

# IMPLEMENTACIÓN DE UN SENSOR DE VIENTO CON ARDUINO



CALDERON ANGEL  
Universidad Nacional de Salta  
Facultad de Ciencias Exactas

# Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Objetivos</b>  | <b>3</b>  |
| <b>2</b> | <b>Contexto y justificación del trabajo</b>                 | <b>3</b>  |
| <b>3</b> | <b>Antecedentes</b>   | <b>4</b>  |
| <b>4</b> | <b>Marco teórico</b>  | <b>5</b>  |
| 4.1      | Concepto meteorológico . . . . .                            | 5         |
| 4.2      | Sensor de viento WH-1081 . . . . .                          | 5         |
| 4.3      | Círculo interno del WH-1081 . . . . .                       | 7         |
| 4.4      | Sensor de viento Belfort model 1074 . . . . .               | 8         |
| 4.5      | Campo magnético terrestre y declinación magnética . . . . . | 10        |
| <b>5</b> | <b>Componentes electrónicos para la experiencia</b>         | <b>13</b> |
| 5.1      | Placa arduino . . . . .                                     | 13        |
| 5.2      | Fuente de alimentación . . . . .                            | 14        |
| 5.3      | Módulo micro SD . . . . .                                   | 16        |
| 5.4      | Magnetómetro HMC5583L de tres ejes . . . . .                | 17        |
| <b>6</b> | <b>Conexión y ensamblado del sensor</b>                     | <b>21</b> |
| 6.1      | Conexión del sensor WH-1081 . . . . .                       | 21        |
| 6.2      | Conexión del sensor Belfort modelo 1074 . . . . .           | 23        |
| <b>7</b> | <b>Metodología de trabajo: sensor WH-1081</b>               | <b>25</b> |
| <b>8</b> | <b>Desarrollo experimental</b>                              | <b>26</b> |

|           |   |           |
|-----------|---|-----------|
| 8.1       | Primera experiencia: Determinación de resistores del sensor de viento . . . . .           | 26        |
| 8.2       | Segunda experiencia: Determinación de tensiones de salida del sensor de viento . . . . .  | 28        |
| 8.3       | Tercera experiencia: Ensamble e instalación de la estación meteorológica . . . . .        | 28        |
| 8.4       | Cuarta experiencia: Calibración del sensor de viento Belfort 1074 usando un potenciómetro | 30        |
| <b>9</b>  | <b>Programación. Adquisición de datos</b>   | <b>30</b> |
| 9.1       | Programa 1: Almacenamiento de datos con Módulo micro SD . . . . .                         | 31        |
| 9.2       | Programa 2: Vision de posición de veleta en tiempo real . . . . .                         | 34        |
| 9.3       | Programa 3: Calibración de la posición del sensor de viento . . . . .                     | 37        |
| <b>10</b> | <b>Errores de medición</b>  | <b>41</b> |
| <b>11</b> | <b>Conclusiones</b>   | <b>42</b> |
| <b>12</b> | <b>Glosario</b>   | <b>45</b> |
|           | <b>Bibliografía</b>   | <b>50</b> |

## Objetivos

1. Hacer uso de una estación meteorológica con el fin de tomar medidas y realizar el seguimiento de la variable dirección de viento.
2. Establecer un sistema automatizado que adquiera los datos del sensor de viento, mediante el uso del programa Arduino y Scilab.
3. Mediante una brújula electrónica, hacer un programa que controle y verifique la posición de la estación meteorológica. De modo que el sensor de dirección de viento apunte siempre al verdadero "Norte".

## Contexto y justificación del trabajo

El fin del proyecto es realizar medidas y hacer un seguimiento de la variable de dirección del viento. Para el seguimiento de esta variable atmosférica se deberá utilizar transductores los cuales transforman la magnitud física en una señal eléctrica. Por lo que, una parte importante de este proyecto es escoger apropiadamente el dispositivo para acondicionar la señal y así, poder interpretarlas y hacer un correcto seguimiento. El encargado de realizar y almacenar los seguimientos será el sistema Arduino.

Podemos decir que este trabajo se divide en dos partes. La primera consiste en trabajar con la estación meteoerológica *WH-1081*, tomar las medidas de la dirección del viento mediante Arduino y luego procesarlas y mostrar en una pantalla usando el programa Scilab (objetivos 1 y 2). La segunda parte consiste en implementar en una estación meteorológica, una brújula digital de Arduino (magnetómetro HCM5883L) para calibrar la posición del "norte" marcado en la estación con el [norte geográfico \[15\]](#); para dicha experiencia se trabaja con el sensor de viento *Belfort model 1074* (ver sección 4.4) (objetivo 3). ¿Porqué corregir la posición? Supongamos que queremos dejar nuestro sensor meteorológico en una región con un alto nivel de actividad sísmica. El ensamblaje y fijación de nuestro sensor en dicho lugar puede tener múltiples fluctuaciones futuras. Es decir, el norte del sensor con lo cual calibrámos desde el comienzo con el norte verdadero puede

cambiar, y de esta manera las mediadas de dirección del viento no serían correcta. Este es uno de los motivos por el cual se decidió implementar este corrector automático de la posición del sensor de viento.

Se escogió el programa Arduino porque, a de más de ser sencillo, flexible y libre para los usuarios que no tienen mucho conocimiento de programación, posee una gran variedad de componentes electrónicos y sensores que son útiles para la adquisición de señales analógicas de muchos sistemas de medición, y sobre todo de una estación meteorológica.

La mayor parte de este proyecto es práctico y sencillo, ya que no se usa nuevos conceptos más de lo vistos en los primeros años de las carreras universitarias en Ciencias Exactas.

Si bien en este proyecto sólo se estudia un sólo sensor de la estación, la implementación de automatización con Arduino se puede extender a todos los sensores restantes (anemómetro, pluviómetro, termómetro, etc), y así crear nuevos proyectos de estudios e **investigaciones futuras**.

Finalmente, cabe aclarar que este informe fué tipeado mediante **LAT<sub>E</sub>X**<sup>[7]</sup> [ver glosario 6].

SECCIÓN 3

## Antecedentes

Hay antecedentes de proyectos que se asemejan a los objetivos 1 y 2. Uno de los cuales se basa en un trabajo de tesis: Diseño e implementación de una estación meteorológica con Raspberry Pi [1] ([docplayer.es/65904272-Diseno-e-implementacion-de-una-estacion-meteorologica-con-raspberry-pi.html](http://docplayer.es/65904272-Diseno-e-implementacion-de-una-estacion-meteorologica-con-raspberry-pi.html)). Realizado en 2016 con la titulación en Master Ingeniería de Telecomunicación electrónica.

En este proyecto de tesis el autor trabaja con todos los sensores de una estación meteorológica (), cuyo objetivo es automatizar el seguimiento de las distintas variables meteorológicas usando el sistema operativo **Raspberry Pi** [20].

Si bien en nuestro trabajo sólo analizamos un solamente el sensor de dirección de viento, la automatización del almacenamiento y seguimiento de la posición de la veleta se lo hace mediante el sistema Arduino y Scilab.

Por otra parte, no se encontraron proyectos similares al objetivo 3. La idea de automatizar la posición de la estación meteorológica sale de un experimento hecho en la terraza de la Facultad de Ciencias (UNSa), antela necesidad de contar con una brújula (digital o analógica) para posicionar el sensor de viento con el [norte geográfico](#) [15].

SECCIÓN 4

## Marco teórico

### 4.1 Concepto meteorológico

El viento es un movimiento natural de masas de aire. Los desequilibrios térmicos entre unos lugares y otros provocan diferencia de presión atmosféricas las cuales producen los vientos. Debido a ello, se desencadena un flujo de aire desde las presiones altas hacia las mas bajas, con tendencia de seguir el gradiente de presión. Los gradientes fuertes de presión que se representan en los mapas meteorológicos con isobaras muy próximas, ocasionan vientos fuertes, mientras que allá donde el gradiente de presión es pequeño y por tanto las isobaras se encuentran muy alejadas entre ellas, los vientos son flojos. Las calmas devienen en aquellas áreas donde no hay diferencia de presión atmosférica especialmente en los centros de los anti ciclones.

El instrumento para medir la velocidad del viento es el **anemómetro** y la dirección del viento se mide mediante una **veleta**. Por convención se toma la dirección del viento allá donde viene y no hacia donde bufa.

### 4.2 Sensor de viento WH-1081

La veleta es un dispositivo giratorio que consta de una placa que gira libremente y un señalador que indica la dirección del viento. Mediante un circuito resistivo, el sensor es capaz de indicarnos con qué punto cardinal coincide la dirección de que viene el viento (si es que se tratara de una veleta de ochos direcciones).

En este proyecto se trabajó con el sensor de la estación meteorológica WH-1081.

En el manual de instalación del equipo [2] se indican los pasos del montaje de los distintos

dispositivos.



Figure 1

En el borde del sensor encontramos una "muestra" que dice "N", la cual tiene que apuntar hacia el norte, con el fin de que el sensor tenga una referencia correcta. Para hacer coincidir la muestra o marca del sensor con el norte verdadero, se hará uso de una brújula analógica o digital. Mas adelante se hablará del sistema de brújula electrónico para evitar esta trabajo manual.

También, en dicho manual se indica [2] se indica como es la conexión para la comunicación de los distintos sensores (termómetro, pluviómetro, anemómetro, etc) con la estación meteorológica. Por ejemplo, el cable que sale del anemómetro tiene que ser insertado en la clavija telefónica También, en dicho manual se indica (RJ-11) del el sensor de dirección de viento. Luego, el cable del sensor de dirección de viento tiene que ser insertado en la clavija telefónica localizada en el sensor de temperatura y humedad, con la marca "*Wind*" encima.

### 4.3 Circuito interno del WH-1081

En la siguiente figura se muestra como es el circuito interior de la veleta y la interface de entrada:

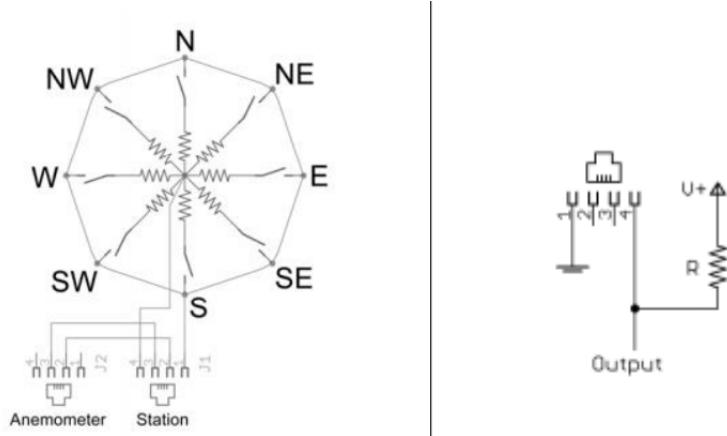


Figure 2: Esquema interno del sensor e interface de entrada a la veleta con el conector RJ-11

En este modelo de veleta se observan ocho resistencias, donde cada una corresponde a una dirección del sistema de puntos cardinales (Norte, Sur, Este, Oeste, Nordeste, Noroeste, Sureste, Sudoeste).

Entonces, se pude hacer una idea de que la veleta se comporta como una resistencia variable  $R_{vel}$  y que mediante un circuito externo podemos medir la tensión de salida (señal analógica del sensor), luego con este dato podemos obtener la dirección del viento. El circuito equivalente del sensor de viento es:

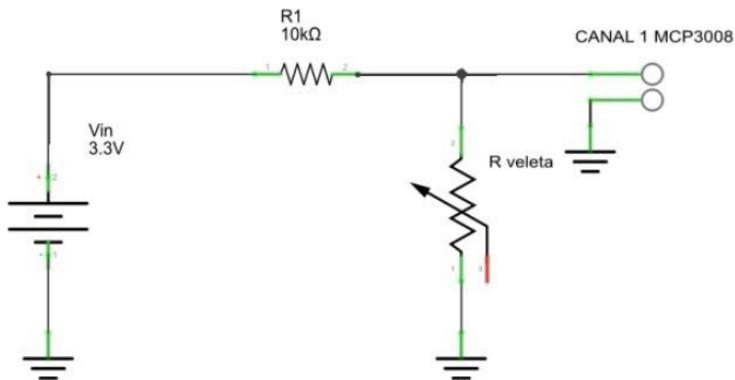


Figure 3: circuito equivalente del sensor de viento

El fabricante en su hoja de características nos indica los valores de las resistencias  $R_{vel}$  dependiendo de donde apunte la veleta, así como la tensión de salida. Esta tabulación se

da para una resistencia de carga  $R_1$  y una fuente de tensión  $V_{in}$  definidos por el fabricante. Como no se cuenta con estos datos tabulados, lo que se hará en la práctica será medir los valores de las ocho resistencias con un multímetro, y fabricar un **circuito externo** para la toma de datos.

Las ecuaciones que nos permiten obtener las tensiones de salida son:

1. Aplico Ley de Voltaje de Kirchhoff

$$V_{in} - I * R_1 - I * R_{vel} = 0 \quad (4.1)$$

$I$  es la corriente que circula en el circuito.

2. Despejo  $I$ .

$$I = \frac{V_{in}}{R_1 + R_{vel}} \quad (4.2)$$

3. La tensión de salida de la resistencia variable es

$$V_{out} = I * R_{vel} \quad (4.3)$$

4. Reemplazo  $I$ .

$$V_{out} = \frac{R_{vel}}{R_1 + R_{vel}} * V_{in} \quad (4.4)$$

Sin embargo, en la práctica las tensiones  $V_{out}$  se las medirá directamente del circuito (sección 8.2).

#### 4.4 Sensor de viento Belfort model 1074

Como se dijo en la sección 2, se usará el sensor de viento Belfort modelo 1074-12 [19] para la experiencia de calibración de la posición de la estación con el [norte geográfico](#) [15]. .

