UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERÍA Y ARQUITECTURA

ESCUELA DE INGENIERÍA ELÉCTRICA

Estación Meteorológica multiparamétrica sincronizada con GPS y monitoreada a través de Internet.

PRESENTADO POR:

VÍCTOR JAMIL PALMA MENJÍVAR

FRANCISCO RODRIGO RAMÍREZ

PARA OPTAR AL TITULO DE:

INGENIERO ELECTRICISTA

CIUDAD UNIVERSITARIA, SEPTIEMBRE 2013

UNIVERSIDAD DE EL SALVADOR

RECTOR :

ING. MARIO ROBERTO NIETO LOVO

SECRETARIA GENERAL :

DRA. ANA LETICIA ZAVALETA DE AMAYA

FACULTAD DE INGENIERÍA Y ARQUITECTURA

DECANO :

ING. FRANCISCO ANTONIO ALARCÓN SANDOVAL

SECRETARIO :

ING. JULIO ALBERTO PORTILLO

ESCUELA DE INGENIERÍA ELÉCTRICA

DIRECTOR :

ING. JOSÉ WILBER CALDERÓN URRUTIA

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERÍA Y ARQUITECTURA

ESCUELA DE INGENIERÍA ELÉCTRICA

Trabajo de Graduación previo a la opción al Grado de:

INGENIERO ELECTRICISTA

Título :

Estación Meteorológica multiparamétrica sincronizada con GPS y monitoreada a través de Internet.

Presentado por :

VÍCTOR JAMIL PALMA MENJÍVAR

FRANCISCO RODRIGO RAMÍREZ

Trabajo de Graduación Aprobado por:

Docente Director :

ING. CARLOS OSMÍN POCASANGRE JIMÉNEZ.

San Salvador, Septiembre 2013

Trabajo de Graduación Aprobado por:

Docente Director :

ING. CARLOS OSMÍN POCASANGRE JIMÉNEZ.

AGRADECIMIENTOS.

"Si te postran diez veces, te levantas
otras diez, otras cien, otras quinientas...
No han de ser tus caídas tan violentas
ni tampoco por ley, han de ser tantas."
Almafuerte, *Avanti.*

"Vosotros me decís: «la vida es difícil de llevar». Pero si no fuera así ¿Para qué tendríais vuestro orgullo por las mañanas y vuestra resignación por las tardes?..."
Friedrich Nietzche, *Así habló Zaratustra.*

Agradezco y dedico este trabajo a mi familia, por el constante cariño, apoyo, paciencia y tolerancia, en las buenas, las malas y las peores. Quizás a veces no lo parece, pero en realidad no tengo palabras para expresar mi gratitud y devoción para con ustedes.

También quiero agradecer a todos los amigos y compañeros con quienes he compartido las luchas y vicisitudes en el desarrollo de la carrera. Únicamente los que transitamos estos *rites de passage* conocemos el nivel de sacrificio necesario para llegar a la meta. A todos ustedes, los saludo. Vires acquirit eundo.

Gracias a todo el plantel docente de la Escuela de Eléctrica, porque contribuyeron a mi formación académica, gracias por cada desafío en la carrera, cada corrección, cada consejo. Nada ha caído en saco roto.

Y especialmente agradezco al Ingeniero Carlos Pocasangre, por darnos la oportunidad de desarrollar este trabajo de graduación y haber tenido confianza en que este proyecto se podía culminar exitosamente, y porque sin su valiosa guía, los resultados obtenidos no habrían sobrepasado las expectativas iniciales.

A todos ellos, Larga Vida y Prosperidad. GRACIAS POR TODO.

Víctor Jamil Palma Menjívar.

AGRADECIMIENTOS

El presente trabajo de graduación fue realizado bajo la supervisión del Ing. Carlos Osmín Pocasangre, a quien me gustaría expresar mi gratitud y agradecimiento por brindar su apoyo para la realización de este.

A mis padres por darme la vida.

A las personas que perdí en el camino pero que dejaron su presencia en la formación de mi carácter y mis primeros valores. Quienes a temprana edad me brindaron su apoyo y me permitieron aprender lo esencial para la vida.

A mis amigos, por apoyarme y animarme a lograr este sueño que se ha hecho realidad. Por compartir mis sueños y luchar por un compromiso en común.

A mis hermanos quienes al ser parte de mi vida me brindaron experiencias que nutrieron mis sueños y me ayudaron a caminar cada nuevo amanecer por lograr mis ideales. Por estar en mis momentos felices y tristes, por apoyarme, por nunca dejarme caer, por estar ahí.
Tania, Juli, Clary, Josué, William.

A Anita, por ser quien creyó en mí quien me permitió desarrollar mi potencial me enseño a mirar la vida con ojos de ternura, me mostro que otro mundo puede ser posible y me enseño a dar de mi más que a pedir de otros.

A Mi esposa, Rosita. Porque tú siempre lo serás para mí. Por estar conmigo en la etapa final pero la más dura. Por sacrificarte por mí, para que yo pueda cumplir este sueño y permitirme así luchar por ti, por estar a mi lado y ayudarme a seguir adelante. Por brindarme tu amor y hacerme tan feliz.

A Marcela, mi hija, quien se convirtió en la razón de mi vida, quien me dio la orientación que necesitaba para alcanzar esta meta. Por brindarme sonrisas y ternura que solo tú me podías dar. Por ser a tu corta edad una hija ejemplar que me llena de orgullo y me motiva a esforzarme cada día más.

FRANCISCO RODRIGO RAMIREZ.

RESUMEN

El presente trabajo documenta el proceso de diseño y construcción de una estación meteorológica, con el objetivo de que además de encontrarse registrando datos de diferentes sensores, la Estación también haga uso de la tecnología GPS, y que además facilite el acceso a los datos empleando una interfaz web para monitorear el estado actual de los sensores, y así poder descargar los archivos que contienen el registro de los parámetros, utilizando hardware y software de naturaleza libre.

El Capitulo 1 describe conceptos básicos de Meteorología, asi como también las características principales de los dispositivos utilizados para la implementación de la Estación Meteorológica, dividiéndose estos en plataformas de microcontroladores, sensores, y otros dispositivos de apoyo del sistema. También se hace una revisión de los protocolos utilizados para establecer comunicación entre los módulos que conforman la estación meteorológica.

El capitulo 2 trata el procedimiento empleado para el diseño de la Estación, describiendo inicialmente los motivos para elegir una plataforma de desarrollo libre. Después se detallan los módulos y dispositivos seleccionados para construir la Estación Meteorológica con las características definidas por los objetivos, especificando los medios que se utilizaran para su interacción, lectura y cualquier acondicionamiento necesario para las señales de salida.

Después se presentan las características del entorno y el lenguaje mediante el cual se programaran los módulos, y luego el proceso de programación propiamente, detallando todas las librerías empleadas, la forma de utilizarlas y cualquier detalle relevante con respecto a la obtención y/o calculo de todos los parámetros de la estación, así como del desarrollo de la interfaz de usuario del sistema.

Finalmente, se muestran las placas que se utilizaron para implementar las conexiones necesarias entre los circuitos, y los criterios empleados para efectuar el diseño.

El Capitulo 3 presenta los resultados obtenidos del sistema construido. Inicia con las características generales, los parámetros que muestra la Estación y las opciones de configuración disponibles. Se incluyen pruebas de registro de sensores en una muestra de los archivos generados. También se presentan los datos disponibles en el Servidor Web interno de la Estación, así como también los datos que se envían hacia un servidor web externo en Internet. Finalmente, se hace un análisis económico del sistema, comparando las ventajas que representa el enfoque utilizado de hardware/software libre, con respecto a las Estaciones Meteorológicas de origen propietario.

Se incluye además un análisis de legalidad de la Estación construida, recalcando la naturaleza libre del sistema y las posibilidades del mismo, teniendo en cuenta los términos legales definidos para poder hacer uso indiscriminado de los diferentes dispositivos empleados en el desarrollo y construcción de la Estación Meteorológica.

Finalmente, se incluye como anexo un manual del usuario, que describe la operación completa de la estación, y además se detalla la utilización de todas las características desarrolladas y las posibilidades de configuración existentes.

INDICE

DESCRIPCIÓN DEL TEMA.

ANTECEDENTES.

Desde que existe la humanidad, el hombre ha tenido una preocupación constante por entender los fenómenos climáticos que lo rodean. Mediante la observación sistemática del entorno se pueden apreciar las características que corresponden a diferentes tipos de comportamiento del clima. Presión atmosférica, temperatura, humedad, radiación solar, velocidad del viento, precipitación de agua, etc., son fenómenos cuya magnitud se puede medir utilizando diferentes métodos, convirtiéndose en variables meteorológicas que en conjunto pueden describir la situación climática de un área geográfica particular.

El estudio de diversos fenómenos geológicos también se sostiene ampliamente en el registro y análisis de estas variables climáticas. Más aun, el registro continuo y metódico de estas variables a través de los años permite deducir y establecer patrones climáticos, y por lo tanto, inferir de manera aproximada el estado del clima en un determinado intervalo futuro. Ya que el clima incide directamente en todas las actividades humanas, formular una predicción del mismo, aunque sea en una forma hipotética, permite prepararse adecuadamente para sobrellevar cualquier fenómeno que pudiera presentarse, incluso puede evitar tragedias, pérdidas humanas y materiales.

PLANTEAMIENTO DEL PROBLEMA.

Para efectuar un registro de las variables climatológicas es necesario contar con el equipo adecuado para automatizar el proceso de medición. Un microcontrolador programado para tal fin optimiza el procesamiento de los datos y se eliminan los errores por inferencia humana.

Las variables que principalmente se miden en el monitoreo meteorológico, son: temperatura, presión, velocidad y dirección del viento, humedad y radiación solar, entre otras. Para obtener los valores de estas magnitudes se utilizan diferentes clases de instrumentos de medición.

Los instrumentos que se usan comúnmente para medir las variables de interés son:

-Termómetro, que registra la temperatura ambiente.
-Barómetro, para obtener lecturas de la presión atmosférica.
-Higrómetro, para medir la humedad del aire, o de otro gas.
-Anemómetro, mide la velocidad del viento.
-Veleta, indica la dirección del viento.
-Piranómetro, mide la radiación solar que incide sobre la superficie terrestre.

Un sensor es un dispositivo que al ser sometido a determinada condición o ambiente, tiene una reacción o señal de salida, que corresponde a una magnitud de interés y puede ser interpretada para asignarle a dicha magnitud un valor especifico. Utilizando adecuadamente la información proporcionada por un sensor se puede construir un instrumento de medición.

La estación meteorológica a construir tendrá un microprocesador como unidad central, con una pantalla para visualizar información y un teclado

para poder usar algunas opciones básicas del instrumento. Utilizando un protocolo de comunicación, los diferentes sensores se conectaran a la unidad central, para que esta tome los datos, los interprete y pueda registrarlos en una memoria de almacenamiento.

Cada vez que la estación inicie su operación, se conectara a la red satelital GPS, de donde obtendrá los datos de fecha, hora y posicionamiento. Con esta información, el dispositivo programara un circuito reloj que llevara un registro constante de la fecha y hora, que se añadirá a la información proporcionada por los sensores.

