidad vectorial, por lo que puede medirse y procesarse como tal, por esta razón es que se utilizan las componentes escalares del viento por separado, las cuales son intensidad (rapidez) y dirección. Las unidades de medida que se utiliza en el SI para la intensidad del viento son metros por segundo $\left[\frac{m}{s}\right]$, aunque en algunos países o instituciones utilizas los nudos $[kt]$, en donde un nudo corresponde a $0{,}515\left[\frac{m}{s}\right]$ [7]. Existe una convención para definir en qué dirección sopla el viento, a esto se le conoce como la convención meteorológica, la cual consiste en que la dirección del viento se expresa en grados y se mide según desde donde sopla el viento a partir del norte y avanzando en dirección horaria, como se muestra a continuación [4]:

$$Norte = 0° \,; Este = 90° \,; Sur = 180° \,; Oeste = 270° \tag{3}$$

La intensidad y dirección del viento son variables, por lo que existe una norma de la WMO (World Meteorological Organization) que indica que la intensidad del viento corresponderá al promedio registrado en los 10 minutos previos y la dirección del viento será la predominante en los 10 minutos [5]. Además es común la utilización del promedio de dos minutos y de la medición instantánea de la intensidad y dirección del viento.

La forma común de medir las dos magnitudes del viento (intensidad y dirección) es a través de un anemómetro de copas con una veleta. El anemómetro de copas es un dispositivo

que mide la intensidad del viento a través del giro de copas o medias circunferencias que se conectan a un eje, las corrientes de viento hacen girar las copas debido a que la fuerza que ejerce el viento es mayor en el lado cóncavo de las copas. En el eje se acopla un transductor eléctrico que produce un voltaje en corriente continua que es proporcional a la velocidad del eje, y por lo tanto a la intensidad del viento. Este voltaje se transforma para obtener el valor de intensidad o rapidez del viento. Estos anemómetros poseen una intensidad mínima y máxima de operación, que corresponden a la intensidad mínima por la cual las copas comenzarán a girar y a la intensidad máxima que pueden girar sin dañar la estructura.

La veleta es una placa plana que gira alrededor de un eje vertical y que se orienta según la dirección que posea la corriente de aire o viento. Utiliza un transductor eléctrico que convierte el ángulo en que se encuentra posicionada la veleta a un voltaje proporcional a este ángulo [5]. La dirección del viento corresponde a una función circular que comprende valores entre $0° \, y \, 360°$, es por esta razón que se requiere un procesamiento especial para obtener un promedio de dirección del viento válido. Existen diversos métodos para la obtención de un promedio de dirección del viento, estos son el método de Mitsuta, Yamartino y Turner [8], pero el que se utilizará será el método vectorial, el cual se explicará a continuación:

Se utiliza la convención meteorológica, la cual se explicó anteriormente. Dadas dos series de tiempo que contienen observaciones (i) de la intensidad del viento (ws) y dirección del viento en grados (wd), se calcula la dirección promedio del viento de la siguiente manera:

$$\vec{u} = promedio\left(ws[i] * \cos\left(wd[i] * \frac{\pi}{180}\right)\right) \tag{4}$$

$$\vec{v} = promedio\left(ws[i] * \sin\left(wd[i] * \frac{\pi}{180}\right)\right) \tag{5}$$

$$WD_{promedio}(°) = arctan2\left(\frac{\vec{v}}{\vec{u}}\right) * \frac{180}{\pi} \tag{6}$$

$$WD_{promedio}(°) = \left(360 + WD_{promedio}\right) \% \, 360 \tag{7}$$

La ecuación (7) se utiliza para que el resultado de la dirección promedio del viento se encuentre en el rango adecuado, el cual es $(0°, 359°)$.

Es importante resaltar la diferencia entre utilizar un promedio simple y este método vectorial, para esto se realizará un ejemplo simple con tres valores de ángulos (355°, 5° y 15°) y una velocidad constante (5 kph) para calcular la dirección promedio del viento.

- Promedio simple entre 355°, 5° y 15°

$$\frac{355 + 5 + 15}{3} = 125°$$

$$Resultado\ esperado = 5°$$

- Utilizando el método vectorial para el cálculo de la dirección promedio del viento:

  Se tienen los siguientes datos:

$$ws = 5;\ wd = [355\ 5\ 15]$$

  Reemplazando en (4) y (5), resulta:

$$\vec{u} = 4{,}9305$$

$$\vec{v} = 0.4314$$

  Finalmente, se calculará la dirección del viento promedio:

$$WD_{promedio}(°) = arctan2\left(\frac{0.4314}{4{,}9305}\right) * \frac{180}{\pi} = 5{,}55°$$

Queda ejemplificado la importancia de la utilización del método vectorial para calcular la dirección promedio del viento.

3.1.5. Presión atmosférica.

La presión atmosférica se produce debido a que la atmósfera ejerce una presión hacia la superficie de la tierra. Esta presión es igual al peso de una columna vertical de aire de sección transversal unitaria. Debido a que el aire es un fluido y a que todas las ciudades o zonas

poseen características geográficas diferentes, la presión se ejerce de manera heterogénea alrededor del planeta. La unidad correspondiente a la presión atmosférica en el SI es el Pascal $[Pa]$, pero en meteorología se utiliza el milibar $[mbar]$ o el Hectopascal $[hPa]$ debido a que son equivalentes $(1\ [hPa] = 1\ [mbar])$ [4], [5]. La presión atmosférica promedio a nivel del mar es de $1.013\ [mbar]$ mientras que a medida que se va aumentando la altura, el valor medido de presión atmosférica va disminuyendo.

Se utilizan distintas técnicas para medir la presión atmosférica, las cuales pueden ser de forma directa o indirecta. En la forma directa se mide la fuerza por unidad de área ejercida por la atmósfera mientras que en la técnica indirecta se mide el punto de ebullición de un líquido que se expone a la presión atmosférica [5].

Los instrumentos que se utilizan para medir de forma directa la presión atmosférica son el barómetro de mercurio y el barómetro aneroide. El barómetro de mercurio mide la altura que alcanza una columna de mercurio que se encuentra bajo el efecto de la presión atmosférica. El modelo que se utiliza comúnmente es conocido como barómetro Fortín, el cual se compone por un tubo graduado de vidrio que se encuentra abierto por un extremo y cerrado por el otro. El tubo se introduce en un recipiente que contiene mercurio, esto se realiza por el extremo abierto. El peso de la columna de mercurio se equilibra con la fuerza que ejerce la atmosfera dando como resultado la medición de presión atmosférica correspondiente al lugar en donde se esté realizando [4], [5].

El barómetro aneroide se compone de una cámara de vacío con un diafragma flexible que según la presión que se le aplique (proveniente de la atmósfera) este se mueve. El barómetro posee una aguja que se puede indexar a un barógrafo registrado con el cual se puede observar el valor de la presión atmosférica medida [4], [5].

El instrumento que se utiliza para medir indirectamente la presión atmosférica es un barómetro de ebullición. Este barómetro utiliza un sensor de temperatura que se utiliza para medir la temperatura del punto de ebullición de un líquido que se calienta con el fin de mantener este proceso constante. El barómetro posee una cámara en donde el líquido se

condensa con el fin de volver a ser introducido en la cámara de combustión. La presión atmosférica puede ser calculada mediante el uso de fórmulas matemáticas. Para realizar pruebas a nivel del mar se debe utilizar un sensor de temperatura de gran precisión debido a que a esta altura las variaciones son muy pequeñas [5].

Una estación meteorológica automática utiliza un sensor para medir la presión atmosférica, estos sensores generalmente además de medir la presión atmosférica miden la temperatura y la humedad del ambiente. Este sensor se basa en un circuito integrado que posee un sensor aneroide el que gracias a otros circuitos permiten convertir los movimientos del diafragma en voltaje. Es necesario un microprocesador para convertir estos voltajes en conjunto con la medición de la temperatura y gracias a ecuaciones de calibración se puede obtener el valor de presión atmosférica [5].

3.2. Estado del Arte

Existen diferentes formas que permiten medir los parámetros meteorológicos, pero con el pasar del tiempo y el avance de la tecnologías, nacen soluciones comerciales para conocer y en algunas ocasiones predecir las condiciones meteorológicas. Junto con la aparición de estas soluciones, comienzan a aparecer estudios y publicaciones provenientes desde el mundo académico, las cuales tienen por objetivo implementar estas soluciones pero a un menor costo basándose en software y hardware libre.

3.2.1. Soluciones Comerciales.

Actualmente en nuestro país existe una variada oferta de estaciones meteorológicas automáticas disponibles en el mercado. Mayormente la oferta se basa en estaciones meteorológicas profesionales, que pueden medir diversos parámetros meteorológicos. La mayoría envía los datos obtenidos a bases de datos mediante internet, pero también existen algunas en que los datos se extraen de forma manual. Algunas utilizan baterías como sistema

de energía y otras utilizan paneles solares junto a baterías como sistema de alimentación. A continuación se podrá observar algunas estaciones meteorológicas que se comercializan en el mercado local.

3.2.1.1. Estación meteorológica agrícola WatchDog 2800.

Esta estación (Figura 1) es comercializada localmente por Veto. Es una empresa que se dedica a la comercialización de instrumentos de medición, control y registro de variables [9]. Es una estación meteorológica profesional de la línea WatchDog desarrollada por Spectrum Technologies que se enfoca en el uso agrícola. Puede medir temperatura ambiental, humedad relativa, punto de rocío, radiación solar, precipitaciones, intensidad y dirección del viento. Los datos obtenidos pueden ser descargados a un computador utilizando un cable de comunicación (incluido) o estos pueden ser enviados a través de radiotransmisores con un alcance de tres kilómetros. Utiliza un sistema de alimentación compuesto por cuatro pilas AA. Además incluyendo módulos específicos se puede agregar la funcionalidad de un modelo predictivo de enfermedades y plagas [10]. Su precio es de $3.246.838 (febrero de 2022).

Figura 1: Estación meteorológica agrícola WatchDog 2800

Fuente: https://vetocl.vteximg.com.br/arquivos/ids/178759-1000-1000/A6093219-1.jpg?v=637415016832030000

3.2.1.2. Estación meteorológica IMETOS 3.3.

Esta estación (Figura 2) es comercializada localmente por Agroprecisión Spa, es una empresa que ofrece servicios y productos orientados a la agricultura de precisión [11]. Es una estación meteorológica profesional desarrollada por Pessl Instruments empresa dedicada a implementar tecnología en agricultura. La estación incluye un pluviómetro, sensor de dirección del viento y todos los sensores que permiten el cálculo de la evapotranspiración: temperatura del aire, humedad relativa, radiación y velocidad del viento. Los valores medidos los registra y envía a la base de datos FieldClimate en donde se puede tener acceso a las mediciones a través de aplicaciones web y móviles. Para enviar los datos es necesario una tarjeta SIM de un proveedor local con acceso a internet mediante las redes 2G, 3G y 4G, además se puede configurar para enviar alertas por mensaje de texto SMS. Posee un sistema de alimentación compuesto por un panel solar y una batería, lo que la hace autosuficiente energéticamente. Adicionalmente se pueden contratar servicios adicionales como modelos de enfermedades de plantas y previsiones meteorológicas hiper-localizadas [12]. La variante IMETOS IMT280 junto con el mástil de tubo galvanizado que se utiliza como soporte, tienen un valor de $2.829.570 (Iva incluido) (Anexo 1).

Figura 2: IMETOS IMT280

Fuente: https://metos.at/es/imetos33/

3.2.1.3. Estación meteorológica Vantage Pro2 plus inalámbrica Davis Instruments.

Esta estación (Figura 3) es comercializada localmente por Improfor, es una empresa importadora y exportadora dedicada a la seguridad industrial y el cuidado del medio ambiente [13]. Es una estación meteorológica profesional que mide las siguientes variables meteorológicas: temperatura, humedad, punto de rocío, precipitaciones, presión atmosférica, radiación solar, evotranspitación, velocidad y dirección del viento. Posee una consola que permite visualizar los parámetros medidos de forma simultánea a través de una pantalla LCD, estos parámetros son enviados de forma inalámbrica a una distancia de hasta 300 metros actualizándose la información cada 2,5 segundos. Implementa un sistema de alimentación compuesto por un panel solar y una batería además de un sistema de respaldo compuesto por una pila de litio. Posee un sistema de alarmas que alertan fenómenos meteorológicos como vientos fuertes, heladas, lluvias intensas, etc. Adicionalmente se puede implementar un datalogger para el registro de los datos medidos [14]. Tiene un valor de $1.249.500 (Iva incluido) (Anexo 2).

Figura 3: Davies Vantage Pro2 Plus inalámbrica

Fuente: https://www.estacionesdavis.es/es/davis-vantage-pro-2/12-davis-vantage-pro2-plus-inalambrica.html

3.2.2. Publicaciones sobre estaciones meteorológicas basadas en Arduino.

En el año 2017, Solano et al. [15] realizaron una estación meteorológica con fines educativos basada en el microcontrolador Atmega8L. Esta estación mide a través de sensores cinco parámetros meteorológicos, los cuales son: intensidad de lluvia, radiación ultravioleta, humedad relativa, temperatura y velocidad del viento. Transmite los datos obtenidos mediante radiofrecuencia a través del módulo nRF24L01+, estos datos son recibidos por una estación base la cual envía los datos meteorológicos a una computadora mediante comunicación serie. Dentro del artículo se menciona que posee un sistema de alimentación autónomo pero no se especifica en mayor detalle los componentes de este sistema de alimentación.

En el mismo año, Üçgün et al. [16] realizaron una estación de pronóstico del tiempo basada en Arduino Uno R3. Esta estación mide los siguientes parámetros meteorológicos: temperatura, humedad, presión barométrica y lluvia. Posee una pantalla LCD en donde se pueden visualizar imágenes que corresponden a sol, nube, paraguas, etc. Estas imágenes tienen relación con los eventos meteorológicos asociados a los valores medidos y a reglas que establecieron en base a inferencias sobre el clima. Realizaron distintas mediciones en la provincia de Bilecik (Turquía) y las compararon con datos proporcionados por meteorología, concluyendo que los valores de temperatura y presión son cercanos, pero en la humedad existen diferencias mayores, lo que se puede ser debido a factores externos.

Posteriormente, Brito et al. [17] desarrollan una estación meteorológica de bajo costo utilizando hardware y software gratuitos, en la Universidad Tecnológica Federal do Paraná (Brasil). Está estación mide a través de sensores temperatura, humedad, presión atmosférica, dirección del viento, a través de un anemómetro mide la intensidad del viento y a través de un pluviómetro la cantidad de precipitación caídas. Utiliza un Arduino para recibir los datos medidos de los sensores, el Arduino se conecta con una Raspberry PI a través del protocolo de comunicación I2C, la cual está conectada a internet a través de Ethernet, con el fin de enviar los datos a un servidor web, esto se realiza mediante una aplicación Java desarrollada por ellos.

También, desarrollaron una aplicación web que muestra un informe de los datos leídos por la estación meteorológica. No se especifica el sistema de alimentación, por ende se intuye que el Arduino y la Raspberry Pi son alimentados mediante conexión USB a un computador.

En el año 2018, Imtiaz et al. [18] diseñan una estación meteorológica portátil de bajo costo con el fin de que pueda ser utilizada en lugares que estén fuera de la cobertura de las aplicaciones meteorológicas convencionales. Esta estación utiliza sensores para medir temperatura, humedad, presión atmosférica y utiliza un anemómetro hecho por ellos con el fin de disminuir aún más los costos. Utiliza un Arduino y una "Ethernet Shield" para conectarse a internet y enviar los datos obtenidos a un servidor web, estos datos pueden ser visualizados a través de una página web creada por los autores en donde se pueden observar gráficos de los parámetros medidos. Se menciona que en caso de que se corte la energía la estación funcionará a través de un panel solar. Una ventaja de esta estación meteorológica es que se puede instalar en cualquier lugar, otorgando datos correspondientes al lugar seleccionado a diferencia de datos proporcionados por sitios web tradicionales debido a que estos entregan datos de áreas más grandes.

En el año 2019, Haque et al. [19] crean una estación meteorológica portátil basada en Arduino Mega con energía renovable. Esta estación mide los siguientes parámetros meteorológicos a través de sensores: temperatura, humedad, precipitación, contaminantes atmosféricos y presión atmosférica. La precipitación se mide mediante un dispositivo de presencia de lluvia (Raindrop) y los contaminantes los mide a través de un dispositivo que detecta humo y monóxido de carbono. Para la comunicación de los valores medidos utiliza una pantalla LCD la cual muestra en directo los valores medidos y un módulo GSM el cual envía un mensaje de texto con los valores, para realizar esta acción el módulo debe recibir previamente un mensaje de texto de la persona interesada en conocer las mediciones de las condiciones meteorológicas medidas por la estación. El sistema de alimentación que posee se basa en un sistema fotovoltaico off-grid compuesto por un panel solar, un regulador de carga y una batería de plomo ácido. Realizan una comparación entre datos medidos por la estación y datos

meteorológicos proporcionados por Accuweather. Esta comparación se realiza en dos lugares, al interior de la ciudad de Dhaka y a las afueras de la ciudad, considerando los datos de temperatura, humedad y presión barométrica. Para la primera comparación se obtiene un porcentaje de error del 0,6% para la temperatura, 4,1% para la humedad y 7,78% para la presión barométrica. Para la segunda comparación se obtiene un porcentaje de error del 3,7% para la temperatura, 4,3% para la humedad y 9,68% para la presión barométrica. Estos resultados indican que las mediciones realizadas por esta estación meteorológica presentan una mínima diferencia con los sitios mencionados y por ende se puede confiar en las mediciones realizadas.

En el año 2020, Kaewwongsri y Silanon [20] diseñan e implementan una estación meteorológica que usa CoAp en la red NB-IoT. Esta estación mide a través de sensores los siguientes parámetros meteorológicos: temperatura, humedad, velocidad y dirección del viento, presión atmosférica, precipitaciones, ozono. Utilizan un Arduino Uno y un Arduino Mega. El primero se encarga de la comunicación con los sensores y la "Weather Shield" a la cual se conecta un anemómetro, un pluviómetro de balancín y posee integrados los sensores de presión barométrica, humedad y temperatura. El segundo se utiliza como controlador principal, maneja el sensor de ozono, un reloj en tiempo real y una "NB-IoT Shield". Ambos Arduino se conectan entre si mediante el protocolo I2C. La estación transmite los datos a través de CoAP en la red NB-IoT, hacia una base de datos (MySQL). Finalmente los datos obtenidos pueden ser visualizados como gráficos a través del software Grafana.

En el año 2021, Ashok Baste et al. [21] crean una estación meteorológica para el uso en plantas de energía solar fotovoltaica basada en Arduino Mega. Esta estación mide a través de sensores los siguientes parámetros meteorológicos: temperatura, humedad, presión atmosférica y radiación solar, mediante un anemómetro velocidad además de la dirección del viento y a través de un pluviómetro de balancín las precipitaciones. Esta estación utiliza el microcontrolador Arduino Mega 2560 conectado a una pantalla LCD en donde pueden ser visualizados los datos en tiempo real además de contar con la opción de transmitir los datos mediante WI-FI y almacenarlos en una base de datos (MySQL). También crearon una aplicación

web mediante la cual pueden ser observados los datos en tiempo real. La estación cuenta con un piranómetro que es el dispositivo responsable de la medición de la radiación solar. Esta estación meteorológica se implementó en una planta solar fotovoltaica, con el fin de obtener los datos meteorológicos de esta y poder planear de forma optimizada las actividades de mantenimiento de la planta, debido a que si se realizan en condiciones meteorológicas desfavorables, resulta una actividad mucho más costosa y menos duradera. Un dato importante además de la radiación es el viento, debido a que fuertes ráfagas de este pueden provocar daños en los paneles fotovoltaicos, es por lo que al monitorear la velocidad del viento constantemente se pueden tomar acciones preventivas con el fin de disminuir los daños a los paneles.

3.3. Semejanzas y diferencias entre estaciones meteorológicas automáticas y de bajo costo basadas en Arduino

En los apartados anteriores, se definieron y explicaron las variables meteorológicas que serán monitoreadas por la estación de bajo costo basada en Arduino. Además, se explicó la forma tradicional de medir dichas variables meteorológicas y como una estación meteorológica profesional puede realizar estas mediciones. Para hacer una comparación entre las estaciones automáticas y una estación meteorológica Arduino, primero se detallarán las semejanzas entre ambas.

Las principales semejanzas estudiadas, pueden ser implementadas para medir las mismas variables meteorológicas. Estas mediciones las realizan a través de sensores, pudiendo llevar un registro de los valores medidos, los cuales pueden ser almacenados mediante el uso de datalogger o transmitidos hacia servidores web, a través de radiofrecuencia o internet. El sistema de alimentación que usan generalmente se compone por paneles solares junto con baterías, las que permiten funcionar cuando las condiciones de generación de energía no sean las adecuadas.

Las principales diferencias entre las estaciones meteorológicas profesionales y las de bajo costo basadas en Arduino son la precisión y el costo. Una estación profesional, que se comercializa en el mercado, posee una precisión mucho mayor en comparación a una de bajo costo, debido a que los sensores que utilizan se encuentran fabricados, testeados y certificados para responder a pequeñas variaciones. La mayoría de los sensores que utilizan las estaciones automáticas se implementan en la industria, mientras que los sensores que se integran en una estación Arduino deben ser compatibles con la plataforma. El costo de una estación meteorológica profesional es mayor, debido a las razones expuestas anteriormente y a que se encuentran fabricadas con elementos certificados que permitan una larga durabilidad, en cambio en una estación meteorológica Arduino siempre se privilegiará el uso de elementos de un menor costo.

4. Metodología

En esta sección, se detallará el proceso de diseño, construcción e implementación de la estación meteorológica de bajo costo basada en Arduino. Se comenzará por la etapa de diseño, en donde se define el hardware, el sistema de alimentación y el software a utilizar. Posteriormente se definirá el emplazamiento en donde será instalada la estación meteorológica, se analizará la radiación disponible y la ubicación óptima del panel solar. Finalmente, se detallará el proceso de construcción de la estación meteorológica e implementación de los componentes.

4.1. Diseño

En este apartado, se explica la selección de componentes que integrarán la estación meteorológica de bajo costo basada en Arduino, junto con sus características, también se detallará la selección de componentes para el sistema de alimentación. Por último, se detallará el software que será utilizado en el proceso de implementación y análisis de resultados.

4.1.1. Hardware.

Se describen los distintos componentes que integrados forman la estación meteorológica de bajo costo basada en Arduino. La parte electrónica, consiste en una placa de desarrollo (Arduino), en los sensores que se utilizarán para medir las distintas variables meteorológicas y en el dispositivo de almacenamiento de los datos obtenidos junto con el dispositivo que los transmitirá a un servidor web.

4.1.1.1. Placa de desarrollo Arduino.

La placa Arduino, es el corazón de la estación meteorológica, ya que lee el programa que carga el usuario realizando las acciones que este requiere, como por ejemplo comunicarse

con sensores para obtener los valores de las variables meteorológicas que mide. Además, es el encargado de mostrar los datos a través del puerto serie y enviarlos tanto al datalogger como al servidor web para su correcta visualización y almacenamiento.

Arduino nace en el año 2005 en el Instituto de diseño interactivo de Ivrea (Italia), por la necesidad de contar con un dispositivo de bajo costo para poder ser utilizado en las aulas. El instituto se vio obligado a cerrar sus puertas en el mismo año. Ante la posibilidad de perder todo el desarrollo de Arduino, se decidió liberarlo y abrirlo al público para que todo el mundo pudiera participar en la evolución del proyecto proponiendo mejoras y sugerencias. Los diseñadores y desarrolladores de la idea fueron Massimo Banzi, David Cuartielles, David Mellis, Tom Igoe y Gianluca Martino.

Arduino es una plataforma de desarrollo que se basa en una placa electrónica de hardware libre, que incorpora un microcontrolador reprogramable y una serie de pines. Estos permiten establecer conexiones entre el microcontrolador y diferentes sensores y actuadores.

Una de las placas más populares desarrolladas por la compañía, es el Arduino Uno, esto se debe a que fue la primera en salir al mercado, por ende, posee un desarrollo más extenso. Sin embargo, para este proyecto, se escogió el Arduino Mega (Figura 4), debido a que posee una mayor memoria volátil (RAM) lo que permite ejecutar códigos más extensos que el Arduino Uno y es compatible con la mayoría de las shields desarrolladas para Arduino Uno, Duemilanove o Diecimila. A continuación, se podrán observar las principales características del Arduino Mega.

Arduino Mega

- Posee un microcontrolador ATmega2560
- Voltaje de entrada $7\,V\ a\ 12\,V$.
- Posee 54 pines de entrada/salida digital de los cuales 14 se pueden utilizar como salidas PWM.

- Posee 16 entradas análogas
- 256k de memoria flash
- Reloj de 16 MHz de velocidad.