El sensor Belfort es usado para medir específicamente velocidad y dirección de viento.

Se trata de un sensor combinado de copa y veleta. Las copas del anemómetro se colocan por encima de la paleta de dirección en una configuración coaxial (es decir, sobre un mismo eje de rotación). Este diseño simplifica los requisitos generales de manipulación, montaje y cableado. La paleta de dirección tiene un diseño de pala única, que incorpora



Figure 4: Belfort wind sensor 1074

equilibrio estático y aerodinámico para una respuesta óptima a flujos de aire muy bajos y precisión en todos los rangos de velocidades de viento. El movimiento de la paleta se transfiere mediante un engranaje loco uno a uno al alojamiento principal. A continuación se muestra las especificaciones del sensor de dirección de viento [ver [19]]:

1. **Rango** [7]: 0-360°.
2. **Precisión** [8]:  $\pm 3.6^\circ$ , para una escala completa de 360°.
3. **Resolución** [9]: Infinita.
4. **Linealidad** [10]: 0.25% de la escala completa.
5. **Umbral** [11]: 0.34 m/s (0.75mph).
6. **Distancia constante** [12]: 1.2m, 50% de recuperación.
7. **Relación de amortiguación** [13]: 0.5 a 0.6.
8. Potencia requerida: 5V, referencia regulada.
9. Dimensiones: 53.3x81.5x17.8cm
10. Peso: 4.3kg.
11. Trajeta de acondicionamiento: 0-5V, 4-20mA.

El sensor de viento cuenta con un sistema de engranajes conectados a un **potenciómetro de multivuelta de  $20k\Omega$  marca Spectrol modelo 708-96** como se muestra en la figura 5. Por ejemplo, el movimiento de la paleta cambia el valor de la resistencia entre los pines 1 y 2 del potenciómetro.

Entonces para obtener los valores de la dirección de la veleta, trabajamos con el divisor de tensión (potenciómetro) alimentando con 5V un extremo y el otro conectándolo a tierra desde Arduino. El pin del medio (pin1) lo conectamos a un pin analógico de Arduino (A0 por ejemplo) quien recibirá la señal analógica de tensión generada por el movimiento de la paleta.

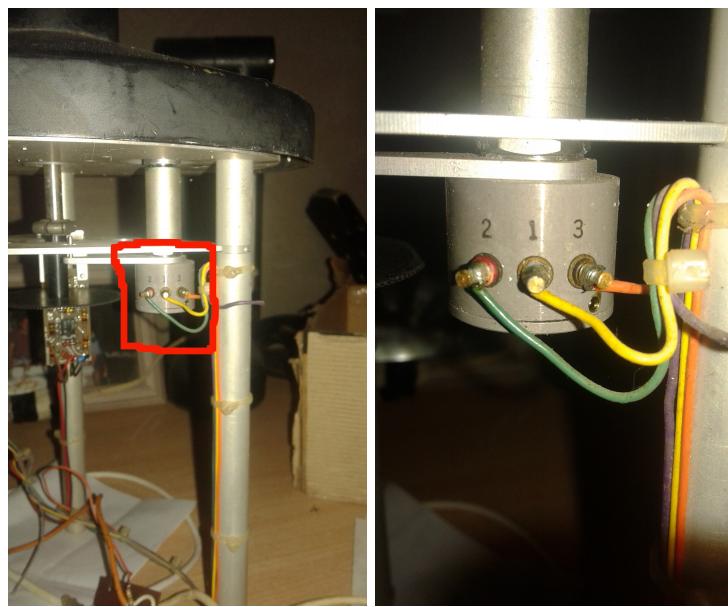


Figure 5: Potenciómetro de  $20k\Omega$  Spectrol conectado a la paleta

Vamos a usar la tarjeta Arduino para condicionar la señal y lograr un formato de datos de dirección de viento de  $360^\circ$  (ver sección 6.2).

#### 4.5 Campo magnético terrestre y declinación magnética

La **declinación magnética** [17]. en un punto de la tierra es el **ángulo** comprendido entre el **norte magnético** [16]. y el **norte geográfico** [15]. (norte verdadero). En otras palabras, es la diferencia entre el norte geográfico y el indicado por una brújula (norte magnético)[15].

Por convención, a la declinación se la considera de valor positivo si el norte magnético se encuentra al este del norte verdadero, y negativa si se ubica al oeste.

La declinación magnética se origina a causa de que los polos magnéticos de la tierra no coinciden con los polos geográficos. Esto hace que la aguja de nuestra brújula, al orientarse con las líneas de fuerza de este "gran íman" que es la tierra, no apunte al norte geográfico (verdadero) sino al norte magnético.

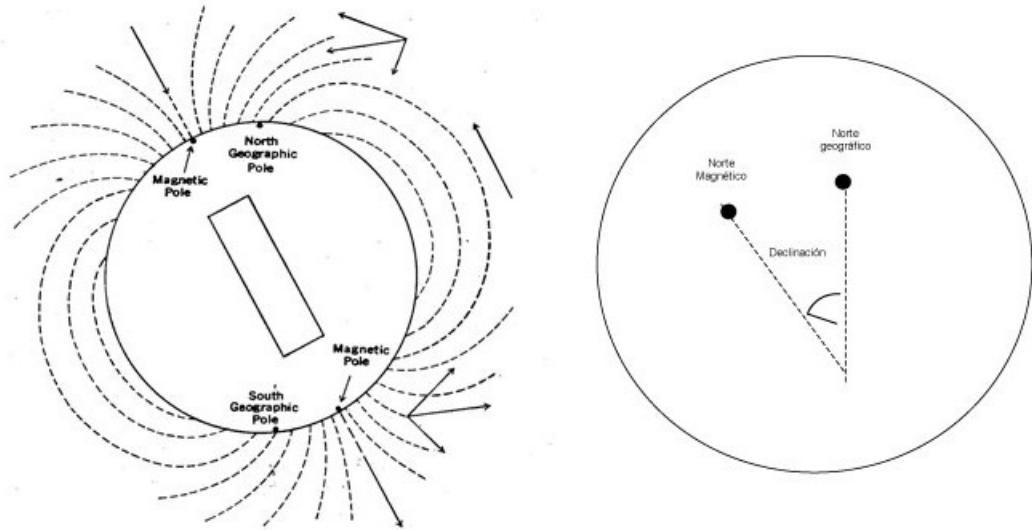


Figure 6

Las líneas de igual declinación magnética se denominan **curvas isogónicas**. A las de valor nulo se denominan **curvas agónicas**, es decir la declinación magnética en un punto de la curva agónica es cero.

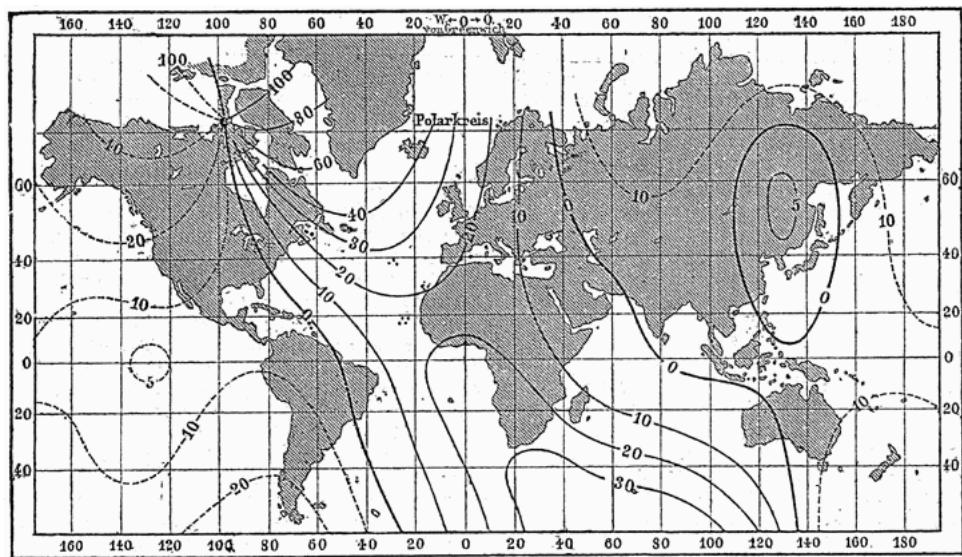


Figure 7: Mapa de curvas isogónicas de 1888

Desde hace muchos tiempo se ha intentado entender el fenómeno del magnetismo terrestre, pero hasta hoy no se ha logrado formar una teoría sólida. Lo que si se sabe es que los **polos magnéticos** cambian permanentemente de posición. Esto hace que la declinación conel

correr de los años una cantidad llamada **”variación anual”**. Existen otras perturbaciones que pueden provocar bruscas variaciones en el registro de la aguja, que en algunos casos pueden superar los  $50^\circ$  o  $60^\circ$ , tales fenómenos son causados por la formación geológica de la zona. Los casos más significativos se dan en zonas con islas de constitución volcánica.

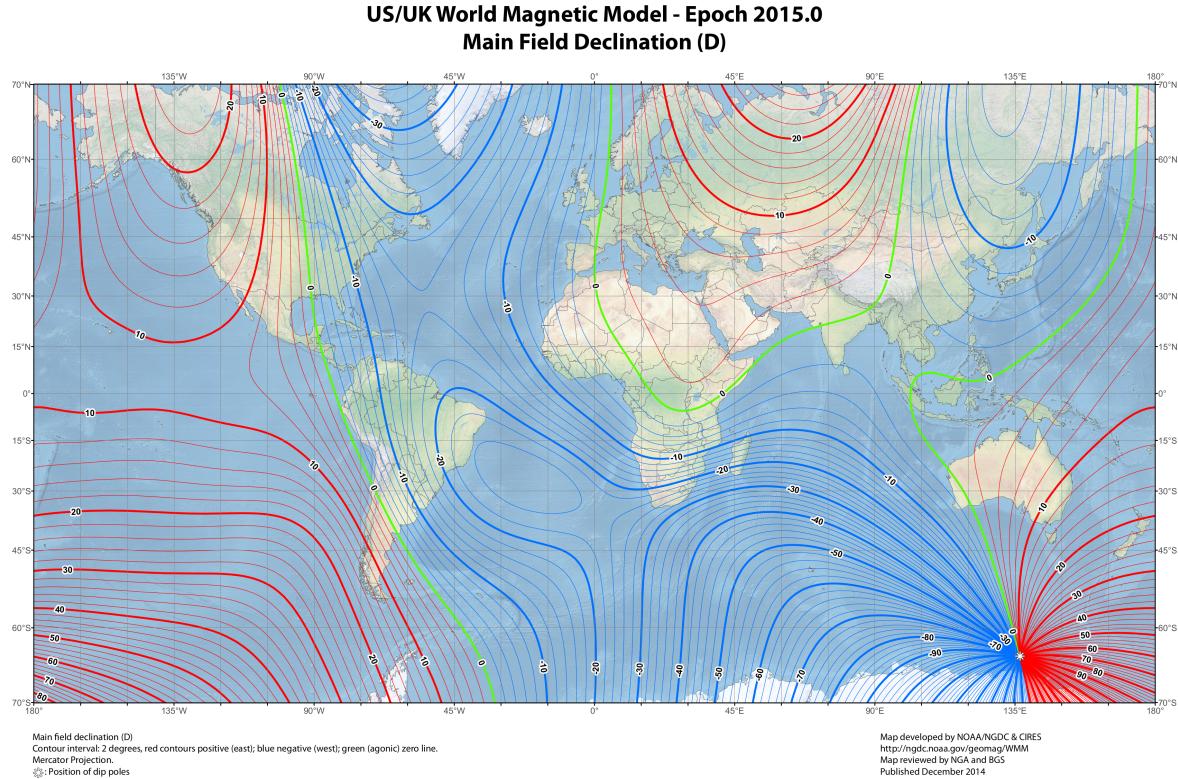


Figure 8: Curva roja: declinación positiva; curva azul: declinación negativa; curva verde: (agónica) declinación nula

Para saber el valor exacto de la declinación magnética en donde estamos ubicados podemos acceder a la siguiente página [www.magnetic-declination.com](http://www.magnetic-declination.com)

Podemos ver en la figura 9 nuestra ubicación y además nos dice que la declinación magnética es negativa y vale  $\beta = -8^\circ 2'$ . Además nos muestra la inclinación (otro parámetro importante en la navegación y Aoeronáutica [ver glosario 5]) la magnitud del campo magnético terrestre en nuestra ubicación.

Entonces para concluir con esta teoría, en la práctica lo que se hará es determinar la dirección del norte magnético mediante una brújula (analógica o digital), y luego adicionar la declinación magnética del lugar en donde nos situamos ( $-8^\circ$ , Salta Capital). Y de esta manera determinar la dirección del norte geográfico (verdadero) que es lo que nos interesa para fijar correctamente la estación meteorológica.

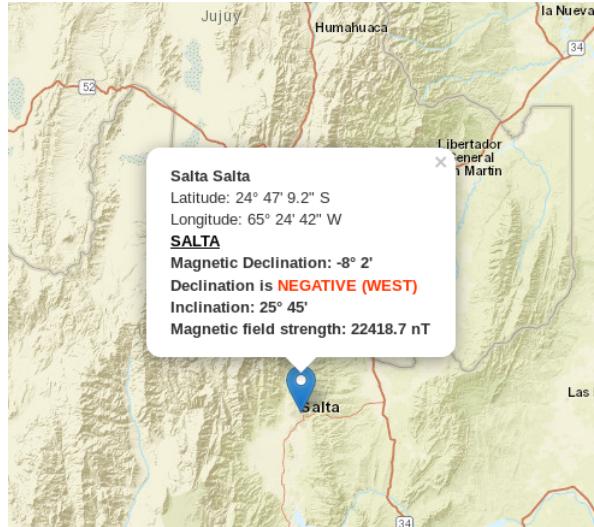


Figure 9

SECCIÓN 5

## Componentes electrónicos para la experiencia

### 5.1 Placa arduino

Arduino es un Software-Hardware de código abierto. Se puede alimentar con una fuente a través de la conexión micro USB o mediante la entrada *Jack* [3].