Algunos sensores producen señales analógicas como salida, por lo que se hace necesario utilizar un convertidor analógico/digital para convertir estas señales en datos digitales que pueda manejar la unidad central.

Una vez que la unidad central obtiene los datos, procede a obtener la lectura de la fecha y hora actuales, mediante el circuito reloj destinado a tal fin, y los añade a los datos obtenidos con el objeto de mantener un registro conciso y metódico. Este proceso se repetirá durante un tiempo establecido para obtener una lectura precisa de las variables a medir.

Los datos se guardaran en el formato de hoja de cálculo que puede ser abierto por cualquier programa, libre o propietario, que soporte la visión y edición de este tipo de archivos. También se podrá acceder a estos datos utilizando un navegador de Internet, mediante la conexión a un servidor web sencillo alojado en la unidad central de la estación meteorológica.

JUSTIFICACIONES.

El equipo que se utiliza tradicionalmente para realizar mediciones climáticas en estaciones meteorológicas, tiende a ser de alto costo económico, y el acceso al mismo se dificulta debido a que los fabricantes constituyen un mercado reducido.

El avance de la tecnología y la tendencia actual de utilizar software y hardware de libre distribución, permite tener acceso a dispositivos relativamente sencillos, pero cuya capacidad de procesamiento se presta adecuadamente a la implementación de diferentes instrumentos y funcionalidades.

Conscientes de esta situación, varias empresas y fabricantes han desarrollado un nuevo mercado, en el que muchos sensores tradicionales, entre otros dispositivos, son ofrecidos en placas prefabricadas, con el objetivo de facilitar la interacción con una unidad central de procesamiento de datos, evitando que el usuario final efectúe el trabajo de diseñar y construir sus propios circuitos.

De esta manera, configurando adecuadamente varios módulos prefabricados, es posible construir un instrumento de medición de un desempeño y una precisión aceptables, con la ventaja de reducir considerablemente el costo económico, y al ser de libre distribución, el producto final no tiene ninguna atadura con respecto a derechos de autoría intelectual.

Utilizar la red de posicionamiento global GPS provee una forma confiable de mantener actualizados los datos de hora y fecha, de suma importancia en cualquier sistema de mediciones. Finalmente, también se aprovechará la disponibilidad de la gran red de comunicación que es Internet, para tener un acceso eficiente a las mediciones realizadas.

ALCANCE.

Al finalizar el trabajo de graduación, se debe tener una estación meteorológica, que permita medir y cuantificar las variables que se han elegido. Entre los parámetros cuantificados tendríamos: temperatura, presión, humedad relativa, velocidad del viento y dirección del viento. Teniendo registros por largos periodos de tiempo para luego ser presentados y/o procesados.

Se trabajara en una plataforma de hardware libre, con un entorno de programación basado también en software libre, como medio de visualización de los datos se puede utilizar cualquier programa que soporte hojas de datos, según la preferencia del usuario final. Además, como un medio alternativo, los registros obtenidos estarán disponibles en un mini-servidor, para el monitoreo de las variables antes mencionadas mediante internet.

El diseño de todo el sistema debe ser escalable o soportar la adición de otros sensores para cuantificar nuevos parámetros y podrá ser mejorado continuamente en posteriores trabajos de graduación.

OBJETIVOS.

OBJETIVO GENERAL.

Diseñar y construir un prototipo de estación meteorológica, que permita cuantificar, entre otras variables: temperatura, presión, humedad relativa, velocidad y dirección del viento.

OBJETIVOS ESPECÍFICOS.

Construir una estación meteorológica utilizando una plataforma de hardware libre con lenguaje de programación de libre distribución y con la potencia de procesado de datos que requiere la aplicación.

Proveer al sistema con comunicación LAN, para mostrar las variables cuantificadas en una página web tipo texto, en un mini-servidor montado en la placa de hardware libre.

Proveer al sistema con tecnología GPS que permita obtener datos de tiempo y posición para ajustar el reloj interno del sistema.

Disponer de suficiente memoria no volátil en el equipo, para poder soportar mediciones a un máximo de 32,000 muestras compuestas por: Hora, fecha, valor de las variables físicas estudiadas.

Guardar un registro diario de las magnitudes medidas por los sensores en un archivo tipo hoja de datos, para análisis posteriores.

OBSERVACIONES

El montaje de la estación meteorológica está basado en hardware de libre distribución, lo que permite que el sistema sea escalable.

En su mayor parte, el sistema el sistema implementa un protocolo de comunicación cuyo fin es optimizar la capacidad de manejo de dispositivos que tiene la unidad central.

CAPITULO 1: MARCO TEORICO.

1.1 METEOROLOGIA.

La meteorología es la ciencia interdisciplinaria, fundamentalmente una rama de la Física de la atmósfera, que estudia el estado del tiempo, el medio atmosférico, los fenómenos allí producidos y las leyes que lo rigen.

Hay que recordar que la Tierra está constituida por tres partes fundamentales: una parte sólida llamada litósfera, recubierta en buena proporción por agua (llamada hidrósfera) y ambas envueltas por una tercera capa gaseosa, la atmósfera. Éstas se relacionan entre sí produciendo modificaciones profundas en sus características. La ciencia que estudia estas características, las propiedades y los movimientos de las tres capas fundamentales de la Tierra, es la Geofísica. En ese sentido, la meteorología es una rama de la geofísica que tiene por objeto el estudio detallado de la envoltura gaseosa de la Tierra y sus fenómenos.

Se debe distinguir entre las condiciones actuales y su evolución a la que se llama tiempo atmosférico, y las condiciones medias durante un largo periodo que se conoce como clima del lugar o región. En este sentido, la meteorología es una ciencia auxiliar de la climatología ya que los datos atmosféricos obtenidos en múltiples estaciones meteorológicas durante largo tiempo se usan para definir el clima, predecir el tiempo, comprender la interacción de la atmósfera con otros subsistemas, etc. El conocimiento de las variaciones meteorológicas y el impacto de las mismas sobre el clima ha sido siempre de suma importancia para el desarrollo de la agricultura, la navegación, las operaciones militares y la vida en general.

Desde la más remota antigüedad se tiene constancia de la observación de los cambios en la atmósfera y de otros componentes asociados con el movimiento de los astros, con las estaciones del año y con fenómenos relacionados. Los antiguos egipcios asociaban los ciclos de crecida del Nilo

con los movimientos de las estrellas explicados por los movimientos de la mitología egipcia, mientras que los babilonios predecían el tiempo guiándose por el aspecto del cielo. Pero el término «meteorología» proviene de Meteorológica, título del libro escrito alrededor del año 340 a. C. por Aristóteles, quien presenta observaciones mixtas y especulaciones sobre el origen de los fenómenos atmosféricos y celestes. Una obra similar, titulada Libro de las señas, fue publicada por Teofrasto, un alumno de Aristóteles; se centraba en la observación misma de los fenómenos más que en la previsión del tiempo.

Los progresos posteriores en el campo meteorológico se centraron en que nuevos instrumentos, más precisos, se desarrollaran y pusieran a disposición. Galileo construyó un termómetro en 1607, seguido de la invención del barómetro por parte de Evangelista Torricelli en 1643. El primer descubrimiento de la dependencia de la presión atmosférica en relación a la altitud fue realizado por Blaise Pascal y René Descartes; la idea fue profundizada luego por Edmund Halley. El anemómetro, que mide la velocidad del viento, fue construido en 1667 por Robert Hooke, mientras que Horace de Saussure completa el elenco del desarrollo de los más importantes instrumentos meteorológicos en 1780 con el higrómetro a cabello, que mide la humedad del aire. Otros progresos tecnológicos, que son conocidos principalmente como parte del progreso de la física, fueron la investigación de la dependencia del volumen del gas sobre la presión, que conduce a la termodinámica, y el experimento de Benjamín Franklin con la cometa y el rayo. Franklin fue asimismo el primero en registrar de modo preciso y detallado las condiciones del tiempo en base diaria, así como en efectuar previsiones del tiempo sobre esa base.

El primero en definir de modo correcto la circulación atmosférica global fue George Hadley, con un estudio sobre los alisios efectuado en 1735. En los inicios, ésta fue una comprensión parcial de cómo la rotación terrestre influye en la cinemática de los flujos de aire. Más tarde (en el siglo XIX), fue

comprendida la plena extensión de la interacción a larga escala tras la fuerza del gradiente de presión y la deflexión causada por la fuerza de Coriolis, que causa el movimiento de las masas de aire a lo largo de las isóbaras. La fuerza de deflexión debe este nombre en los primeros años del siglo XIX, con referencia a una publicación de Gaspard-Gustave Coriolis en 1835, que describía los resultados de un estudio sobre la energía producida por la máquina con partes en rotación, como la ruta del agua de los molinos. En 1856, William Ferrel hipotetizó la existencia de una «célula de circulación» a latitudes intermedias, en las cuales el aire se deflecta por la fuerza de Coriolis creando los principales vientos occidentales. La observación sinóptica del tiempo atmosférico era aún compleja por la dificultad de clasificar ciertas características climáticas como las nubes y los vientos. Este problema fue resuelto cuando Luke Howard y Francis Beaufort introdujeron un sistema de clasificación de las nubes (1802) y de la fuerza del viento (1806), respectivamente. El verdadero punto de cambio fue la invención del telégrafo en 1843 que permitía intercambiar información sobre el clima a velocidades inigualables.

1.1.1 RAMAS DE LA METEOROLOGÍA.

La meteorología incluye el estudio (descripción, análisis y predicción) de las variaciones diarias de las condiciones atmosféricas a gran escala o Meteorología sinóptica, el estudio de los movimientos en la atmósfera y su evolución temporal basada en los principios de la mecánica de fluidos (Meteorología dinámica, muy relacionada actualmente con la meteorología sinóptica), el estudio de la estructura y composición de la atmósfera, así como las propiedades eléctricas, ópticas, termodinámicas, radiactivas y otras (Meteorología física), la variación de los elementos meteorológicos cerca de la Tierra en un área pequeña (Micrometeorología) y otros muchos fenómenos. El estudio de las capas más altas de la atmósfera (superiores a los 20 km o 25 km) generalmente implica el uso de técnicas y disciplinas especiales, y recibe el nombre de aeronomía. El término aerología se aplica al estudio de las condiciones atmosféricas a cualquier altura.

1.1.2 OBJETOS DE ESTUDIO.

Es objeto de estudio de la meteorología todo lo concerniente a la climatología y la previsión del tiempo. Esto abarca, por ejemplo, las repercusiones en la Tierra de los rayos solares, la radiación de energía calorífica por el suelo terrestre, los fenómenos eléctricos que se producen en la ionosfera, los de índole física, química y termodinámica que afectan a la atmósfera, los efectos del tiempo sobre el organismo humano, etc.