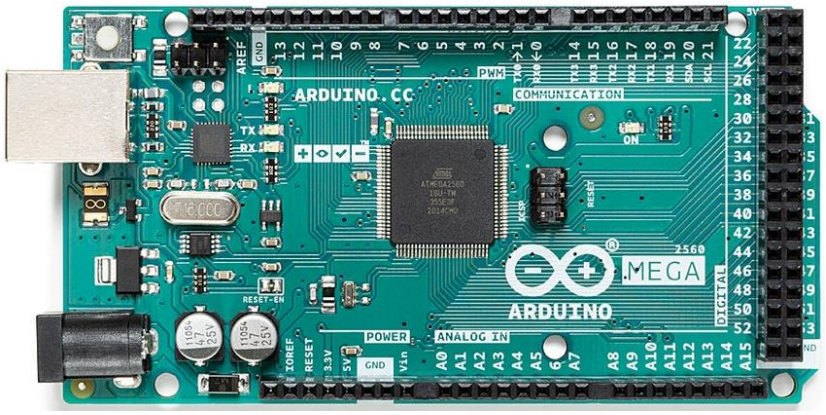


Figura 4: Arduino Mega

Fuente: https://store-usa.arduino.cc/products/arduino-mega-2560-rev3?selectedStore=us

Disposición y conexiones

En la Figura 5, se observa la disposición y conexiones que posee el Arduino Mega.

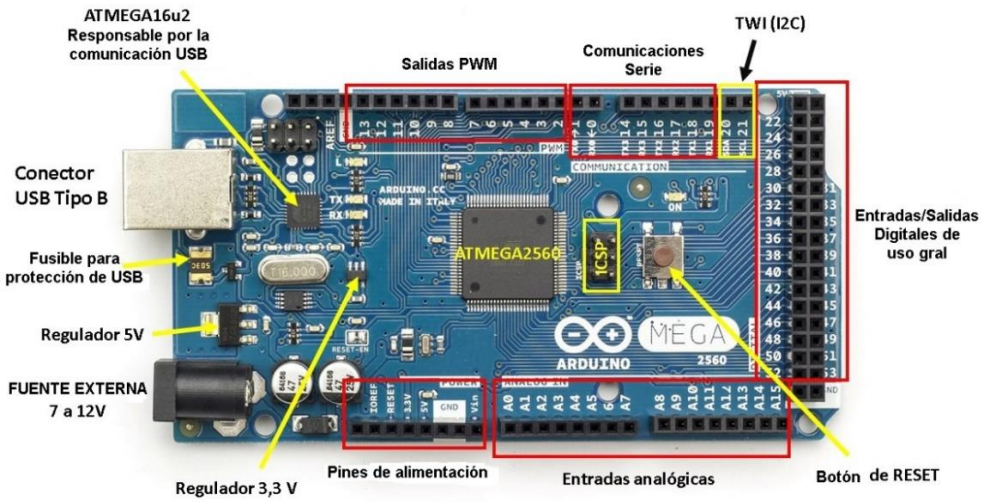


Figura 5: Disposición y conexiones Arduino Mega

Fuente: Google imágenes

4.1.1.2. Datalogger.

Un datalogger (Figura 6) es un dispositivo, que permite monitorear y registrar datos. En la estación meteorológica, será el encargado de llevar un registro de los datos obtenidos a través de los sensores, registrando sus lecturas cada cierto periodo de tiempo.

El datalogger seleccionado es totalmente compatible con Arduino, se monta encima encajando perfectamente y permitiendo el acceso a los terminales mediante conectores. Esta placa, dispone de un reloj de tiempo real RTC y cuenta con una pequeña batería que le permite mantener el registro de fecha y hora. Además posee un lector de tarjeta SD, la cual almacenará los datos. Debe ser alimentada con $5V$, el reloj utiliza un protocolo de conexión I2C mientras que el lector de tarjeta SD utiliza un protocolo de conexión SPI [22].

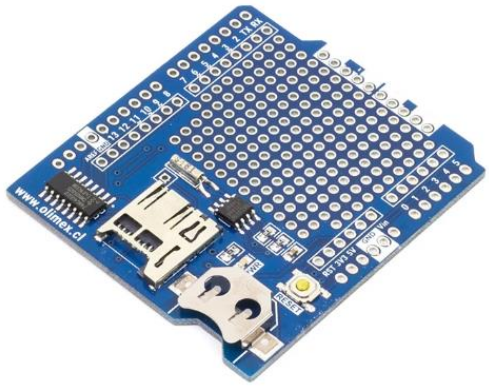


Figura 6: Datalogger Shield

Fuente: https://www.mcielectronics.cl/shop/product/mci-arduino-logger-shield-usd-rtc-mci-electronics-10355

4.1.1.3. Ethernet Shield W5100 WizNet.

La ethernet Shield[2] (Figura 7), permite conectar el Arduino a internet a través de un cable ethernet estándar RJ45. Al ser una Shield encaja perfectamente sobre la placa Arduino, la cual proporciona la alimentación necesaria para su funcionamiento. Con esta Shield y la librería adecuada, se puede montar un pequeño servidor web o un cliente, enviando los datos recopilados a un servidor web. Posee un zócalo para tarjetas de memorias micro-SD para el almacenamiento de ficheros. Se comunica con el Arduino Mega mediante el bus SPI (mediante el conector ICSP) utilizando los pines digitales 50, 51 y 52 (no utiliza el pin 53, pero este debe ser configurado como salida para que funcione correctamente el bus SPI) [23]. Posee las siguientes especificaciones:

- Voltaje de Operación: 5V DC
- Chip Ethernet: Wiznet W5100
- Velocidad Ethernet: 10/100 Mbps
- Conector RJ45
- Interface: SPI
- Compatible con Arduino Uno, Mega, Leonardo
- Lector MicroSD Card

[2] Una Shield es una placa de circuito impreso que se encuentra diseñada para ser colocada sobre la placa Arduino, permitiendo el acceso a todos los pines de este. Una Shield agrega o amplia capacidades del Arduino utilizado.

Figura 7: Ethernet Shield W5100 WizNet

Fuente: https://altronics.cl/ethernetshield-w5100

4.1.1.4. Sensores.

Para realizar una comunicación del Arduino con el entorno y poder recibir datos de este, se utilizan los sensores. El sensor, es uno de los elementos más importantes en las estaciones meteorológicas, debido a que son los que interactúan con el entorno y generan una señal correspondiente según la lectura que puede tomar en la interacción. Los sensores en una estación meteorológica, se montan en una estructura fija normalmente protegida de la radiación y la humedad (esto depende del tipo de sensor) [5].

En la estación meteorológica Arduino, se implementarán distintos sensores, los cuales se encargarán de proveer los datos meteorológicos del lugar en donde se instale la estación. A continuación, se entregarán los detalles y características de cada sensor seleccionado.

Sensor de temperatura y humedad AM2315

Sensor de temperatura y humedad relativa de tipo capacitivo (Figura 8) desarrollado por AOSONG. Es un sensor de alta precisión y de bajo consumo de energía, utiliza el protocolo de comunicación I2C para conectarse a la placa Arduino, es utilizado en estaciones meteorológicas, equipos de aire acondicionado y electrodomésticos. Debe ser alimentado por un voltaje en el rango $3{,}3\,V$ a $5{,}5\,V$ pero se recomienda un voltaje de alimentación de $5\,V$. Posee

cuatro cables, VDD (power), GND (ground), SDA y SCL. Puede medir humedad relativa en el rango $0 \sim 99{,}9\,\%$, mientras que el rango de temperatura que puede medir es $-40\,°C\ a\ 80\,°C$. Posee $\pm 2\,\%$ de precisión de humedad relativa y $\pm 0{,}3\,°C$ de temperatura [24].

Figura 8: Sensor de temperatura y humedad AM2315

Fuente: https://www.mcielectronics.cl/shop/product/sensor-de-temperatura-y-humedad-am2315-25767?search=MCI05578

Arduino weather Shield

Esta Shield (Figura 9) desarrollada por la empresa SparkFun, está construida para trabajar con Arduino Uno y Mega. Posee tres sensores incorporados, los cuales son: sensor de temperatura y humedad Si7021, sensor de presión barométrica MPL3115A2 y sensor de luz ALS-PT19.

Para la estación meteorológica, se utilizarán los sensores de presión barométrica, temperatura y humedad, con la distinción que este último se utilizará con el fin de monitorear la temperatura interna de la caja en donde estarán alojados los componentes electrónicos de la estación.

El MPL3115A2 es un sensor digital que puede medir presión barométrica, altitud y temperatura, posee integrado un conversor análogo digital, con el cual proporciona una salida correspondiente a la medición en °C y Pascales. Utiliza, el protocolo de comunicación I2C con el microcontrolador Arduino. Debe ser alimentado con un voltaje en el rango de $1{,}6\ a\ 3{,}6\,V$. El

sensor opera en los siguientes rangos: $20\ kPa\ a\ 110\ kPa$ de presión, $-698\ m\ a\ 11.774\ m$ de altitud y $-40°C\ a\ 85°C$ de temperatura [25].

El Si7021 es un sensor digital de humedad y temperatura de bajo costo. Utiliza, el protocolo de comunicación I2C con el microcontrolador Arduino, debe ser alimentado por un voltaje de $3{,}3\ V$. El sensor opera en los siguientes rangos: $0 \sim 100\%$ RH, $-10\ a\ 85\ °C$ de temperatura y posee una precisión de $\pm 3\%$ RH y $\pm 0{,}4°C$ de temperatura [26].

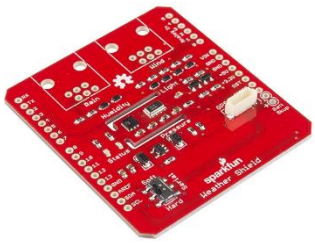


Figura 9: Arduino weather Shield

Fuente: https://www.mcielectronics.cl/shop/product/estacion-meteorologica-shield-sparkfun-10902

Kit de sensores para Arduino weather Shield

Este kit se compone de un anemómetro de copas, una veleta y un pluviómetro de balancín (Figura 10). La ventaja de este kit es que es totalmente compatible con la Arduino weather Shield (Figura 9) y Arduino. Para realizar la conexión necesita conectores RJ11 de seis pines. El anemómetro de copas utiliza un interruptor de láminas lo que permite medir la intensidad del viento a través de detección de la frecuencia en un cierto periodo de tiempo. La veleta posee un potenciómetro que le permite medir la dirección del viento. El pluviómetro utiliza el mismo principio que los mencionados en el inciso 3.1.3 [27].

Figura 10: Anemómetro, veleta y pluviómetro para estación meteorológica

Fuente: https://www.mcielectronics.cl/shop/product/sensores-para-estacion-meteorologica-10008

4.1.2. Sistema de alimentación.

Un punto importante dentro del diseño de la estación meteorológica es la alimentación. Para esto, se debe tener en cuenta que al momento de instalar la estación esta debe funcionar de manera autónoma, ya que variando el uso que se le requiera entregar, se instalará generalmente en zonas en donde no se cuente con acceso a energía eléctrica de manera sencilla. Es por esto, que se determinó que la alimentación de la estación meteorológica será a través de un sistema fotovoltaico off-grid compuesto por un panel solar, una batería y un regulador de carga.

4.1.2.1. Panel Solar

El panel solar (Figura 11) será el encargado de proveer la energía necesaria para el correcto funcionamiento de la estación meteorológica. El panel seleccionado, es desarrollado por la empresa SparkFun y posee las siguientes características, las cuales fueron extraídas desde su hoja de datos proporcionada por la empresa [28].

- Dimensiones: $210mm\ x\ 113mmx5mm$
- Celdas monocristalinas - 19% efficiencia
- Open Circuit Voltage: 7.0V
- Peak Voltage: 6.0V

- Peak Current: 615mA
- Peak Power: 3.69 Watt
- Cuatro tornillos en la parte trasera del panel permitiendo múltiples opciones de montaje
- Revestimiento de uretano impermeable
- Resistente a los rayos UV

Figura 11: Panel solar

Fuente: https://www.mcielectronics.cl/shop/product/panel-solar-3-5w-22995?search=panel+solar

4.1.2.2. Regulador de carga solar Lipo rider pro.

Esta tarjeta (Figura 12) funciona como un regulador de carga recibiendo la energía que proporciona el panel solar, con esta energía carga una batería y posee una salida USB, por la cual se alimentará la placa Arduino, posee las siguientes características:

- Potencia máxima de carga 1A
- Fuente de alimentación USB estable a 5V
- Conector para panel solar y batería JST 2.0
- Carga baterías de polímero de litio a través de energía solar o USB.
- Indicadores LED que permiten visualizar el estado de carga de la batería

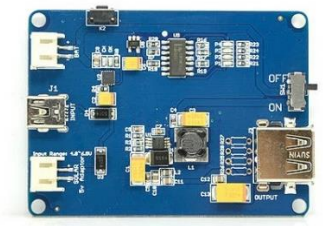


Figura 12: Adaptador para paneles solares lipo rider pro

Fuente: https://www.mcielectronics.cl/shop/product/adaptador-para-paneles-solares-lipo-rider-pro-con-salida-usb-10587?search=Bateria

4.1.2.3. Batería.

La batería de litio ion (Figura 13) permitirá proveer la energía necesaria para el funcionamiento de la estación meteorológica cuando el panel deje de producirla. Esta batería de iones de litio es desarrollada por SparkFun, posee las siguientes características, las cuales fueron extraídas desde la página del fabricante [29]:

- Capacidad nominal: 6000mAh
- Voltaje nominal: 3.7V
- Corriente de carga máxima: 1C (6000mA)
- Corriente de carga normal: 0.2C (1200mA)
- Circuito de protección

Figura 13: Batería de Litio Ion 6.000 mAh

Fuente: https://www.mcielectronics.cl/shop/product/bateria-de-litio-ion-6000mah-11293?search=Bateria

4.1.2.4. Sensor de voltaje FZ0430.

El sensor de voltaje FZ0430 (Figura 14), consiste en un divisor de tensión que posee dos resistencias de $30K\ \Omega\ y\ 7.5K\ \Omega$. La tensión máxima que puede medir es de $25\ V\ DC$ alimentándolo con $5V$ (Vcc) y $16.5V\ DC$ recibiendo una alimentación de $3{,}3V$ (Vcc). Incluye tres pines, dos de alimentación (Vcc y GND) y uno de salida (S) [30]. Posee las siguientes especificaciones técnicas:

- Rango de entrada de voltaje: $0V$ a $25V$ DC
- Voltaje detección entrada máximo: $25V\ (5V\ x\ 5 = 25V)\ o\ 16{,}5V\ (3{,}3V\ x\ 5 = 16{,}5V)$
- Rango de detección de voltaje: $24{,}41mV$ a $25V$
- Resolución analógica de tensión: $0{,}00489V$
- Voltaje detección entrada mínimo: $24{,}45mV (4{,}89mV\ x\ 5 = 24{,}45mV)$

Figura 14: Sensor de voltaje FZ0430

Fuente: https://altronics.cl/sensor-voltaje-fz0430

4.1.2.5. Sensor de corriente ACS712

El sensor de corriente ACS712 (Figura 15) posee un circuito integrado que permite medir la cantidad de corriente que fluye a través de un circuito AC o DC. El método de medición de la corriente es a través de un sensor de efecto hall[3], que entrega un voltaje de salida que es

[3] El efecto Hall produce una diferencia de voltaje a través de un conductor en presencia de un campo magnético.

proporcional a la corriente que fluye por el circuito. Incluye tres pines, dos de alimentación (Vcc y GND) y uno de salida (OUT). Posee las siguientes características técnicas:

- Alimentación: $4{,}5\ V\ a\ 5{,}5\ V\ DC$
- Rango de medida: $0\ a\ 5\ A$ AC/DC
- Voltaje de salida: $66\ mV\ /\ A$
- Salida de voltaje sin corriente: $VCC\ /\ 2$
- Resistencia interna: $1{,}2\ m\Omega$
- Salida: Analógica $185\ mV/A$
- Medidas de la placa: $31mm\ x\ 13\ mm$
- Tiempo de respuesta: $5\ \mu s$
- Rango de Error: 1,5 %

Figura 15: Sensor de corriente ACS712

Fuente: Google imágenes

4.1.3. Software.

En esta sección, se describen los principales softwares que se utilizarán en la etapa de implementación de la estación meteorológica de bajo costo basada en Arduino.

4.1.3.1. IDE Arduino.

Para desarrollar programas en Arduino que establezcan la comunicación con los dispositivos externos que se conecten, es necesario instalar el entorno de desarrollo integrado o IDE (Integrated Development Environment) que es el software necesario para programar cualquier placa Arduino o similares. Una vez dentro de la IDE (Figura 16), se puede comenzar a escribir un algoritmo, el cual consiste en un conjunto de instrucciones o reglas ordenadas y definidas que le permiten al Arduino llevar a cabo una acción. El lenguaje de programación que se utiliza para programar un Arduino se basa en el lenguaje C/C++. Una de las principales características de este lenguaje, es que permite escribir programas de manera simplificada. Otro aspecto importante en la creación de programas Arduino es el uso de librerías, las cuales ofrecen funciones o métodos preprogramados, que permiten realizar desarrollos más complejos pero de forma más intuitiva y cómoda, de forma que al utilizar una librería existente en Arduino o descargada, se omite la estructura o código de estas funciones permitiendo ser utilizadas directamente. Para realizar esta acción es necesario incluir previamente la librería en el inicio del sketch o programa [31].

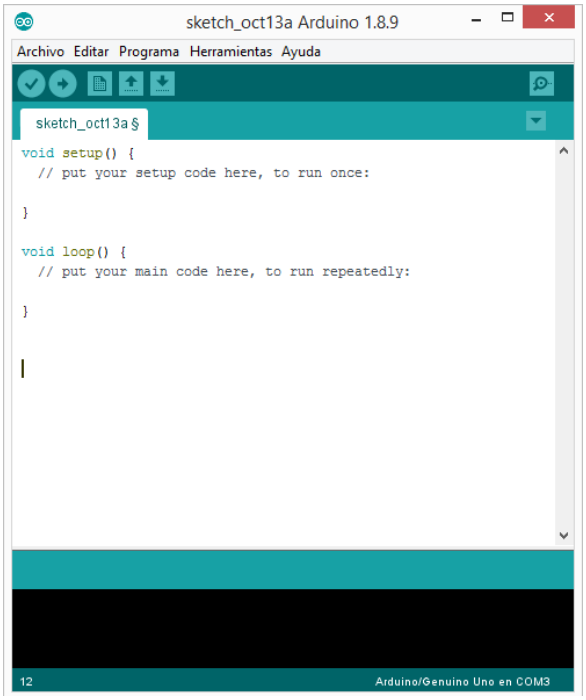


Figura 16: Arduino IDE

Dentro de la IDE (Figura 16), se pueden visualizar dos funciones principales Setup() y Loop() las cuales serán explicadas en el siguiente apartado. También posee una franja de color negro la cual corresponde a la consola, la que permite visualizar información sobre el estado de la compilación o de posibles fallos o errores que se pueden producir.

Un apartado importante dentro de la IDE, es el monitor serie (Figura 17), esta ventana permite visualizar acciones específicas programadas en la función loop(). En la configuración de la estación meteorológica, nos permitirá visualizar los valores obtenidos por los sensores que posteriormente serán almacenados en el datalogger y enviados hacia el servidor.

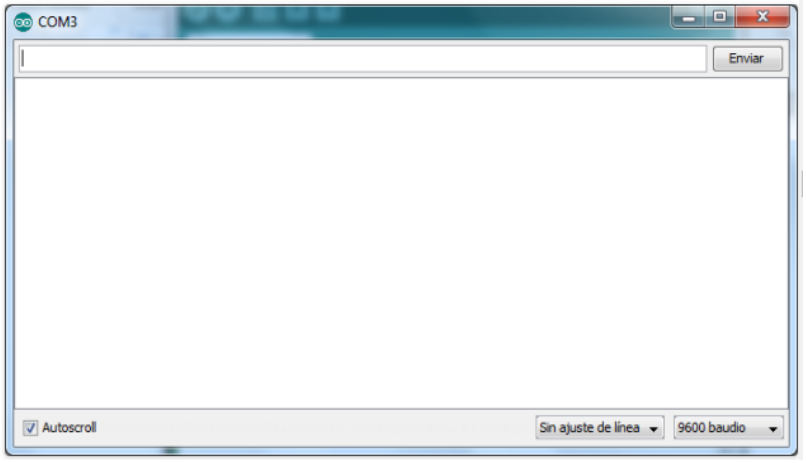


Figura 17: Monitor Serie

Estructura general de un sketch o programa Arduino

Los sketch o programas que se desarrollan dentro de la IDE poseen tres secciones principales:

- Declaración de variables: Se realiza en el espacio sobre la función Setup() y tiene por objetivo la asignación de variables o constantes que serán utilizadas a lo largo del desarrollo del programa.

- Función setup (): Esta función se ejecuta solo una vez al momento de encender la placa Arduino, dentro de esta función se define la utilización de pines como entradas o salidas y se inicia la comunicación serial.
- Función loop (): Esta función se ejecuta tras la función setup (), dentro de esta se debe definir el código que se repetirá infinitamente hasta que el Arduino se desconecte o pierda su fuente de alimentación [31].

4.1.3.2. Microsoft Excel.

Es una aplicación desarrollada por Microsoft para el uso en computadoras y derivados. Se basa en una hoja de cálculo que permite la realización de cálculos matemáticos utilizando fórmulas o realizar gráficos de información proporcionada en tablas [32]. En el proyecto, se utilizará esta aplicación para manejar y leer los datos almacenados por el datalogger y en el servidor , los cuales poseen un formato compatible con Excel. Una vez organizados los datos en tablas, estos pueden ser visualizados a través de gráficos para poder analizar las variables meteorológicas con mayor precisión y ver las variaciones en determinados periodos de tiempo.

4.1.3.3. ThingSpeak.

ThingSpeak es un servicio de plataforma de análisis para Internet of Things (IoT) que permite agregar, visualizar y analizar flujos de datos en vivo en la nube [33]. En el proyecto, se utilizará como un servidor para almacenar y visualizar los datos enviados por el Arduino a través de internet, pudiendo realizar un seguimiento de estos a través de la página web o de la aplicación ThingView. ThingSpeak, tiene la capacidad de ser compatible con el software Matlab, lo que permite analizar y procesar datos en línea o desde la misma aplicación de escritorio de Matlab.

4.1.3.4. Matlab.

Es uno de los softwares utilizados para tareas relacionadas a la ingeniería más populares a nivel mundial. Es un entorno de programación y computación numérica que se utiliza para analizar datos, desarrollar algoritmos y crear modelos [34]. En el proyecto, este programa se utilizará para el análisis de los datos meteorológicos obtenidos por la estación y proporcionados por el servidor ThingSpeak.

4.2. Emplazamiento

En la presente sección, se detalla el emplazamiento en donde será construida e instalada la estación meteorológica Arduino. Considerando que el sistema de alimentación está compuesto por un panel solar, es importante analizar la radiación disponible en el emplazamiento seleccionado junto con la ubicación óptima del panel solar, con el fin de maximizar la generación de energía.

4.2.1. Selección del emplazamiento.

El lugar seleccionado para la instalación de la estación meteorológica basada en Arduino es la azotea del edificio A de la Universidad de O'Higgins (Figura 18), ubicada en Libertador Bernardo O'Higgins 611, Rancagua, Región del Libertador Bernardo O'Higgins, Chile. Se escoge este lugar debido a que posee un fácil acceso, es cercano a la escotilla de acceso a la azotea y es seguro. Además, se encuentra a una distancia prudente de los equipos instalados, como se puede apreciar en la Figura 18.