Figure 10

Posee 20 pines de entrada / salida digital (de los cuales 7 se pueden usar como salidas PWM y 12 como entradas analógicas), un oscilador de cristal de 16MHz, un microcontrolador ATmega32u4, una conexión micro USB, un encabezado ICSP y un botón de reinicio. En la tabla siguiente se especifican sus características:

|                                     |   |
|-------------------------------------|---|
| Microcontrolador                    | ATmega32u4  |
| Tensión de funcionamiento           | 5V  |
| Voltaje de entrada<br>(recomendado) | 7-12V   |
| Voltaje de entrada (límites)        | 6-20V   |
| Pernos digitales de E / S           | 20  |
| Canales PWM                         | 7   |
| Canales de entrada<br>analógica     | 12  |
| Corriente DC por Pin de E /<br>S    | 40 mA   |
| Corriente DC para 3.3V Pin          | 50 mA   |
| Memoria flash                       | 32 KB (ATmega32u4) de los cuales 4 KB<br>utilizados por el cargador de arranque |
| SRAM                                | 2.5 KB (ATmega32u4)   |
| EEPROM                              | 1 KB (ATmega32u4)   |
| Velocidad de reloj                  | 16 MHz  |
| Longitud                            | 68.6 mm   |

## 5.2 Fuente de alimentación

Para alimentar nuestro proyecto vamos a usar baterías de Litio que proveen un tensión de 3.4V cada una y una potencia de 7800mA. Conectados en serie proveen una tensión de 7.8V.



Figure 11

La placa viene diseñada para aceptar alimentación mediante el **jack estandar**. Al tratarse de una entrada de corriente directa, la conexión del eliminador tiene una polaridad que debe ser respetada: el positivo va en el centro del conector.

La placa posee un regulador de tensión interno cuyo voltaje de entrada es de 7 a 12V DC. Valores menores que 7V pueden causar que el regulador no pueda trabajar correctamente. Valores mayores que 12V pueden causar un rápido sobrecalentamiento del regulador, aunque la cantidad de accesorios (demanda de corriente) conectados no sea grande.

Otra manera de alimentar el Arduino es a través del **pin VIN**. El pin VIN se localiza en

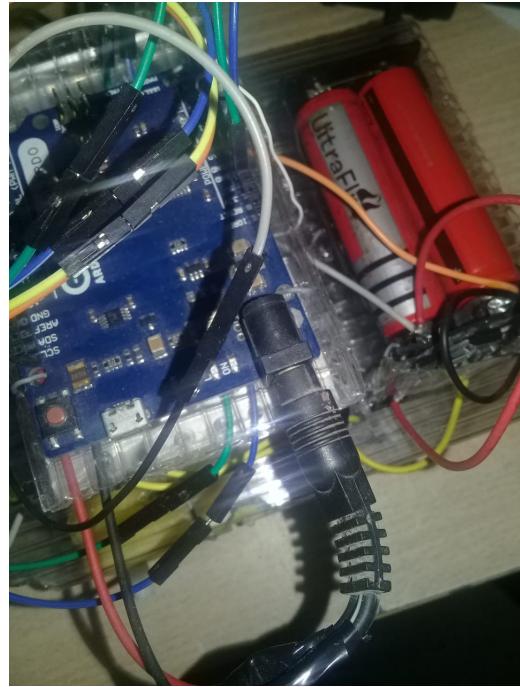


Figure 12

el grupo de pines de alimentación y tierras, y cumple una doble función:

1. Permite aplicar una alimentación externa en un rango de 6 a 12Vdirectamente a la entrada del regulador de la tarjeta. En este caso no se cuenta con protección contra inversión de polaridad ni contra sobre-corriente.
2. Funciona como salida de voltaje cuando el arduino se está alimentando a través del jack estándar. En este caso el voltaje presente en el VIN será aquiel que estamos aplicando en el jack, restando la caída de tensión en el diodo de protección de inversion de polaridad (alrededor de 0.7V).

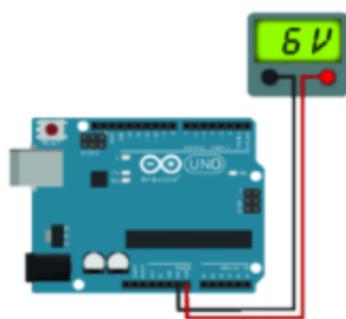


Figure 13: Entrada de alimentación VIN

No se recomienda conectar cargas mayores a 1000mA en este pin, ya que se puede dañar

el diodo de protección.

### 5.3 Módulo micro SD

Las memorias SD son las más usadas por dispositivos portátiles, por su gran capacidad y su reducido tamaño. Estas características nos dan una buena alternativa de almacenamiento para usarlo en arduino, sobre todo cuando necesitamos guardar gran cantidad de información.

La comunicación de la memoria con Arduino es por SPI, para ello necesitamos módulos externos. Traen los componentes necesarios para adaptar los voltajes a TTL y poder conectarlo de forma fácil a nuestro arduino.

Estos módulos nos permiten insertar una memoria micro SD (nosotros vamos a trabajar con una memoria de 16GB, muy grande para pocos datos). Estos dispositivos se pueden alimentar con 3.3V o 5V.

La transferencia de datos entre la placa Arduino y el módulo de micro SD se hará mediante la comunicación SPI[6].

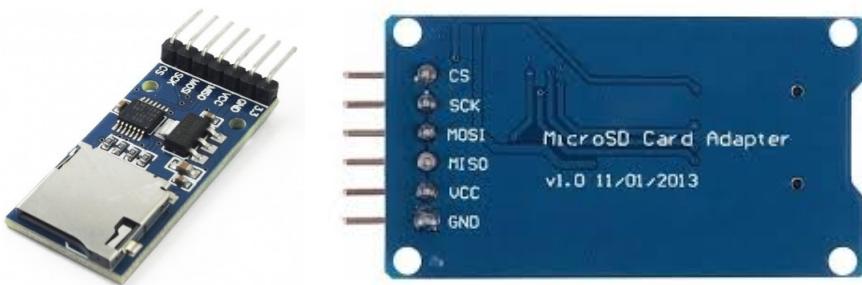


Figure 14

El bus SPI (del inglés *Serial Peripheral Interface*) es un estándar de comunicaciones, usado principalmente para la transferencia de información entre circuitos integrados en equipos electrónicos. Tiene interés como medio de comunicación porque una gran variedad de sensores y dispositivos comerciales disponen de un **interface SPI como medio de comunicación**.

El bus SPI tiene una arquitectura de **maestro-esclavo**. El dispositivo maestro puede iniciar la comunicación con varios dispositivos esclavos, y enviar o recibir datos de ellos. La comunicación de datos entre maestros y esclavos se realiza en dos líneas independientes,

una del maestro a los esclavos, y otra de los esclavos al maestro.

Otra característica SPI es que es un bus **síncrono**. El dispositivo maestro proporciona una señal de reloj, que mantiene a todos los dispositivos sincronizados. Por lo tanto el bus SPI requiere un mínimo de tres líneas:



Figure 15

1. **MOSI** (master-out, slave-in) comunicación del maestro al esclavo.
2. **MISO** (master-in, slave-out) comunicación del esclavo al maestro.
3. **SLK** (clock) señal del reloj enviada por el maestro.

Arduino **dispone de soporte SPI por hardware** vinculado físicamente a ciertos pines. Los pines asociados a SPI varían de un modelo a otro.

Los pines para el Arduino Leonardo son:



Figure 16: Pines para la comunicación SPI con el Arduino

#### 5.4 Magnetómetro HMC5583L de tres ejes

Se les dice *magnetómetro* a los dispositivos que sirven para cuantificar en fuerza o dirección la señal magnética de una muestra.

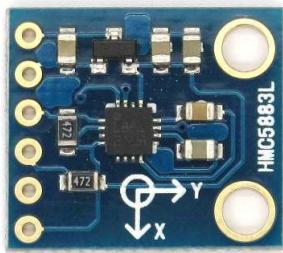


Figure 17

El HMC5883L [9] es un magnetómetro de tres ejes digital, también conocido como brújula digital. Diseñado para medir campos magnéticos débiles (como los que hay presente de forma natural en nuestro planeta). El rango de escala completa es de  $+/- 8$  gauss y la resolución del sensor es de hasta 5 milli-gauss [ver glosario 1].

El rango de alimentación del HCM5883L es de 2.16 a 3.6 V (aunque en el módulo un regulador de voltaje que permite la alimentación en un rango de 3 a 5V) y se comunica de forma digital con el microcontrolador a través de una interface I<sup>2</sup>C [13].

Para entender un poco como funciona este dispositivo veamos su estructura interna.

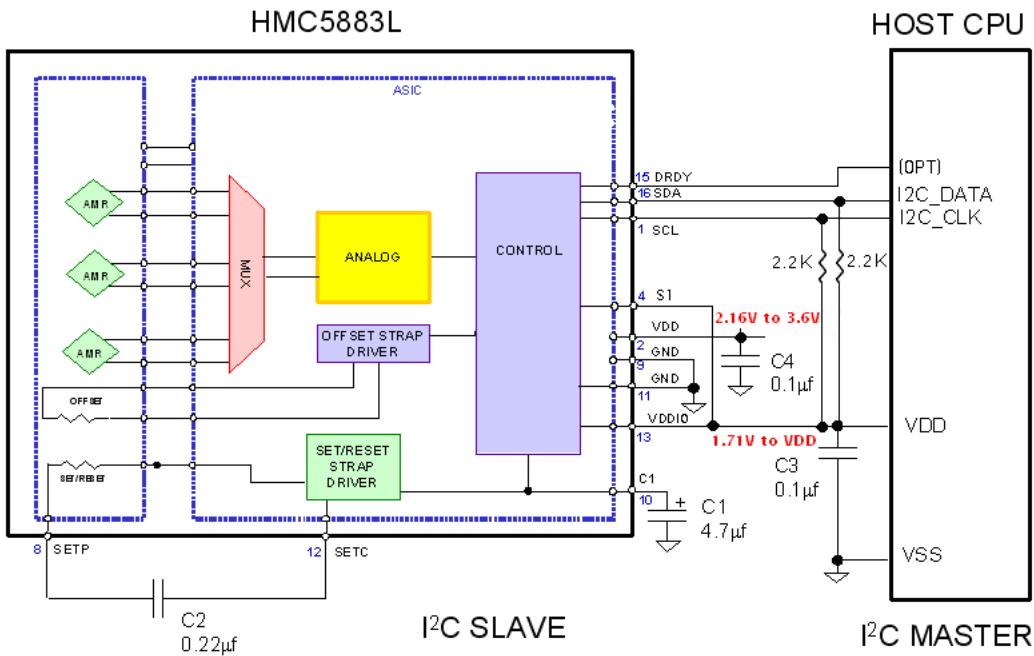


Figure 18

Vemos en la figura 18 un diagrama en bloque del magnetómetro. En la parte izquierda

hay un bloque que tiene los elementos sensores basados en un material magnéto-resistivo, esto significa que cada una de estas resistencias que forman parte de un puente varían su valor en función del campo magnético al cual se expone. Cada puente mide un eje, por eso hay tres puentes que miden las componentes del campo magnético. El resto del circuito integrado convierte las pequeñas variaciones de las resistencias en tensión, y luego convierte estas señales a digital mediante el ADC. Arduino accede a los datos tomados mediante el bus I2C (usando los pines SCL y SDA) [ver glosario 3].

Entonces, lo que faremos en la **experiencia** es detectar el campo magnético terrestre de la misma forma que lo hace una brújula convencional. Se dice que el circuito integrado es de tres ejes porque mide las tres componentes (x,y,z) del campo magnético presente. El eje *x* y el eje *y* están sobre el plano del circuito impreso. En la experiencia sólo nos va a interesar las componentes (x,y), con ellos podremos determinar la posición de nuestro sensor (eje *x*) respecto del **norte geográfico** [15]. .

¿Como vamos a determinara esta dirección? Veamos la figura 19. En color rojo se encuentra la resultante del campo magnético (en nuestra ubicación) de la tierra que apunta hacia el **norte magnético** [16]. .

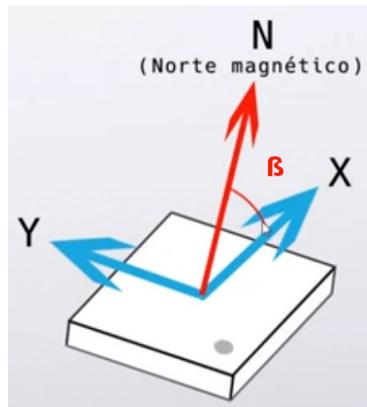


Figure 19

El HMC5883L dará dos valores, es decir las magnitudes de las componentes (x,y) del campo magnético en el punto donde nos situamos. Para obtener la resultante (flecha roja) debemos calcular el ángulo  $\beta$  que se produce respecto del eje x. El eje x del circuito impreso actuará como la aguja de una brújula, entonces si estamos completamente alineado con el norte magnético el ángulo  $\beta$  sería cero. A este ángulo también se lo denomina *azimut magnético* [ver glosario 4]

El ángulo  $\beta$  se calcula como el *arctg* de la división entre la magnitud de la componente x

del campo y la magnitud de la componente y del campo:

$$\beta = \arctg\left(\frac{y}{x}\right) \quad (5.1)$$

De esta manera obtenemos el ángulo  $\beta$  del campo magnético respecto del eje x. Esta fórmula está incluida en la librería que usaremos en la programación.