Los temas de la meteorología teórica están fundamentados, en primer lugar, sobre un conocimiento preciso de las distintas capas de la atmósfera y de los efectos que producen en ella los rayos solares. En particular, los meteorólogos establecen el balance energético que compara la energía solar absorbida por la Tierra con la energía irradiada por ésta y disipada en el espacio interestelar. Todo estudio ulterior implica, por lo demás, un conocimiento de las repercusiones que tienen los movimientos de la Tierra sobre el tiempo, los climas, la sucesión de las estaciones. También dan lugar a profundos estudios teóricos los dos parámetros principales relativos al aire atmosférico: la presión y la temperatura, cuyos gradientes y variaciones han de ser conocidos con la mayor precisión.

1.1.3 EQUIPOS E INSTRUMENTOS METEOROLÓGICOS.

En general, cada ciencia tiene su propio equipamiento e instrumental de laboratorio. Sin embargo, la meteorología es una disciplina corta en equipos de laboratorio y amplia en los equipos de observación en campo. En algunos aspectos esto puede parecer bueno, pero en realidad puede hacer que simples observaciones se desvíen hacia una afirmación errónea.

En la atmósfera, hay muchos objetos o cualidades que pueden ser medidos. La lluvia, por ejemplo, ha sido observada en cualquier lugar, siendo uno de los primeros fenómenos en ser medidos históricamente.

1.1.4 LA PREVISIÓN DEL TIEMPO.

Varias veces por día, a horas fijas, los datos procedentes de cada estación meteorológica, de los barcos y de los satélites llegan a los servicios regionales encargados de centralizarlos y analizarlos, tanto para hacer progresar a la meteorología como para establecer previsiones sobre el tiempo clave que hará en los días venideros. Como las observaciones se repiten cada 3 horas (según el horario sinóptico mundial), la sucesión de los mapas y diagramas permite apreciar la evolución sinóptica, se ve cómo las perturbaciones se forman o se resuelven, si están subiendo o bajando la presión y la temperatura, si aumenta o disminuye la fuerza del viento o si cambia éste de dirección, si las masas de aire que se dirigen hacia tal región son húmedas o secas, frías o cálidas, etc.

Parece así bastante fácil prever la trayectoria que seguirán las perturbaciones y saber el tiempo que hará en determinado lugar al cabo de uno o varios días. En realidad, la atmósfera es una gigantesca masa gaseosa tridimensional, turbulenta y en cuya evolución influyen tantos factores que uno de éstos puede ejercer de modo imprevisible una acción preponderante que trastorne la evolución prevista en toda una región. Así, la previsión del tiempo es tanto menos insegura cuando menor es la anticipación y más reducido el espacio a que se refiere. Debido a ello la previsión es calificada de micrometeorológica, mesometeorológica o macrometeorológica, según se trate, respectivamente, de un espacio de 15 km, 15 a 200 km o más de 200 km.

Las previsiones son formuladas en forma de boletines, algunos de los cuales se destinan a la ciudadanía en general y otros a determinados ramos de la actividad humana y navegación aérea y marítima, agricultura, construcción, turismo, deportes, regulación de los cursos de agua, ciertas industrias, prevención de desastres naturales, etc.

1.1.5 ESTACIÓN METEOROLÓGICA.

Una estación meteorológica es una instalación destinada a medir y registrar regularmente diversas variables meteorológicas. Estos datos se utilizan tanto para la elaboración de predicciones meteorológicas a partir de modelos numéricos como para estudios climáticos.
Está equipada con los principales instrumentos de medición, entre los que se encuentran los siguientes:

-Anemómetro (mide la velocidad del viento).
-Veleta (señala la dirección del viento).
-Barómetro (mide la presión atmosférica).
-Heliógrafo (mide la insolación recibida en la superficie terrestre).
-Higrómetro (mide la humedad).
-Piranómetro (mide la radiación solar).
-Pluviómetro (mide el agua caída).
-Termómetro (mide la temperatura).

Estos instrumentos se encuentran protegidos en una casilla ventilada, denominada abrigo meteorológico o pantalla de Stevenson, la cual mantiene la luz solar lejos del termómetro y al viento lejos del higrómetro, de modo que no se alteren las mediciones de éstos.

Cuanto más numerosas sean las estaciones meteorológicas, más detallada y exactamente se conoce la situación. Hoy en día, gran cantidad de ellas cuentan con personal especializado, aunque también hay un número de estaciones automáticas ubicadas en lugares inaccesibles o remotos, como regiones polares, islotes deshabitados o cordilleras. Además existen fragatas meteorológicas, barcos que contienen a bordo una estación meteorológica muy completa y a los cuales se asigna una posición determinada en pleno océano. Sin embargo, es necesario recalcar que, con el gran crecimiento de la población urbana desde fines del siglo XIX, la mayor parte de las estaciones meteorológicas están actualmente situadas en zonas urbanas, bien porque

se ubican en ciudades nuevas o bien porque se encuentran en poblaciones rurales absorbidas por los grandes núcleos urbanos en su proceso de expansión, con lo que existe un sesgo introducido por los microclimas urbanos que dan pie para corroborar, de manera errónea, el aumento de las temperaturas a escala mundial (lo que sería una prueba del calentamiento global), lo cual está muy lejos de ser un hecho comprobado sin lugar a dudas.

1.1.6 MAGNITUDES CUANTIFICABLES.

Los instrumentos más comunes y las variables que se miden en una estación meteorológica incluyen:

-Termómetro, instrumento que mide la temperatura en diversas horas del día.
-Termómetros de subsuelo (geotermómetro), para medir la temperatura a 5, 10, 20, 50 y 100 cm de profundidad.
-Termómetro de mínima junto al suelo, mide la temperatura mínima a una distancia de 15 cm sobre el suelo.
-Termógrafo, registra automáticamente las fluctuaciones de la temperatura.
-Barómetro, medida de presión atmosférica en la superficie.
-Pluviómetro, medida de la cantidad de agua caída sobre el suelo en forma de lluvia, nieve o granizo.
-Psicrómetro o higrómetro, medida de la humedad relativa del aire y la temperatura del punto de rocío.
-Piranómetro, medida de la radiación solar global (directa + difusa).
-Heliógrafo, medida de las horas de luz solar.
-Anemómetro, medida de la velocidad del viento.
-Veleta, que indica la dirección del viento.
-Nefobasímetro, medida de la altura de las nubes, pero sólo en el punto donde éste se encuentre colocado.

1.1.7 CAMPOS DE APLICACIONES.

- Agrometeorología.- Tiene por objetivo contribuir a la generación de información meteorológica que permite desarrollar investigaciones específicas en el campo de la bioclimatología y la agroclimatología, y aportar información útil para los sectores productivos, organismos oficiales de investigación, y empresas ó instituciones no vinculadas directamente a la producción agropecuaria.
- Meteorología teórica.- se ocupa del estudio de los fenómenos meteorológicos a través de teorías científicas.
- Meteorología física.- se interesa en el estudio de las propiedades físicas de la atmósfera.
- Meteorología Dinámica.- estudia la atmósfera desde el punto de vista de las leyes dinámicas que gobiernan los sistemas meteorológicos.
- Meteorología experimental.- estudia los fenómenos y procesos meteorológicos en laboratorios y campos de experimentación.
- Meteorología aplicada.- en su aplicación a todas las actividades sociales, económicas y, en general, a todas las actividades humanas.
- Meteorología Sinóptica.- se ocupa de los fenómenos atmosféricos sobre la base de análisis de cartas en la que previamente se han asentado observaciones sinópticas con el propósito de hacer un diagnóstico o un pronóstico de condiciones meteorológicas.
- Meteorología Aeronáutica.- estudia el efecto que los fenómenos meteorológicos tienen sobre las aeronaves y todo lo concerniente a la aeronavegación.
- Hidrometeorología. Rama de la Meteorología que se relaciona con la Hidrología.
- Meteorología Marítima.- que consta a su vez de dos áreas:

a) Meteorología oceánica: Estudia la interacción entre la atmósfera y el mar.

b) Estrictamente Meteorología marítima: Se ocupa de suministrar servicios, desde el punto de vista meteorológico, a todas las actividades marinas.

- Meteorología Médica.- Meteorología relacionada con la salud humana.

- Micrometeorología.- estudia las condiciones meteorológicas a pequeña escala. Este tipo de estudio normalmente implica mediciones de parámetros meteorológicos y estudios cuidadosos de cerca de superficie en períodos cortos de tiempo.

- Mesometeorología.- estudia las condiciones meteorológicas a escala media. El tamaño del área que cubren estos fenómenos es desde algunos km2 hasta decenas de km2.

- Macrometeorología.- estudia las condiciones meteorológicas a gran escala. El área que ocupan estos fenómenos meteorológicos se relaciona con amplias regiones geográficas, tales como partes de un continente, un continente completo o, incluso, el planeta entero.

De acuerdo a lo establecido por la Organización Meteorológica Mundial (OMM), las estaciones meteorológicas se clasifican de la manera mostrada en la tabla 1.1.

Según su Finalidad	Clasificación
Sinóptica	-Climatológica. -Agrícolas. -Especiales. -Aeronáuticas. -Satélites.
De acuerdo a la magnitud de las observaciones.	-Principales. -Ordinarias. -Auxiliares o adicionales.
Por el nivel de observación.	-Superficie -Altitud
Según el lugar de observación.	-Terrestre. -Aéreas. -Marítimas.

Tabla 1.1 Clasificación de Estaciones Meteorológicas.

Como se puede observar una estación meteorológica puede tener diferentes fines, dependiendo de los propósitos para los cuales fue instalada. La información se utiliza en varias aplicaciones u observaciones adicionales que

le dan sus características. Por consiguiente, en una estación meteorológica pueden conjugarse dos o más categorías simultáneamente.

1.1.8 OBSERVACIONES METEOROLÓGICAS.

La observación meteorológica consiste en la medición y determinación de todos los elementos que en su conjunto representan las condiciones del estado de la atmósfera en un momento dado y en un determinado lugar utilizando instrumental adecuado.

Estas observaciones realizadas con métodos y en forma sistemática, uniforme, ininterrumpida y a horas establecidas, permiten conocer las características y variaciones de los elementos atmosféricos, los cuales constituyen los datos básicos que utilizan los servicios meteorológicos, tanto en tiempo real como diferido.

Las observaciones deben hacerse, invariablemente, a las horas preestablecidas y su ejecución tiene que efectuarse empleando el menor tiempo posible. Es de capital importancia que el observador preste preferente atención a estas dos indicaciones, dado que la falta de cumplimiento de las mismas da lugar, por la continua variación de los elementos que se están midiendo u observando, a la obtención de datos que, por ser tomados a distintas horas o por haberse demorado demasiado tiempo en efectuarlos, no sean sincrónicos con observaciones tomadas en otros lugares. La veracidad y exactitud de las observaciones es imprescindible, ya que de no darse esas condiciones se lesionan los intereses, no solo de la meteorología, sino de todas las actividades humanas que se sirven de ella. En este sentido, la responsabilidad del observador es mayor de lo que generalmente él mismo supone.