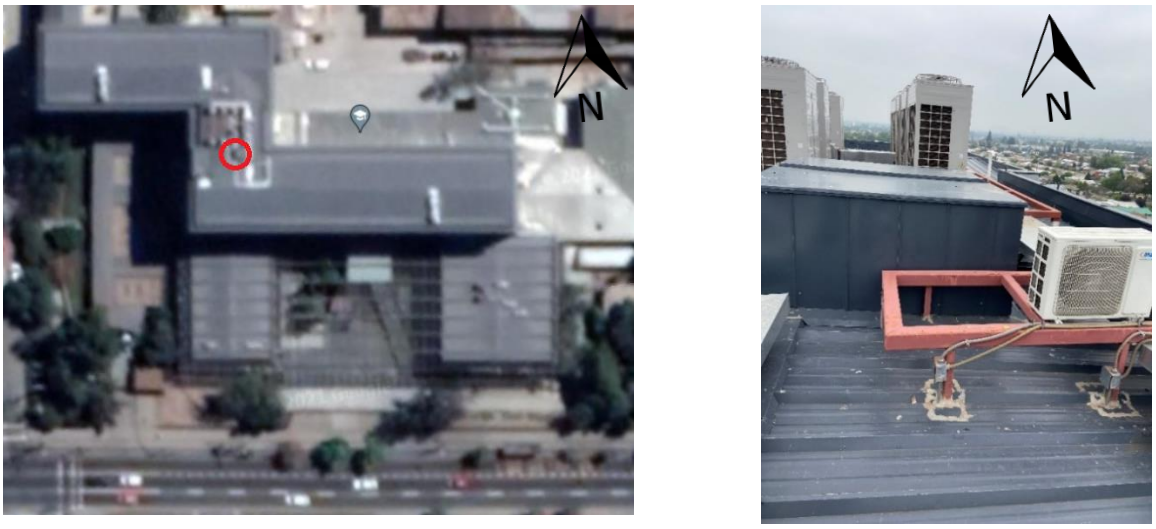


Figura 18: Emplazamiento estación meteorológica basada en Arduino

Fuente: Google Earth

La zona de instalación de la estación meteorológica Arduino, será por encima de la estructura de fierro de color rojo. A esta se fijará un gabinete de tablero eléctrico para exterior. Sobre este, se instalará el panel solar y se acoplará un tubo galvanizado encargado de cargar la estructura que sostiene al pluviómetro de balancín junto con los sensores de velocidad y dirección del viento. El montaje de la estructura será proporcionado por el área de mantención del campus Rancagua de la Universidad de O´Higgins.

Considerando el sistema de alimentación seleccionado, existen dos aspectos interesantes a investigar sobre el emplazamiento seleccionado, los cuales son: la irradiancia del emplazamiento y la ubicación óptima del panel solar, los que serán investigados a continuación.

4.2.2. Radiación del emplazamiento.

Un aspecto importante a considerar es la radiación solar que posee el emplazamiento seleccionado. Con el fin de conocerla, se consultó el sitio del ministerio de energía Explorador Solar [35] el cual proporciona los siguientes datos para el emplazamiento seleccionado:

Tabla 1: Ubicación del sitio seleccionado

Nombre	Universidad de O'Higgins
Latitud	-34.1644 °S
Longitud	-70.7416 °O
Elevación	503 m

Para poder obtener la cantidad de radiación disponible en el emplazamiento, se utilizó la herramienta de cálculo de generación del sistema fotovoltaico. Para esto, se ingresó la capacidad instalada $(3{,}5\ W)$, el tipo de arreglo (fijo inclinado), el tipo de montaje (estructura aislada) y se realizó una optimización de los ángulos (inclinación y azimut) con el fin de poder comparar la configuración mencionada con una configuración del panel instalado en forma horizontal.

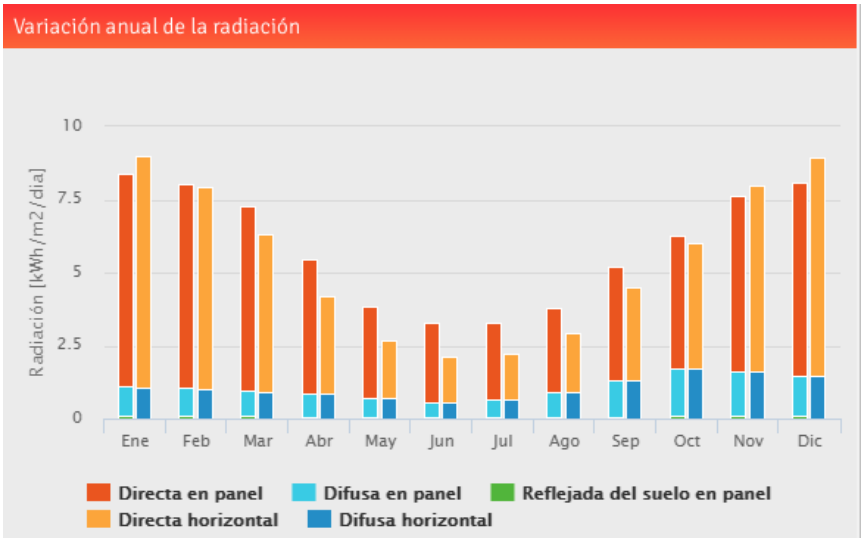


Figura 19: Promedio anual de la radiación solar para el emplazamiento seleccionado

Fuente: Explorador Solar

Como se aprecia en la Figura 19, la radiación directa horizontal es mayor en los meses de mayor generación, los cuales corresponden a diciembre y enero, pero en el resto de los

meses del año la radiación directa en el panel (con ángulos optimizados) es bastante mayor sobre todo en los meses de menor generación, que corresponden a junio y julio.

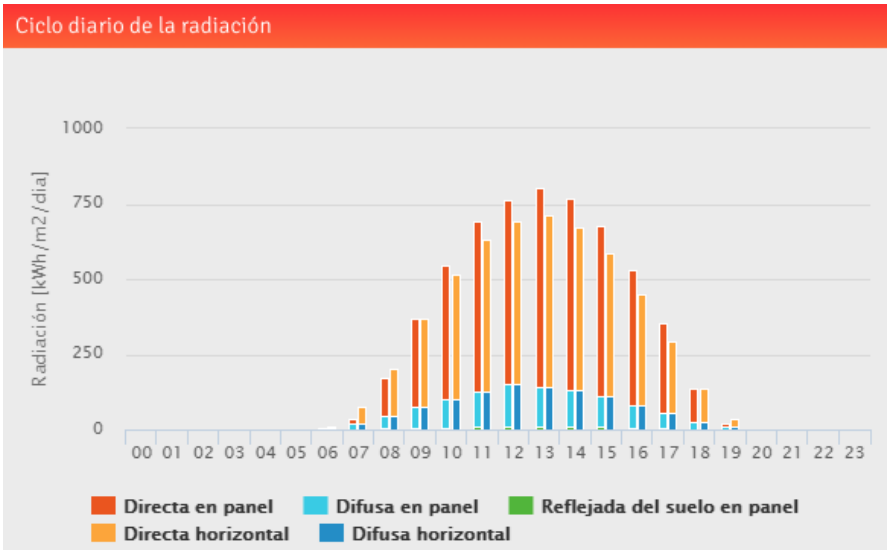


Figura 20: Promedio horario de la radiación solar para el emplazamiento seleccionado

Fuente: Explorador Solar

En la Figura 20, se puede apreciar el promedio horario de la radiación para el emplazamiento seleccionado, en donde la radiación directa en panel (para ángulos optimizados) es mayor a la radiación directa horizontal durante prácticamente la mayor parte del día, sobre todo en horarios de mayor generación. Es por las razones descritas anteriormente, que la configuración óptima para la instalación del panel con el fin de maximizar la producción de energía es fijo inclinado, con los ángulos de inclinación y azimut optimizados.

4.2.3. Ubicación óptima del panel solar.

Otro aspecto importante a considerar en la instalación del panel solar, es la ubicación óptima con la cual se puede maximizar la generación de energía eléctrica. Para esto, se utilizó la herramienta Explorador solar, la cual en uno de sus apartados entrega la opción de optimizar los ángulos para la instalación del panel solar según la ubicación entregada. Una vez realizado este proceso, seleccionando un tipo de arreglo fijo inclinado y un tipo de montaje aislado, entrega por resultado lo siguiente:

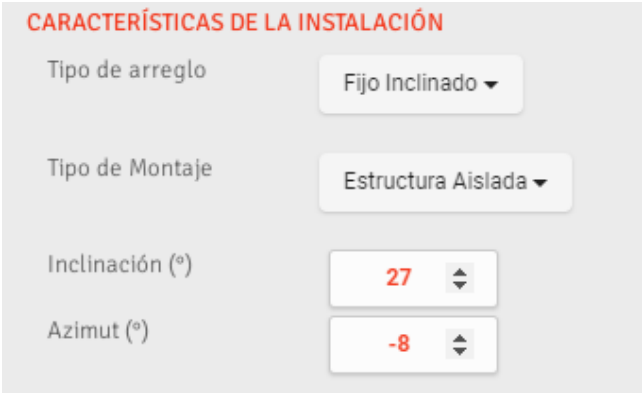


Figura 21: Optimización de ángulos para la instalación del panel solar según el emplazamiento seleccionado

Fuente: Explorador Solar

Por lo tanto, con una inclinación de $27°$ y un azimut de $-8°$ en la instalación del panel solar, se puede obtener un rendimiento óptimo de generación de energía del panel solar. Con esta configuración, se obtienen los siguientes datos de generación fotovoltaica mensual promedio:

$$Total\ diario = 0{,}01\ kWh$$
$$Total\ Anual = 5\ kWh$$
$$Factor\ de\ planta = 17\%$$

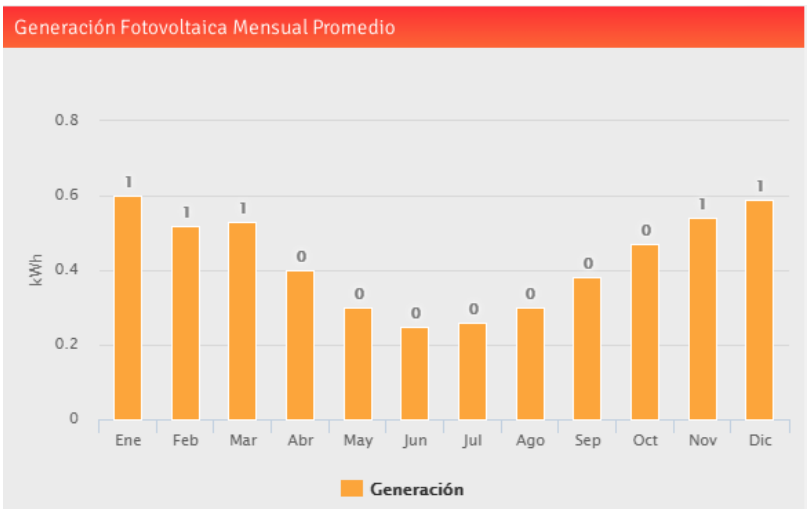


Figura 22: Generación fotovoltaica mensual promedio

Fuente: Explorador Solar

Como se aprecia en la Figura 22, los meses de mayor generación corresponden a diciembre y enero con un promedio de $0{,}6\ kWh$ y $0{,}59\ kWh$ respectivamente, en cambio los meses de menor generación corresponden a junio y julio con un promedio de $0{,}25\ kWh$ y $0{,}26\ kWh$ respectivamente.

4.3. Construcción e implementación

El objetivo principal de este proyecto, es la construcción de una estación meteorológica de bajo costo basada en Arduino, por lo que la presente sección tiene por objetivo explicar detalladamente este proceso junto con el de implementación del hardware que compone la estación. Esta última, se basa en la configuración de cada componente por separado, de modo que al ir avanzando con la implementación se facilite el proceso de anexar cada sketch correspondiente a cada componente hasta lograr la integración completa, formando finalmente la estación meteorológica de bajo costo basada en Arduino.

4.3.1. Configuración sensor de temperatura y humedad exterior AM2315.

El primer componente de la estación meteorológica Arduino que se configuró es el sensor de temperatura y humedad exterior, el cual corresponde al sensor AM2315.

Para comenzar, se realizó una búsqueda de información correspondiente al sensor en la cual destaca su hoja de datos, consultada en página del fabricante, etc. Pero es en la página Cactus [36] en donde se encuentra información relevante con respecto al sensor, como un esquema de conexión y el código proporcionado por el fabricante.

La conexión del sensor al Arduino se realiza siguiendo las instrucciones del fabricante detalladas en la página mencionada anteriormente, de la siguiente forma:

Tabla 2: Disposición y conexiones sensor AM2315

Cable	Pin Arduino
VCC (Rojo)	5V
GND (Negro)	GND
SDA (Amarillo)	A4
SCL (Blanco)	A5

Se deben conectar dos resistencia Pull-up de $10K\,\Omega$, la primera entre SDA y VCC, la segunda entre SCL y VCC. Esto para evitar el fenómeno conocido como "flotación"[4], es por esta razón que una resistencia Pull-up asegura que el pin trabaje en un estado conocido (HIGH o LOW) disminuyendo también la corriente de operación [37].

En la Figura 23 se puede observar el esquema de conexión del sensor AM2315.

[4] El fenómeno de flotación se produce cuando al leer el estado de un pin este se encuentra en un valor intermedio entre HIGH o LOW, por lo que no se puede determinar de manera exacta su verdadero estado. [37]

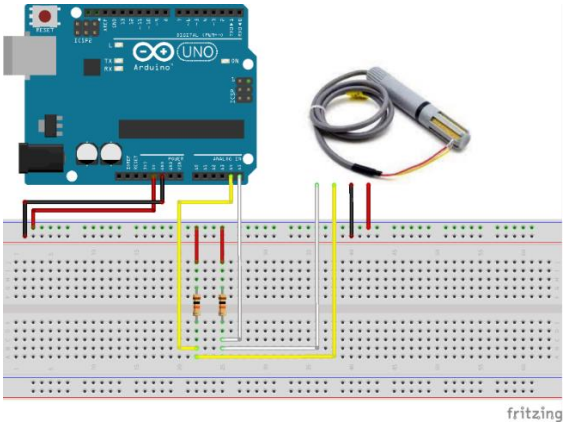


Figura 23: Esquema de conexión sensor AM2315

El sketch toma como base el entregado en [36] junto con la librería utilizada, realizando pequeñas modificaciones. Este código puede ser visualizado en Anexo 2 y entrega por resultado las siguientes mediciones de temperatura $[°C]$ y humedad $[HR]$:

```
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.40
Humedad ext(%):55.10 Tem ext(°C):22.40
Humedad ext(%):55.10 Tem ext(°C):22.40
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.40
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.40
Humedad ext(%):55.10 Tem ext(°C):22.30
Humedad ext(%):55.10 Tem ext(°C):22.40
```

Figura 24: Resultado prueba sensor de temperatura y humedad exterior

4.3.2. Configuración Arduino weather Shield.

El segundo componente de la estación meteorológica Arduino en configurarse es la "Arduino weather Shield". Como se menciona en 4.1.1.4 esta Shield posee integrada tres sensores, de los cuales se configuraron el de presión atmosférica (MPL3115A2) y el de temperatura y humedad (Si7021), este último será el encargado de medir internamente las temperatura interior del gabinete.

Para comenzar se realizó una búsqueda de información con respecto a la Arduino weather Shield. Es en la página del fabricante SparkFun [38] en donde se encuentra información relevante sobre esta Shield.

Al ser una Shield, está diseñada para montarse sobre el Arduino Mega mediante headers encajando perfectamente. En un principio no se poseían "headers hembra" por lo que se comenzó a configurar utilizando "headers macho" pero sin realizar la correcta sujeción mediante soldadura de estaño. El sketch reconocía correctamente la Shield pero entregaba valores incorrectos, por lo que al momento de conseguir los "headers hembra" y ser solados, la Shield (Figura 25) comienza a entregar datos correctos.

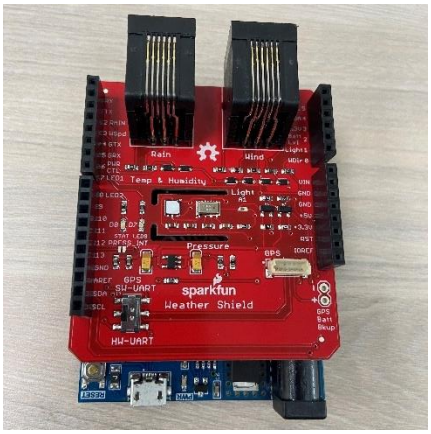


Figura 25: Implementación Arduino weather Shield

El sketch utiliza como base el código de Nathan Seidle actualizado por Joel Barlett que proporciona el fabricante en [38], pero solo toma algunas de las funciones que ahí se presentan. Es por esta razón, que se parte por configurar el sensor de temperatura y humedad (Si7021). Una vez configurado correctamente, se realiza el mismo procedimiento para el sensor de presión atmosférica (MPL3115A2). Una vez configurados los dos sensores, se integran ambos códigos en un solo sketch, que puede ser visualizado en Anexo 2, el cual entrega como resultado las siguientes mediciones de temperatura $[°C]$, humedad $[HR]$ y presión atmosférica $[Pa]$:

```
Presion(Pa):98789.25 Temp int(°C):22.77 Humedad int(%):53.59
Presion(Pa):98792.50 Temp int(°C):22.77 Humedad int(%):53.59
Presion(Pa):98792.50 Temp int(°C):22.77 Humedad int(%):53.60
Presion(Pa):98789.50 Temp int(°C):22.75 Humedad int(%):53.60
Presion(Pa):98788.75 Temp int(°C):22.76 Humedad int(%):53.60
Presion(Pa):98792.25 Temp int(°C):22.76 Humedad int(%):53.60
Presion(Pa):98791.00 Temp int(°C):22.77 Humedad int(%):53.62
Presion(Pa):98788.00 Temp int(°C):22.77 Humedad int(%):53.62
Presion(Pa):98793.50 Temp int(°C):22.77 Humedad int(%):53.59
Presion(Pa):98789.75 Temp int(°C):22.76 Humedad int(%):53.59
Presion(Pa):98790.75 Temp int(°C):22.77 Humedad int(%):53.60
Presion(Pa):98790.00 Temp int(°C):22.76 Humedad int(%):53.61
Presion(Pa):98790.00 Temp int(°C):22.78 Humedad int(%):53.60
Presion(Pa):98792.75 Temp int(°C):22.78 Humedad int(%):53.62
Presion(Pa):98794.25 Temp int(°C):22.76 Humedad int(%):53.62
Presion(Pa):98789.25 Temp int(°C):22.77 Humedad int(%):53.62
```

Figura 26: Resultados mediciones sensores de presión atmosférica, temperatura y humedad interior estación meteorológica Arduino

4.3.3. Configuración Datalogger.

El tercer componente de la estación meteorológica en configurarse es el datalogger. Tal como se menciona en la sección 4.1.1.2, el datalogger se basa en una Shield, por ende encaja perfectamente con el Arduino Mega, posee un lector de tarjeta SD en el cual se introduce una tarjeta para almacenar las lecturas de los sensores, además de un reloj de tiempo real RTC encargado de mantener la fecha y hora correspondiente.

Para poder implementar el datalogger, se necesitan realizar mediciones, es por lo que se configura almacenando los datos provistos por los tres sensores configurados anteriormente, los cuales son: AM2315 (temperatura y humedad exterior), MPL3115A2 (presión atmosférica) y Si7021 (temperatura y humedad interior). La implementación de estos sensores junto con el datalogger puede ser visualizado en la Figura 27.

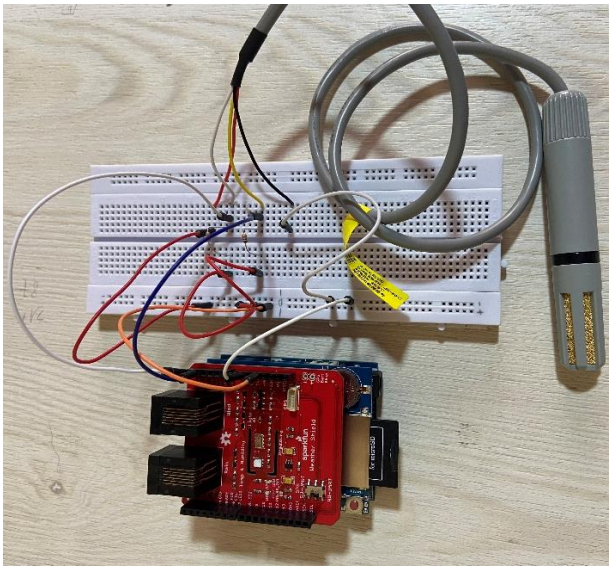


Figura 27: Implementación datalogger, Arduino weather Shield y AM2315

El sketch se basa en un tutorial de Prometec [39], el que considera solo la parte de configuración y escritura de datos en la tarjeta SD. Una vez verificado que la integración de los programas de los sensores funciona correctamente junto con la inclusión de la fecha y hora correspondiente (Figura 28), se realiza la prueba para comprobar que los valores obtenidos por los sensores se registran de forma correcta en un archivo Excel (Figura 29).

El sketch puede ser visualizado en el Anexo 2.

```
Fecha/Hora, Temp ext[°C], Humedad ext[HR], Presión[Pa], Temp int[°C], Humedad int[HR]
2021/11/12 13:30:4 22.00, 55.80, 98631.00, 23.44, 51.88
2021/11/12 13:30:5 21.90, 55.80, 98635.25, 23.41, 51.79
2021/11/12 13:30:6 21.90, 55.80, 98636.75, 23.39, 51.78
2021/11/12 13:30:8 22.00, 55.80, 98633.75, 23.39, 51.78
2021/11/12 13:30:9 21.90, 55.80, 98634.00, 23.39, 51.73
2021/11/12 13:30:10 22.00, 55.90, 98636.75, 23.40, 51.75
2021/11/12 13:30:11 21.90, 55.90, 98635.50, 23.39, 51.75
2021/11/12 13:30:12 21.90, 55.90, 98632.00, 23.40, 51.73
2021/11/12 13:30:14 21.90, 55.90, 98632.00, 23.40, 51.74
2021/11/12 13:30:15 22.00, 55.90, 98632.50, 23.38, 51.72
2021/11/12 13:30:16 22.00, 55.90, 98630.75, 23.40, 51.69
```

Figura 28:Resulta prueba de integración sensores

Fecha, Hora, Temp ext[C], Humedad ext[HR], Presion[Pa], Temp int[C], Humedad int[HR]		
2021/11/12 13:30:4 22.00, 55.80, 98631.00, 23.44, 51.88		
2021/11/12 13:30:5 21.90, 55.80, 98635.25, 23.41, 51.79		
2021/11/12 13:30:6 21.90, 55.80, 98636.75, 23.39, 51.78		
2021/11/12 13:30:8 22.00, 55.80, 98633.75, 23.39, 51.78		
2021/11/12 13:30:9 21.90, 55.80, 98634.00, 23.39, 51.73		
2021/11/12 13:30:10 22.00, 55.90, 98636.75, 23.40, 51.75		
2021/11/12 13:30:11 21.90, 55.90, 98635.50, 23.39, 51.75		
2021/11/12 13:30:12 21.90, 55.90, 98632.00, 23.40, 51.73		
2021/11/12 13:30:14 21.90, 55.90, 98632.00, 23.40, 51.74		
2021/11/12 13:30:15 22.00, 55.90, 98632.50, 23.38, 51.72		
2021/11/12 13:30:16 22.00, 55.90, 98630.75, 23.40, 51.69		

Figura 29: Resultado prueba de registro de mediciones

De esta forma se concluye que el datalogger cumple con la función de llevar el registro de los valores obtenidos por los sensores.

4.3.4. Configuración del sistema de alimentación.

El sistema de alimentación está compuesto por un panel solar, un regulador de carga y una batería, tal como se menciona en 4.1.2. Adicionalmente, se le añadió al sistema de alimentación tres sensores: dos sensores de voltaje, el primero es el encargado de monitorear el voltaje de salida del panel solar y entrada al regulador de carga, mientras que el segundo será el encargado de monitorear el voltaje de la batería. El otro sensor añadido es un sensor de corriente, encargado de monitorear la corriente a la salida del panel solar y a la entrada del regulador de carga. El regulador posee una salida USB por la cual se alimentará el Arduino (Figura 30).

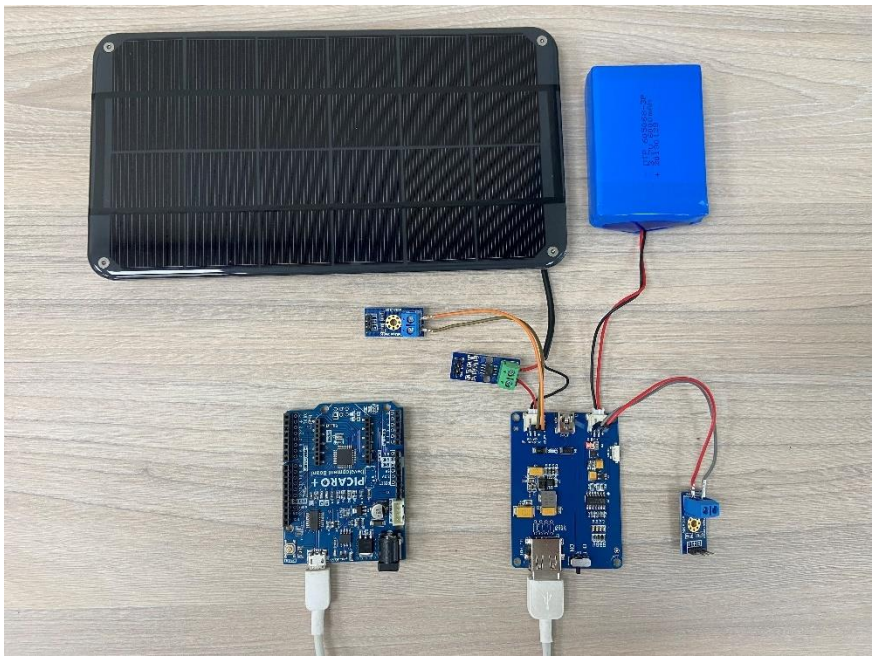


Figura 30: Sistema de alimentación

Se realizó una prueba para comprobar el estado de los sensores del sistema de alimentación y analizar los valores de voltaje del panel solar, voltaje de la batería y corriente producida por el panel solar entregados por los sensores, dando por resultado lo expuesto en la Figura 31. Estas pruebas, fueron realizadas en un ambiente cerrado, con poca luz natural y con la batería totalmente descargada.

```
Sensores de voltaje                Sensor de corriente Panel
Voltaje panel solar[V]:            Corriente: 0.07 A
3.81                               Corriente: 0.15 A
Voltaje Bateria[V]:                Corriente: 0.15 A
1.64                               Corriente: 0.15 A
Voltaje panel solar[V]:            Corriente: 0.15 A
3.74                               Corriente: 0.15 A
Voltaje Bateria[V]:                Corriente: 0.15 A
1.66                               Corriente: 0.15 A
Voltaje panel solar[V]:            Corriente: 0.15 A
3.81                               Corriente: 0.15 A
Voltaje Bateria[V]:                Corriente: 0.15 A
1.64                               Corriente: 0.15 A
```

Figura 31: Resultados prueba del sistema de alimentación

4.3.5. Kit de sensores para estación meteorológica.

Este kit se compone de tres instrumentos, los cuales son: un anemómetro, el que será el encargado de medir la intensidad del viento. Una veleta, con la cual se determinará la dirección en que sopla el viento y un pluviómetro, el cual será el encargado de contabilizar la cantidad de precipitación caída. Se comenzó por configurar los dos primeros instrumentos, partiendo por el montaje de estos en la estructura, lo que puede ser observado en la Figura 32.