Entonces, el **HCM5883L** (junto con el programa que presentaremos mas adelante) devuelve azimut magnético ángulo del campo magnético respecto del eje x (impreso en el circuito) en radianes [ver glosario 4]. Para pasar a grados sexagesimal debemos multiplicar por 180 y dividir por  $\pi$ :

$$\beta = \arctg\left(\frac{y}{x}\right) * \frac{180}{\pi} \quad (5.2)$$

Ahora vamos a ver conexión del módulo HCM5883L con la placa Arduino (figura 20).

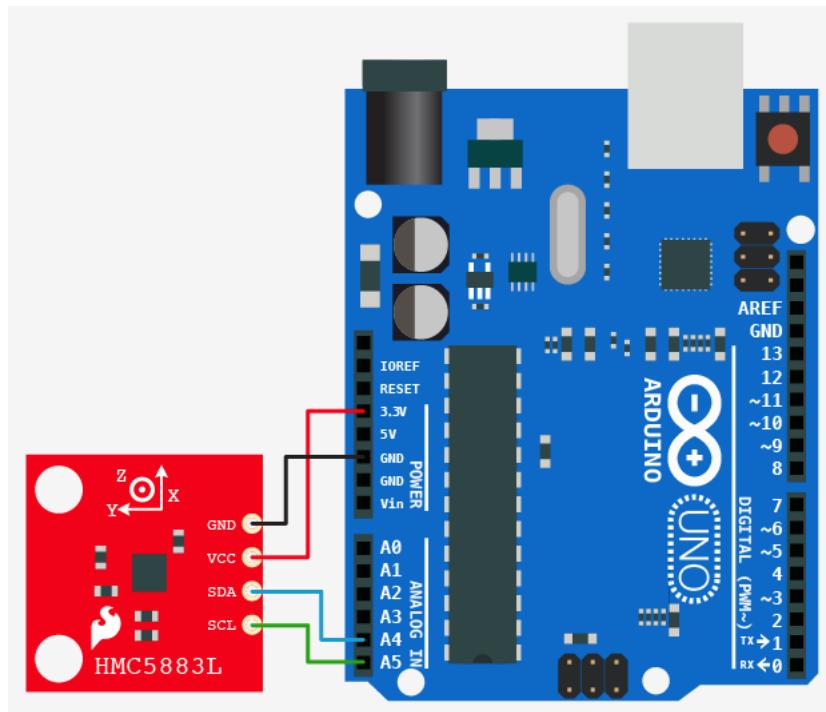


Figure 20

Como estamos usando un Arduino Leonardo, en el pin *VCC* se conecta directamente a 3.3 o 5 V de Arduino, recordemos que el módulo funciona con 3.3V pero también incluye un regulador de tensión para que se pueda usar el riel de 5V de forma directa. El pin *GND* lo conectaremos directamente al *GND* de Arduino. Luego el pin *SCL* del módulo

conectaremos al pin 3 del Arduino Leonardo. Y finalmente el pin *SDA* del módulo lo conectaremos a al pin 2 de Arduino [11].

| HMC5883L | Arduino Uno, Nano, Mini. | Arduino Mega , DUE | Arduino Leonardo |
|----------|--------------------------|--------------------|------------------|
| VCC      | 5V                       | 5V                 | 5V               |
| GND      | GND                      | GND                | GND              |
| SCL      | A5                       | 21                 | 3                |
| SDA      | A4                       | 20                 | 2                |

Figure 21: Conexión I2C entre HCM5883L y Arduino

Para trabajar de forma simple en la programación y reducir la cantidad de código usaremos una librería específica para el HCM5883L, esta librería no está disponible mediante el gestor de librerías de Arduino. Para su descarga puede ingresar al siguiente enlace: [github.com/keepworking/Mecha-QMC5883L](https://github.com/keepworking/Mecha-QMC5883L)

Al trabajar con esta librería lo que vamos a obtener son los valores de las componentes ( $x,y,z$ ) del campo magnético terrestre (en nuestra zona) y el azimut magnético  $\beta$ , que es lo que nos interesa [ver glosario 4].

Mostraremos con más detalle este programa en la sección de programación (sección 9.3).

## SECCIÓN 6

### Conexión y ensamblado del sensor

#### 6.1 Conexión del sensor WH-1081

La veleta cuenta con dos conectores que los tomamos como positivo y tierra. Estos vienen conectado a una ficha RJ-11 de cuatro pines. Los dos pines externos corresponden a la veleta, y los otros dos (internos) están corresponden al anemómetro.

Luego se arma un divisor de tensión, donde los pines que salen del RJ-11 o sensor de viento son los de una resistencia variable, en serie con una resistencia de carga fija  $R = 10K\Omega$ . Se usa una fuente de tensión de 5V proporcionada por la placa Arduino. Para hacer un

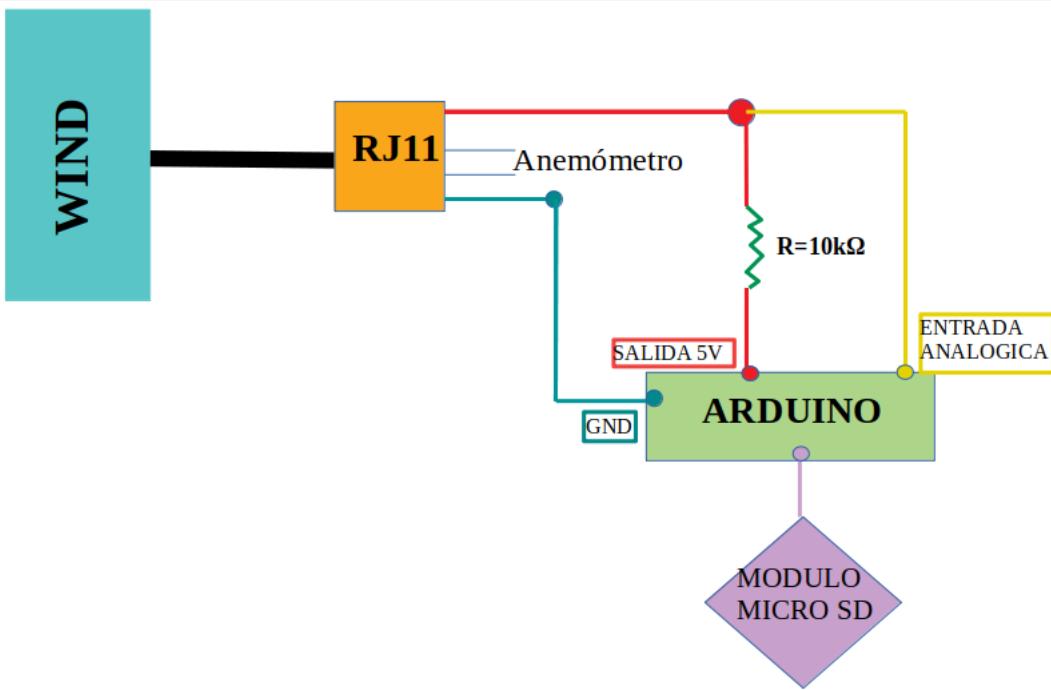


Figure 22: Conexión de la veleta

trabajo prolíjo y organizado, las conexiones del divisor de tensión y la placa Arduino se realizan en un protoboard (chico).

Se conecta un de los pines del sensor de viento a GND (masa proporcionado por Arduino) y el otro estará conectado el divisor de tensión y a un pin analógico de la placa Arduino. Esta entrada analógica se encargará de tomar las señales analógicas (tensiones) dadas por el sensor de viento, la señal recibida cambia al cambiar la posición de la velata.

Y el módulo de micro SD se conecta al Arduino mediante las entradas ISCP.

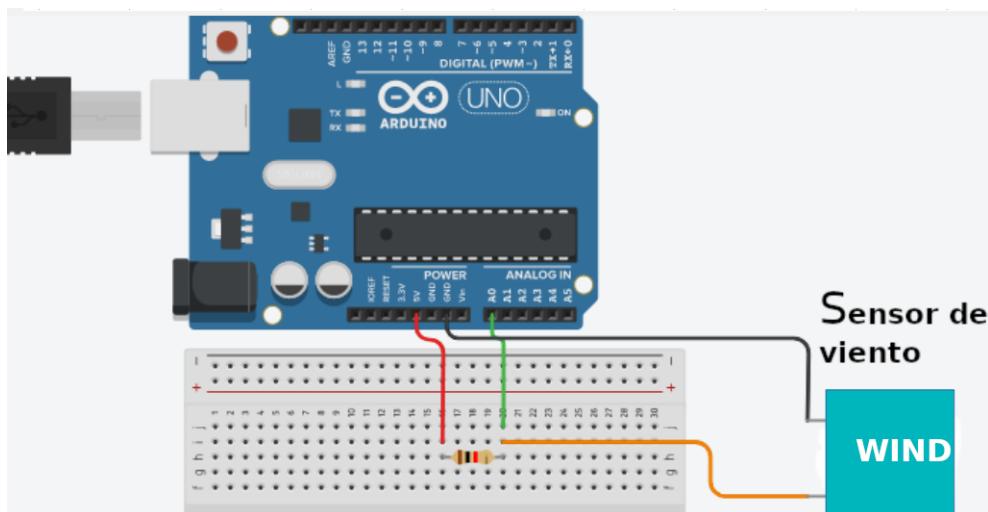


Figure 23: Simulación de conexiones hecho con [TINKERCAD \[19\]](#). [8]

## 6.2 Conexión del sensor Belfort modelo 1074

Como se dijo en la sección 4.4, el sensor de viento Belfort 1074 cuenta con un sistema de engranajes conectados a un potenciómetro de  $20k\Omega$ .

Entonces para obtener los valores de la dirección de la veleta, trabajamos con el divisor de tensión (potenciómetro Spectrol de  $20k\Omega$ ) alimentando con 5V un extremo y el otro conectándolo a tierra desde Arduino. El pin del medio (pin1) lo conectamos a un pin analógico de Arduino (A0 por ejemplo) quien recibirá la señal analógica de tensión generada por el movimiento de la paleta. El esquema es el siguiente:

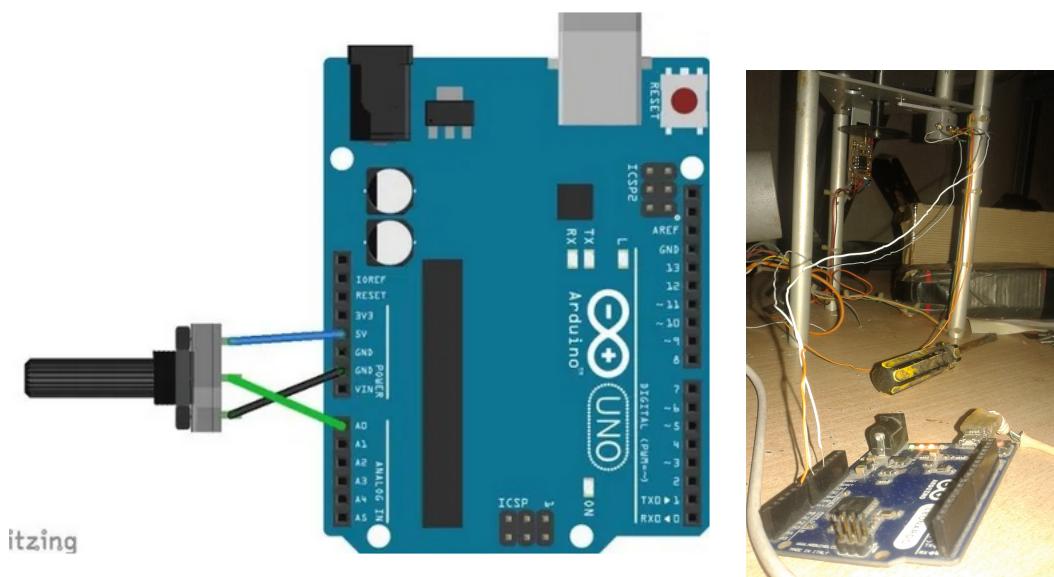


Figure 24

En la sección 6.2 mostraremos la programación de este dispositivo junto con el magnetómetro HCM5883l.

Además de esto el magnetómetro debe ser instalado y fijado en alguna superficie (no móvil) horizontal del sensor de viento, vemos un ejemplo en la figura 25.

Entonces, las dos parámetros importantes que mediremos con Arduino serán:

1. Dirección de la veleta, ángulo de  $0^\circ$ - $360^\circ$  respecto del "norte" marcado en el cuerpo de dicho sensor.
2. Azimut magnético, ángulo de  $0^\circ$ - $360^\circ$  respecto del norte magnético.

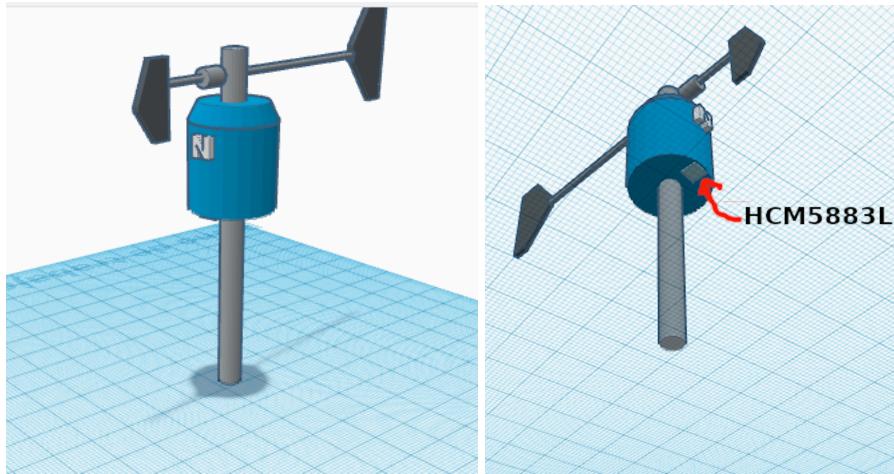


Figure 25: Diseño 3D echo con TINKERCAD [19].