1.1.8.1 OBSERVACIONES SINÓPTICAS.

Son observaciones que se efectúan en forma horaria (horas fijas del día) remitiéndolas inmediatamente a un centro recolector de datos, mediante

mensajes codificados, por la vía de comunicación más rápida disponible. Estas observaciones se utilizan para una multitud de fines meteorológicos, en general en tiempo real, es decir, de uso inmediato, y especialmente para la elaboración de mapas meteorológicos para realizar el correspondiente diagnóstico y formular los pronósticos del tiempo para las diferentes actividades.

1.1.8.2 OBSERVACIONES CLIMATOLÓGICAS.

Son observaciones que se efectúan para estudiar el clima, es decir, el conjunto fluctuante de las condiciones atmosféricas, caracterizados por los estados y las evaluaciones del tiempo en una porción determinada del espacio.

Estas observaciones difieren muy poco de las sinópticas en su contenido y se realizan también a horas fijas, tres o cuatro veces al día (por lo menos) y se complementan con registros continuos diarios o semanales, mediante instrumentos registradores

1.1.8.3 OBSERVACIONES AERONÁUTICAS.

Se trata de observaciones especiales que se efectúan en las estaciones meteorológicas instaladas en los aeródromos, esencialmente para satisfacer las necesidades de la aeronáutica, aunque comúnmente se hacen también observaciones sinópticas. Estas observaciones se comunican a otros aeródromos y, frecuentemente, a los aviones en el vuelo, pero en los momentos de despegue y aterrizaje, el piloto necesita algunos elementos esenciales de la atmósfera, como el tiempo presente, dirección y velocidad del viento, visibilidad, altura de las nubes bajas, reglaje altimétrico, etc., para seguridad de la nave, tripulación y pasajeros

1.1.8.4 OBSERVACIONES MARÍTIMAS.

Son observaciones que se realizan sobre buques fijos, móviles, boyas ancladas y a la deriva. Estas dos últimas son del tipo automático. Estas

observaciones constituyen una fuente vital de datos y son casi las únicas observaciones de superficie fiables procedentes de los océanos, que representan más de los dos tercios de la superficie total del globo. Esas observaciones se efectúan en base a un plan, según el cual se imparte una formación a determinados observadores seleccionados entre las tripulaciones de las flotas de buques, especialmente mercantes, para que puedan hacer observaciones sinópticas durante el viaje y transmitirlas a las estaciones costeras de radio.

1.1.8.5 OBSERVACIONES AGRÍCOLAS.

Son observaciones que se hacen de los elementos físicos y biológicos del medio ambiente, para determinar la relación entre el tiempo y la vida de plantas y animales.

Con estas observaciones, se trata de investigar la acción mutua que se ejerce entre los factores meteorológicos e hidrológicos, por una parte, y la agricultura en su más amplio sentido, por otra. Su objeto es detectar y definir dichos efectos para aplicar después los conocimientos que se tienen de la atmósfera a los aspectos prácticos de la agricultura. Al mismo tiempo se trata de disponer de datos cuantitativos, para las actividades de planificación, predicción e investigación agrometeorológica y para satisfacer, plenamente, la función de ayuda a los agricultores, para hacer frente a la creciente demanda mundial de alimentos y de productos secundarios agrícolas.

1.1.8.6 OBSERVACIONES DE LA PRECIPITACIÓN.

Son observaciones relativas a la frecuencia, intensidad y cantidad de precipitación, ya sea en forma de lluvia, llovizna, aguanieve, nieve o granizo y constituyen elementos esenciales de diferentes tipos de observaciones. Dada la gran variabilidad de las precipitaciones, tanto desde el punto de vista espacial como temporal, se debe contar con un gran número de estaciones suplementarias de observación de la precipitación.

1.1.8.7 OBSERVACIONES DE ALTITUD.

Son observaciones de la presión atmosférica, temperatura, humedad y viento que se efectúan a varios niveles de la atmósfera, llegándose generalmente hasta altitudes de 16 a 20 km y, muchas veces, a más de 30 km. Estas mediciones se hacen lanzando radiosondas, que son elevadas al espacio por medio de globos inflados con gas más liviano que el aire, y a medida que van subiendo, transmiten señales radioeléctricas, mediante un radiotransmisor miniaturizado, que son captadas en tierra por receptores adecuados y luego procesadas para convertirlas en unidades meteorológicas.
La observación de la dirección y velocidad del viento puede efectuarse con la misma radiosonda, haciendo uso del "Sistema de Posicionamiento Global (GPS)" y recibiendo los datos, en tierra, mediante radioteodolitos siguiendo la trayectoria de un globo inflado con gas helio o hidrógeno, mediante un teodolito óptico o, para mayor altura, un radar aerológico.

1.1.8.8 OTRAS OBSERVACIONES.

También se recopilan otras observaciones efectuadas a partir de las aeronaves en vuelo y diversos tipos de observaciones especiales, tales como las que se refieren a la radiación, al ozono, a la contaminación, hidrológicas, evaporimétricas, temperatura y humedad del aire a diversos niveles hasta 10 m. de altura y del suelo y subsuelo.

1.1.8.9 HORAS EN QUE SE REALIZAN LAS OBSERVACIONES.

La hora observacional depende del tipo, finalidad y uso de cada observación. Es importante que las observaciones sean sincrónicas y continuadas durante varios años, para que puedan utilizarse en cualquier estudio o investigación Para determinado tipo de observaciones, en especial las sinópticas, la OMM ha establecido horas fijas, en tiempo universal coordinado (UTC).

Las horas principales, para efectuar observaciones sinópticas de superficie son: 00:00 - 06:00 - 12:00 - 18:00 UTC.
Las horas sinópticas intermedias son: 03:00 - 09:00 - 15:00 - 21:00 UTC.

Las horas fijas para la observación sinóptica en altitud son: 00:00 - 12:00 UTC.

Las observaciones aeronáuticas se realizan en forma horaria, las de despegue y aterrizaje en el momento mismo en que el piloto efectúa dichas operaciones, y en vuelo en cualquier momento.

1.2 COMPONENTES DE IMPLEMENTACION Y DESARROLLO DE LA ESTACION METEOROLOGICA.

1.2.1 ARDUINO.

Arduino es una plataforma de desarrollo, de libre distribución, para la creación de prototipos basados en software y hardware flexibles y fáciles de usar. Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede afectar aquello que le rodea controlando luces, motores y otros actuadores. El microcontrolador en la placa Arduino se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing). Los proyectos hechos con Arduino pueden ejecutarse sin necesidad de conectar a un ordenador.

El hardware consiste en una placa con un microcontrolador Atmel AVR y puertos de entrada/salida. Los microcontroladores más usados son el Atmega168, Atmega328, Atmega1280, Atmega2560 y ATmega8 por su sencillez y bajo costo que permiten el desarrollo de múltiples diseños. Por otro lado el software consiste en un entorno de desarrollo que implementa el lenguaje de programación Processing/Wiring y el cargador de arranque (boot loader) que corre en la placa, y mediante el cual se cargan los códigos en el microcontrolador.

Una de las principales ventajas de este modulo para ser elegido como parte central del diseño es la factibilidad que ofrece con: captura de datos, acondicionamiento de señal, transmisión de datos y disponibilidad de sensores compatibles con la plataforma.

Otra de las ventajas considerables es la cantidad de Shields, o módulos, que se encuentran disponibles para la plataforma Arduino y que permiten añadir diversas funcionalidades, de forma que se puedan solventar las necesidades específicas de un diseño particular.

1.2.2 ARDUINO MEGA.

El Arduino Mega es una placa con un microcontrolador ATmega2560. Tiene 54 entradas/salidas digitales (de las cuales 14 proporcionan salida PWM), 16 entradas digitales, 4 UARTS (puertos serie por hardware), un cristal oscilador de 16MHz, conexión USB, entrada de corriente, conector ICSP y botón de reset.

DESCRIPCION	CARACTERISTICAS
Microcontrolador	ATmega2560
Voltaje de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (limite)	6-20V
Pines E/S digitales	54 (14 proporcionan salida PWM)
Pines de entrada analógica	16
Intensidad por pin	40 mA
Intensidad en pin 3.3V	50 mA
Memoria Flash	256 KB (8 KB usados por (bootloader)
SRAM	8 KB
EEPROM	4 KB
Velocidad de reloj	16 MHz

Tabla 1.2 Resumen de Características principales del Arduino Mega.

Alimentación: El Arduino Mega puede ser alimentado mediante la conexión USB o con una fuente de alimentación externa. El origen de la alimentación

se selecciona automáticamente. La placa puede trabajar con una alimentación externa de entre 6 a 20 voltios. Si el voltaje suministrado es inferior a 7V, el pin de 5V puede proporcionar menos voltaje de su capacidad nominal y la placa puede volverse inestable, si se usan más de 12V los reguladores de voltaje se pueden sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios.
Los pines de alimentación son los siguientes:

VIN: La entrada de voltaje a la placa Arduino cando se está usando una fuente externa de alimentación (en opuesto a los 5 voltios de la conexión USB). Se puede proporcionar voltaje a través de este pin, o si se está alimentado a través de la conexión de 2.1mm, acceder a ella a través de este pin.
5V: La fuente de voltaje estable que es usada para alimentar el microcontrolador y otros componentes de la placa. Esta puede provenir de VIN a través de un regulador integrado en la placa, o proporcionada directamente por el USB u otra fuente estabilizada de 5V.

3V3: Una fuente de voltaje a 3.3 voltios generada en el chip FTDI integrado en la placa. La corriente máxima soportada es 50mA.

GND: Pines de toma de tierra.

Memoria: El ATmega2560 tiene 256KB de memoria flash para almacenar código, de los cuales 8KB son usados para el arranque del sistema (bootloader). Tiene 8 KB de memoria RAM y 4KB de memoria EEPROM, a la cual se puede acceder para leer o escribir.

Entradas y Salidas: Cada uno de los 54 pines digitales del Mega pueden utilizarse como entradas o como salidas usando las funciones pinMode(), digitalWrite(), y digitalRead().

Las E/S operan a 5 voltios. Cada pin puede proporcionar o recibir una intensidad máxima de 40mA y tiene una resistencia interna (desconectada por defecto) de 20-50kOhms.

El Mega tiene 16 entradas analógicas, y cada una de ellas proporciona una resolución de 10bits (1024 valores).

Puertos Serie: En los pines: 0 (RX) y 1 (TX), 19 (RX) y 18 (TX); 17 (RX) y 16 (TX); 15 (RX) y 14 (TX). Usado para recibir (RX) y transmitir (TX) datos a través de puertos seriales TTL. Los pines Serie: 0 (RX) y 1 (TX) están conectados a los pines correspondientes del chip FTDI USB-to-TTL.

Interrupciones Externas: Pines 2 (interrupción 0), 3 (interrupción 1), 18 (interrupción 5), 19 (interrupción 4), 20 (interrupción 3), y 21 (interrupción 2). Estos pines se pueden configurar para lanzar una interrupción en un valor LOW (0V), en flancos de subida o bajada (cambio de LOW a HIGH (5V) o viceversa).

PWM: Pines de 0 a 13. Proporcionan una salida PWM de 8 bits de resolución (valores de 0 a 255).