Figura 32: Anemómetro y veleta

Una vez montados los instrumentos, se obtuvo un sketch que utiliza como base el código de Nathan Seidle actualizado por Joel Barlett que proporciona el fabricante en [38]. Se comenzó por configurar el anemómetro, para esto se estudió el sketch mencionando anteriormente con el fin de extraer las funciones y configuraciones útiles, como por ejemplo el cálculo de la rapidez instantánea del viento y el promedio de dos minutos.

Para la veleta resultó un poco más difícil, debido a que se tuvo que acudir a otro método mucho más preciso, el cual se detalla en 3.1.4. De todas maneras se extrajeron configuraciones útiles, como la dirección instantánea del viento y la idea general del promedio de dos minutos.

En el caso del pluviómetro, este solo se montó y se realizó una prueba de su correcto funcionamiento, esto debido a razones que se mencionarán en el apartado 4.3.13.

Las pruebas para la configuración de estos instrumentos se realizaron en un ambiente controlado de laboratorio, es por lo que en este apartado no se presentan resultados de los datos obtenidos por estos instrumentos, pero sí podrán ser observados en los apartados 5.4 y 5.5.

4.3.6. Ethernet Shield y servidor ThingSpeak.

Como se mencionó en 4.1.1.3, la ethernet Shield permite al Arduino conectarse a internet mediante un cable ethernet RJ-45. Se seleccionó esta opción debido a que una conexión por ethernet es más estable que, por ejemplo, una conexión wifi. De este modo, se utiliza ThingSpeak para poder almacenar los datos proporcionados por lo sensores y que estos sean visualizados en el momento que se estime conveniente en la página web o en la aplicación para smartphone ThingView.

Para la configuración, se comienza por colocar la Shield sobre el Arduino Mega, luego se conecta el cable ethernet. Para comenzar a utilizar ThingSpeak es necesario registrarse, iniciar sesión, crear un canal y configurar los "fields" o campos a utilizar. Debido a que, se utiliza una licencia gratuita de ThingSpeak, esta permite almacenar datos en ocho campos diferentes, es por esta razón que los campos seleccionados son los siguientes:

- Field 1: Temperatura exterior (sensor AM2315)
- Field 2: Humedad exterior (Sensor AM2315)
- Field 3: Temperatura interior (Sensor Si7021)
- Field 4: Intensidad del viento, promedio de dos minutos (Anemómetro)
- Field 5: Dirección del viento promedio de dos minutos (Veleta)
- Field 6: Voltaje panel solar (Sensor FZ0430)
- Field 7: Corriente panel solar (Sensor ACS712)

- Field 8: Voltaje Batería (Sensor FZ0430)

Una vez realizada la configuración de los fields, es necesario obtener el código del canal y el apikeys, que es un código que permite la escritura de datos en el canal. Para poder enviar los datos de las mediciones obtenidas por el Arduino, es necesario instalar la librería ThingSpeak en la IDE de Arduino. Una vez realizado esto, se utilizó un sketch de ejemplo de la librería para configurar el sketch que enviará los datos al canal de ThingSpeak. Estos datos pueden ser enviados como mínimo cada veinte segundos. En la Figura 33 se puede observar el resultado de una medición de temperatura y humedad realizada durante dos horas por el sensor AM2315 con datos enviados cada dos minutos.

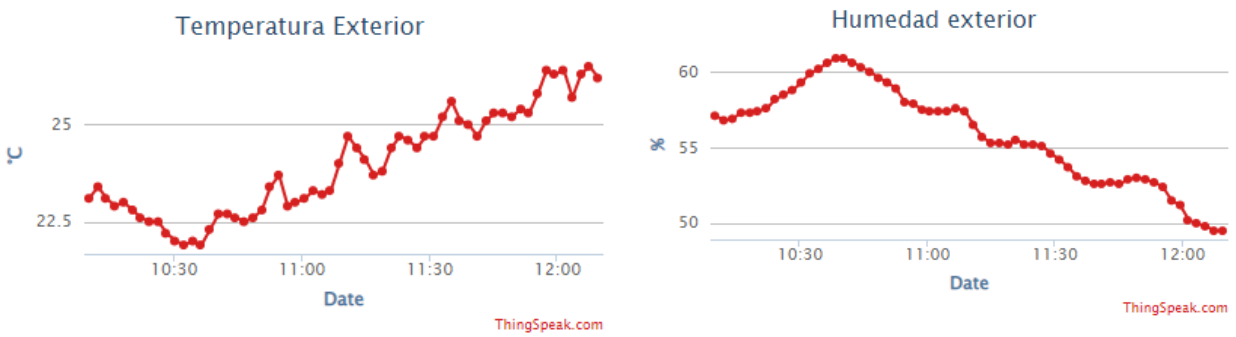


Figura 33: Visualización de temperatura y humedad exteriores en ThingSpeak

4.3.7. Carcasa meteorológica.

Esta carcasa cumple la función de proteger de la radiación solar y de la lluvia al sensor de humedad y temperatura AM2315, debido a que en las especificaciones entregadas por el fabricante, detalla que el sensor no se encuentra calificado como resistente a la intemperie [40]. Se realizó una búsqueda de diseños de carcasas de código abierto hechas para el sensor, es por lo que de la página Thingverse [41] se seleccionó un diseño para impresión en 3D. Una vez obtenidos los archivos, estos fueron impresos en la Fábrica Digital O'Higgins. El filamento utilizado es PLA 850 de color negro, el cual posee las siguientes características, las cuales fueron extraídas desde la hoja de datos [42] proporcionada por el fabricante:

Propiedades térmicas:

- Heat distorsión temperature (HDT): comprendida entre $80\ a\ 90\ °C$ para una pieza 100% rellena, cristalizada a $110°C/15min$. Esto significa que al exponer una pieza a una temperatura igual o menor, esta sufrirá una nula o mínima deformación. Lo que comparado con el PLA estándar o ABS, los cuales poseen una HDT con valores entre $50\ a\ 80\ °C$, resulta ser mucho más resistente y confiable.

Una vez impresas las partes que componen la carcasa, estas fueron recubiertas con pintura en spray de color blanco, con el motivo de disminuir al máximo la absorción de radiación evitando de esta forma que el sensor arroje lecturas erróneas de la temperatura. Una vez secas las piezas, estas fueron ensambladas, como se puede apreciar en la Figura 34.

Figura 34: Carcasa meteorológica

4.3.8. Montaje de componentes en placa MDF.

El Arduino Mega es el corazón de la estación meteorológica de bajo costo, pero junto a este existen componentes que también necesitan ser instalados al interior del gabinete, tales como: regulador de carga, batería, sensores de voltaje, sensor de corriente, protoboard. Estos componentes fueron montados en una placa MDF, la cual se colgará al interior del gabinete con un sistema de ganchos, con el fin de que su extracción sea de manera sencilla (Figura 35).

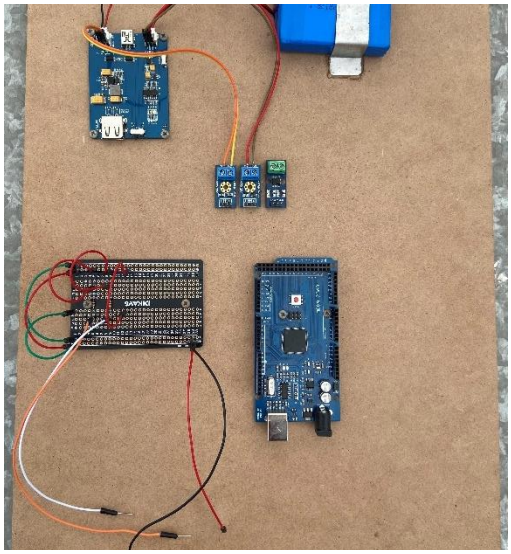


Figura 35: Montaje de componentes en placa MDF

4.3.9. Montaje de gabinete y componentes exteriores e interiores.

Tal como se mencionó en 4.2.1, el gabinete fue instalado en el emplazamiento seleccionado por el equipo de mantención de la universidad. Este gabinete, posee todas las condiciones para ser instalado y ser expuesto a la intemperie. Sobre este, se instaló el panel solar, a la izquierda se montó la estructura que sostiene al pluviómetro de balancín, al anemómetro y la veleta (estos últimos con orientación norte), mientras que al lado derecho se montó la carcasa del sensor AM2315 junto a este (Figura 36). Al interior del gabinete, se instaló

la placa MDF y se realizaron las conexiones de alimentación del Arduino, de comunicación con los sensores exteriores y la conexión de ethernet, como se puede apreciar en la Figura 36, además se implementó un pequeño ventilador, el cual tiene como finalidad realizar una circulación de aire que permita reducir la temperatura del interior del gabinete.

Figura 36: Montaje gabinete y componentes exteriores e interiores

4.3.10. Determinación del consumo de la estación meteorológica Arduino.

Para medir el consumo de la estación meteorológica de bajo costo basada en Arduino, se alimentó con una fuente continua de $5\,V$ y $1\,A$ directamente a los pines V_{in} y GND del Arduino Mega. Con un multímetro, se realizó una medición del voltaje y de la corriente proporcionados por la fuente, resultando las siguientes mediciones:

Figura 37: Medición del consumo de la estación meteorológica de bajo costo basada en Arduino

Como se aprecia en la Figura 37, el voltaje proporcionado por la fuente es de $5{,}18\,V$ mientras que la corriente proporcionada por la fuente es de $0{,}195\,A$. De esta forma, se determinó que el consumo de la estación meteorológica Arduino es de $1{,}01\,W$.

4.3.11. Modo ahorro de energía.

Con el fin de disminuir el consumo de energía de la estación meteorológica, se implementa el modo ahorro de energía o "Sleep mode" en Arduino. Para esto, es necesario instalar la librería "LowPower" que junto al modo "Power Down" ponen al Arduino Mega en modo ahorro de energía durante cuatro segundos, una vez transcurrido el tiempo determinado, el Arduino ejecuta el resto de código que se encuentra en la función Loop. El "Sleep Mode" dispone de seis configuraciones para poner al Arduino en modo ahorro de energía, dentro de estas, la configuración seleccionada "Power Down" es la que presenta un menor consumo de energía en relación con las otras configuraciones, debido a que mantiene encendida una menor cantidad de componentes del Arduino.

4.3.12. Determinación del consumo de la estación meteorológica Arduino (modo ahorro de energía).

Para medir el consumo de la estación meteorológica en modo ahorro de energía, se realizó el mismo procedimiento seguido en 4.3.10, resultando las siguientes mediciones:

Figura 38: Medición del consumo de la estación meteorológica de bajo costo basada en Arduino (modo ahorro de energía)

Como se muestra en la Figura 38, el voltaje proporcionado por la fuente es de $5{,}18\,V$, mientras que la corriente proporcionada por esta es de $0{,}172\,A$. De esta forma, se determina que el consumo de la estación meteorológica Arduino es de $0{,}89\,W$. Es en esta última medición, se puede visualizar el efecto que causa la implementación del modo ahorro de energía, en donde el consumo por parte de la estación meteorológica se reduce en un $13{,}5\,\%$.

4.3.13. Cambios con respecto a la implementación inicial

Tras la realización de las primeras pruebas de funcionamiento de la estación meteorológica de bajo costo basada en Arduino, surge la necesidad de realizar modificaciones a la implementación detallada anteriormente, debido a que los resultados no fueron los esperados. Las modificaciones que se realizaron son las siguientes:

- Se trasladó de lugar la estructura que sostiene al anemómetro, veleta y pluviómetro, desde la izquierda hacia la derecha del gabinete, montándose a un costado de la carcasa meteorológica. Este cambio se debe a que durante la tarde, la estructura generaba sombra sobre el panel solar, reduciendo la generación de energía de la estación meteorológica. Tras la realización de este cambio se solucionó el problema.
- Se decidió no integrar el datalogger, el motivo es que tras realizar la configuración mostrada en 4.3.3, en pruebas realizadas posteriormente, presentó una desconfiguración de la fecha y la hora, problema que no pudo ser resuelto. Otro motivo por el cual se decide no integrar el datalogger es que se implementa la Arduino ethernet Shield, la cual permite al Arduino conectarse a internet, enviar los datos medidos a un servidor y lograr realizar un seguimiento de forma remota del estado y mediciones de la estación meteorológica Arduino.
- Se decidió no agregar la medición de precipitaciones realizada por el pluviómetro de balancín, esto debido a que actualmente nos encontramos en la estación de verano, en donde existe muy poca probabilidad de lluvia, además como el servidor ThingSpeak cuenta con 8 fields disponibles para almacenar datos, se decidió dar prioridad a otras variables, es por esta razón que se desmontó el pluviómetro de balancín de la estación meteorológica.
- Tras varias pruebas de rendimiento del sistema de alimentación y tras concluir que un panel solar no generaba el excedente de corriente para realizar la carga de la batería, se decidió agregar otro panel solar de las mismas características en paralelo, logrando así

sumar las corrientes que generan ambos paneles y mantener el voltaje. Para realizar esta nueva configuración, fue necesario agregar dos diodos a la salida de cada panel, con el fin de tener una configuración de diodo de bloqueo. Este circuito se realizó en la protoboard existente. Tras la realización de esta modificación, se solucionó el problema logrando que el sistema de alimentación opere de la forma esperada.

- De la misma forma, se agregó otra batería de las mismas características que la que se implementó inicialmente, en una configuración paralela, con el fin de mantener el voltaje pero aumentar al doble la corriente que pueden proveer al Arduino. Este circuito se realizó en una nueva protoboard. Si bien el sistema funcionaba de manera correcta con una batería, esta modificación se realizó pensando en la disminución de la producción de energía durante los meses de invierno, como se evidencia en los apartados 4.2.2 y 4.2.3.
- Por último, tras la realización de la tercera prueba de rendimiento del sistema de alimentación, se decide implementar el modo de ahorro de energía en Arduino, con el fin de disminuir el consumo de energía de la estación meteorológica y poder lograr un mejor rendimiento del sistema de alimentación.

Todos los cambios con respecto a la implementación inicial se pueden apreciar en la Figura 39 y Figura 40.

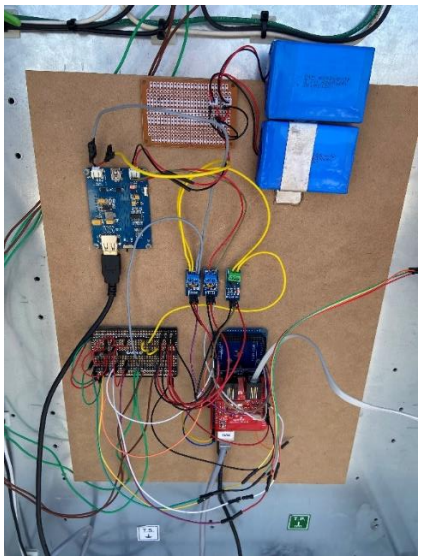


Figura 39: Cambios con respecto a la implementación inicial

Figura 40: Cambios con respecto a la implementación inicial

5. Resultados y análisis

En esta sección, se analizarán los datos obtenidos por la estación meteorológica de bajo costo basada en Arduino. Estos datos se obtuvieron durante seis días corridos comenzando el viernes 24 de diciembre del año 2021 y culminando el viernes 31 de diciembre del año 2021. Los datos a analizar corresponden a los almacenados en el servidor ThingSpeak en cada field configurado.

5.1. Temperatura del aire

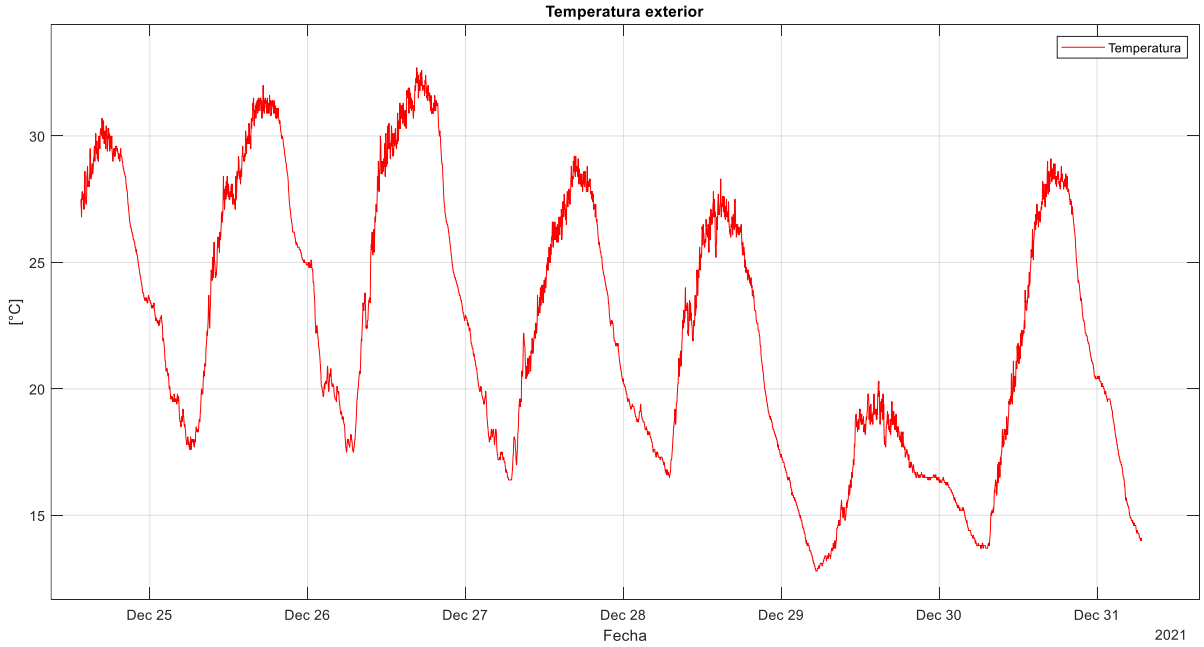


Figura 41: Registro de temperatura del aire

En la Figura 41 se observa, el registro a lo largo del tiempo señalado de las mediciones de la temperatura del aire. Los valores obtenidos corresponden a valores normales para la estación de verano, en la mayoría de los días sigue un patrón casi constante, salvo en el día 29 de diciembre. En este día la temperatura disminuyó considerablemente, alcanzando una temperatura máxima de $20{,}3°C$ y una temperatura mínima de $12{,}8°C$. El día que la temperatura alcanzó un mayor valor corresponde al 26 de diciembre, alcanzando un valor máximo de $32{,}7°C$.

En la siguiente tabla se podrá observar el registro de la temperatura máxima y mínima registradas por la estación meteorológica Arduino.

Tabla 3: Registros máximos y mínimos de la temperatura del aire

Día	Registro (Max/Min)
24 de diciembre	30,7°C/NA
25 de diciembre	32°C/17,6°C
26 de diciembre	32,7°C/17,5°C
27 de diciembre	29,2°C/16,4°C
28 de diciembre	28,3°C/16,5
29 de diciembre	20,3°C/12,8°C
30 de diciembre	29,1°C/13,7°C

5.2. Humedad relativa del aire

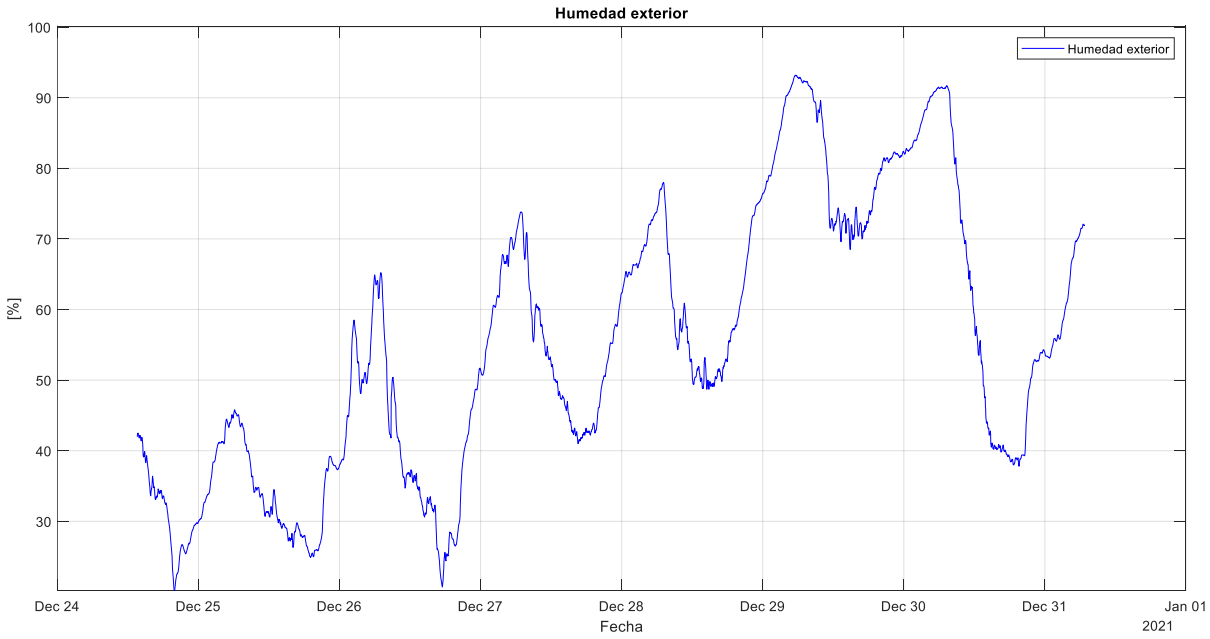


Figura 42: Registro de humedad relativa del aire

En la Figura 42, se observa el registro a lo largo del periodo de tiempo señalado de la humedad relativa del aire. Los valores obtenidos corresponden a un comportamiento normal de la variable, es decir que la humedad relativa se comporta de manera inversa a la temperatura, mejor dicho, qué a mayor temperatura disminuye la humedad relativa y cuando desciende la temperatura, la humedad relativa aumenta. El día que presentó una mayor humedad tanto en el inicio del día como al final de este es el 29 de diciembre, lo que refleja que hubo una disminución en la temperatura del aire, tal como se menciona en el apartado 5.1.

En la siguiente tabla se podrá observar el registro de la humedad relativa máxima y mínima registradas por la estación meteorológica Arduino.