Ambos valores se usarán para determinar el ángulo correcto de la veleta respecto del norte geográfico (norte verdadero).

Para terminar con esta sección veremos como es el esquema de los componentes electrónicos para esta experiencia:

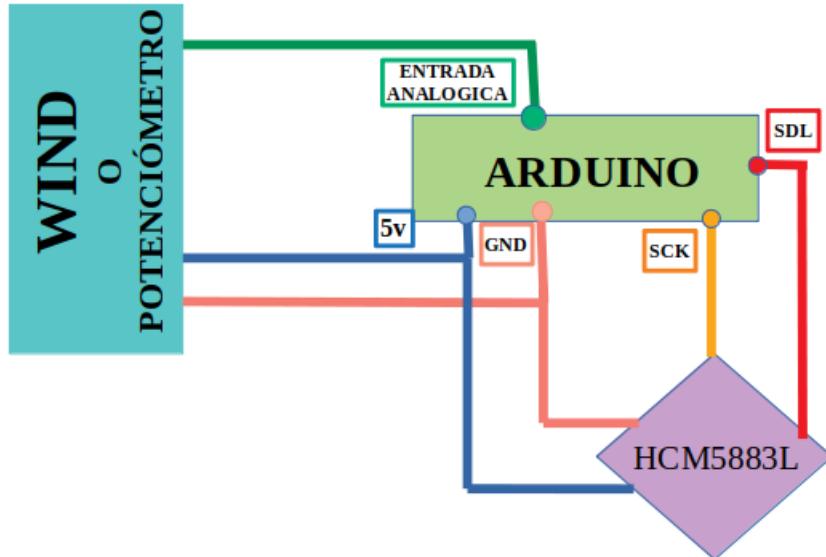


Figure 26

La veleta cuenta con un potenciómetro marca spectrol) de  $20k\Omega$  que posee tres pines. El pin 1 lo conectamos a GND, el pin 2 va a 5V del Arduino, y el pin 3 va al pin de entrada analógico de Arduino. EL pin 3 recibirá la señal analógica en voltios y por medio del AD [18] lo convertirá en una señal digital de 0-1023.

Luego, el magnetómetro HCM5883L cuenta con cuatro pines de los cuales se usa cuatro. El primer pin se conecta a los 5V de Arduino, el segundo al GND (tanto el sensor de viento como el magnetómetro deben estar conectado a la misma masa), el tercero (SCK) se conecta al pin digital 3 de Arduino, y el cuarto (SDA) se conecta al pin digital 2.

Las especificaciones estan dadas en el programa de la sección [9.3](#).

SECCIÓN 7

## Metodología de trabajo: sensor WH-1081

Se realizará la experiencia de acuerdo a los siguientes pasos:

1. Medir, con un multímetro, los ocho valores de las resistencia del sensor de dirección de viento.
2. Armar un divisor de tensión con una fuente de alimentación de 5V porporcionada por la placa Arduino. Medir la tensión de salida del sensor con un multímetro.
3. Hacer dos programa en Arduino:
  - (a) Uno que tome los datos y muestre la posición de la veleta en pantalla en tiempo real. Mediante el programa *Scilab* se crea un scrip que procese los datos medidos. Debe generar un gráfico estadístico de la cantidad de medida en función de la posición de la veleta.
  - (b) Otro que almacene los datos en una memoria micro SD para su posterior evaluación. Mediante el programa *Scilab* generar una comunicación serial entre Scilab y el puerto serial de Arduino. Aqui se procesan los datos del sensor en tiempo real, y mediante una interface gráfica se muestra en pantalla la posición actual de la veleta.
4. Hacer un experimento prototipo usando un potenciómetro. El segundo programa tomará los datos cada 15min durante dos días. Estos se almacenarán en una tarjeta micro SD.
5. Armar una caja de protección portátil para el circuito de medición. Y para alimentar el circuito se usa una fuente, baterias de litio 7.8V y 7800mA.

6. Instalar el sensor en una zona al aire libre para realizar las medidas durante dos días. Verificar que el soporte del sensor quede bien fijo; usar una brújula para que la mueca de la veleta coincida con el norte verdadero.
7. Para la experiencia de calibración de la posición con el sensor de viento belfort 1074 (ver sección 4.4) se mide el valor de la resistencia del potenciómetro de multivuelta, ubicado en el interior de la cabina del sensor.
8. Se conecta a la placa Arduino el HCM5883L y el sensor de acuerdo al esquema de conexión de la sección 6.2.
9. ubicar el magnetómetro y la placa Arduino en un lugar fijo dentro de la caja de protección del sensor de viento. El magnetómetro debe quedar fijo sobre una superficie paralelo al plano del horizonte. Es importante este paso ya que la lectura del azimut magnético tomada por el HCM5883l determina la corrección de la posición de la veleta respecto del norte geográfico (ver sección 6.2).
10. Subir el programa de Arduino que automatiza el sistema del sensor de viento. Este programa permite medir la posición de la veleta respecto del norte geográfico, y además corrige la posición del sensor de viento debido a alguna perturbación en su posición (ver sección 9.3).

SECCIÓN 8

## Desarrollo experimental

### 8.1 Primera experiencia: Determinación de resistores del sensor de viento

Lo que se hizo en primer lugar, fue observar y analizar la conexión de salida de la veleta (conector RJ-11) cuatro pines.



Figure 27

Los pines que van hacia la veleta son los externos, los dos del medio son para la conexión del anemómetro. De acuerdo a esto se construyó un adaptador RJ-11 (hembra) tal como se muestra en la figura.

Luego, se conectó estos cables al multímetro y se midió las resistencias (en escala de los  $k\Omega$ ) del sensor posicionando la veleta en distintas direcciones.

Se obtuvieron los siguientes resultados:

Table 1: Valores de resistencias

| Punto cardinal | Dirección | $R(k\Omega)$ |
|----------------|-----------|--------------|
| NORTE          | 0         | 32.8         |
| NOROESTE       | 45        | 8.1          |
| ESTE           | 90        | 0.9          |
| SUDESTE        | 135       | 2.1          |
| SUR            | 180       | 3.8          |
| SUDOESTE       | 225       | 15.8         |
| OESTE          | 270       | 199.6        |
| NOROESTE       | 315       | 66.5         |

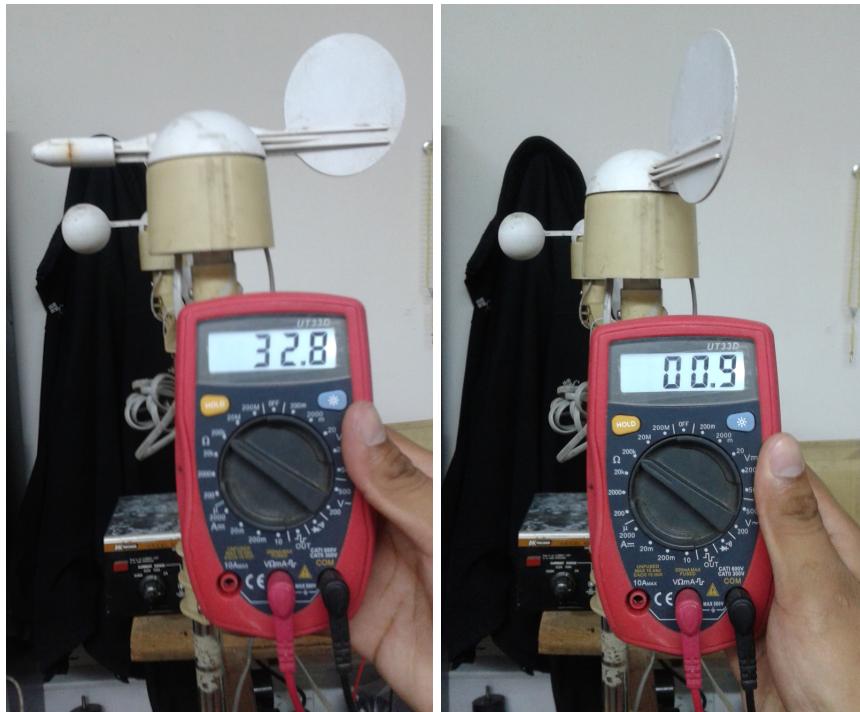


Figure 28: Medición de las ocho resistencias del sensor.

Se debe tener en cuenta que, los valores de las resistencias son para ocho posiciones de la veleta.

## 8.2 Segunda experiencia: Determinación de tensiones de salida del sensor de viento

Se armó un divisor de voltaje con una resistencia de carga de  $1k\Omega$  y una tensión de entrada de 5V proporcionada por la placa Arduino.

Se midió la tensión de salida del sensor de viento con un multímetro y se obtuvieron los siguientes resultados:

Table 2: Tensiones de salida

| Punto cardinal | R( $k\Omega$ ) | Tensión (V) |
|----------------|----------------|-------------|
| NORTE          | 32.8           | 3.84        |
| NOROESTE       | 8.1            | 2.27        |
| ESTE           | 0.9            | 0.45        |
| SUDESTE        | 2.1            | 0.9         |
| SUR            | 3.8            | 1.4         |
| SUDOESTE       | 15.8           | 3.08        |
| OESTE          | 199.6          | 4.64        |
| NOROESTE       | 66.5           | 4.35        |

## 8.3 Tercera experiencia: Ensamble e instalación de la estación meteorológica

Se instaló la estación meteorológica en un soporte ubicado en una zona libre de la terraza de la facultad de Ciencias Exactas (UNSa). Se corrige la inclinación de la estación mediante un "nivel de mano". Con un brújula comparamos la dirección del norte con la mueca "N" de la veleta.

Se armó el sistema electrónico descrito como en la figura 22. Se conecta las dos baterías de Lítio al Arduino a través de la entrada Jack. El protoboard se usa para conectar la resistencia de carga de  $10k\Omega$  y otros conectores (ver figura 30).

También se implanta un adaptador de ficha RJ-11, este se conecta al sensor de viento. De



Figure 29: Nivelación de la estación meteorológica

este adaptador salen dos cables (tierra y positivo) que mandaran las señales de la posición de la veleta al Arduino.

Se conecta el módulo de memoria micro SD al Arduino mediante los pines ICSP para realizar la comunicación SPI.

Y luego, se construye una caja protectora para cubrir el circuito de sol, lluvia, etc.

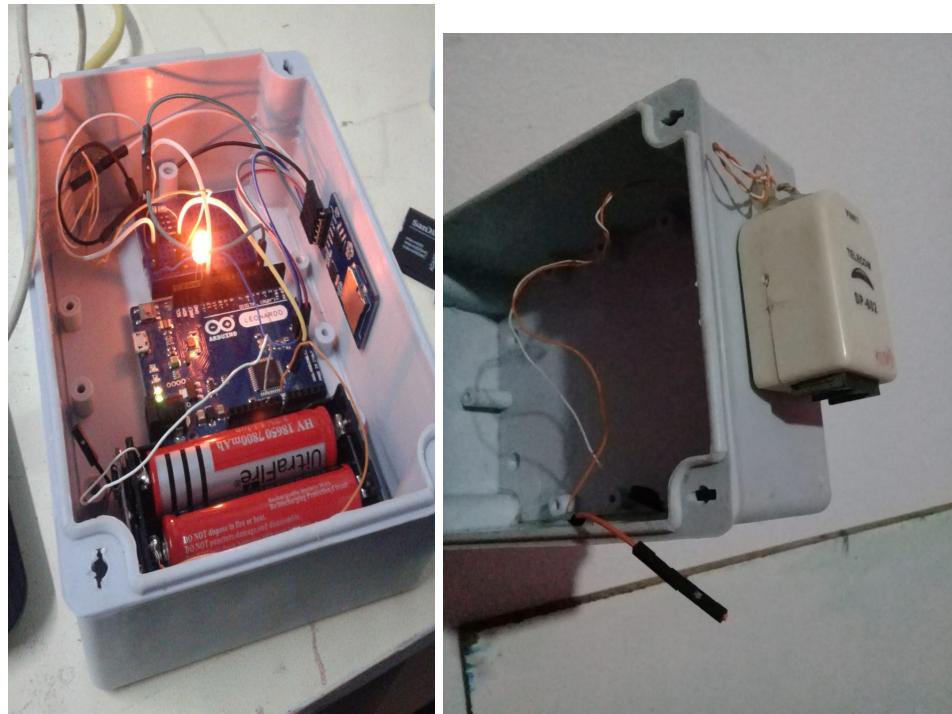


Figure 30: Caja de protección con el circuito ensamblado y adaptador de ficha RJ-11

Una vez realizada la experiencia, se obtuvieron 37 medidas en un periodo de 10 hs, estos

datos se registraron en una memoria micro SD. Se analizaron dichos datos en una planilla Libre Office Calc, resultando así el siguiente histograma:

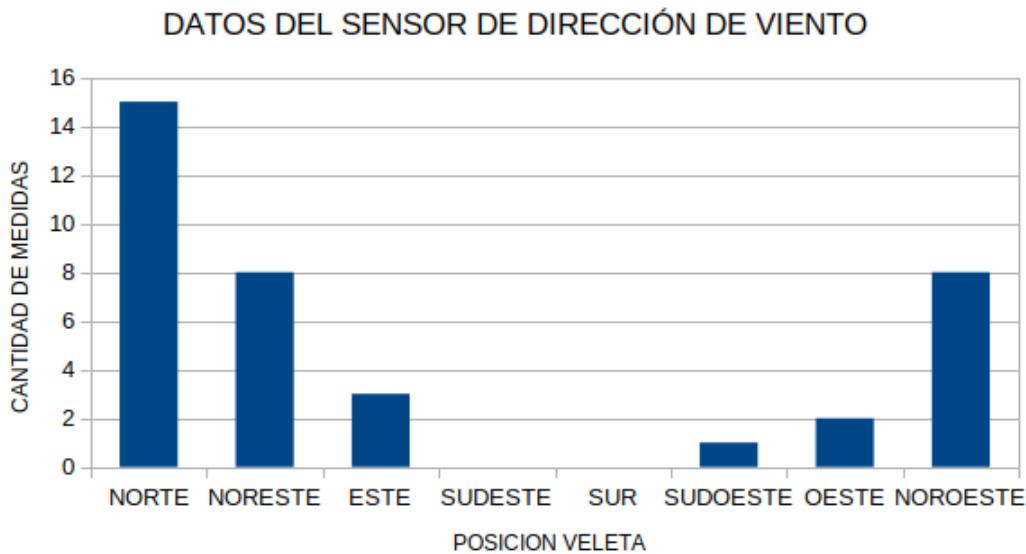


Figure 31: Histograma: posición de la veleta vs cantidad de veces que se obtuvo esa posición en un periodo de 10 hs

#### 8.4 Cuarta experiencia: Calibración del sensor de viento Belfort 1074 usando un potenciómetro

En esta parte se hace el ensamble todos los componentes electrónicos de acuerdo al esquema 26 para realizar la corrección de la posición del "norte" de la veleta con el norte geográfico [15].