Comunicaciones: El Arduino Mega facilita en varios aspectos la comunicación con el ordenador, otro Arduino u otros microcontroladores. Proporciona cuatro puertos de comunicación vía serie UART TTL (5V). Un chip FTDI FT232RL integrado en la placa canaliza esta comunicación serie a traes del USB y los drivers FTDI (incluidos en el software de Arduino) proporcionan un puerto serie virtual en el ordenador. El software incluye un monitor de puerto serie que permite enviar y recibir información textual de la placa Arduino.

SPI: 50 (SS), 51 (MOSI), 52 (MISO), 53 (SCK). Estos pines proporcionan comunicación SPI.

I2C: 20 (SDA) y 21 (SCL). Soporte del protocolo de comunicaciones I2C (TWI).

Programación: El ATmega2560 en el Arduino Mega viene precargado con un gestor de arranque (bootloader) que permite cargar nuevo código sin necesidad de un programador por hardware externo. Se comunica utilizando el protocolo STK500 original.

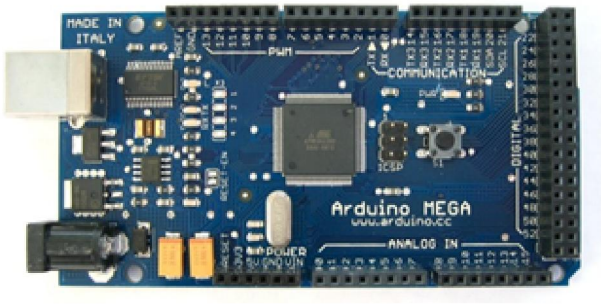


Fig. 1.1, Arduino Mega. (Fuente: http://arduino.cc/es/Main/ArduinoBoardMega)

1.2.3 ARDUINO UNO.

El Arduino Uno es una placa electrónica basada en el microprocesador Atmega328. Tiene 14 pines digitales de entrada/salida (de las cuales 6 se pueden utilizar como salidas PWM), 6 entradas analógicas, un cristal oscilador de 16 MHz, y una conexión USB.

Alimentación: Al igual que el ARDUINO MEGA la tarjeta puede funcionar con un suministro externo de 6 a 20 voltios. Si se proporcionan menos de 7V, sin embargo, el pin de 5V puede suministrar menos de cinco voltios y la junta puede ser inestable. Si se utiliza más de 12V, el regulador de voltaje se puede sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios.

VIN: El voltaje de entrada a la placa Arduino cuando se trata de utilizar una fuente de alimentación externa (en lugar de 5 voltios de la conexión USB u otra fuente de alimentación regulada). Puede suministrar tensión a través de este pin, o, si el suministro de tensión a través de la toma de alimentación, el acceso a través de este pin.

5V: Este pin actúa como una salida estable de 5V que provienen de un regulador en la placa.

3V3: Un suministro de 3.3 voltios generados por el regulador. La máxima corriente entregada es de 50 mA.

GND: Pin a tierra.

Descripción	Características
Microcontroladores	ATmega328
Voltaje de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límites)	6-20V
Pines E / S digitales	14 (de los cuales 6 proporcionan PWM)
Pines de entrada analÃ³gica	6
DC Corriente por I / O Pin	40 mA
Corriente CC para Pin 3.3V	50 mA
Memoria Flash	32 KB y 0,5 KB utilizado por boot loader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Velocidad del reloj	16 MHz

Tabla 1.3, Resumen de Características principales del Arduino UNO.

IOREF: Este pin de la placa Arduino proporciona la referencia de tensión con la que opera el microcontrolador. Un shield configurado puede leer el voltaje del pin IOREF y seleccionar la fuente de alimentación adecuada o habilitar transductores de voltaje en las salidas para trabajar con los 5V o 3.3V.

Memoria: El ATmega328 tiene 32 KB (con 0.5 KB utilizado por el gestor de arranque). También dispone de 2 KB de SRAM y 1 KB de EEPROM (que puede ser leído y escrito con la librería EEPROM).

Entrada y salida: Cada uno de los 14 pines digitales en el Uno se puede utilizar como una entrada o salida. Funcionan a 5 voltios. Cada pin puede proporcionar o recibir un máximo de 40 mA y tiene una resistencia pull-up interna (desconectada por defecto) de 20 a 50 kOhm.

El Uno tiene 6 entradas analógicas, etiquetadas de A0 a A5, cada uno de las cuales proporcionan 10 bits de resolución. El rango de operación por defecto va desde GND hasta 5 voltios, aunque es posible cambiar el extremo superior del rango con el pin AREF, definiendo así un nuevo voltaje de referencia.
Interrupciones externas: Pines 2 y 3. Pueden ser configurados para activar una interrupción en un nivel lógico bajo, un flanco ascendente o descendente, o un cambio de nivel lógico.

PWM: Pines 3, 5, 6, 9, 10, y 11 proporcionan una salida PWM de 8 bits.
Comunicación: El Arduino Uno tiene puertos para la comunicación con una computadora, otro Arduino, u otros microcontroladores.

Bus Serial: Pines 0 (RX) y 1 (TX) Se utiliza para recibir (RX) y transmitir (TX) datos a través de un bus serial TTL. Un Arduino utilizando este puerto serial para comunicarse con una computadora a través de USB aparece como un puerto COM virtual.

SPI: Pines 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK) Para comunicación SPI.

I2C: Pines A4 y A5, o los pines SDA y SCL (en los modelos R3) soportan comunicación I2C mediante la librería Wire.

Fig. 1.2 Arduino UNO. (Fuente: http://arduino.cc/en/Main/arduinoBoardUno)

1.2.4 ETHERNET SHIELD.

El Ethernet Shield le permite a una placa Arduino conectarse a internet. Está basada en el chip Ethernet Wiznet W5100, que es capaz de usar los protocolos TCP y UDP. En modo Servidor soporta hasta cuatro clientes con conexiones simultáneas.

Arduino usa los pines digitales 10, 11, 12, y 13 (SPI) para comunicarse con el W5100 en el Ethernet shield. Si se utilizan con este fin, estos pines no pueden ser utilizados para e/s genéricas.

El shield provee un conector Ethernet estándar RJ45. También incluye LEDs que proveen información sobre la operación del modulo, estos son:

-PWR: indica que el Arduino y el shield están alimentados.

-RX: parpadea cuando el shield recibe datos.

-TX: parpadea cuando el shield envía datos.

El jumper soldado marcado como "INT" puede ser conectado para permitir a la placa Arduino recibir notificaciones de eventos por interrupción desde el W5100, pero esto no está soportado por la librería Ethernet. El jumper conecta el pin INT del W5100 al pin digital 2 de Arduino.

Este shield se coloca directamente sobre el Arduino, y dispone de unos conectores que permiten a su vez, apilar otras placas encima. Entonces, para utilizar este modulo solo hay que montarlo sobre el Arduino, y conectar el Ethernet Shield a un ordenador, a un switch o a un router utilizando un cable Ethernet standard (CAT5 o CAT6 con conectores RJ45). La conexión al ordenador puede requerir el uso de un cable cruzado (aunque muchos ordenadores actuales, pueden hacer el cruce de forma interna).
Al shield se le debe asignar una dirección MAC y una IP. Una dirección MAC es un identificador global único para cada dispositivo en particular, asignar una al azar suele funcionar, pero no se debe utilizar la misma para más de una placa.
Una dirección IP válida depende de la configuración de la red. Es posible usar DHCP para asignar una IP dinámica a la placa. Opcionalmente se pueden especificar un Gateway de salida la máscara de subred.

La última versión de la Ethernet Shield incluye un slot para microSD, que puede ser accedido desde el Arduino, lo que le permite al modulo almacenar datos de interés.

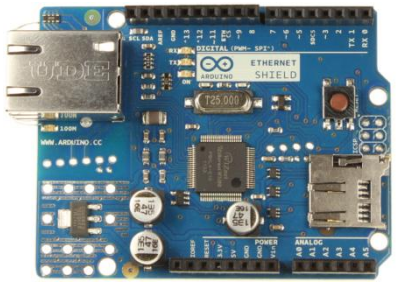


Fig 1.3. Ethernet Shield. (Fuente: http://arduino.cc/en/Main/ArduinoEthernetShield)

1.2.5 RTC SHIELD.

Un RTC o reloj de tiempo real es básicamente un circuito dedicado a proveer fecha y hora, además generalmente funciona con una batería, de manera que mantiene la fecha/hora incluso cuando hay un corte de energía.

La mayoría de los microcontroladores, incluyendo el Arduino, tienen incorporada una función que lleva el conteo del tiempo transcurrido desde que se energizo el dispositivo. En el caso del Arduino, esta función es llamada millis() y devuelve un valor en milisegundos. Esta es una forma de realizar un seguimiento de períodos de tiempo como minutos u horas.

Sin embargo, esto significa que cuando esta energizado, el temporizador de millis se ajusta inicialmente a 0, sin disponer de mayor información sobre la fecha y hora real, lo único que se sabe es la cantidad de milisegundos desde la última vez que se encendió.

Así que si se desea ajustar la hora en el Arduino, se tendrá que programar la fecha y hora, y se tendrá que contar desde ese momento. Pero si se corta la alimentación, habrá que reajustar la hora.

Este enfoque es suficiente para aplicaciones básicas, pero un proyecto serio necesita un cronometraje consistente que no desaparezca con un corte de energía. Ese es el motivo principal para utilizar un RTC independiente.

La figura 1.4 muestra el reloj de tiempo real, DS1307. Es de bajo costo, fácil de soldar, puede funcionar durante varios años con una batería de litio, y mantiene el registro de la hora y fecha incluso cuando el Arduino pierde energía o se reprograma.

Fig 1.4 Integrado RTC DS1307. (Fuente: http://learn.adafruit.com/adafruit-data-logger-shield/using-the-real-time-clock)

Fig. 1.5 RTC Shield + SD Card. (Fuente: http://www.robot-italy.com/en/adafruit-assembled-data-logging-shield-for-arduino.html)

1.2.5 SENSORES.

1.2.5.1 BMP085.

Este sensor de precisión de Bosch es la mejor solución de detección de bajo costo para medir la presión atmosférica y la temperatura. Debido a los cambios de presión con la altitud también se puede utilizar como un altímetro. El sensor está soldado a un PCB con un regulador de 3.3V, se conecta al bus I2C y ya incluye resistencias de pull-up en los pines SDA y SCL.

Fig.1. 6 Sensor de presión BMP085. (Fuente: http://learn.adafruit.com/bmp085)

Descripción	Característica
Alimentación	3 a 5 VCC
Lógica	3 a 5V compatible
Rango de detección de Presión	300-1100 hPa (9000m a -500m sobre el nivel del mar)
Resolución	0.03hPa / 0,25 m
Rango de funcionamiento	-40 A +85 Â° C
Precisión de temperatura	± 2 ° C

Tabla 1.4, Características del BMP085.