Tabla 4: Registros máximos y mínimos de la humedad relativa del aire

Día	Registro (Max/Min)
24 de diciembre	NA/20 %
25 de diciembre	45,8 %/24,9 %
26 de diciembre	65,2 %/20,7 %
27 de diciembre	73,8 %/41 %
28 de diciembre	78 %/48,7 %
29 de diciembre	93,2 %/68,5 %
30 de diciembre	91,7 %/37,8 %

5.3. Temperatura interior

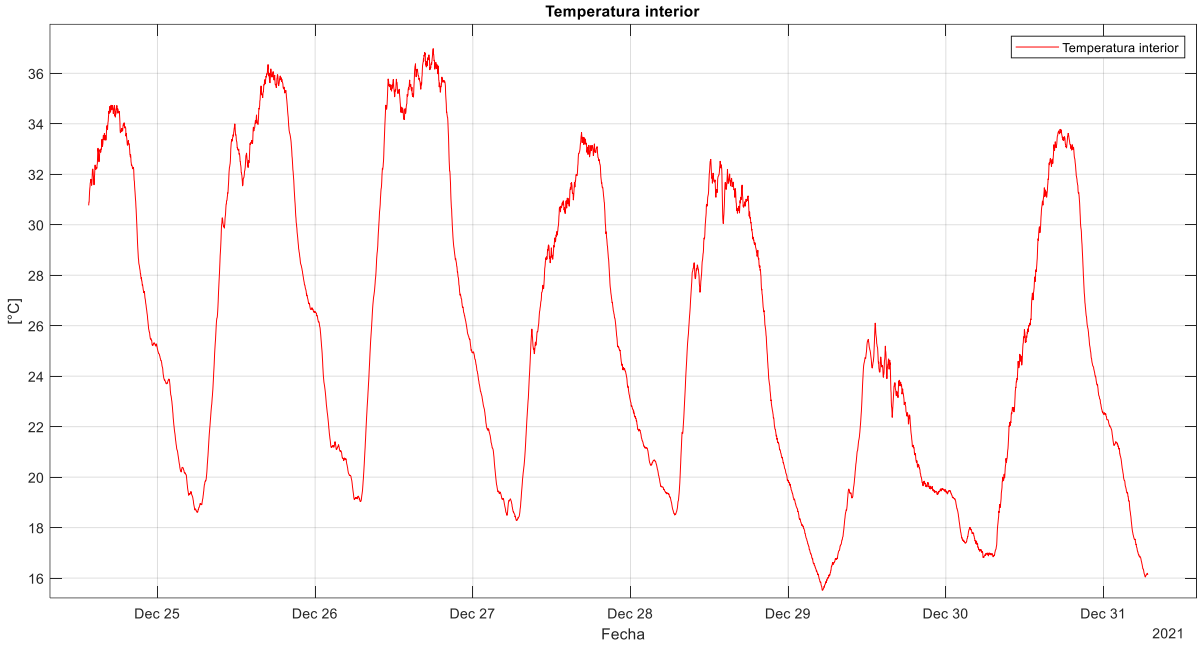


Figura 43: Registro de temperatura interior

En la Figura 43, se observa el registro a lo largo del periodo señalado de las mediciones de la temperatura interior de la estación meteorológica Arduino. Los valores obtenidos corresponden a valores esperados, debido a que el sensor está expuesto al calor que generan los componentes del microcontrolador Arduino, además de la temperatura que se genera al interior del gabinete. Es importante señalar, que los valores obtenidos durante la medición cumplen con los rangos de temperatura de funcionamiento permitidos para los componentes internos de la estación meteorológica Arduino.

5.4. Intensidad del viento (promedio dos minutos)

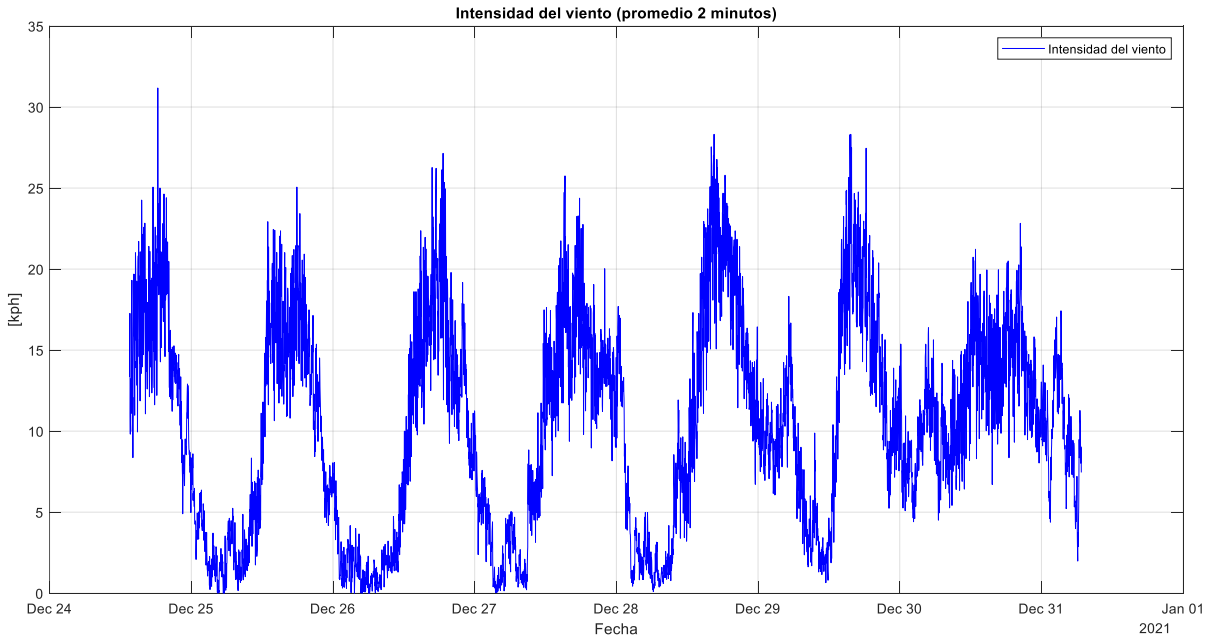


Figura 44: Registro de la intensidad del viento (promedio 2 minutos)

En la Figura 44 se observa, el registro a lo largo del periodo señalado de las mediciones del promedio de dos minutos de la intensidad del viento. Como se puede apreciar, la velocidad del viento no sigue un patrón reiterativo y puede cambiar entre mediciones.

En la Figura 45 se observa, un histograma de variación del promedio de dos minutos de la intensidad del viento, en donde en el eje Y corresponde al número de mediciones de cada intensidad y el eje X corresponde a la intensidad en kilómetros por hora (kph). Cada barra acumula un rango de intensidades, por ejemplo 1 kph corresponde a las intensidades comprendidas entre [0,1] kph. Según la información proporcionada por el histograma, las intensidades predominantes que presentan mayor frecuencia son las comprendidas entre [0,3] kph y [11,15] kph con más de 300 mediciones para cada intervalo.

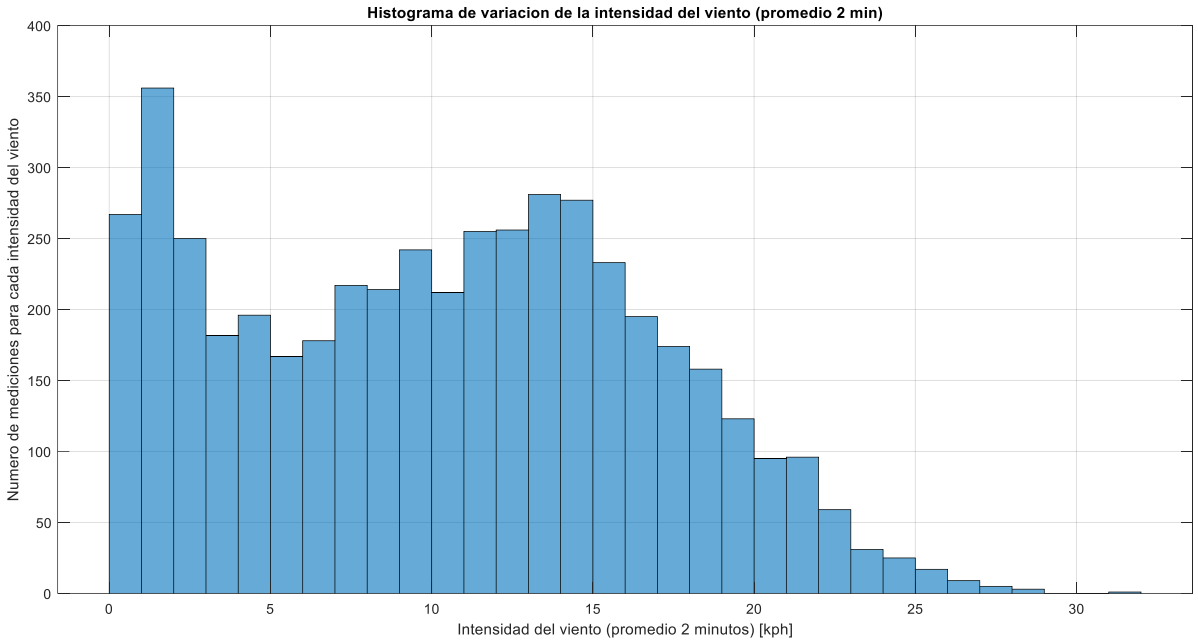


Figura 45: Histograma de la intensidad del viento (promedio 2 minutos)

5.5. Dirección del viento (promedio dos minutos)

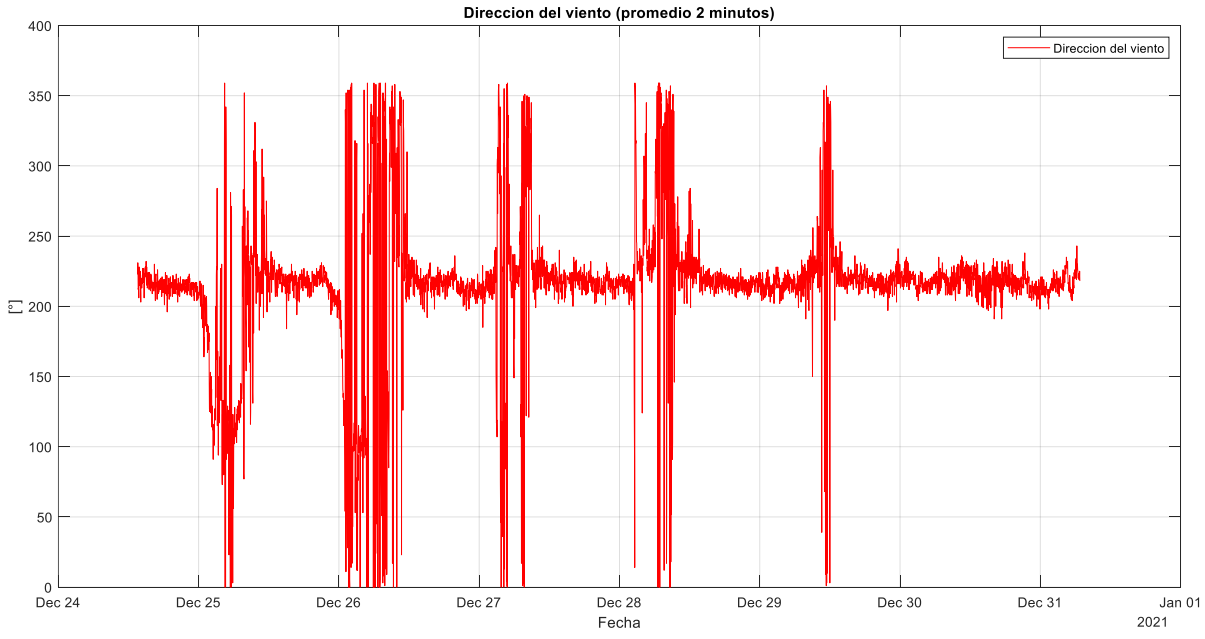


Figura 46: Registro de la dirección del viento (promedio 2 minutos)

En la Figura 46 se puede observar el registro a lo largo del periodo señalado de las mediciones del promedio de dos minutos de dirección del viento. En el gráfico, se evidencia que la dirección del viento posee una dirección preponderante, pero que puede variar por largos periodos como se aprecia en el día 26 de diciembre.

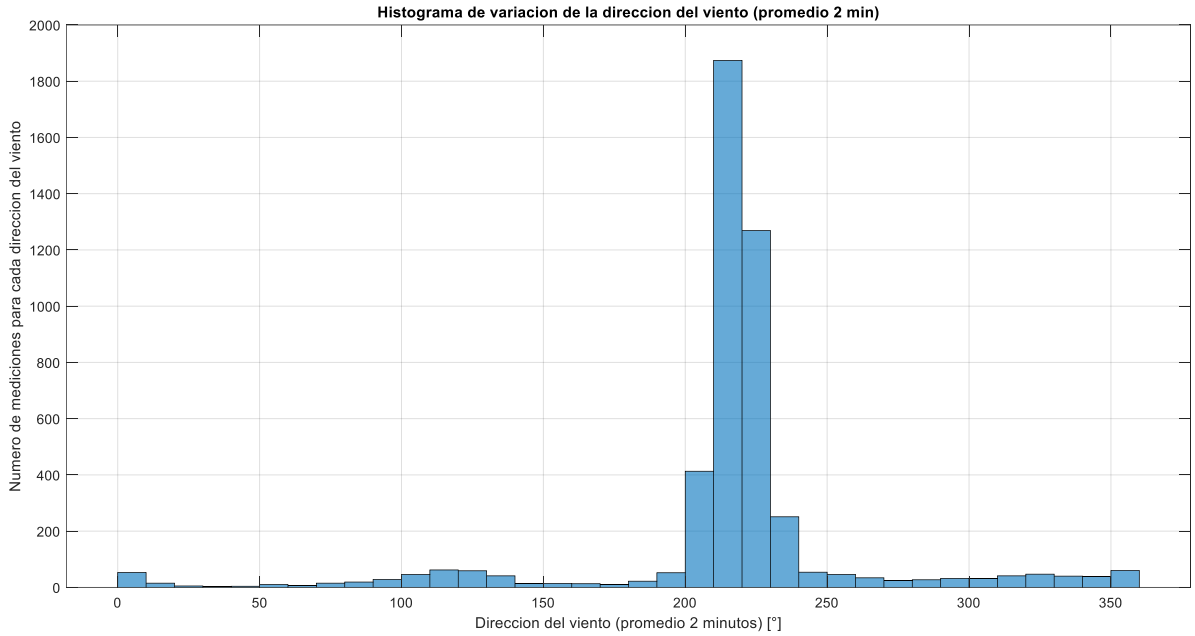


Figura 47: Histograma de la dirección del viento (promedio dos minutos)

En la Figura 47 se observa, un histograma de variación del promedio de dos minutos de la dirección del viento, en donde en el eje Y corresponde al número de mediciones de cada dirección y el eje X corresponde a un intervalo de 10° de dirección del viento. Como se puede apreciar, la dirección del viento preponderante se encuentra en el intervalo de 210° a 220° con más de 1800 mediciones, seguido por el intervalo de 220° a 230°, el cual posee más de 1200 mediciones a lo largo del periodo de tiempo señalado.

5.6. Voltaje panel solar

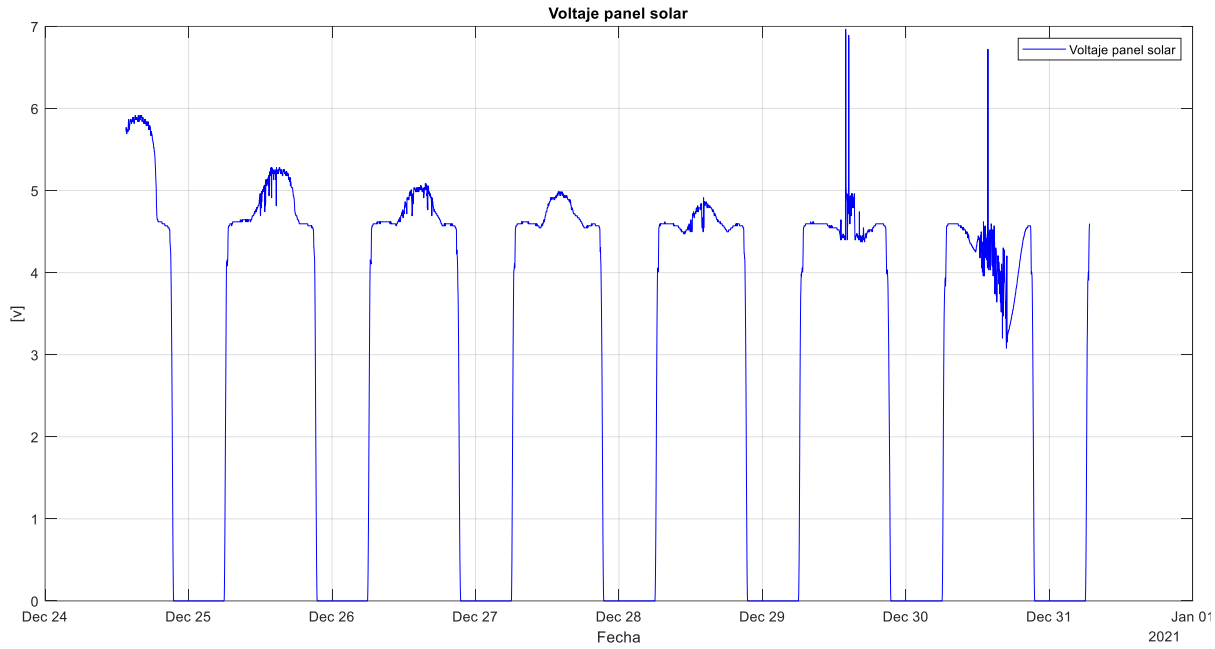


Figura 48: registro del voltaje producido por el panel solar

En la Figura 48 se observa, el registro a lo largo del periodo señalado de las mediciones del voltaje producidos por los paneles solar. Como se puede apreciar, el voltaje sigue un comportamiento casi constante entre los días 24 y 28 de diciembre, debido a que las condiciones meteorológicas fueron similares para estos días. No así, los días 29 y 30 de diciembre, en donde se puede observar una disminución importante de los valores de voltaje producidos. En el día 29, se presentó nubosidad gran parte del día, lo que implicó los cambios en la temperatura y humedad durante el transcurso del día (Figura 41 y Figura 42), disminuyendo considerablemente la producción de energía de los paneles solares debido a que la radiación que absorben no es directa, si no que mayormente captan la radiación difusa. Para el día 30 las condiciones meteorológicas volvieron a valores normales para la estación de verano, pero la disminución y variación del voltaje producido por el panel solar durante gran parte del día se debe a la presencia de incendios forestales en la comuna vecina de Machalí, lo que provoca un aumento de los aerosoles (partículas microscópicas que flotan en el aire) en la atmósfera. Los aerosoles, disminuyen la radiación solar que llega a la superficie del panel solar

de manera directa, provocando una disminución importante en la producción de energía por parte de estos [43].

5.7. Corriente panel solar

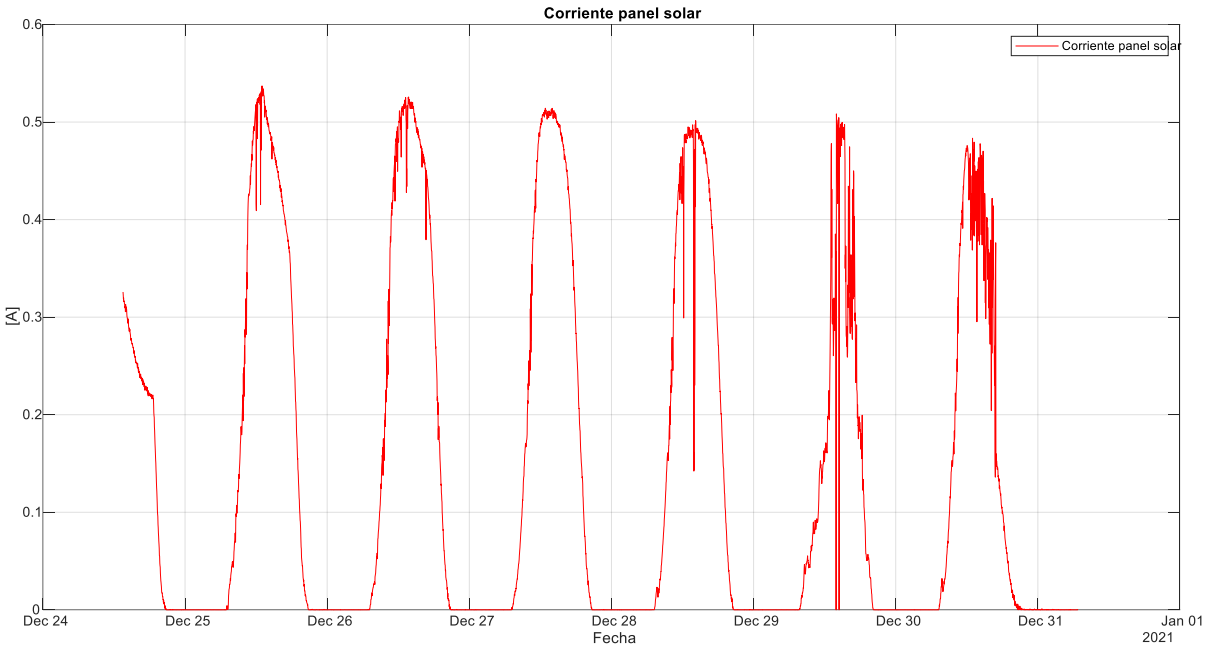


Figura 49: Registro de la corriente producida por el panel solar

En la Figura 49 se observa, el registro a lo largo del periodo señalado de las mediciones de la corriente producida por los paneles solares. Como se puede apreciar, al igual que el voltaje generado por los paneles, la corriente sigue un patrón casi constante entre los días 25 y 28 de diciembre, con la diferencia que se nota una pequeña disminución en la corriente máxima alcanzada en cada día. En el día 29 de diciembre se observa una mayor variabilidad de la corriente producida siendo cero en algunos periodos del día. Mismo caso ocurre en día 30 de diciembre pero con la diferencia que la corriente se mantiene en un rango mayor que el día anterior. Se puede prever, que este comportamiento a lo largo del periodo de muestreo afecta a la carga de las baterías, el cual podrá ser observado en el siguiente apartado.

5.8. Voltaje batería

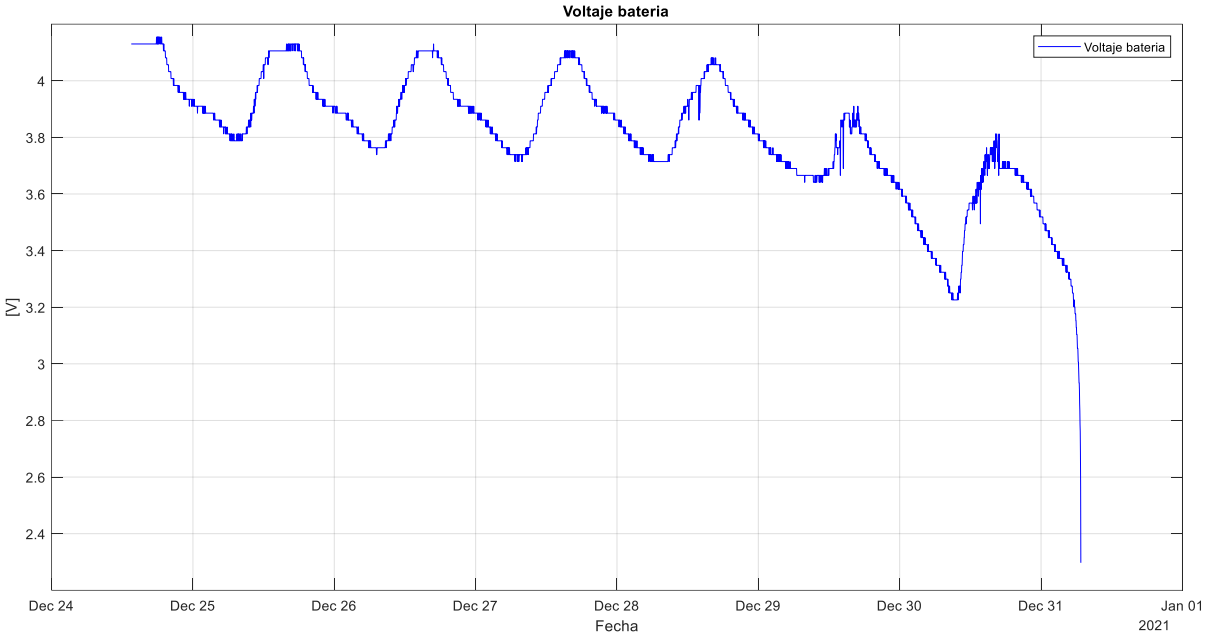


Figura 50: Registro del voltaje de la batería

En la Figura 50 se observa el registro a lo largo del periodo señalado de las mediciones del voltaje de la batería. Como se puede apreciar el comportamiento del voltaje de la batería es el esperado, cargándose durante el día mientras los paneles solares proporcionan la energía y descargándose cuando los paneles dejan de proporcionar esta energía. Entre los días 24 y 28 de diciembre, el voltaje de la batería presenta un comportamiento similar, con el detalle que durante el transcurso de este periodo, el voltaje máximo alcanzado por la batería se mantiene por periodos cada vez más breves, lo que provoca que día a día el voltaje mínimo alcanzado en el periodo mencionado sea menor en comparación al día anterior. Esto es provocado por el comportamiento de la corriente generada por el panel solar, mencionado en 5.7. En los días 29 y 30 se observa que la batería alcanza los menores valores de voltaje, debido a los factores que afectaron la producción de energía por los paneles solares (nubosidad e incidencia de los efectos provocados por incendios forestales). Por esta razón, en el día 31 de diciembre es donde el voltaje de la batería alcanza sus valores mínimos lo que indica que la batería se

encuentra totalmente descargada. A continuación, se presenta una tabla con los valores máximos y mínimos de voltajes alcanzados por la batería durante el periodo descrito:

Tabla 5: Registro de voltajes máximos y mínimos de la batería

Día	Voltaje máximo [V]	Voltaje mínimo [V]
24 de diciembre	4,15	NA
25 de diciembre	4,13	3,78
26 de diciembre	4,10	3,73
27 de diciembre	4,10	3,71
28 de diciembre	4,08	3,71
29 de diciembre	3,91	3,64
30 de diciembre	3,81	3,22
31 de diciembre	NA	2,29

5.9. Rendimiento sistema de alimentación

Antes de obtener la configuración final del sistema de alimentación, se realizaron pruebas de rendimiento en las cuales se variaba la cantidad de variables a medir, la frecuencia de comunicación con el servidor ThingSpeak y la composición del sistema de alimentación, a continuación se presentan los resultados obtenidos.