SECCIÓN 9

#### Programación. Adquisición de datos

Uno de los objetivos de este proyecto es establecer un sistema automatizado que adquiera las mediciones del sensor de viento. En esta sección se crea todo lo referente al software que se necesita para la lectura de las mediciones del sensor. Para realizar dicho sistema se hace uso de los software Arduino y Scilab.

Se crea dos programas. El primero toma las medidas por el sensor y lo almacena en una memoria micro SD cada 15 min. El segundo muestra la posición de la veleta en tiempo



Figure 32

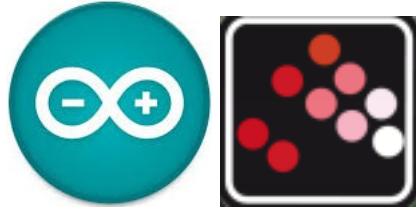


Figure 33

real mediante una interface gráfica generada mediante el software libre Scilab.

### 9.1 Programa 1: Almacenamiento de datos con Módulo micro SD

Este programa toma las señales del sensor de dirección de viento. La magnitudes a leer son una serie de voltajes directos (ochos señales analogicas) Estas serán tomadas mediante un pin analógico de arduino que convierte la señal analogica (0v-5v) en digital(0-1023)

El software Arduino opera con un lenguaje de programación idéntico a C. Mostraremos como funciona mediante unas líneas de flujos:

```
1 //Leemos la señal analógica transformada a digital.  
2 vol_dig=analogRead(pin_lec);  
  
3 /*Tenemos la lista de las ocho señales analógicas del sensor  
4 en voltios (medidas con un multímetro digital).
```

```

5 Estas son: VAMOS A TRABAJAR CON LAS MEDIDAS DEL DIA 15/05
6 3.84 - 2.27 - 0.45 - 0.9 - 1.40 - 3.08 - 4.62 - 4.35 [V]
7 norte-nordeste-este-sureste-sur-suroeste- oeste-noroeste*/
8 /*Las senales digitales correspondientes tambien se las
9 pueden calcular como:
10 vol_dig=v_real*1023/5
11 donde el "v_real" es la tension medida con el multmetro*/
12 /*Como las senales son discretas y ademas estan un poco
13 distantes unas con otras, puedo poner como condicion que
14 el valor digital de la senal este en un rango de valores , de
15 esta manera puedo evitar errores en la lectura cuando haya
16 fluctuaciones en la senal. Asigno a cada una de las ocho
17 senales posibles un numero: "posicion=num"*/
18     if (vol_dig>=775 & vol_dig<=795)
19     {
20 //El valor de la senal digital es 785.664 "NORTE"
21 posicion=1;
22 }
23     if (vol_dig>=459 & vol_dig<=469)
24     {
25 //El valor de la senal digital es 464.442 "NORESTE"
26     posicion=2;
27 }
28     if (vol_dig>=87 & vol_dig<=97)
29     {
30 //El valor de la senal digital es 92.07 "ESTE"
31     posicion=3;
32 }
33     if (vol_dig>=179 & vol_dig<=189)
34     {
35 //El valor de la senal digital es 184.14 "SUDESTE"
36     posicion=4;

```

```

37 }
38     if (vol_dig>=281 & vol_dig<=291)
39     {
40 //El valor de la senal digital es 286.44 "SUR"
41 posicion=5;
42 }
43     if (vol_dig>=627 & vol_dig<=637)
44     {
45 //El valor de la senal digital es 632.214 "SUROESTE"
46 posicion=6;
47 }
48     if (vol_dig>=940 & vol_dig<=950)
49     {
50 //El valor de la senal digital es 945.252 "OESTE"
51 posicion=7;
52 }
53     if (vol_dig>=880 & vol_dig<=900)
54     {
55 //El valor de la senal digital es 890.1 "NOROESTE"
56 posicion=8;
57 }

```

```

58 /*Luego, guardo en la memoria micro SD el valor de la
59 posicion, en un archivo bajo el nombre "dir.txt".
60 Este archivo tendra dos columnas: una sera el tiempo
61 (minutos) y otra la posicion de la veleta en dicho tiempo*/

```

```

62 mi_archivo=SD.open("dir.txt",FILE_WRITE);
63 if (mi_archivo)
64 {
65     digitalWrite(led1,HIGH);
66 mi_archivo.print(tiempo);
67 mi_archivo.print("	");
68 mi_archivo.println(posicion);
69 mi_archivo.close();

```

```

70 }

71 /*Haremos que el programa se detenga 15 minutos y luego
72 mida otra señal pasado los 15 minutos*/

73 //Quiero tomar medidas cada 15min.
74 for (i=0;i<=60*15; i++)
75 {
76   if (i==2)
77   {
78     digitalWrite(led1,LOW);
79   }
80   delay(1000);
81 }
82 tiempo=tiempo+15; //Aumento 15 min.

83 /*BASICAMENTE ESTO ES TODO EL PROGRAMA 1*/

```

## 9.2 Programa 2: Vision de posición de veleta en tiempo real

Lo que hace este programar es mostrar en pantalla de un computador (y posiblmente en el celular, otro proyecto), la posición de la veleta en tiempo real. Esto es posible realizar mediante una comunicación serial entre el puerto serie de Arduino y el programa Scilab.

El programa en Arduino es el mismo que el presentado en la sección anterior. La única diferencia es que en vez de leer las señales del sensor cada 15 min, será cada 1 seg.

```

1 //Tomo los datos cada 1 seg.
2   delay(1000);

```

En esta parte se analizará como la estructura del programa Scilab.

```

3 //VAMOS A ESTABLECER LA COMUNICACION SERIAL .
4   errcatch(-1,"continue")
5   comun=openserial('/dev/ttyACM0','9600,n,8,1');
6 //SI OCURRE UN ERROR AL LEER EL PUERTO ENVIAR MENSAJE .
7   err_comun=iserror()

```

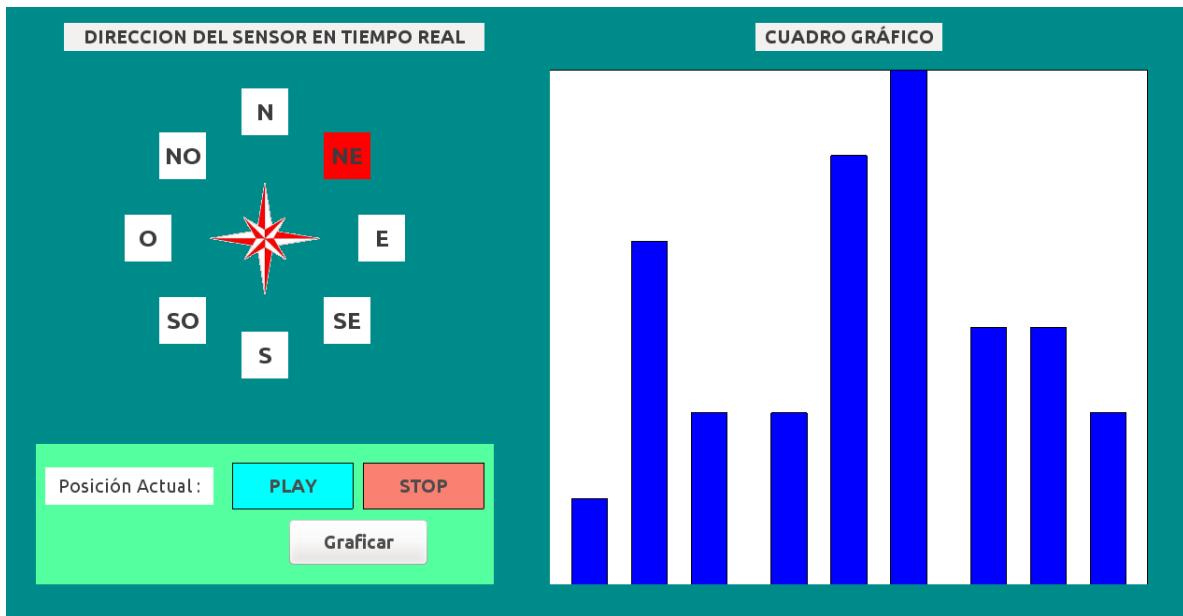


Figure 34: Interface gráfica para mostrar la posición de la veleta

```

8     if (err_comun==0) then
9         errclear()
10 //Mandamos la orden "M" para que ARDUINO comience a medir.
11     orden='M' //Medir
12     writeserial(comun , 'M')

13 /*Leemos el valor tomado por la placa Arduino y la
14 guardamos en la variable "lect"*/
15     while (estado_leer=="off")
16         lect=readserial(comun)
17 /*convierzo la lectura en valor numerico. Y esta representara
18 la posicion de la veleta "pos"*/
19         pos=strtod(lect)

```

Hasta aqui esta claro, ahora en otro scrip se realiza la interface gráfica mediante los comandos GUI para mostrar la posición de la veleta. Se crean ocho casillas cada cual corresponde a una dirección de los ocho puntos cardinales.

Entonces de acuerdo al valor de ”pos”, el programa lo que hará es cambiar el color (blanco a rojo) de la casilla correspondiente a la posición actual del sensor de viento

```

20 select pos
21     case 1

```

```

22      set(norte,"backgroundcolor",name2rgb("red")/255)
23      sleep(500) //suspende scilab por 0.5seg
24      set(norte,"backgroundcolor",pro1)
25  case 2
26      set(noreste,"backgroundcolor",name2rgb("red")/255)
27      sleep(500) //suspende scilab por 0.5seg
28      set(noreste,"backgroundcolor",pro2)
29  case 3
30      set(este,"backgroundcolor",name2rgb("red")/255)
31      sleep(500) //suspende scilab por 0.5seg
32      set(este,"backgroundcolor",pro3)
33  case 4
34      set(sudeste,"backgroundcolor",name2rgb("red")/255)
35      sleep(500) //suspende scilab por 0.5seg
36      set(sudeste,"backgroundcolor",pro4)
37  case 5
38      set(sur,"backgroundcolor",name2rgb("red")/255)
39      sleep(500) //suspende scilab por 0.5seg
40      set(sur,"backgroundcolor",pro5)
41  case 6
42      set(sudoeste,"backgroundcolor",name2rgb("red")/255)
43      sleep(500) //suspende scilab por 0.5seg
44      set(sudoeste,"backgroundcolor",pro6)
45  case 7
46      set(oeste,"backgroundcolor",name2rgb("red")/255)
47      sleep(500) //suspende scilab por 0.5seg
48      set(oeste,"backgroundcolor",pro7)
49  case 8
50      set(noroeste,"backgroundcolor",name2rgb("red")/255)
51      sleep(500) //suspende scilab por 0.5seg
52      set(noroeste,"backgroundcolor",pro8)
53 end
54 estado_leer=get(leer,"enable")
55 end

```

Finalmente se muestra la posición de la veleta en tiempo real, a través de una interface gráfica generada por Scilab.

Otro proyecto que se puede hacer más adelante es, obtener los datos del sensor y analizarlos en la pc o celular mediante una comunicación vía Bluetooth. Esto se puede realizar implementando un módulo Bluetooth (maestro - esclavo) que trabaja con Arduino.

### 9.3 Programa 3: Calibración de la posición del sensor de viento

En la sección 5.4 vimos todas las características y funcionamiento del sensor HCM5883L (magnetómetro de tres ejes). También se mostró todos los detalles necesarios del sensor de viento Benfort (sección 4.4).

Recordemos que unos de los objetivos de este proyecto es calibrar la posición del sensor de viento con el verdadero norte ([Norte geográfico \[15\]](#)). Para ello, debemos conocer dos parámetros:

1. Dirección de la veleta, 0-360°.
2. Azimut magnético, 0-360° [4](#).

Ambos son obtenidos mediante el hardware Arduino.