1.2.5.2 DHT22.

El DHT22 es un sensor digital de temperatura y humedad, de bajo costo. Se utiliza un sensor de humedad capacitivo y un termistor para medir el aire circundante, y da una señal digital en el pin de datos (no hay pines de entrada analógica). Es bastante simple de usar, pero requiere sincronización cuidadosa para tomar datos. El único inconveniente de este sensor es que sólo se pueden obtener lecturas una vez cada 2 segundos. La figura 1.7 muestra el esquema de conexión. Simplemente se conecta el primer pin de la izquierda al Vcc 3-5V, el segundo pin al pin de entrada de datos (En este caso, el pin 2 del Arduino) y el pin del extremo derecho a tierra.

características	Descripción
Alimentación	3 a 5V de alimentación y de E / S
Rango de Humedad/ error	0-100 % / 2-5 %
Rango Temperatura / error	para -40 a 80 °C / ± 0,5 ° C
Velocidad de muestreo	0,5 Hz

Tabla 1.5 Características sensor de temperatura DHT22.

Se omite el pin 3, pues no se utiliza. Además se coloca una resistencia pull up de 10K entre VCC y el pin de datos.

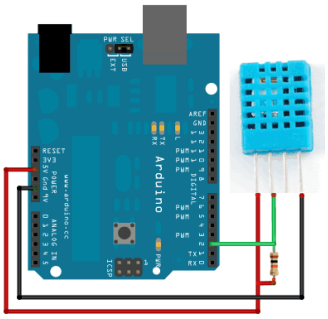


Fig. 1.7 Conexión DHT22 a Arduino. (Fuente: http://learn.adafruit.com/dht/connecting-to-a-dhtxx-sensor)

1.2.5.3 ANEMÓMETRO DAVIS.

Un anemómetro es un aparato para medir la velocidad del viento. Por lo general, utiliza copas montadas sobre un eje vertical, que giran a medida que el viento fluye. Una veleta ubicada cerca de las copas tiene como función obtener la dirección del viento.

El anemómetro y la veleta forman un dispositivo analógico pasivo. No se enciende. Responde a un breve pulso de excitación en la dirección. Los pulsos de velocidad son creados por un interruptor magnético sellado. De hecho, todo el conjunto es una unidad sellada y no se pretende que se le de mantenimiento, más allá de una realineación de las piezas externas. En la sección superior se encuentra el interruptor magnético en una placa de circuito. La dirección del viento se obtiene mediante un potenciómetro.

Este anemómetro utiliza tres copas con sus ejes a presión en un tubo de plástico. Cada taza puede reemplazarse si llegara a producirse algún daño. Las tres copas se mantienen en su lugar por un tapón de plástico presionado en el extremo del tubo.

Fig. 1.8 Anemómetro Davis. (Fuente: http://www.lexingtonwx.com/anemometer/)

La velocidad de viento y las funciones de dirección del viento tiene circuitos separados, pero el cable rojo es conexión "ground" para ambos. El interruptor del reed switch está conectado al cable rojo y al cable negro. Por lo tanto, cuando el interruptor de láminas se cierra, la tensión de cable negro pasará de 5 voltios a cero mientras que el interruptor magnetico (reed switch) está cerrado. Este pulso se produce una vez por revolución cuando el dispositivo está funcionando con el viento.

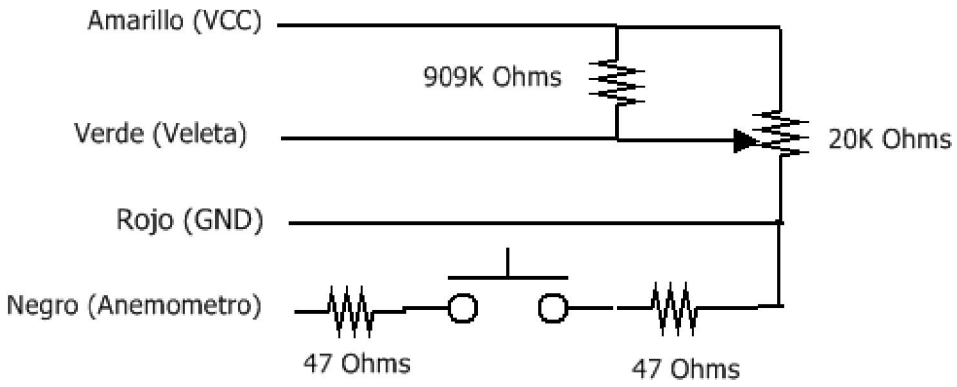


Fig. 1.9 Esquemático de conexión del Anemómetro DAVIS.

En el esquema de la figura 1.9, se muestra el diagrama de conexión del Anemómetro/Veleta DAVIS, hay dos resistencias de 47 ohmios en serie con el interruptor magnético. Estas resistencias protegen al reed switch en caso de producirse conexiones erróneas.

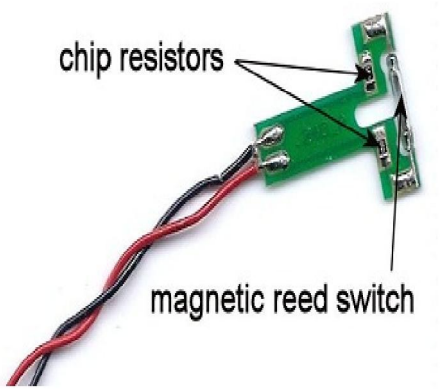


Fig. 1.10 Switch Magnético. (Fuente: http://www.lexingtonwx.com/anemometer/)

El circuito de la dirección del viento utiliza un potenciómetro lineal de 20K Ohmios para detectar la posición de la veleta. La señal de salida corresponde entonces al valor resistivo que marque el potenciómetro, según sea el desplazamiento angular de la Veleta.

1.2.6 PERIFERICOS.

1.2.6.1 PANTALLA LCD.

La pantalla es un modulo de comunicación serial e I2C que proporciona las características básicas de presentación de caracteres que proporciona una pantalla LCD estándar.

Tiene una estructura de comunicación simple para presentar texto en la pantalla. Cuenta con 20 columnas x 4 Filas y se pueden definir caracteres especiales. El modulo incluye un puerto de entrada para un teclado matricial 4X4, 8 botones separados, o un botón con un puerto de control IR.
La luz de fondo del LCD y el contraste puede ajustarse por control en la programación para proporcionar las diferentes condiciones de luz que sean requeridas, teniendo hasta 254 niveles de intensidad.

El modulo soporta dos interfaces de comunicación:

I2C:

-Compatible con el Estándar Philips bus I2C.

-Data rate 100 kbps.

Serial TTL:

- 9600 baudios.

-8 Bits por carácter

-1 Bit de parada.

-No control de flujo.

-No se debe conectar directamente con RS232. Se debe Usar un chip MAX232 o equivalente para convertir los niveles de la señal RS232 a 5v.

Modulo de conexión:

El modulo tiene 4 puertos de conexión: Interfaz I2C, Interfaz Serial , conector para teclado y conector para control IR.

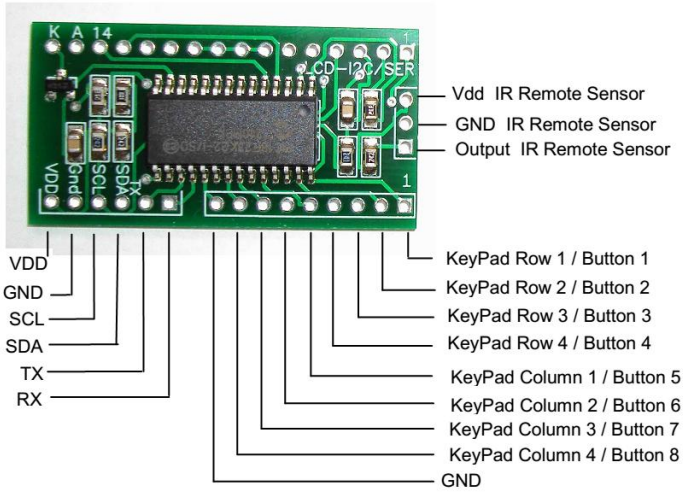


FIG. 1.11 Modulo principal pantalla LCD. (Fuente: http://www.web4robot.com/files/SerialLCDCtrN.pdf)

Conexión de interface I2C/serial TTL:

Pin No.	Pin Name	Description
1	RX	RX – LCD controller serial receive line
2	TX	TX – LCD controller serial transmit line
3	SDA	I2C SDA signal
4	SCL	I2C SCL signal
5	GND	Ground connection
6	VDD	Supply voltage

Tabla 1.6 Pinout para comunicación I2C. (Fuente: http://www.web4robot.com/files/SerialLCDCtrN.pdf)

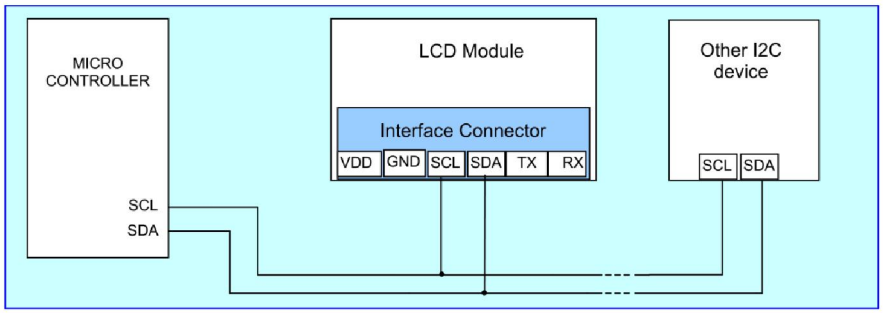


Fig. 1.12 Conexión de modulo LCD a bus I2C. (Fuente: http://www.web4robot.com/files/SerialLCDCtrN.pdf)

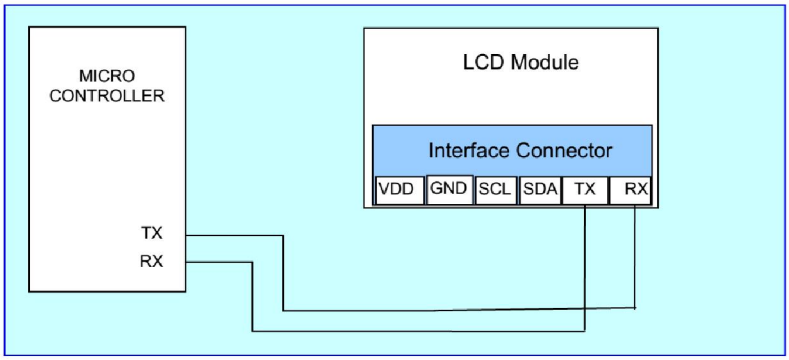


Fig. 1.13 Conexión de modulo LCD a bus serial TTL. (Fuente: http://www.web4robot.com/files/SerialLCDCtrN.pdf)

1.2.6.2 TECLADO MATRICIAL 4X4.

Los teclados matriciales son en realidad una técnica de interfaz, en la que las Entradas y Salidas se dividen en dos secciones: las columnas y las filas. Se puede imaginar una matriz como una hoja de Excel. En la figura 1.14 es una matriz de 4 x 4.