5.9.1. Prueba de rendimiento n°1.

La primera prueba de rendimiento se realizó midiendo todas las variables meteorológicas, además de las variables correspondientes al sistema de energía. La frecuencia de comunicación utilizada para el envío de datos hacia el servidor ThingSpeak es de un minuto. El sistema de alimentación se compone por un panel solar y una batería. A continuación, se

puede observar el registro de las mediciones de voltaje producido por el panel solar y el voltaje de la batería.

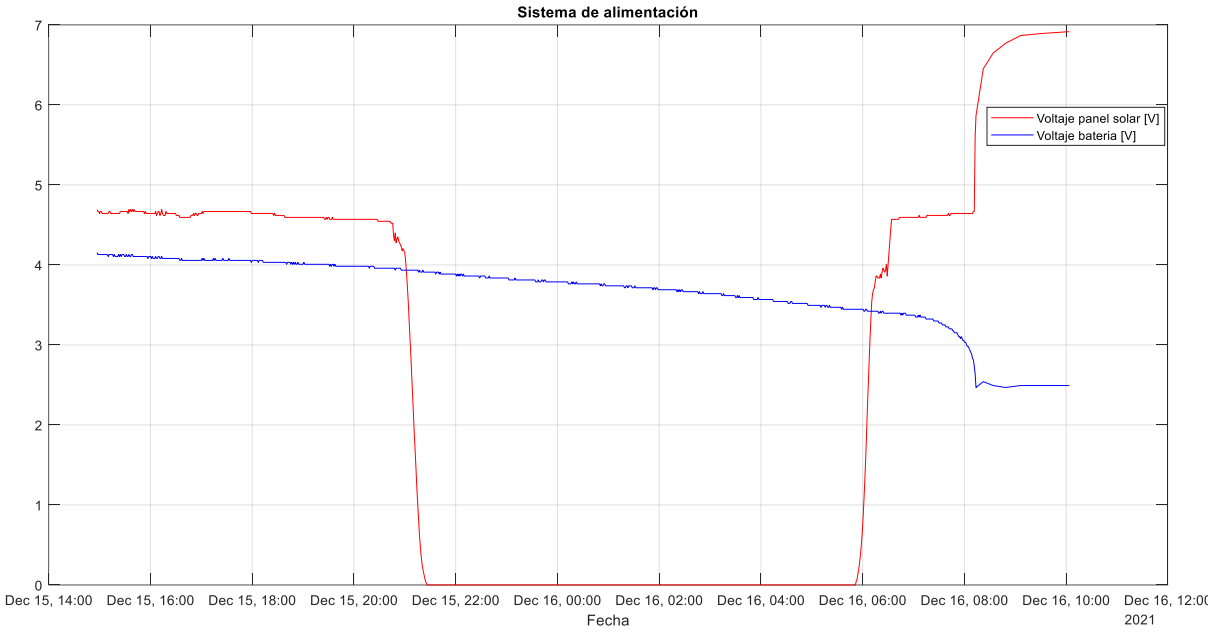


Figura 51: Prueba de rendimiento n°1

Como se aprecia en la Figura 51, la prueba comienza alrededor de las 15:00 horas del día 15 de diciembre con la batería totalmente cargada. El rendimiento del sistema de alimentación con la configuración mencionada no es el esperado, debido a que el voltaje de la batería comienza a bajar inmediatamente tras dejar funcionando la estación meteorológica Arduino, en donde el comportamiento del voltaje de la batería es similar al de una recta con pendiente negativa, descargándose totalmente pasadas las 8:00 del día 16 de diciembre. Es por esta razón, que se determina la utilización de un panel solar adicional de las mismas características que el utilizado inicialmente, con el propósito de que con esta configuración se logre producir la energía necesaria para la carga de la batería.

5.9.2. Prueba de rendimiento n°2.

La segunda prueba de rendimiento se realizó midiendo las variables meteorológicas temperatura y humedad del aire exterior, además de las variables correspondientes al sistema de energía (voltaje del panel solar y voltaje de la batería). La frecuencia de comunicación utilizada para el envío de datos hacia el servidor ThingSpeak es de dos minutos. El sistema de alimentación se compone por dos paneles solares conectados en paralelo y de una batería. A continuación, se puede observar el registro de las mediciones de voltaje producido por los paneles solares y el voltaje de la batería.

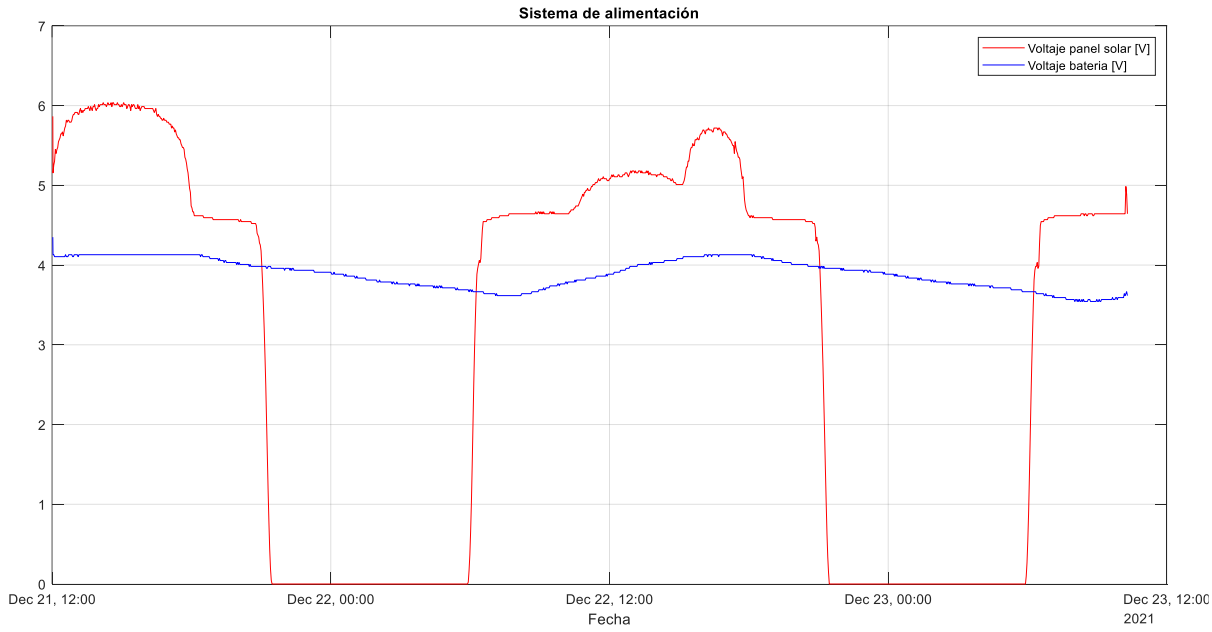


Figura 52: Prueba de rendimiento n°2

Como se aprecia en la Figura 52, la prueba comienza alrededor de las 12:00 del día 21 de diciembre con la batería totalmente cargada. El rendimiento del sistema de energía es el esperado, el voltaje de la batería al comienzo de la prueba se mantiene constante y cuando los paneles disminuyen el voltaje debido a la disminución de la radiación solar directa, la batería se comienza a descargar poco a poco. Se aprecia que, durante la noche la batería se descarga pero comienza a cargarse con la salida del sol lo que genera producción de energía. Se aprecia que,

la batería alcanza su carga máxima alrededor de las 16:00 del día 22 de diciembre, logrando repetir el ciclo. Se culmina la prueba de rendimiento el día 23 de diciembre alrededor de las 10:00 en donde se observa que la batería se encontraba cargándose nuevamente. La carga de la batería se debe a que los paneles solares son capaces de generar un excedente de energía que logra cargar la batería, caso contrario a la configuración vista anteriormente en donde el panel solar no generaba dicho excedente.

5.9.3. Prueba de rendimiento n°3.

La tercera prueba de rendimiento se realizó midiendo todas las variables meteorológicas, además de las variables correspondientes al sistema de energía. La frecuencia de comunicación utilizada para el envío de datos hacía el servidor ThingSpeak fue de dos minutos. El sistema de alimentación se compone por dos paneles solares conectados en paralelo y de dos baterías conectadas en paralelo (permite aumentar la capacidad de almacenamiento de energía a $12.000\,mAh$). A continuación, se puede observar el registro de las mediciones de voltaje producido por los paneles solares y el voltaje de la batería.

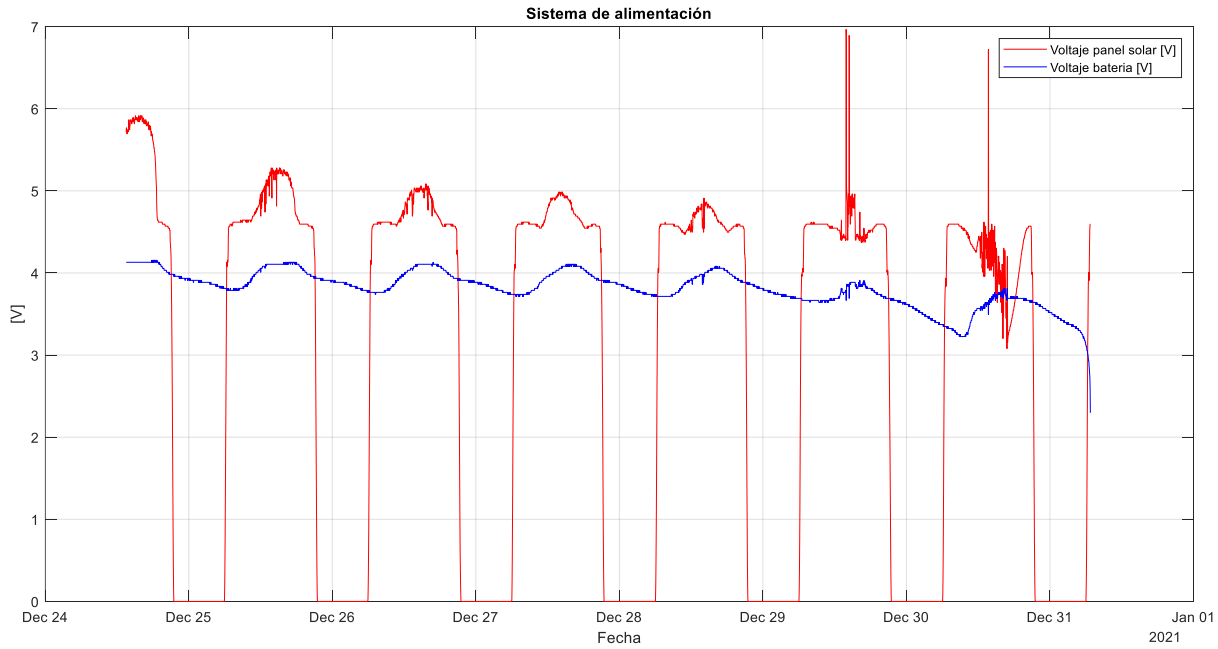


Figura 53: Prueba de rendimiento n°3

Como se aprecia en la Figura 53, la prueba comienza alrededor de las 13:30 del día 24 de diciembre con la batería totalmente cargada. El rendimiento del sistema de energía es el esperado, el voltaje de la batería al comienzo de la prueba se mantiene constante y cuando el panel disminuye el voltaje debido a la disminución de la radiación solar, la batería se comienza a descargar. Se aprecia que durante la noche la batería se descarga pero comienza a cargarse con la salida del sol lo que genera producción de energía. Se observa, que la batería alcanza voltajes cercanos al máximo durante los primeros tres días pero cada vez por menos tiempo, lo que tiene directa relación con la corriente generada por los paneles solares durante estos días. Este ciclo, se mantiene hasta el día 29 de diciembre que baja el voltaje, pero este no vuelve a subir completamente durante el día, lo que provoca una gran merma que se ve reflejada aún más el día 30 de diciembre, en donde se vuelve a cargar la batería pero sin alcanzar un valor mayor que el día anterior, debido a los efectos de la nubosidad e incendios forestales. Finalmente en el día 31 de diciembre es cuando la batería se descarga completamente.

5.9.3.1. Potencia consumida vs potencia generada.

Para analizar la potencia consumida por la estación meteorológica Arduino versus la potencia generada por el sistema de energía, se utilizarán los valores medidos en el apartado 4.3.10 junto con la potencia generada por el sistema de alimentación, los cuales pueden ser apreciados en la Figura 54.

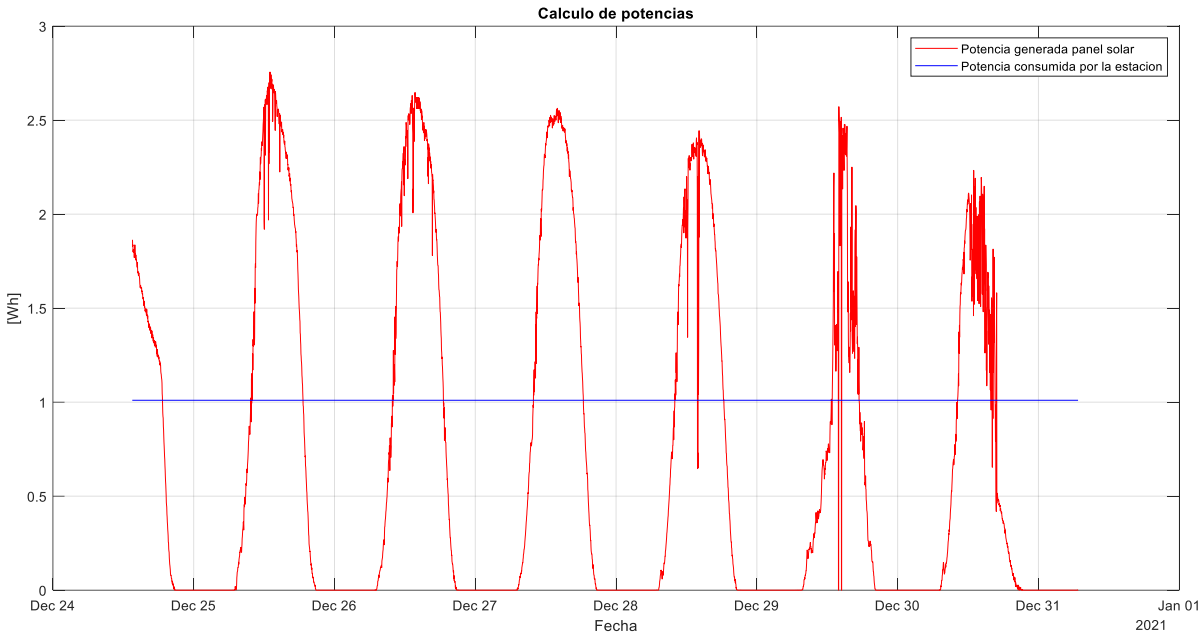


Figura 54: cálculo de potencias

Se observa, que la potencia consumida por la estación meteorológica Arduino es menor a la potencia generada por el sistema de alimentación en el periodo de generación, es por lo que este excedente de potencia es el responsable de realizar la carga de la batería, siendo esta la encargada de suplir la potencia que consume la estación cuando el sistema de alimentación no se encuentra en su periodo de generación.

5.9.4. Prueba de rendimiento n°4 (modo ahorro de energía).

La cuarta prueba de rendimiento, se realiza midiendo todas las variables meteorológicas junto con las variables correspondientes al sistema de energía. La frecuencia de comunicación para el envío de datos hacia el servidor ThingSpeak es de dos minutos para las variables relacionadas al viento y de cuatro minutos para el resto de variables. El sistema de alimentación se compone de la misma forma que la prueba realizada anteriormente. El registro de las mediciones correspondiente a voltaje del panel solar y voltaje de la batería se pueden ser apreciados a continuación.

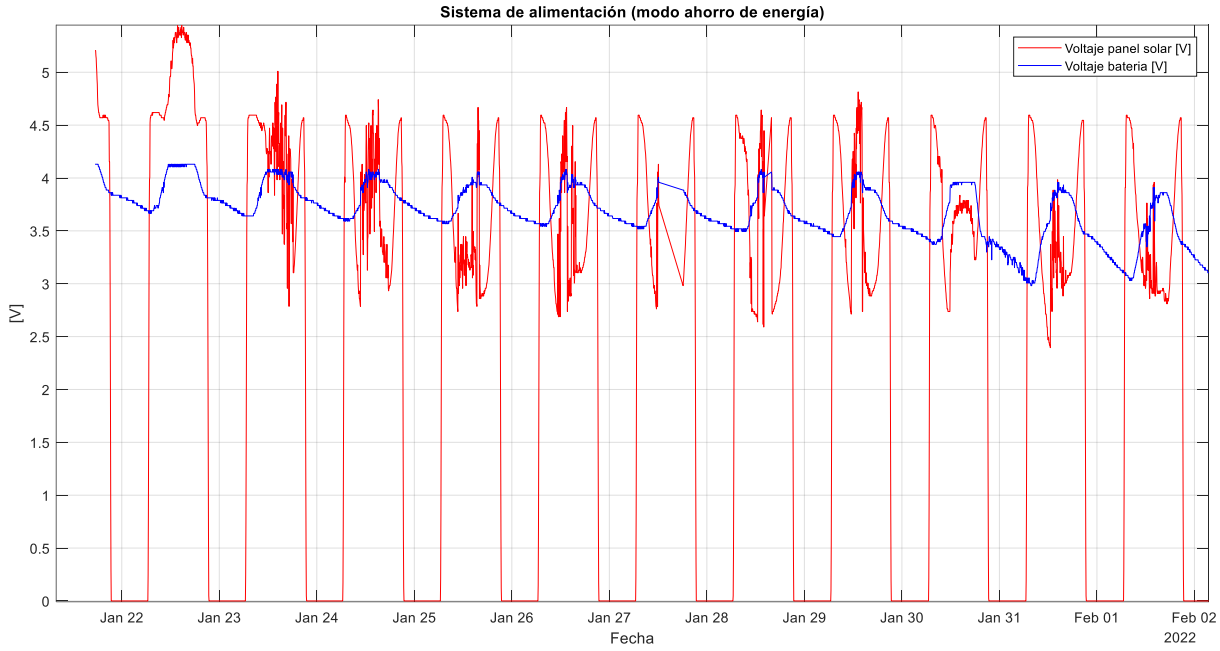


Figura 55: prueba de rendimiento n°4

Como se aprecia en la Figura 55, la prueba comienza el día 21 de enero alrededor de las 12:00 con la batería totalmente cargada. Durante todo el registro, se observa el ciclo de descarga y carga de esta con relación a las condiciones de generación de energía correspondientes a cada día. Durante el periodo, se aprecia que el voltaje del panel solar se comporta de una manera mucho más irregular con relación a las pruebas realizadas anteriormente, esto se debe a que las condiciones para la generación de energía a contar del tercer día no fueron los ideales, lo que se debe a la presencia de nubosidad y la presencia de polvo en la superficie de los paneles solares durante el transcurso de los días, lo que disminuye considerablemente la generación de energía. Con respecto al voltaje de la batería, se aprecia que con la implementación del modo ahorro de energía, la batería se carga en un tiempo mucho menor (cuando existen las condiciones ideales de generación) en comparación a las prueba realizada anteriormente, lo que tiene relación a la disminución del consumo de energía que realiza la estación meteorológica Arduino. Si bien las condiciones de generación no fueron las ideales, los resultados de la cuarta prueba de rendimiento son satisfactorios, siendo este el modo en el cual la estación meteorológica de bajo costo basada en Arduino se mantendrá operando.

5.9.4.1. Potencia consumida vs potencia generada.

En la Figura 56 se muestra el análisis de potencia consumida por la estación meteorológica Arduino, versus la potencia generada por el sistema de alimentación. Para analizar la potencia consumida, se utilizaron los valores calculados en el apartado 4.3.12. Para el sistema de alimentación se utilizaron los valores medidos durante la prueba de rendimiento n°4.

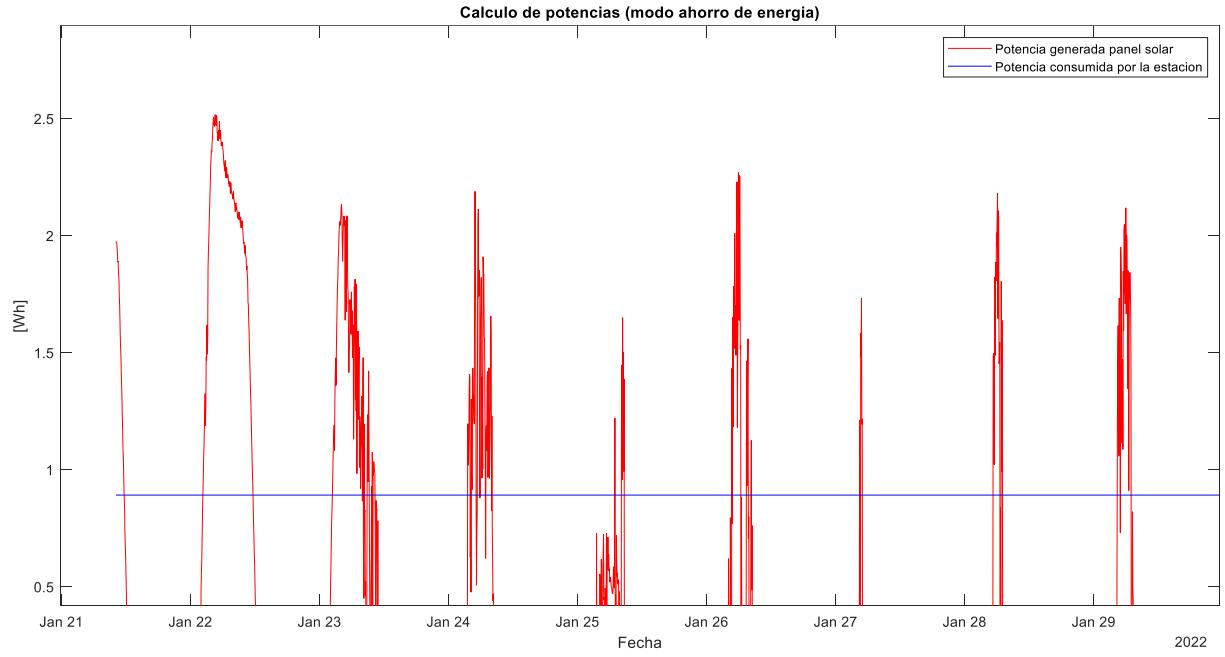


Figura 56: cálculo de potencias

Se observa, que la potencia generada por el sistema de alimentación es mayor a la potencia consumida por la estación meteorológica, lo que genera un excedente de energía el cual se utiliza para cargar las baterías. También se evidencia que, las condiciones de generación de energía no fueron las normales o ideales, lo que generó una disminución considerable en la potencia generada por el sistema de energía, lo que influyo en la carga de las baterías, pero en una menor proporción menor debido a la implementación del modo de ahorro de energía, como se pudo apreciar en la Figura 55.

6. Conclusiones y trabajo futuro

6.1. Resumen del trabajo realizado

La realización de este proyecto contempló el diseño, construcción e implementación de una estación meteorológica de bajo costo basada en Arduino, que mide temperatura y humedad del aire, presión atmosférica, intensidad y dirección del viento. También contempló el diseño e implementación de un sistema de energía autónomo compuesto de dos paneles solares, un regulador de carga y dos baterías, con el fin de que la estación fuera autosuficiente energéticamente.

Se comenzó por realizar una revisión bibliográfica en donde se investigaron las variables meteorológicas a medir por la estación, luego se detalló la forma tradicional y con qué instrumentos se realizan las mediciones de dichas variables, además se expuso como una estación meteorológica automática realiza dichas mediciones. Tras esto, se realizó una comparación entre estaciones meteorológicas automáticas y estaciones basadas en Arduino, resaltando similitudes y diferencias entre ambas. Posteriormente, se analizaron algunas estaciones meteorológicas presentes en el mercado junto con publicaciones en base a estaciones meteorológicas basadas en Arduino.