Entonces vamos a analizar algunas líneas de código para entender el programa:

Primero incluimos las librerías necesarias.

```
1 #include <Wire.h> /*incluimos esta libreria necesaria para la
2 comunicacion I2C.*/
3 #include <MechaQMC5883.h> //Libreria del magnetometro.
```

Ahora definimos las variables que vamos a usar. Notar que se trabaja con variables de tipo enteras, es decir los valores de los ángulos, declinación, etc serán enteros. Se puede trabajar variables de tipo flotante, pero aun no se ha aprobado.

```
4 int valor_sensor; /*variable que guardara el voltaje leido
5 por Arduino proveniente del sensor de viento.*/
6 int dir; //direccion de veleta. Angulo 0-360.
```

```

7 int decli=-8; /*declinacion magnetica. El
8 valor en nuestra ubicacion
9 es de es -8 2', se lo aproxima a 8 *|.
10 int azimuth; /*angulo respecto del norte magnetico dado
11 por el HCM5883L.*/
12 int geo; //angulo respecto del norte geografico.
13 int x, y, z; /*declaro las variables de tipo entero.
14 En estas variales se guardaran los valores de las componentes
15 del campo magnetico terrestre medido por nuestro sensor*/
16 int corr; //Posicion corregida del sensor de viento.

```

Luego Arduino lee la señal analógica proveniente del sensor de viento o veleta, y lo guarda en la variable *valor sensor* que será un valor digital de 0 a 1023.

```

17 valor_sensor=analogRead(A3);

```

Usamos la función *map* (ver glosario 14) para crear un rango de 360 valores a partir de otro rango de 0 a 1023 que son los valores de la señales analógica, lo que devuelve esta función es la posición de la veleta en grados. Por ahora el [norte geográfico \[15\]](#). lo haremos coincidir con el 0V y por lo tanto, con 0°.

```

18 dir=map(valor_sensor ,1 ,1023 ,0 ,290);
19 /*dir: es la dirección o posicion de la veleta respecto
20 del "norte fijo" en el sensor. Este norte puede o no
21 coincidir con el norte geográfico (verdadero norte),
22 que es justamente lo que tenemos que calibrar.*/

```

Luego, obtenemos los valores del azimut magnético medidos por el HCM5883L.

```

23 hcm.read(&x, &y, &z,&azimuth); /*Con esta funcion se
24 hace dos procesos en simultaneo
25 1- Leer los valores x,y,z del campo magnetico.
26 2- Calcular el azimut magnetico
27 y asignarlos a las variables que recien creamos.
28 El simbolo & significa que estaremos pasando los
29 valores de las variables por referencia.*/
30

```

```
31  /*azimuth: angulo respecto del norte magnetico ,  
32 en sentido horario.*/
```

Luego, sumo al [azimuth magnético \[4\]](#). el valor de la [declinación magnética \[17\]](#). (de donde estamos  $8^\circ$ ). Y de esta forma obtenemos el valor de *geo* que es el ángulo de la brújula (HCM5883l) respecto del [norte geográfico \[15\]](#). (norte verdadero). Este va de  $0^\circ$  a  $360^\circ$  en sentido horario.

```
33 geo=azimuth+decli;
```

Puede pasar que el magnetómetro apunte exactamente al norte geográfico, y como la declinación es negativa el ángulo resultante será negativo. Para ello pongo una condición: si el valor de *geo* es negativo le sumo 360. Es decir, la aguja esta del lado izquierdo del norte geográfico.

```
34 if (geo<0){  
35     geo=geo+360;  
36 }
```

Finalmente vamos a corregir el la dirección del norte de la veleta. Supongamos que la posición de la veleta esta en  $0^\circ$ , es decir *dir*= $0^\circ$ . Pero la posición del cuerpo del sensor de viento (suponemos que ya esta calibrado) se mueve  $10^\circ$  sentido horario, es decir em *geo*= $10^\circ$  (ver sección [6.2](#)). De esta manera el valor de *dir*= $0^\circ$  ya no es correcto, ya que no coincide con el norte geográfico. Como conozco los valores de *geo* y *dir*, los sumo y obtengo que la posición de la veleta es de  $10^\circ$ , y lo guardo en la variable *corr* (ángulo corregido).

```
37 corr=geo+dir; //Este es el angulo de la veleta corregido. Y as la p  
38 del sensor est calibrada.
```

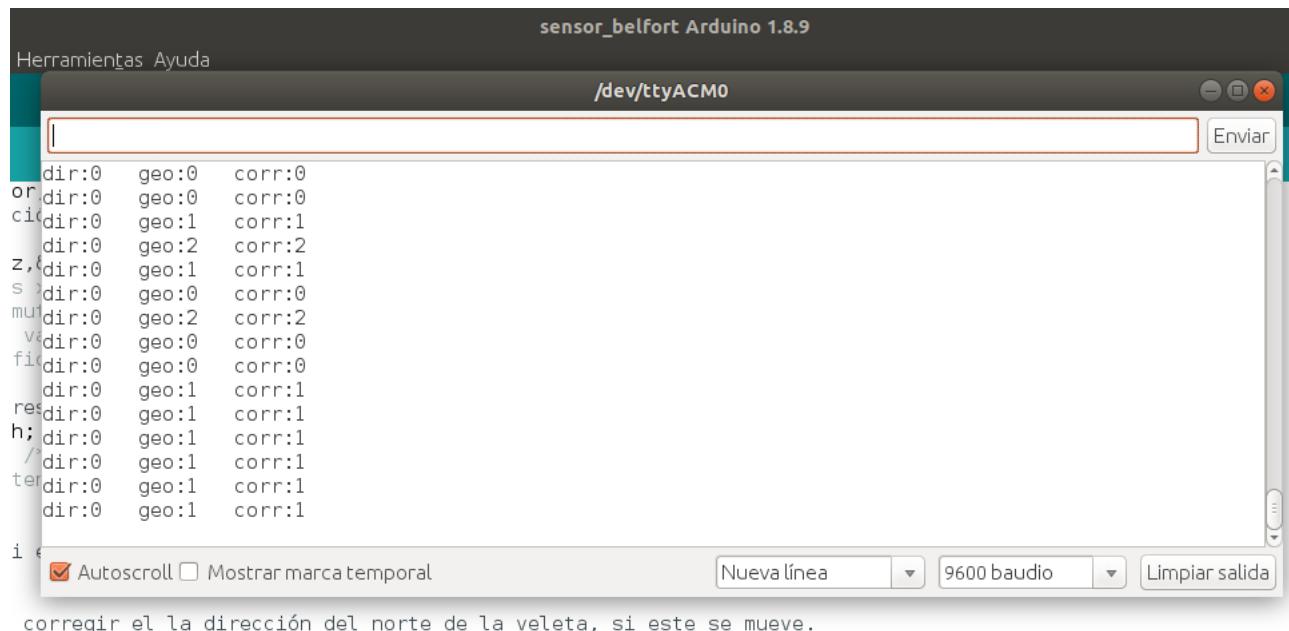
Para el caso en que, por ejemplo, *geo*= $10^\circ$  y *dir*= $355^\circ$  tome un valor alto, realizo la siguiente condición.

```
39 if (corr>360){  
40     corr=geo+dir-360;  
41 }
```

Y este proceso de calibración se actualiza cada segundo.

```
42 delay(1000);
```

Vamos a verificar que el programa funciona mostrando por el monitor serial los valores de **dir** (dirección de la veleta respecto del "norte" marcado en el sensor), **geo** (posición de HCM, y por lo tanto del cuerpo del sensor de viento, respecto del norte geográfico), y **corr** (ángulo corregido o verdadero ángulo respecto del norte geográfico)



The screenshot shows the Arduino Serial Monitor window titled "sensor\_belfort Arduino 1.8.9". The port is set to "/dev/ttyACM0". The data window displays a series of lines of text representing sensor data. Each line consists of three values separated by spaces: "dir:0 geo:0 corr:0". This pattern repeats 15 times. Below the data window, there is a status bar with several buttons and settings. One of the buttons is checked, labeled "Autoscroll". To the right of the status bar, there are buttons for "Nueva línea", "9600 baudio", and "Limpiar salida". At the bottom of the window, there is a message in Spanish: "corregir el la dirección del norte de la veleta, si este se mueve."

Figure 35

Podemos ver en la figura 35 **geo=0°** coincide aproximadamente con el [norte geográfico \[15\]](#), **dir=0°** esta en la posición del "norte" marcado en el sensor, y vemos que **corr=0°** coincide con el valor de **geo**, este quiere decir que la veleta esta apuntando en la dirección del verdadero.

Vemos en la figura 36 que el valor de **geo≈ 22°** mientras **dir** sigue en 0°, esto quiere decir que el magnetómetro y por lo tanto el cuerpo del sensor de viento se movió 20° respecto del norte geográfico. Por lo tanto, aunque la posición de la veleta respecto del "norte" marcado dicho sensor no cambie (**dir=0°**), el ángulo correcto de la posición de la veleta respecto del norte geográfico es **corr=22°**.

En la figura 37 se muestra que **geo=332°**, es decir que la aguja del magnetómetro y por lo tanto el cuerpo del sensor de viento apunta a la izquierda del norte geográfico. La veleta esta a 30° respecto de su norte "marcado". Sin embargo, la posición dela veleta apunta aproximadamente hacia el norte geográfico, por eso **corr≈0°**.

```

dir:0 geo:22 corr:22
or dir:0 geo:24 corr:24
ci dir:0 geo:22 corr:22
    dir:0 geo:22 corr:22
z, dir:0 geo:22 corr:22
s > dir:0 geo:21 corr:21
mu dir:0 geo:21 corr:21
v dir:0 geo:21 corr:21
fi dir:0 geo:23 corr:23
    dir:0 geo:20 corr:20
res dir:0 geo:23 corr:23
; dir:0 geo:21 corr:21
/ dir:0 geo:22 corr:22
ter dir:0 geo:21 corr:21
    dir:0 geo:21 corr:21

```

i  Autoscroll  Mostrar marca temporal Nueva línea 9600 baudio Limpiar salida

corregir el la dirección del norte de la veleta, si este se mueve.

Figure 36

```

dir:30 geo:333 corr:3
or dir:30 geo:331 corr:1
ci dir:30 geo:332 corr:2
    dir:30 geo:332 corr:2
z, dir:30 geo:333 corr:3
s > dir:30 geo:331 corr:1
mu dir:30 geo:332 corr:2
v dir:30 geo:331 corr:1
fi dir:30 geo:331 corr:1
    dir:30 geo:331 corr:1
res dir:30 geo:332 corr:2
; dir:30 geo:333 corr:3
/ dir:30 geo:332 corr:2
ter dir:30 geo:331 corr:1
    dir:30 geo:332 corr:2

```

i  Autoscroll  Mostrar marca temporal Nueva línea 9600 baudio Limpiar salida

corregir el la dirección del norte de la veleta, si este se mueve.

Figure 37

SECCIÓN 10

## Errores de medición

En la experiencia de calibración de la posición del sensor de viento Belfort 1074 (ver sección 4.4) se usó un magnetómetro digital HCM5883L (ver sección 5.4).

Hay que tener en cuenta que para obtener valores correctos con el HCM5883L hay que

estar alejados de fuentes de campo magnéticos. la presencia de materiales ferromagnéticos (por ejemplo, Hierro, aceros, Cobalto, Níquel) pueden causar perturbaciones en las lecturas del sensor del campo magnético terrestre.

Por otra parte, en la experiencia cuando Arduino lee el valor del [azimuth magnético \[4\]](#) se observa pequeñas fluctuaciones en dicho ángulo (ver figura 38). Podemos ver que el ángulo **dir** permanece todo el tiempo  $0^\circ$ , no hay perturbación en la medida, pero el ángulo **geo** varía de  $21^\circ$  a  $24^\circ$ , y por lo tanto el ángulo correcto **corr** también va a variar. Por lo tanto, establecemos un error en la medición del ángulo de la veleta respecto del norte geográfico, este será:

$$\Delta corr = \pm 2^\circ \quad (10.1)$$

```

/dev/ttyACM0

dir:0 geo:22 corr:22
rdir:0 geo:24 corr:24
idir:0 geo:22 corr:22
dir:0 geo:22 corr:22
,dir:0 geo:22 corr:22
;dir:0 geo:21 corr:21
udir:0 geo:21 corr:21
vdir:0 geo:21 corr:21
idir:0 geo:23 corr:23
dir:0 geo:20 corr:20
esdir:0 geo:23 corr:23
;dir:0 geo:21 corr:21
/dir:0 geo:22 corr:22
erdir:0 geo:21 corr:21
dir:0 geo:21 corr:21

```

Autoscroll  Mostrar marca temporal      Nueva línea ▾ 9600 baudio ▾ Limp

corregir el la dirección del norte de la veleta, si este se mueve.

Figure 38

## SECCIÓN 11

### Conclusiones

1. La parte experimental del proyecto se hizo en un lapso de tres semanas, tres días por semana. Si bien la experiencia era sencilla, la parte de automatización de la estación meteorológica fué todo un desafío.
2. Se logró cumplir con los con el objetivo del trabajo, de obtener medidas del sensor guardando los datos en una memoria micro SD, y también establecer una interface

gráfica que muestre la posición de la veleta en tiempo real.

3. En la experiencia que consistía en tomar los datos del sensor al aire libre y almacenarlos en una memoria, hubo inconvenientes. La idea era dejar al sensor con el Arduino tomando datos durante dos días. Se hizo una evaluación del consumo de energía de la fuente de baterías. Estas proporcionan 7800mA los cuales la placa Arduino consume 50mA y el módulo de micro SD consume aprox 50mA. El calculo nos dice que las baterías de Litio pueden proporcionar energía, para este consumo, durante 6 días.



Figure 39

Pero en la experiencia las baterías duraron sólo 10 hs. Esto se debe, a que no se conectó correctamente la fuente de alimentación a la placa Arduino. Se conectó las terminales de las baterías en el pin VIN. Este pin soporta un amperaje de 1000mA. Y la batería que se uso proporciona 7800mA. En realidad, la placa se debía haber dañado. La conexión correcta de la fuente de alimentación al Arduino, para esta capacidad de voltaje, debió hacerse con la entrada Jack, que posee un regulador de tensión que soporta un amperaje mayor.