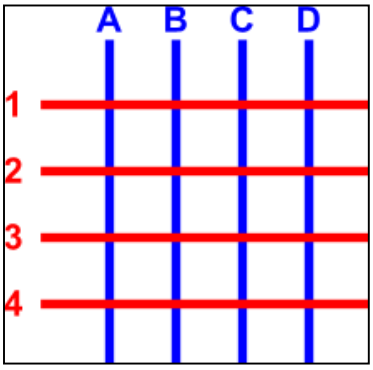


Fig. 1.14 Matriz 4x4. (Fuente: http://pcbheaven.com/wikipages/How_Key_Matrices_Works/)

Las líneas azules son las columnas y las líneas rojas las filas. Hay 16 nodos en donde las filas y columnas se cruzan. En los nodos donde se cruzan las filas y las columnas se conecta un botón del tipo push-button para hacer contacto. Cuando el operador pulsa este botón, se conectará la columna y la fila que corresponde. La figura 1.15 muestra los botones para conformar el teclado matricial.

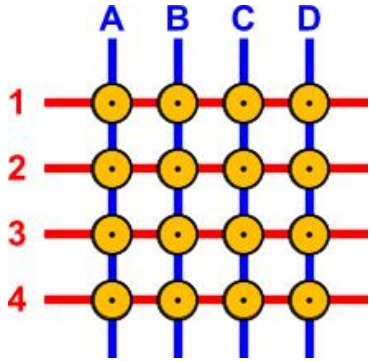


Fig. 1.15 Matriz 4x4. (Fuente: http://pcbheaven.com/wikipages/How_Key_Matrices_Works/)

Un microcontrolador se encarga de detectar e interpretar adecuadamente la combinación de fila/columna que corresponde a cada botón pulsado.

1.2.6.3 ADAFRUIT ULTIMATE GPS.

Es un módulo basado en el chipset MTK3339, un módulo GPS de alta calidad que puede rastrear hasta 22 satélites en 66 canales. Cuenta con un

excelente receptor de alta sensibilidad (-165 dB) y una antena cerámica incorporada. En alta velocidad, el dispositivo puede obtener datos hasta 10 veces por segundo.

El consumo de corriente es muy bajo, inferior a los 20 mA durante operación. Cuenta además con un regulador de 3.3V para que se pueda elegir la alimentación ya sea con 3.3V ó 5V DC. Cuenta con un LED de color rojo brillante, que indica el estado del rastreo de señal del GPS. Cuando el dispositivo inicia su operación, este LED se enciende cada segundo de forma intermitente, mientras está en busca de satélites, y una vez que obtiene coordenadas geográficas, el GPS entra en modo de ahorro de energía, y el LED se enciende una vez cada 15 segundos.

El modulo GPS también incluye un conector uFL para conectar una antena activa a 3V, en caso de ser necesario. El módulo detectará automáticamente la antena externa.

Este dispositivo también tiene la capacidad de registrar de datos en una memoria Flash interna de 64 KB. Si esta opción se encuentra activa, el GPS registrara hora, fecha, latitud, longitud y altura, cada 15 segundos, siempre y cuando estos datos se hayan recibido satisfactoriamente. El modulo puede almacenar aproximadamente 16 horas de datos.

CARACTERÍSTICA	DESCRIPCIÓN
Capacidad de satélites	22
Frecuencia de actualización	1 a 10 Hz
Precisión de posición	1.8 m
Precisión de velocidad	0.1 m/s
Tiempo de inicio	34 s
Potencia	-145 dBm
Altura máxima	27 000 m
Voltaje de entrada	3.3 a 5 V
Corriente nominal	20 mA

Tabla 1.9, Características del GPS.

La conexión con el Arduino se hace mediante el puerto serial, como se muestra en la figura 1.16, de forma que únicamente se utilizan 4 líneas para conectar el microcontrolador con el GPS.

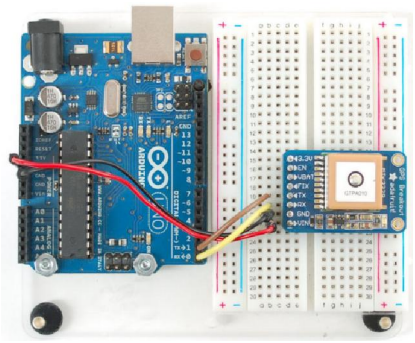


Fig. 1.16 Conexión de GPS con Arduino UNO. (Fuente: learn.adafruit.com/adafruit-ultimate-gps/direct-computer-wiring)

Los pines se conectan de la manera siguiente:

GPS	ARDUINO
Vin	5 V
GND	GND
Tx	Rx
Rx	Tx

Tabla 1.10, Conexiones del GPS.

1.3 PROTOCOLOS DE COMUNICACIÓN.

1.3.1 PROTOCOLO DEL BUS I2C.

El I2C (Inter Integrated Circuits) es un bus de comunicación serial síncrono de dos líneas que fue originalmente desarrollado por Philips Semiconductors (ahora nxp semiconductors) desde los inicios de los '80. Hoy es un estándar aceptado y respaldado por los fabricantes de dispositivos semiconductores.

El bus I2C permite la comunicación entre múltiples dispositivos (en teoría más de 1000), todos conectados paralelamente a las dos líneas que lo componen. Las transferencias de datos siempre se realizan entre dos dispositivos a la vez y en una relación maestro – esclavo.

Los dispositivos maestros son normalmente los microcontroladores y los dispositivos esclavos pueden ser memorias, conversores DAC y ADC, controladores de LCD, sensores de todos los tipos, etc.

Para que todos los dispositivos se puedan comunicar sin entorpecerse unos y otros, sin que haya pérdidas o colisiones en las transferencias de datos, sin que los dispositivos rápidos se desentiendan de los dispositivos lentos, etc., se deben de seguir ciertas reglas estándar.

Todas las especificaciones software y hardware del protocolo del bus I2C están descritas en el "I2C-bus specification and user manual". La figura 1.20 muestra una simplificación del estándar I2C.

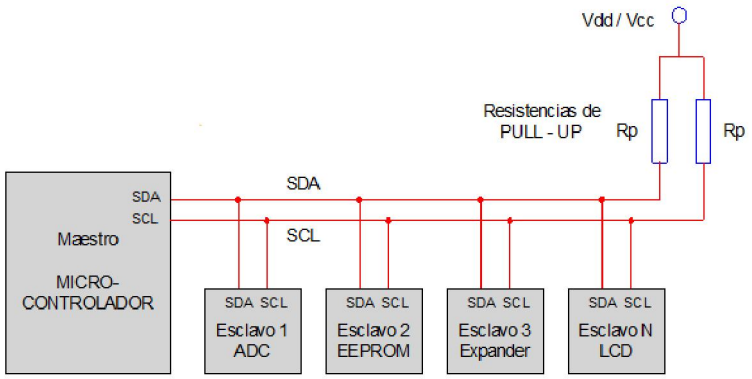


Fig. 1.17 Topología del Bus I2C. (Fuente: http://www.cursomicros.com/avr/bus-i2c/protocolo-bus-i2c.html)

Las transferencias de datos se llevan a cabo mediante dos líneas: línea serial de datos SDA y línea serial de reloj SCL. Ambas son bidireccionales. SDA se encarga de conducir los datos entre el dispositivo maestro y los esclavos. SCL es la señal de reloj que sincroniza los datos que viajan por la línea SDA.

El dispositivo maestro (microcontrolador) es quien siempre tiene la iniciativa de la comunicación, el maestro genera la señal de reloj y controla cuando se transmiten o reciben los datos.
Puede haber varios esclavos en la red I2C, pero el maestro solo se comunica con uno a la vez. Por eso cada dispositivo esclavo debe ser identificado por una dirección única.

1.3.1.1 TRANSFERENCIAS DE DATOS.

Los datos que se transfieren por el bus I2C deben viajar en forma de paquetes, aquí llamados transferencias. Como se ve en la figura 1.18, una transferencia empieza con un START y termina con un STOP. Entre estas señales van los datos propiamente dichos. Cada dato debe ser de 8 bits (1 byte) y debe ir seguido de un noveno bit, llamado bit de reconocimiento (ACK o NACK).

La transferencia de la figura 1.18 tiene dos bytes pero puede varios más (sin restricción) o puede haber un solo byte por paquete.
Los datos son transferidos por la línea SDA y son acompañados y sincronizados por los pulsos de reloj de la línea SCL. Para transmitir un bit primero hay que poner la línea SDA a 1 ó 0 según sea el caso, y luego colocar un pulso en la línea SCL.

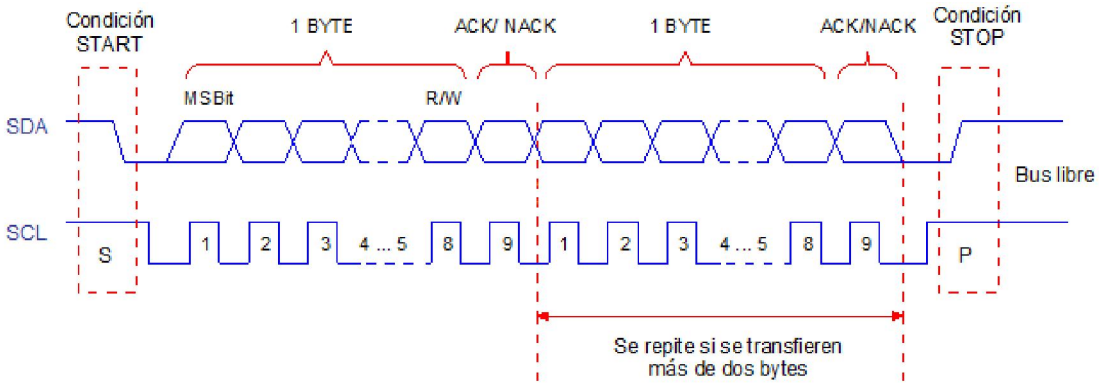


Fig. 1.18 Transferencias de datos sobre el bus I2C. (Fuente: http://www.cursomicros.com/avr/bus-i2c/protocolo-i2c-condicion-start-stop.html)

Los datos pueden viajar de ida y de vuelta por SDA sin colisionar porque es el maestro quien controla cuándo se transmite o recibe un dato. De ese modo, el control de SDA puede ser asumido tanto por el maestro como por el esclavo y ambos dispositivos podrán intercambiar los roles de transmisor o receptor. Eso sí, en cualquier caso, el control de la línea SCL siempre (excepto en el Clock Stretching) es asumido por el maestro.

1.3.1.2 CONDICIONES START, STOP Y START REPETIDA.

Los paquetes de datos transferidos por el bus I2C deben ir enmarcados por un Start y un Stop. Ambas señales son generadas por el maestro.

Una condición START es una transición de Alto a Bajo en la línea SDA cuando SCL está en Alto. Se le representa por la letra S. Después de Start el bus se considera ocupado.
Una condición STOP es una transición de Bajo a Alto en la línea SDA mientras SCL está en Alto. Está simbolizada por la letra P. Después de Stop las dos líneas están en Alto y el bus se considera libre. Se usa Stop para cerrar la transferencia de un paquete de datos o para abortar una transferencia previa que quedó truncada.
La señal de una condición START repetida es exactamente igual a la de START. La diferencia es de tipo "ocasional": aunque en principio cada transferencia debe ir enmarcada por un Start y un Stop, el estándar contempla la posibilidad de iniciar una nueva transferencia sobre una anterior que no ha sido cerrada con un Stop. El Start de la nueva transferencia se llama entonces Start Repetida y su símbolo es Rs.