Posteriormente, se realizó una descripción de la plataforma de trabajo y los componentes seleccionados para realizar las mediciones de las variables meteorológicas descritas anteriormente, una descripción del sistema de almacenamiento y transferencia de datos, y del sistema de alimentación que provee la energía necesaria para el funcionamiento autónomo de la estación meteorológica Arduino.

Luego, se analizaron los detalles técnicos del emplazamiento, los cuales son la ubicación de la estación meteorológica, la irradiancia disponible en el emplazamiento

seleccionado y la ubicación óptima de los paneles solares, con el fin de aprovechar al máximo la radiación disponible para una mayor generación de energía.

Después, se explicó detalladamente el proceso de construcción e implementación de cada componente de la estación meteorológica, culminando con la implementación final y una explicación de los cambios con respecto a la implementación inicial.

Finalmente, se analizaron los datos obtenidos en las diferentes pruebas de funcionamiento de la estación meteorológica Arduino y del rendimiento del sistema de alimentación.

6.2. Conclusiones generales

Se realizaron cuatro pruebas de funcionamiento de la estación meteorológica Arduino, las cuales consistieron en dejar funcionando la estación, manteniendo un constante monitoreo de las mediciones y el desempeño del sistema de alimentación. Como se analizó en 5.9.1, la primera prueba de funcionamiento para las variables meteorológicas resultó satisfactoria, lo que indica que la implementación de todos los sensores e instrumentos se realizó de forma correcta. Es en el rendimiento del sistema de alimentación, en donde se evidenciaron problemas, los cuales fueron que durante el transcurso del periodo de prueba la batería no logró cargarse, de hecho desde el momento que se dejó funcionando la estación, esta comenzó a descargarse. Es por esta razón, que se decidió implementar un segundo panel solar de las mismas características que el utilizado, con el fin de que con esta configuración se logre la carga de la batería. Es en la segunda prueba de funcionamiento de la estación, analizada en 5.9.2 en donde se analiza la nueva configuración del sistema de alimentación, la cual entrega resultados satisfactorios, en donde se evidencia que al agregar otro panel solar de las mismas características que el implementado inicialmente, logra generar el excedente de energía que permite la carga de la batería. Adicional a esto, se decidió implementar otra batería de iguales características, con el fin de lograr un mayor almacenamiento de energía considerando la disminución de producción de energía durante los meses de invierno, evidenciados en 4.2.3, en

donde esta decae en alrededor de un 58%. Se realizó una tercera prueba de funcionamiento de la estación, en donde las variables meteorológicas se analizaron desde el apartado 5.1 al 5.8. Los resultados obtenidos para las mediciones de las variables meteorológicas son los esperados, logrando llevar un registro de seis días de mediciones. El sistema de alimentación para esta prueba funcionó de forma correcta durante los primeros días, logrando realizar la carga de la batería de manera correcta y sin inconvenientes. Es en el día 29 de diciembre en donde la producción de energía disminuye debido a la presencia de nubosidad, lo que implica que la radiación que reciben los paneles solares no es de forma directa, si no que difusa. Para estos días el registro de temperatura máxima fue alrededor de 20 °C con una humedad superior al 80% para gran parte del periodo de producción de energía. Esto, afectó la producción de corriente por parte de los paneles solares, como se evidencio en 5.7, lo que significó una disminución importante de la potencia generada y del voltaje alcanzado por la batería durante este día. Entre los días 28 y 30 de diciembre, ocurrió un gran incendio forestal en la comuna aledaña de Machalí, lo que afecto la producción de energía durante estos días, efectos que se evidenciaron en las mediciones de voltaje y corriente del panel solar (aparatados 5.6 y 5.7) durante este periodo, estos efectos se suman a los ocurridos el día 29 de diciembre explicados anteriormente. Para el día 30 de diciembre, las condiciones meteorológicas volvieron a sus valores normales (estación de verano), pero la producción de energía se vio afectada mayormente por los efectos de los aerosoles liberados por los incendios forestales. Como se analiza en [43], el efecto de los aerosoles no es solo de atenuación de la radiación, si no que de dispersión de esta. Además, el efecto sobre la producción fotovoltaica es de una disminución en la potencia de salida, en alrededor un -4,5 %. Finalmente, son esta seguidilla de efectos adversos a la producción de energía, los que terminan por afectar la carga de las baterías, por lo que es en el día 31 de diciembre en donde estas se descargan completamente. Es por estos motivos, que se implementó el modo de ahorro de energía en Arduino, lo que disminuyó el consumo de energía por parte de la estación en un $13{,}5\,\%$, pasando de consumir $0{,}195\,Ah$ a $0{,}172\,Ah$ con la nueva configuración, de acuerdo con las mediciones realizadas en los apartados 4.3.10 y 4.3.12. Se realizó una cuarta prueba de funcionamiento, analizada en el apartado

5.9.4, la cual entregó resultados positivos con respecto al rendimiento del sistema de alimentación con la implementación del modo de ahorro de energía. Como se evidencio, las baterías con esta nueva implementación lograr realizar la carga completa en un 116 % más rápido que con la implementación realizada en la prueba de funcionamiento n°3 (considerando días con condiciones de generación similares). Durante el periodo de prueba, se presentaron variaciones en las condiciones de generación, las cuales se evidencian en el apartado 5.9.4, estas se debieron principalmente a la presencia de nubosidad y al efecto de la presencia de polvo en la superficie de los paneles solares, lo que provocó la disminución de la potencia generada por estos, como se evidenció en el apartado 5.9.4.1. A pesar de la presencia de variaciones en la generación de energía, las baterías proporcionaron la energía necesaria para el correcto funcionamiento de la estación meteorológica en el periodo de prueba, por lo que se definió que la actual configuración tanto del sistema de energía como de la implementación del modo de ahorro de energía es la que se mantendrá en funcionamiento.

De esta forma, se concluye que en el presente trabajo se logra el diseño, construcción e implementación de una estación meteorológica basada en Arduino, junto con el diseño e implementación de un sistema de energía autónomo, compuesto por dos paneles solares y dos baterías. Está estación meteorológica, se emplazó en la azotea del edificio A de la Universidad de O'Higgins. Tras la realización de pruebas de funcionamiento y rendimiento, se logra encontrar una configuración que permite el correcto funcionamiento de la estación meteorológica, culminando el trabajo realizado de forma satisfactoria.

6.3. Trabajo futuro

Concluido el proyecto tras el análisis de resultados, se presentan las siguientes mejoras con el fin de complementar lo logrado en la realización del presente proyecto y lograr mejores resultados.

Con respecto al rendimiento energético de la estación meteorológica de bajo costo basada en Arduino, se sugiere realizar un seguimiento a la implementación final de la estación (modo ahorro de energía). Si los resultados no son los óptimos, se sugiere implementar un sistema de alimentación de mayor potencia, que posea paneles solares de mayor capacidad de generación junto con una batería de mayor capacidad que permita el correcto funcionamiento de la estación meteorológica.

Como se mencionó, no se implementó el uso del pluviómetro debido a la restricción de cantidad de fields disponibles en el servidor ThingSpeak. Se sugiere, agregar esta medición debido a que se cuenta con el instrumento y todo lo necesario para su correcto funcionamiento.

Si bien se siguieron todas las recomendaciones e instrucciones para la configuración e implementación del anemómetro y la veleta, se sugiere realizar una calibración de los valores medidos por estos, debido a que no se contaba con los instrumentos para realizar este proceso.

Se sugiere implementar una mayor cantidad de sensores que permitan medir una mayor cantidad de variables, tanto meteorológicas como del sistema de alimentación compuesto por los paneles solares. Con el propósito de complementar el trabajo realizado en el presente proyecto y que la estación meteorológica de bajo costo basada en Arduino pueda ser incluida en los registro de estaciones en línea de la Dirección Meteorológica de Chile (DMC).

Referencias

[1] S. Tenzin, S. Siyang, T. Pobkrut, y T. Kerdcharoen, «Low cost weather station for climate-smart agriculture», en *2017 9th International Conference on Knowledge and Smart Technology (KST)*, feb. 2017, pp. 172-177. doi: 10.1109/KST.2017.7886085.

[2] Y. P. Jayasuriya, C. S. Elvitigala, K. Wamakulasooriya, y B. Sudantha, «Low Cost and IoT Based Greenhouse with Climate Monitoring and Controlling System for Tropical Countries», en *2018 International Conference on System Science and Engineering (ICSSE)*, jun. 2018, pp. 1-6. doi: 10.1109/ICSSE.2018.8519997.

[3] U. Ramani, R. Nithya, S. Sathieshkumar, y T. Santhoshkumar, «Automatic Weather Monitoring Analysis for Renewable Energy System», en *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, jul. 2020, pp. 924-928. doi: 10.1109/ICESC48915.2020.9155865.

[4] R. Garreaud y C. Meruane, «Instrumentos Meteorológicos y Humedad Atmosférica». 2005. [En línea]. Disponible en: http://mct.dgf.uchile.cl/AREAS/meteorologia.html

[5] F. V. Brock, P. E. of M. F. V. Brock, S. J. Richardson, y S. J. Richardson, *Meteorological Measurement Systems*. Oxford University Press, 2001.

[6] I. Z. LÓPEZ y E. C. D. ARCO, *Meteorología y climatología*. UNED, 2010.

[7] V. Herrera, L. Restrepo, N. Quevedo, P. Crespo, y R. Portocarrero, «Manual Teórico práctico del observador meteorológico de superficie», *México Serv. Meteorológico Nac. CONAGUA*, pp. 70-73, 2012.

[8] D. T. Bailey, *Meteorological monitoring guidance for regulatory modeling applications*. DIANE Publishing, 2000. [En línea]. Disponible en: https://www.epa.gov/sites/default/files/2020-10/documents/mmgrma_0.pdf

[9] «Sobre-nosotros - vetocl». https://www.veto.cl/Sobre-nosotros/La-empresa (accedido 4 de octubre de 2021).

[10] «Estación meteorológica agrícola - vetocl». https://www.veto.cl/estacion-meteorologica-agricultura-a6093219/p (accedido 1 de octubre de 2021).

[11] «Agroprecisión». https://agroprecision.cl/ (accedido 5 de octubre de 2021).

[12] «iMETOS 3.3», *METOS by Pessl instruments*. https://metos.at/es/imetos33/ (accedido 5 de octubre de 2021).

[13] «Empresa - IMPROFOR». https://www.improfor.cl/empresa/ (accedido 5 de octubre de 2021).

[14] «6162 Davis Vantage Pro2™ Plus Inalámbrica», *estacionesdavis.es*. https://www.estacionesdavis.es/es/davis-vantage-pro-2/12-davis-vantage-pro2-plus-inalambrica.html (accedido 5 de octubre de 2021).

[15] G. Solano, F. Lama, J. Terrazos, y J. Tarrillo, «Weather station for educational purposes based on Atmega8L», en *2017 IEEE XXIV International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*, Cusco, Peru, ago. 2017, pp. 1-4. doi: 10.1109/INTERCON.2017.8079728.

[16]H. Üçgün y Z. K. Kaplan, «Arduino based weather forecasting station», en *2017 International Conference on Computer Science and Engineering (UBMK)*, oct. 2017, pp. 972-977. doi: 10.1109/UBMK.2017.8093397.

[17]R. C. Brito, F. Favarim, G. Calin, y E. Todt, «Development of a low cost weather station using free hardware and software», en *2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR)*, nov. 2017, pp. 1-6. doi: 10.1109/SBR-LARS-R.2017.8215292.

[18]A. Imtiaz, S. G. Omar, y T. A. Ali, «Efficient Design of a Low Cost Portable Weather Station», en *2018 International Conference on Computer Communication and Informatics (ICCCI)*, ene. 2018, pp. 1-7. doi: 10.1109/ICCCI.2018.8441207.

[19]Md. I. Haque, A. H. MD. Shatil, A. N. Tusar, M. Hossain, y Md. H. Rahman, «Renewable Powered Portable Weather Update Station», en *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*, ene. 2019, pp. 374-377. doi: 10.1109/ICREST.2019.8644330.

[20]K. Kaewwongsri y K. Silanon, «Design and Implement of a Weather Monitoring Station using CoAP on NB-IoT Network», en *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, jun. 2020, pp. 230-233. doi: 10.1109/ECTI-CON49241.2020.9158290.

[21]P. Ashok Baste, S. R. Jadkar, y A. M. Pathak, «Weather Station for Solar PV Power Plant Using Arduino Mega», en *2021 International Conference on Computer Communication and Informatics (ICCCI)*, ene. 2021, pp. 1-6. doi: 10.1109/ICCCI50826.2021.9402478.

[22]«Arduino Logger Shield (uSD + RTC) | MCI Electronics.cl». https://www.mcielectronics.cl/shop/product/mci-arduino-logger-shield-usd-rtc-mci-electronics-10355 (accedido 8 de octubre de 2021).

[23]«Ethernet-Shield-Arduino-W5100-WizNet». https://altronics.cl/ethernetshield-w5100 (accedido 30 de noviembre de 2021).

[24]«Sensor de Temperatura y Humedad AM2315 | MCI Electronics.cl». https://www.mcielectronics.cl/shop/product/sensor-de-temperatura-y-humedad-am2315-25767?search=MCI05578 (accedido 8 de octubre de 2021).

[25]«Sensor Altitud/Presión - MPL3115A2 Breakout | MCI Electronics.cl». https://www.mcielectronics.cl/shop/product/sensor-altitud-presion-mpl3115a2-breakout-sparkfun-10944 (accedido 8 de octubre de 2021).

[26]«SparkFun Humidity and Temperature Sensor Breakout - Si7021 - SEN-13763 - SparkFun Electronics». https://www.sparkfun.com/products/13763?_ga=2.37778211.1053741286.1634560026-1870478506.1631718348 (accedido 19 de octubre de 2021).

[27]«Sensores para Estación Meteorológica | MCI Electronics.cl». https://www.mcielectronics.cl/shop/product/sensores-para-estacion-meteorologica-10008 (accedido 8 de octubre de 2021).

[28]«Solar Panel - 3.5W - PRT-13782 - SparkFun Electronics». https://www.sparkfun.com/products/13782 (accedido 12 de noviembre de 2021).

[29]«Lithium Ion Battery - 6Ah - PRT-13856 - SparkFun Electronics». https://www.sparkfun.com/products/13856 (accedido 12 de noviembre de 2021).
[30]«sensor, voltaje, FZ0430». https://altronics.cl/sensor-voltaje-fz0430 (accedido 1 de diciembre de 2021).
[31]E. Lopez Aldea, *Arduino: guia practica de fundamentos y simulacion*. RA-MA Editorial, 2015. [En línea]. Disponible en: https://elibro-net-uoh.knimbus.com/es/lc/bibliouoh/titulos/106492
[32]«Software de hojas de cálculo Microsoft Excel | Microsoft 365». https://www.microsoft.com/es-es/microsoft-365/excel (accedido 13 de octubre de 2021).
[33]«Más información - ThingSpeak IoT». https://thingspeak.com/pages/learn_more (accedido 29 de diciembre de 2021).
[34]«MATLAB - El lenguaje del cálculo técnico». https://la.mathworks.com/products/matlab.html (accedido 13 de octubre de 2021).
[35]«Explorador Solar». http://solar.minenergia.cl/exploracion (accedido 22 de octubre de 2021).
[36]«Hookup | Code | Arduino to AM2315 Temperature - Humidity Sensor Tutorial». http://cactus.io/hookups/sensors/temperature-humidity/am2315/hookup-arduino-to-am2315-temp-humidity-sensor (accedido 9 de noviembre de 2021).
[37]«Pull-up Resistors - learn.sparkfun.com». https://learn.sparkfun.com/tutorials/pull-up-resistors?_ga=2.231957151.1621089847.1636512391-1870478506.1631718348 (accedido 10 de noviembre de 2021).
[38]«Arduino Weather Shield Hookup Guide V12 - learn.sparkfun.com». https://learn.sparkfun.com/tutorials/arduino-weather-shield-hookup-guide-v12 (accedido 10 de noviembre de 2021).
[39]designthemes, «Data logger shield V1 | Tienda y Tutoriales Arduino». https://www.prometec.net/data-logger-shield/ (accedido 12 de noviembre de 2021).
[40]A. Industries, «AM2315 - Encased I2C Temperature/Humidity Sensor». https://www.adafruit.com/product/1293 (accedido 28 de diciembre de 2021).
[41]Thingiverse.com, «Cocito Weather Station by teo9700». https://www.thingiverse.com/thing:1564828 (accedido 28 de diciembre de 2021).
[42]«TDS PLA 850.pdf», *Dropbox*. https://www.dropbox.com/s/rlob21txtyciups/TDS%20PLA%20850.pdf?dl=0 (accedido 29 de diciembre de 2021).
[43]M. F. A. del Hoyo Peña, «Efecto de los aerosoles en la radiación solar y la producción fotovoltaica en Santiago», 2016, Accedido: 8 de enero de 2022. [En línea]. Disponible en: https://repositorio.uchile.cl/handle/2250/143594
metos.at/home/terms-of-use/

Anexos

A1 Cotizaciones

Cotización estación meteorológica IMETOS 3.3 (Variante IMT280)

AgroPrecisión
Precisión para el nuevo Agro

COTIZACION N° 1
Fecha: 05/10/2021

Spectrum Technologies, Inc.
PMS Instrument
AgroPro – Sensors
Pessl Instruments

SRES. **Anibal Cerda** **Dirección:** **Teléfono**
ATENCION **Rut:** **Fax**
PRESENTE **e-mail:**
V21.09.07

Item	Descrpción	Valor Unitario	Descuento	Cantidad	Valor Total
IMT280	- iMetos ag 280 / iMetos ag(Temp., Rel. Humidity, Radiation, Rainfall,	2.321.770		1,0	2.321.770
LOC1512	- Mastil de tubo galvanizado (1 y 2 pulgadas) /	56.020		1,0	56.020
					-
					-
					-
					-
					-
					-
					-
					-
					-
					-
					-
					-
					-
					-
				Subtotal	$ 2.377.790
	IVA			19%	$ 451.780
					-
					-
				TOTAL	$ 2.829.570

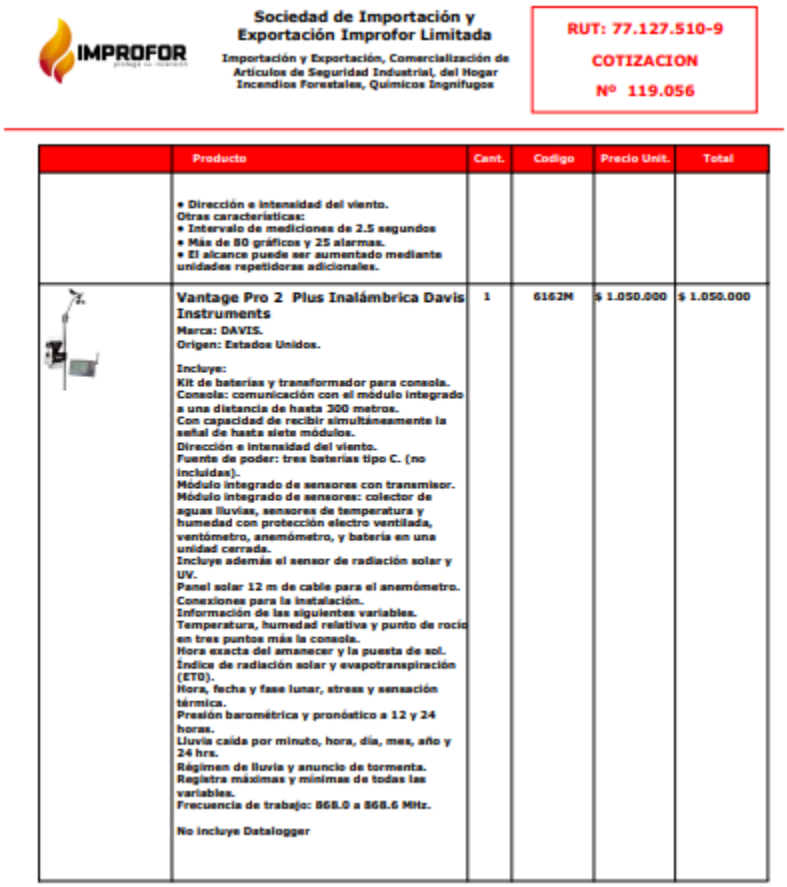
Condiciones y comentarios
- Los servicios realizados fuera de Santiago tendrán un recargo por distancia ($400/km).
- Plazo de entrega, 15 días hábiles desde la recepción de la orden de compra.
- La presente cotización tiene una validez de 7 días.
- Pago contra vencimiento de factura.

Alex San Martin Duarte
Soporte Tecnico Electronico - Ventas
cel: +569 98459797 email: asanmartin@agroprecision.cl

Servicios Agroprecisión SpA / R.U.T.: 77.600.040-K / Giro: Servicios Agrícolas / Av. José Manuel Infante 1183 / Providencia - SantiagoTeléfono: (56 - 2) 274 6658
www.agroprecision.cl

Anexo 1: Cotización IMT280

Cotización estación meteorológica Davis Vantage Pro2 Plus inalámbrica



IMPROFOR

Sociedad de Importación y Exportación Improfor Limitada

Importación y Exportación, Comercialización de Artículos de Seguridad Industrial, del Hogar Incendios Forestales, Químicos Ingnífugos

RUT: 77.127.510-9
COTIZACION
Nº 119.056

	Producto	Cant.	Codigo	Precio Unit.	Total
	• Dirección e intensidad del viento. Otras características: • Intervalo de mediciones de 2.5 segundos • Más de 80 gráficos y 25 alarmas. • El alcance puede ser aumentado mediante unidades repetidoras adicionales.				
	Vantage Pro 2 Plus Inalámbrica Davis Instruments Marca: DAVIS. Origen: Estados Unidos. Incluye: Kit de baterías y transformador para consola. Consola: comunicación con el módulo integrado a una distancia de hasta 300 metros. Con capacidad de recibir simultáneamente la señal de hasta siete módulos. Dirección e intensidad del viento. Fuente de poder: tres baterías tipo C. (no incluidas). Módulo integrado de sensores con transmisor. Módulo integrado de sensores: colector de aguas lluvias, sensores de temperatura y humedad con protección electro ventilada, ventómetro, anemómetro, y batería en una unidad cerrada. Incluye además el sensor de radiación solar y UV. Panel solar 12 m de cable para el anemómetro. Conexiones para la instalación. Información de las siguientes variables. Temperatura, humedad relativa y punto de rocío en tres puntos más la consola. Hora exacta del amanecer y la puesta de sol. Índice de radiación solar y evapotranspiración (ETO). Hora, fecha y fase lunar, stress y sensación térmica. Presión barométrica y pronóstico a 12 y 24 horas. Lluvia caída por minuto, hora, día, mes, año y 24 hrs. Régimen de lluvia y anuncio de tormenta. Registra máximas y mínimas de todas las variables. Frecuencia de trabajo: 868.0 a 868.6 MHz. No incluye Datalogger	1	6162M	$ 1.050.000	$ 1.050.000