4. Por otro lado, la experiencia realizada con el sensor de viento Belfort 1074 fué satisfactorio. Este fué realizado en un periodo de una semana, contando la creación del programa.

Se obtuvo los resultados de la corrección de la posición de la veleta tal como se lo

planeó .

5. La prueba anterior fué realizada con un sensor de viento con un potenciómetro continuo, este se puede adaptar a otros tipos de sensores, por ejemplo uno de ocho posiciones.
6. Si bien, el objetivo fué realizar medidas con un único sensor, se puede ampliar la cantidad de sensores para tomar medidas de otras variables atmosféricas. Haría falta hacer un análisis más profundo de como funcionan los demás sensores de la estación. Pero sería un desafío alcanzable y efectivo.
7. Viendo la versatilidad de la trabajo hecho. Se puede proponer como un futura experiencia una toma de datos del sensor mediante una comunicación Bluetooth.
8. Este trabajo me sirvió para aumentar mis conocimientos de electrónica y programación. Y también para aumentar mi perspectiva del trabajo experimental de la ciencia en el mundo real.

## Glosario

**Definición 1 (Gauss)** Es una unidad del campo magnético del Sistema Cégesimal de Unidades (SCG). 1 gauss equivale a  $10^{-4}$  Teslas. El campo magnético terrestre es del orden de los 0.5 gauss, para tener una idea, un imán pequeño posee 100 gauss.

**Definición 2 (Magnetómetro)** Se les dice magnetómetros a los dispositivos que sirven para cuantificar en fuerza o dirección la señal magnética de una muestra.

**Definición 3 (I2C)** Significa circuito integrado pos sus siglas en inglés (Inter Integrated Circuit) es un protocolo de comunicación serial sincrónico parecido el protocolo SPI. Con el I2C se pueden tener varios maestros controlando uno o múltiples esclavos. Este utiliza dos vías de comunicación, SDA (Serial Data) vía de comunicación entre maestro y esclavo para enviar información, y SCL (Serial Clock) vía por donde viaja la señal del reloj.

**Definición 4 (Azimut Magnético)** Es el ángulo medido desde el norte magnético en sentido de las agujas del reloj (horario), que va desde  $0^\circ$  a  $360^\circ$ . Veamos el siguiente ejemplo, si la aguja coincide exactamente con el norte magnético entonces el azimut magnético será de cero grados.

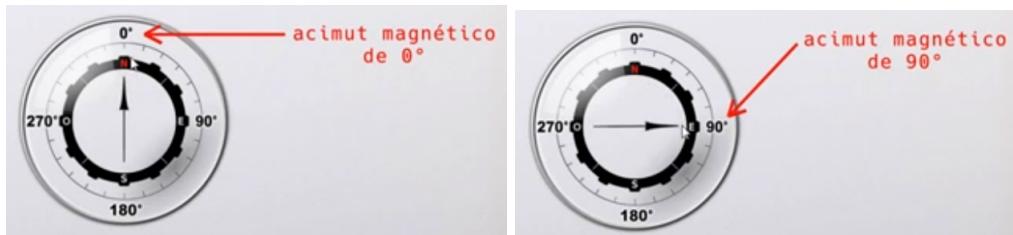


Figure 40: Azimut magnético

Para el caso del magnetómetro HCM5883L la aguja estará representada por la flecha x impresa en dicho sensor. Y el azimut magnético será el ángulo medido desde el norte magnético en dirección de las agujas del reloj.

**Definición 5 (Inclinación)** Es un ángulo formado por la línea de fuerza magnética terrestre de la zona y el plano del horizonte.

**Definición 6 (Latex)** Es un sofisticado programa para la composición tipográfica de texto científicos y es la mejor opción disponible para la edición de texto con contenido matemático tales como artículo, reportes, libros, etc. LATEX no es un procesador de texto, es un lenguaje que nos permite preparar automáticamente un documento de apariencia estándar y de alta calidad.

# LATEX

Figure 41

**Definición 7 (Rango)** Es el intervalo entre los límites de medición inferior y superior de un instrumento, por ejemplo, un termómetro con un rango de -35° a 50 °C (comparar con el intervalo)

**Definición 8 (Precisión)** Grado con el que un instrumento mide una variable en términos de un valor estándar aceptado o un valor verdadero; generalmente medido en términos de inexactitud pero expresado como precisión; a menudo expresado como un porcentaje del rango de escala completa.

**Definición 9 (Resolución)** Es el cambio más pequeño en el parámetro que se está midiendo que causa un cambio detectable en la salida del instrumento.

**Definición 10 (Linealidad)** Desviación máxima de la curva de respuesta real de un instrumento respecto de la línea de recta que mejor ajusta los datos. Se aplica sólo a instrumento con una respuesta más o menos lineal, y generalmente se indica como un porcentaje del rango de escala completa.

**Definición 11 (Umbral)** Valor de entrada más pequeño para un sensor que hará que el sensor responda. Comúnmente utilizados con sensores mecánicos de viento para describir la velocidad de viento necesaria para hacer que el anemómetro y la veleta giren.

**Definición 12 (Distancia constante)** Paso de aire necesario sobre un sensor de velocidad de viento para hacer que el sensor responda al 63% de un cambio de función escalonada en la velocidad del viento.

**Definición 13 (Relación de amortiguación)** Parámetro utilizado para describir la respuesta de la veleta a un cambio repentino en la dirección de viento. Se define como la relación entre la amortiguación real y la amortiguación crítica, donde la amortiguación crítica es el valor de amortiguación que proporciona la respuesta transitoria más rápida sin sobreimpulso.

**Definición 14 (Map)** Vuelve a mapear un número de un rango a otro. Es decir, un valor de `fromLow` se asignaría a `toLow`, un valor de `fromHigh` a `toHigh`, valores intermedios a valores intermedios, etc.

`map(value, fromLow, fromHigh, toLow, toHigh)`

Tenga en cuenta que los "límites inferiores" de cualquiera de los rangos pueden ser mayores o menores que los "límites superiores", por lo que la función `map()` puede usarse para invertir un rango de números, por ejemplo

`y = map(x, 1, 50, 50, 1);`

La función también maneja bien los números negativos, por lo que este ejemplo

`y = map(x, 1, 50, 50, -100);`

También es válido y funciona bien.

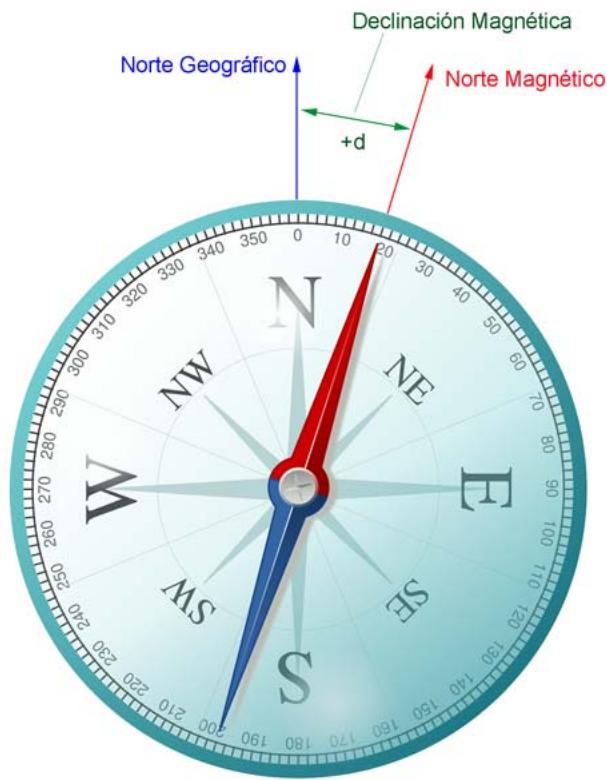
La función `map()` usa matemática entera, por lo que no generará fracciones, cuando la matemática podría indicar que debería hacerlo. Los restos fraccionarios se truncan y no se redondean ni promedian.

**Definición 15 (Norte Geográfico)** También conocido como norte verdadero, coincide con el eje de rotación de la tierra y define en dos puntos su intersección con la superficie de la tierra; el norte geográfico en el polo norte y el sur geográfico en el polo sur.

**Definición 16 (Norte magnético)** El norte magnético está definido por el campo magnético de la tierra y es a donde apuntan las brújulas y compases náuticos. Su posición no coincide exactamente con el norte geográfico, y se mueve de año en año tanto que durante el siglo XX su posición varió 1100 kilómetros de distancia. Actualmente el norte magnético varía su posición unos 60 kilómetros por año.

**Definición 17 (Declinación magnética)** La declinación magnética es el ángulo en el plano horizontal que forman el norte magnético y el norte geográfico o verdadero.

La declinación magnética además de variar con los años, varía según el lugar donde nos encontremos; por ello en las cartas náuticas viene dado su valor y su incremento o decrecimiento anual.



Si el norte magnético está a la derecha del norte geográfico, la declinación magnética es positiva, y si el norte magnético está a la izquierda del norte geográfico la declinación magnética es negativa [[20]].

**Definición 18 (AD)** Se trata de un convertidor de señal analógica a digital. Es un dispositivo electrónico capaz de convertir una señal analógica, ya sea de tensión o corriente, en una señal digital mediante un cuantificador y codificándose en muchos casos en un código binario en particular. Un código es la representación unívoca de los elementos, en este caso, cada valor numérico binario hace corresponder a un sólo valor de tensión o corriente.

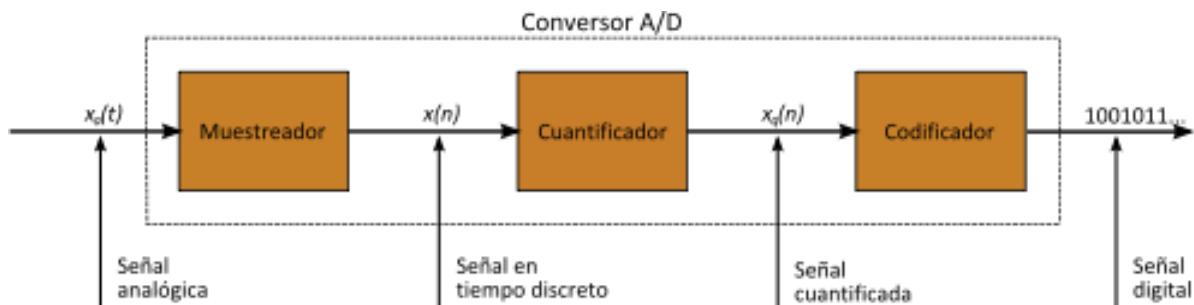


Figure 42

**Definición 19 (Tinkercad)** Traducción del inglés-Tinkercad es un programa gratuito de modelado 3D en línea que se ejecuta en un navegador web, conocido por su simplicidad y facilidad de uso



Figure 43

**Definición 20 (Raspberry Pi)** Es una placa computadora (SBC) de bajo coste, se podría decir que es un ordenador de tamaño reducido, del orden de una tarjeta de crédito, desarrollado en el Reino Unido por la fundación Raspberry Pi (Universidad de Cambridge) en 2011, con el objetivo de estimular la enseñanza de la informática en las escuelas, aunque no comenzó su comercialización hasta el año 2012.

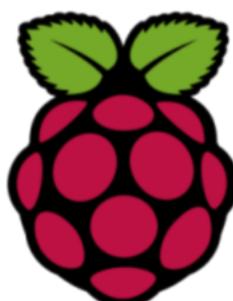


Figure 44

Está formada por una placa que soporta varios componentes necesarios en un ordenador común y es capaz de comportarse como tal.

## References

- [1] Diseño e implementación de una estación meteorológica con Raspberry Pi
- [2] Manual Estación meteorológica PCE-FWS20.pdf
- [3] [www.arduino.cc](http://www.arduino.cc) página oficial.
- [4] Geek Factory: Alimentar el Arduino la guía definitiva
- [5] Sensores y acondicionadores de señal. Ramón Pallas Areny.
- [6] SPI: Serial Peripheral Interface, Wikipedia
- [7] Latex 2017 Alexander Bobón. Latex descarga pdf
- [8] TINKRCAD Simulador web de circuitos con Arduino.
- [9] Arduino: video de magnetómetro ([youtube](https://www.youtube.com/watch?v=Q31mkKY1iM8)) [www.youtube.com/watch?v=Q31mkKY1iM8](https://www.youtube.com/watch?v=Q31mkKY1iM8)
- [10] Magnetómetro definición Wikipedia [es.wikipedia.org/wiki/Magnetómetro](https://es.wikipedia.org/wiki/Magnetómetro)
- [11] [naylampmechatronics.com/blog/49-tutorial-magnetometro-hmc5883l.html](http://naylampmechatronics.com/blog/49-tutorial-magnetometro-hmc5883l.html)
- [12] Descripción del HMC5883L Geek Factory
- [13] Protocolo de comunicación I2C [www.teslabem.com/nivel-intermedio/fundamentos](http://www.teslabem.com/nivel-intermedio/fundamentos)
- [14] Campo magnético terrestre y declinación: Instituto superior de navegación [www.isndf.com.ar/que-es-la-declinacion-magnetica](http://www.isndf.com.ar/que-es-la-declinacion-magnetica)
- [15] Declinación magnética wikipedia
- [16] [www.magnetic-declination.com](http://www.magnetic-declination.com)
- [17] [github.com/jrowberg/i2cdevlib/tree/master/Arduino/HCM5883L](https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/HCM5883L)
- [18] [github.com/jrowberg/i2cdevlib/tree/master/Arduino/I2Cdev](https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/I2Cdev)
- [19] [www.belfort-inst.com/Model-1074.htm](http://www.belfort-inst.com/Model-1074.htm)
- [20] [sailandtrip.com/declinacion-magnetica/](http://sailandtrip.com/declinacion-magnetica/)