Anexo 2: Cotización Davis Vantaje Pro2 Plus inalámbrica

A2 Códigos

Código datalogger

```
#include <Wire.h>
#include "cactus_io_AM2315.h"
#include <SD.h>
#include <SPI.h>      // Necesario para la SD card
#include "RTClib.h"


AM2315 am2315;
RTC_DS1307 RTC;      // Definimos el reloj
File logfile;        // Fichero a escribir en la SD
const int chipSelect = 10;  // SD card pin select
const bool eco = true ;
```

```
int count = 0 ;       // Controla cada cuanto tiemepo se vuelcan los datos a
la SD

void setup() {
  Serial.begin(9600);
  pinMode(chipSelect, OUTPUT);        // Definicion del pin de la SD

  if (!am2315.begin()) {
     Serial.println("Sensor not found, check wiring & pullups!");
     while (1);
  }

  if (!SD.begin(chipSelect))
        Serial.println("error(No hay tarjeta SD.)");
    else
        Serial.println("Tarjeta SD inicializada.");

  // Creamos el fichero de registro
  char filename[] = "LOGGER00.CSV";
  for (uint8_t i = 0; i < 100; i++)
     {
        filename[6] = i/10 + '0';
        filename[7] = i%10 + '0';
        if (! SD.exists(filename))      // Si no existe el fichero, lo
creamos
            {
              logfile = SD.open(filename, FILE_WRITE);
              break;  // leave the loop!
            }
     }
  if (! logfile)
      logfile.println("error(No se pudo crear el fichero de registro)");

  Serial.print("Registrando en: ");   Serial.println(filename);

  // Conexion del reloj RTC
  Wire.begin();
  if (!RTC.begin())
      logfile.println("No hay RTC.");
  else
      Serial.println("RTC correcto. Iniciando captura de datos");

  //Se guardan en la tarjeta SD los encabezados
  logfile.print("Fecha") ;
  logfile.print(", ");
  logfile.print("Hora") ;
  logfile.print(", ");
  logfile.print("Temp ext[°C],") ;
  logfile.print(", ");
  logfile.println("Humedad ext[HR]") ;

  //Se muestran por el puerto serie los encabezados
  Serial.print("Fecha/Hora") ;
  Serial.print(", ");
  Serial.print("Temp ext[°C],") ;
  Serial.println(" Humedad ext[HR]") ;
}
```

```
void loop()
{
  am2315.readSensor();

    DateTime now = RTC.now();

        int err  ;
        float temp, hum;

        logfile.print(now.year(), DEC);
        logfile.print('/');
        logfile.print(now.month(), DEC);
        logfile.print('/');
        logfile.print(now.day(), DEC);
        logfile.print(' ');
        logfile.print(now.hour(), DEC);
        logfile.print(':');
        logfile.print(now.minute(), DEC);
        logfile.print(':');
        logfile.print(now.second(), DEC);
        logfile.print(' ');
        logfile.print(am2315.getTemperature_C());
        logfile.print(", ");
        logfile.println(am2315.getHumidity());


        if ( count++ > 64 )
            {   logfile.flush(); // Para forzar la escritura en la SD
                     count = 0 ;       // Cada 64 lecturas
            }
        if (eco)
            {

            Serial.print(now.year(), DEC);
            Serial.print('/');
            Serial.print(now.month(), DEC);
            Serial.print('/');
            Serial.print(now.day(), DEC);
            Serial.print(' ');
            Serial.print(now.hour(), DEC);
            Serial.print(':');
            Serial.print(now.minute(), DEC);
            Serial.print(':');
            Serial.print(now.second(), DEC);
            Serial.print(' ');

            Serial.print(am2315.getTemperature_C());
            Serial.print(", ");
            Serial.println(am2315.getHumidity());
            }

  delay(1000);

}
```

```
void error(char *str)
{
  Serial.print("error: ");
  Serial.println(str);

  while(1);
}
```

Código estación meteorológica de bajo costo basada en Arduino

Se debe incluir el siguiente código en un archivo txt y se debe guardar junto al código en la misma carpeta.

```
// Utilice este archivo para almacenar todas las credenciales privadas y
detalles de conexión

// Ingrese una dirección MAC para su controlador a continuación.
// Los protectores de Ethernet más nuevos tienen una dirección MAC impresa en
una etiqueta en el protector
#define SECRET_MAC {0x90, 0xA2, 0xDA, 0x10, 0x40, 0x4F}

#define SECRET_CH_ID 1582031
#define SECRET_WRITE_APIKEY "HA01JU1D0YYZ0NKH"
```

Código

```
#include <Ethernet.h>
#include "secrets.h"
#include "ThingSpeak.h"
#include <Wire.h>
#include "SparkFun_Si7021_Breakout_Library.h"
#include "cactus_io_AM2315.h"
#include "math.h"

//Thingspeak configuración
byte mac[] = SECRET_MAC;

//Configure la dirección IP estática que se utilizará si el DHCP no puede
asignar
IPAddress ip(172, 16, 15, 249/22);
IPAddress myDns(172, 16, 0, 13);

EthernetClient client;

unsigned long myChannelNumber = SECRET_CH_ID;
const char * myWriteAPIKey = SECRET_WRITE_APIKEY;

// Crea la instancia de los sensores utilizados
Weather placa_roja; //sensor de temperatura y humedad interior SI7021
AM2315 am2315;

// Definiciones de pines de hardware
```

```
const byte WSPEED_PIN = 3;
const byte WDIR = A0;

// Definición de variables globales

float tempc_int = 0;
float HR_ext = 0;
float temp_ext = 0;
float voltaje_bateria = 0;
float voltaje_panel = 0;
float corriente_panel = 0;
//Sensor de corriente
float sensibilidad = 0.135;
float sum_cos = 0;
float sum_sin = 0;
float avg_cos = 0;
float avg_sin = 0;
int winddir_avg2m_sincos = 0;

long lastSecond; // Contador de milisegundos para ver cuando pasa un segundo
byte seconds; // Cuando llega a 60, aumenta el minuto actual
byte seconds_2m; // Realiza un seguimiento de la "velocidad del viento /
promedio" durante los últimos 2 minutos de datos
byte minutes; // Realiza un seguimiento de dónde estamos en varias matrices
de datos
byte minutes_10m; // Realiza un seguimiento de dónde estamos en ráfagas de
viento / dir durante los últimos 10 minutos de matriz de datos

long lastWindCheck = 0;
volatile long lastWindIRQ = 0;
volatile byte windClicks = 0; //Contabiliza la cantidad de clics para la
velocidad del viento

unsigned long tiempo_anterior = 0; //Usado para configurar el tiempo de
muestreo de subida de los datos a ThingSpeak
const long intervalo = 120000; // Usado para configurar el tiempo de muestreo
de subida de los datos a ThingSpeak

#define WIND_AVG_SIZE 120

byte windspdavg[WIND_AVG_SIZE];
int winddiravg[WIND_AVG_SIZE];

//Estos son todos los valores del anemómetro y veleta
int winddir = 0; // [0-360 dirección del viento instantánea en grados]
float windspeedkph = 0; // [velocidad del viento instantánea en kph]
float windspdkph_avg2m = 0; // [promedio de dos minutos de la velocidad del
viento en kph]
int winddir_avg2m = 0; // [0-360 promedio de dos minutos de la dirección en
grados]

void wspeedIRQ()
// Activado por el imán en el anemómetro (2 tics por rotación), conectado a
la entrada D3
{
```

```
    if (millis() - lastWindIRQ > 10) // Ignora los fallos de rebote del
interruptor de menos de 10 ms (lectura máxima de 228 kph) después de que se
cierre el interruptor de lengüeta
    {
        lastWindIRQ = millis(); // Guarda la hora actual
        windClicks++; //cantidad de kph por clic por segundo
    }
}

void setup() {

  Serial.begin(9600);

//Inicializacion de los sensores I2C
  placa_roja.begin();
  am2315.begin();

  pinMode(WSPEED_PIN, INPUT_PULLUP); // Entrada de los medidores de velocidad
del viento

//Configuración anemómetro y dir. viento

  seconds = 0;
  lastSecond = millis();

// Definición de pines de interrupción internos
  attachInterrupt(1, wspeedIRQ, FALLING);

  interrupts();

 //Thingspeak configuración
  Ethernet.init(10);
  Serial2.begin(115200);
  while (!Serial) {
    ;
  }

  // Comienzo de la conexión por ethernet:
  Serial.println("Initialize Ethernet with DHCP:");
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    // Check for Ethernet hardware present
    if (Ethernet.hardwareStatus() == EthernetNoHardware) {
      Serial.println("Ethernet shield was not found.  Sorry, can't run
without hardware. :(");
      while (true) {
        delay(1); //
       }
    }
    if (Ethernet.linkStatus() == LinkOFF) {
      Serial.println("Ethernet cable is not connected.");
    }
    // Configuración de la dirección IP en vez de DHCP
    Ethernet.begin(mac, ip, myDns);
  } else {
    Serial.print("  DHCP assigned IP ");
    Serial.println(Ethernet.localIP());
```

```
  }

  delay(1000);

  ThingSpeak.begin(client);  // Iniciando ThingSpeak

}

void thingspeak() {

    // Definicion de los fields y de que valores almacenaran
  ThingSpeak.setField(1, temp_ext);
  ThingSpeak.setField(2, HR_ext);
  ThingSpeak.setField(3, tempc_int);
  ThingSpeak.setField(4, windspdkph_avg2m);
  ThingSpeak.setField(5, winddir_avg2m_sincos);
  ThingSpeak.setField(6, voltaje_panel);
  ThingSpeak.setField(7, corriente_panel);
  ThingSpeak.setField(8, voltaje_bateria);


  // Escritura en el canal ThingSpeak
  int x = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
  if(x == 200){
    Serial.println("Channel update successful.");
  }
  else{
    Serial.println("Problem updating channel. HTTP error code " + String(x));
  }

}

void Interior()
{
  //Sensor de temperatura y humedad interior SI7021
  tempc_int = placa_roja.getTemp();
}

void exterior() {
  am2315.readSensor();
  temp_ext=am2315.getTemperature_C();
  HR_ext=am2315.getHumidity();
}

void Superior() {

    //Registra en que minuto esta
  if(millis() - lastSecond >= 1000)
    {
        lastSecond += 1000;

        //Toma una lectura de velocidad y dirección cada segundo durante un
promedio de 2 minutos
        if(++seconds_2m > 119) seconds_2m = 0;

        //Calcule la velocidad y la dirección del viento cada segundo durante
120 segundos para obtener un promedio de 2 minutos
```

```
        float currentSpeed = get_wind_speed();
        windspeedkph = currentSpeed;//actualizar la variable global para la
velocidad del viento

        int currentDirection = get_wind_direction();
        windspdavg[seconds_2m] = (int)currentSpeed;
        winddiravg[seconds_2m] = currentDirection;

        if(++seconds > 59)
        {
            seconds = 0;

            if(++minutes > 59) minutes = 0;
            if(++minutes_10m > 9) minutes_10m = 0;
        }
    }
  delay(100);
}

//Esta función calcula cada una de las variables
void calcWeather()
{
    //Calc winddir
    winddir = get_wind_direction();

    //Calc windspdkph_avg2m
    float temp = 0;
    for(int i = 0 ; i < 120 ; i++)//120
        temp += windspdavg[i];
    temp /= 120.0;
    windspdkph_avg2m = temp;
}

//Retorna la velocidad instantánea del viento
float get_wind_speed()
{
    float deltaTime = millis() - lastWindCheck; //

    deltaTime /= 1000.0; //Convierte a segundos

    float windSpeed = (float)windClicks / deltaTime;

    windClicks = 0; //Reinicia y comienza a medir la nueva velocidad del
viento
    lastWindCheck = millis();

    windSpeed *= 2.4; //cantidad de kph por clic del anemómetro

    return(windSpeed);
}

//Lee la posición de la veleta y regresa la posición en grados
int get_wind_direction()
{
    unsigned int adc;

    adc = analogRead(WDIR); // obtiene la lectura actual del sensor
```

```
    // La siguiente tabla son las lecturas de ADC para la salida del sensor
de dirección del viento, ordenadas de menor a mayor.
    // Cada umbral es el punto medio entre encabezados adyacentes. La salida
son grados para esa lectura de ADC.
    // Tenga en cuenta que estos no están en orden de grados de la brújula.
Consulte la hoja de datos para obtener más información.

    if (adc < 380) return (113);
    if (adc < 393) return (68);
    if (adc < 414) return (90);
    if (adc < 456) return (158);
    if (adc < 508) return (135);
    if (adc < 551) return (203);
    if (adc < 615) return (180);
    if (adc < 680) return (23);
    if (adc < 746) return (45);
    if (adc < 801) return (248);
    if (adc < 833) return (225);
    if (adc < 878) return (338);
    if (adc < 913) return (0);
    if (adc < 940) return (293);
    if (adc < 967) return (315);
    if (adc < 990) return (270);
    return (-1); // error, desconectado?
}
// En esta función se calcula vectorialmente el promedio de dos minutos de
dirección del viento
void compute_windir()
{
    sum_cos = 0;
    sum_sin = 0;
    for(int i = 1 ; i < WIND_AVG_SIZE ; i++)
    {
        sum_cos += windspdavg[i]*cos(winddiravg[i]*3.1416/180);
        sum_sin += windspdavg[i]*sin(winddiravg[i]*3.1416/180);
    }

    avg_cos = sum_cos/WIND_AVG_SIZE;
    avg_sin = sum_sin/WIND_AVG_SIZE;

    winddir_avg2m_sincos = atan2(avg_sin,  avg_cos)*180/3.1416;
    winddir_avg2m_sincos = (360 + winddir_avg2m_sincos)% 360;
}

// Lee los valores obtenidos de los sensores del sistema de alimentación
void sensores() {

  voltaje_bateria = (float)25*analogRead(A8)/1023;
  voltaje_panel = (float)25*analogRead(A9)/1023;

  corriente_panel=get_corriente(500);//obtenemos la corriente promedio de 500
muestras

}

float get_corriente(int n_muestras)
```

```
{
  float voltajeSensor;
  float corriente=0;
  for(int i=0;i<n_muestras;i++)
  {
    voltajeSensor = analogRead(A10) * (5.0 / 1023.0);////lectura del sensor
//A10
    corriente=corriente+(voltajeSensor-2.5)/sensibilidad; //Ecuación para
obtener la corriente

    if (corriente < 0){ //Se eliminan lecturas negativas por ruido
    corriente = 0;
    }
  }
  corriente=corriente/n_muestras;
  return(corriente);
}



void loop() {

    //Configuración del tiempo de muestreo
    unsigned long tiempo_actual = millis();

    if (tiempo_actual - tiempo_anterior >= intervalo)  {

    tiempo_anterior = tiempo_actual;

    Interior();
    exterior();
    sensores();
    Superior(); //# crea los arrays de speed y dirección
    calcWeather(); //# calcula los promedios de speed y dirección
    compute_windir();
    thingspeak();

    }

    else {
      Superior();
      calcWeather();
      compute_windir();
      delay(1000);
    }


}
```

Código modo ahorro de energía

```
#include <Ethernet.h>
#include "secrets.h"
#include "ThingSpeak.h"
#include <Wire.h>
#include "SparkFun_Si7021_Breakout_Library.h"
#include "cactus_io_AM2315.h"
#include "math.h"
#include <LowPower.h>



//Thingspeak configuracion
byte mac[] = SECRET_MAC;

// Set the static IP address to use if the DHCP fails to assign
IPAddress ip(172, 16, 15, 249/22);
IPAddress myDns(172, 16, 0, 13);

EthernetClient client;

unsigned long myChannelNumber = SECRET_CH_ID;
const char * myWriteAPIKey = SECRET_WRITE_APIKEY;

//Crea la instancia de los sensores utilizados
Weather placa_roja; //sensor de temperatura y humedad interior SI7021
AM2315 am2315;

//Hardware pin definitions
// digital I/O pins
const byte WSPEED_PIN = 3;

// analog I/O pins
const byte WDIR = A0;

float tempc_int = 0;
float HR_ext = 0;
float temp_ext = 0;
float voltaje_bateria = 0;
float voltaje_panel = 0;
float corriente_panel = 0;
//Sensor de corriente
float sensibilidad = 0.135;
float sum_cos = 0;
float sum_sin = 0;
float avg_cos = 0;
float avg_sin = 0;
int winddir_avg2m_sincos = 0;

//Global Variables
//-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
long lastSecond; //The millis counter to see when a second rolls by
byte seconds; //When it hits 60, increase the current minute
byte seconds_2m; //Keeps track of the "wind speed/dir avg" over last 2
minutes array of data
```

```
byte minutes; //Keeps track of where we are in various arrays of data
byte minutes_10m; //Keeps track of where we are in wind gust/dir over last 10
minutes array of data

long lastWindCheck = 0;
volatile long lastWindIRQ = 0;
volatile byte windClicks = 0;

unsigned long tiempo_anterior = 0;
unsigned long tiempo_anterior2 = 0;
const long intervalo = 300000;
const long intervalo2 = 120000;

//We need to keep track of the following variables:
//Wind speed/dir each update (no storage)
//Wind speed/dir, avg over 2 minutes (store 1 per second)

#define WIND_AVG_SIZE 120

byte windspdavg[WIND_AVG_SIZE]; //120 bytes to keep track of 2 minute average
int winddiravg[WIND_AVG_SIZE]; //120 ints to keep track of 2 minute average

//These are all the weather values that wunderground expects:
int winddir = 0; // [0-360 instantaneous wind direction]
float windspeedkph = 0; // [kph instantaneous wind speed]
float windspdkph_avg2m = 0; // [2 minute average wind speed kph]
int winddir_avg2m = 0; // [0-360 2 minute average wind direction]

void wspeedIRQ()
// Activated by the magnet in the anemometer (2 ticks per rotation), attached
to input D3
{
    if (millis() - lastWindIRQ > 10) // Ignore switch-bounce glitches less
than 10ms (142MPH max reading) after the reed switch closes
    {
        lastWindIRQ = millis(); //Grab the current time
        windClicks++; //There is 1.492MPH for each click per second.
    }
}

void setup() {

// put your setup code here, to run once:
  Serial.begin(9600);   // open serial over USB at 9600 baud

//Initialize the I2C sensors and ping them
  placa_roja.begin();
  am2315.begin();

  pinMode(WSPEED_PIN, INPUT_PULLUP); // input from wind meters windspeed
sensor

//Configuracion anemometro, pluviometro y dir. viento

  seconds = 0;
  lastSecond = millis();
```

```
// attach external interrupt pins to IRQ functions
  attachInterrupt(1, wspeedIRQ, FALLING);

// turn on interrupts
  interrupts();

// //Thingspeak configuracion
//  Ethernet.init(10);  // Most Arduino Ethernet hardware
//  Serial2.begin(115200);  //Initialize serial
//  while (!Serial) {
//    ; // wait for serial port to connect. Needed for Leonardo native USB
port only
//  }
//
//  // start the Ethernet connection:
//  Serial.println("Initialize Ethernet with DHCP:");
//  if (Ethernet.begin(mac) == 0) {
//    Serial.println("Failed to configure Ethernet using DHCP");
//    // Check for Ethernet hardware present
//    if (Ethernet.hardwareStatus() == EthernetNoHardware) {
//      Serial.println("Ethernet shield was not found.  Sorry, can't run
without hardware. :(");
//      while (true) {
//        delay(1); // do nothing, no point running without Ethernet hardware
//      }
//    }
//    if (Ethernet.linkStatus() == LinkOFF) {
//      Serial.println("Ethernet cable is not connected.");
//    }
//    // try to congifure using IP address instead of DHCP:
//    Ethernet.begin(mac, ip, myDns);
//  } else {
//    Serial.print("  DHCP assigned IP ");
//    Serial.println(Ethernet.localIP());
//  }
//  // give the Ethernet shield a second to initialize:
//  delay(1000);
//
//  ThingSpeak.begin(client);  // Initialize ThingSpeak

}

void thingspeak_5min() {

    // set the fields with the values
  ThingSpeak.setField(1, temp_ext);
  ThingSpeak.setField(2, HR_ext);
  ThingSpeak.setField(3, tempc_int);
//  ThingSpeak.setField(4, windspdkph_avg2m);
//  ThingSpeak.setField(5, winddir_avg2m_sincos);
  ThingSpeak.setField(6, voltaje_panel);
  ThingSpeak.setField(7, corriente_panel);
  ThingSpeak.setField(8, voltaje_bateria);


  // write to the ThingSpeak channel
  int x = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
```

```
  if(x == 200){
    Serial.println("Channel update successful.");
  }
  else{
    Serial.println("Problem updating channel. HTTP error code " + String(x));
  }

}

void thingspeak_2min() {

    // set the fields with the values
//  ThingSpeak.setField(1, temp_ext);
//  ThingSpeak.setField(2, HR_ext);
//  ThingSpeak.setField(3, tempc_int);
  ThingSpeak.setField(4, windspdkph_avg2m);
  ThingSpeak.setField(5, winddir_avg2m_sincos);
//  ThingSpeak.setField(6, voltaje_panel);
//  ThingSpeak.setField(7, corriente_panel);
//  ThingSpeak.setField(8, voltaje_bateria);


  // write to the ThingSpeak channel
  int x = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
  if(x == 200){
    Serial.println("Channel update successful.");
  }
  else{
    Serial.println("Problem updating channel. HTTP error code " + String(x));
  }

}

void Interior()
{
  //Sensor de temperatura y humedad interior SI7021
  tempc_int = placa_roja.getTemp();
}

void exterior() {
  am2315.readSensor();
  temp_ext=am2315.getTemperature_C();
  HR_ext=am2315.getHumidity();
}

void Superior() {

    //Keep track of which minute it is
  if(millis() - lastSecond >= 1000)
    {
        lastSecond += 1000;

        //Take a speed and direction reading every second for 2 minute
average
        if(++seconds_2m > 119) seconds_2m = 0;
```

```
        //Calc the wind speed and direction every second for 120 second to
get 2 minute average
        float currentSpeed = get_wind_speed();
        windspeedkph = currentSpeed;//update global variable for windspeed
when using the printWeather() function

        int currentDirection = get_wind_direction();
        windspdavg[seconds_2m] = (int)currentSpeed;
        winddiravg[seconds_2m] = currentDirection;

        if(++seconds > 59)
        {
            seconds = 0;

            if(++minutes > 59) minutes = 0;
            if(++minutes_10m > 9) minutes_10m = 0;
        }
    }
  delay(100);
}

//Calculates each of the variables that wunderground is expecting
void calcWeather()
{
    //Calc winddir
    winddir = get_wind_direction();

    //Calc windspdkph_avg2m
    float temp = 0;
    for(int i = 0 ; i < 120 ; i++)//120
        temp += windspdavg[i];
    temp /= 120.0;
    windspdkph_avg2m = temp;
}

//Returns the instataneous wind speed
float get_wind_speed()
{
    float deltaTime = millis() - lastWindCheck; //750ms

    deltaTime /= 1000.0; //Covert to seconds

    float windSpeed = (float)windClicks / deltaTime; //3 / 0.750s = 4

    windClicks = 0; //Reset and start watching for new wind
    lastWindCheck = millis();

    windSpeed *= 2.4; //km/h

    return(windSpeed);
}

//Read the wind direction sensor, return heading in degrees
int get_wind_direction()
{
    unsigned int adc;
```

```
    adc = analogRead(WDIR); // get the current reading from the sensor

    // The following table is ADC readings for the wind direction sensor
output, sorted from low to high.
    // Each threshold is the midpoint between adjacent headings. The output
is degrees for that ADC reading.
    // Note that these are not in compass degree order! See Weather Meters
datasheet for more information.

    if (adc < 380) return (113);
    if (adc < 393) return (68);
    if (adc < 414) return (90);
    if (adc < 456) return (158);
    if (adc < 508) return (135);
    if (adc < 551) return (203);
    if (adc < 615) return (180);
    if (adc < 680) return (23);
    if (adc < 746) return (45);
    if (adc < 801) return (248);
    if (adc < 833) return (225);
    if (adc < 878) return (338);
    if (adc < 913) return (0);
    if (adc < 940) return (293);
    if (adc < 967) return (315);
    if (adc < 990) return (270);
    return (-1); // error, disconnected?
}

void compute_windir()
{
    sum_cos = 0;
    sum_sin = 0;
    for(int i = 1 ; i < WIND_AVG_SIZE ; i++)
    {
        sum_cos += windspdavg[i]*cos(winddiravg[i]*3.1416/180);
        sum_sin += windspdavg[i]*sin(winddiravg[i]*3.1416/180);
    }

    avg_cos = sum_cos/WIND_AVG_SIZE;
    avg_sin = sum_sin/WIND_AVG_SIZE;

    winddir_avg2m_sincos = atan2(avg_sin,  avg_cos)*180/3.1416;
    winddir_avg2m_sincos = (360 + winddir_avg2m_sincos)% 360;
}

void sensores() {

  voltaje_bateria = (float)25*analogRead(A8)/1023; //A8
  voltaje_panel = (float)25*analogRead(A9)/1023;  //A9

  corriente_panel=get_corriente(500);//obtenemos la corriente promedio de 500
muestras

}

float get_corriente(int n_muestras)
{
```

```
  float voltajeSensor;
  float corriente=0;
  for(int i=0;i<n_muestras;i++)
  {
    voltajeSensor = analogRead(A10) * (5.0 / 1023.0);////lectura del sensor
//A10
    corriente=corriente+(voltajeSensor-2.5)/sensibilidad; //Ecuación  para
obtener la corriente

    if (corriente < 0){
    corriente = 0;
    }
  }
  corriente=corriente/n_muestras;
  return(corriente);
}



void loop() {

    unsigned long tiempo_actual = millis();
    unsigned long tiempo_actual2 = millis();

    if (tiempo_actual2 - tiempo_anterior2 >= intervalo2){

      tiempo_anterior2 = tiempo_actual2;
      thingspeak_2min();

    }

    if (tiempo_actual - tiempo_anterior >= intervalo)  {

      tiempo_anterior = tiempo_actual;

      Interior();
      exterior();
      sensores();
//      Superior(); //# crea los arrays de speed y direction
//      calcWeather(); //# calcula los promedios de speed y direction
//      compute_windir();
      thingspeak_5min();
//    printInfo();
    }

    else {
      Superior();
      calcWeather();
      compute_windir();
      LowPower.idle(SLEEP_4S, ADC_OFF, TIMER5_OFF, TIMER4_OFF, TIMER3_OFF,
                    TIMER2_OFF, TIMER1_OFF, TIMER0_ON, SPI_OFF, USART3_OFF,
                    USART2_OFF, USART1_OFF, USART0_ON, TWI_OFF);
//      LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF);
    }
}
```