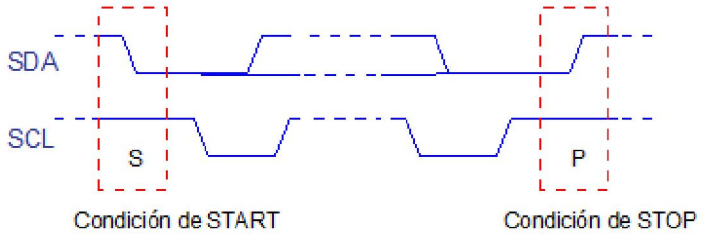


Fig. 1.19 Condiciones START y STOP. (Fuente: http://www.cursomicros.com/avr/bus-i2c/protocolo-i2c-condicion-start-stop.html)

1.3.1.3 EL BIT DE RECONOCIMIENTO (ACK O NACK).

Según las figuras mostradas, cada byte transferido debe ir seguido de un noveno bit, llamado "Acknowledge bit" (bit de reconocimiento, en inglés). Este bit siempre debe ser devuelto por el dispositivo receptor (maestro o esclavo) tras cada byte recibido.

Si el bit de reconocimiento es 0 significa que el dato fue reconocido y aceptado. Este bit se denomina ACK (Acknowledge).
Si el bit de reconocimiento es 1 significa que el dato recibido aún no es aceptado. Se usa este mecanismo para indicar que el receptor está ocupado realizando alguna tarea interna. Este bit se denomina NACK (Not Acknowledge).

1.3.1.4 EL BYTE DE CONTROL.

El primer byte de cada transferencia es conocido como byte de control. Los primeros 7 bits del byte de control contienen la dirección del esclavo con el cual se desea entablar la comunicación y el bit R/W establece si los siguientes bytes serán de lectura o escritura. Como siempre, R/W = 0 indica una escritura y R/W = 1 indica una lectura.

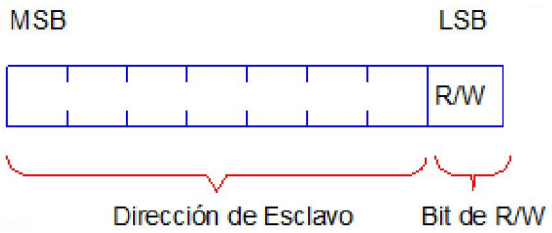


Fig. 1.20 Formato del byte de control (primer byte). (Fuente: http://www.cursomicros.com/avr/bus-i2c/protocolo-i2c-byte-de-control.html)

Todos los esclavos deben recibir el byte de control, pero solo el que halle su dirección en él será el que prosiga la comunicación. Los demás esclavos se deben mantener al margen hasta un nuevo aviso (otra condición de Start).

1.3.1.5 VELOCIDAD DE TRANSFERENCIA DE DATOS.

La velocidad de transferencia está determinada por la frecuencia de la señal de SCL. Por ejemplo, una velocidad de 100 kbits/s implica que cada bit se transmite en 1s/100k = 10μs, lo que significa que cada semiperiodo de la señal de reloj vale en promedio 5 μs. Estos datos se detallan en el Estándar I2C y también suelen ir indicados en los datasheets de los dispositivos I2C.

El estándar del bus I2C soporta cuatro modos de operación:
-Standard Mode, con una velocidad de hasta 100 kbit/s.
-Fast mode, con una velocidad de hasta 400 kbit/s.
-Fast mode plus, con una velocidad de hasta 1 Mbit/s.
-High-speed mode, con una velocidad de hasta 3.4 Mbit/s.
Los valores límites implican que los dispositivos más rápidos son compatibles con los dispositivos más lentos. Lo contrario, por supuesto, no es factible.
Hay dos versiones de direccionamiento en I2C: 7 y 8 bits. 7 bits identifican al dispositivo, y el octavo bit determina si se está leyendo o escribiendo. En Arduino se usan 7 bits de direccionamiento siempre. Si se tiene un dispositivo o un código que usa 8 bits de direccionamiento, se tendrá que renunciar al bit más bajo (además se debe desplazar el valor un bit a la derecha), cediendo una dirección entre 0 y 127.

1.3.2 COMUNICACIÓN SERIAL.

La comunicación serial es un protocolo muy común (no hay que confundirlo con el Bus Serial de Comunicación, o USB) para comunicación entre dispositivos que se incluye de manera estándar en prácticamente cualquier computadora. La mayoría de las computadoras incluyen dos puertos seriales RS-232. La comunicación serial es también un protocolo común utilizado por varios dispositivos para instrumentación; existen varios dispositivos compatibles con GPIB que incluyen un puerto RS-232. Además, la comunicación serial puede ser utilizada para adquisición de datos si se usa en conjunto con un dispositivo remoto de muestreo.

El concepto de comunicación serial es sencillo. El puerto serial envía y recibe bytes de información un bit a la vez. Aun y cuando esto es más lento que la comunicación en paralelo, que permite la transmisión de un byte completo por vez, este método de comunicación es más sencillo y puede alcanzar mayores distancias. Por ejemplo, la especificación IEEE 488 para la comunicación en paralelo determina que el largo del cable para el equipo no puede ser mayor a 20 metros, con no más de 2 metros entre cualesquier dos dispositivos; por el otro lado, utilizando comunicación serial el largo del cable puede llegar a los 1200 metros.

Típicamente, la comunicación serial se utiliza para transmitir datos en formato ASCII. Para realizar la comunicación se utilizan 3 líneas de transmisión: (GND) Tierra (o referencia), (Tx) Transmitir y (Rx) Recibir.

Debido a que la transmisión es asincrónica, es posible enviar datos por una línea mientras se reciben datos por otra. Existen otras líneas disponibles para realizar handshaking, o intercambio de pulsos de sincronización, pero no son requeridas. Las características más importantes de la comunicación serial son la velocidad de transmisión, los bits de datos, los bits de parada, y la paridad. Para que dos puertos se puedan comunicar, es necesario que las características sean iguales.

La comunicación serial en computadores ha seguido los estándares definidos en 1969 por el RS-232 (Recommended Standard 232) que establece niveles de voltaje, velocidad de transmisión de los datos, etc. Por ejemplo, este protocolo establece un nivel de -12v como un uno lógico y un nivel de voltaje de +12v como un cero lógico (por su parte, los microcontroladores emplean por lo general 5v como un uno lógico y 0v como un cero lógico).

Existen en la actualidad diferentes ejemplos de puertos que comunican información de manera serial (un bit a la vez). El conocido como “puerto serial” ha sido gradualmente reemplazado por el puerto USB (Universal Serial Bus) que permite mayor versatilidad en la conexión de múltiples dispositivos. Aunque en naturaleza serial, no suele referenciarse de esta

manera ya que sigue sus propios estándares y no los establecidos por el RS-232.

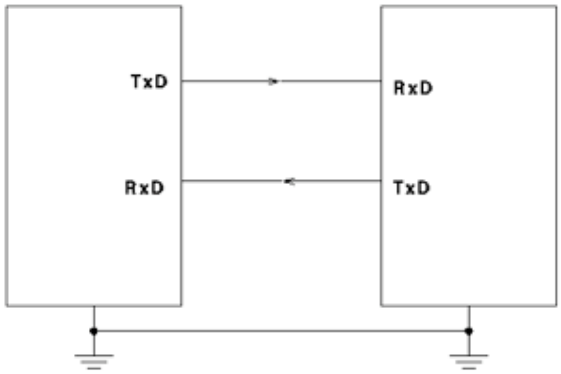


Fig. 1.21 Comunicación asincrónica. (Fuente: http://www.dtic.upf.edu/~jlozano/interfaces/interfaces6.html)

Algunas de las características de la comunicación serial son:

-Velocidad de transmisión (baud rate): Indica el número de bits por segundo que se transfieren, y se mide en baudios (bauds). Por ejemplo, 300 baudios representan 300 bits por segundo. Cuando se hace referencia a los ciclos de reloj se está hablando de la velocidad de transmisión. Las velocidades de transmisión más comunes para las líneas telefónicas son de 14400, 28800, y 33600. Es posible tener velocidades más altas, pero se reduciría la distancia máxima posible entre los dispositivos. Las altas velocidades se utilizan cuando los dispositivos se encuentran uno junto al otro, como es el caso de dispositivos GPIB.

-Bits de datos: Se refiere a la cantidad de bits en la transmisión. Cuando la computadora envía un paquete de información, el tamaño de ese paquete no necesariamente será de 8 bits. Las cantidades más comunes de bits por paquete son 5, 7 y 8 bits. El número de bits que se envía depende del tipo de información que se transfiere. Por ejemplo, el ASCII estándar tiene un rango de 0 a 127, es decir, utiliza 7 bits; para ASCII extendido es de 0 a 255, lo que utiliza 8 bits. Si el tipo de datos que se está transfiriendo es texto simple (ASCII estándar), entonces es suficiente con utilizar 7 bits por

paquete para la comunicación. Un paquete se refiere a una transferencia de byte, incluyendo los bits de inicio/parada, bits de datos, y paridad. Debido a que el número actual de bits depende del protocolo que se seleccione, el término paquete se usar para referirse a todos los casos.

-Bits de parada: Usado para indicar el fin de la comunicación de un solo paquete. Los valores típicos son 1, 1.5 o 2 bits. Debido a la manera como se transfiere la información a través de las líneas de comunicación y que cada dispositivo tiene su propio reloj, es posible que los dos dispositivos no estén sincronizados. Por lo tanto, los bits de parada no sólo indican el fin de la transmisión sino además dan un margen de tolerancia para esa diferencia de los relojes. Mientras más bits de parada se usen, mayor será la tolerancia a la sincronía de los relojes, sin embargo la transmisión será más lenta.

-Paridad: Es una forma sencilla de verificar si hay errores en la transmisión serial. Existen cuatro tipos de paridad: par, impar, marcada y espaciada. La opción de no usar paridad alguna también está disponible. Para paridad par e impar, el puerto serial fijará el bit de paridad (el último bit después de los bits de datos) a un valor para asegurarse que la transmisión tenga un número par o impar de bits en estado alto lógico. La paridad marcada y espaciada en realidad no verifican el estado de los bits de datos, simplemente se fija el bit de paridad en estado lógico alto para la paridad marcada, y en estado lógico bajo para la paridad espaciada. Esto permite al dispositivo receptor conocer de antemano el estado de un bit, lo que serviría para determinar si hay ruido que esté afectando de manera negativa la transmisión de los datos, o si los relojes de los dispositivos no están sincronizados.

1.3.2.1 COMUNICACIÓN SERIAL ARDUINO.

Se utiliza para la comunicación entre la placa Arduino y un ordenador u otros dispositivos. Todas las placas Arduino tienen al menos un puerto serial (también conocido como UART o USART). Se comunica a través de los pines

digitales 0 (RX) y 1 (TX), y con la computadora mediante USB. Por lo tanto, si se utilizan estas funciones, no se pueden usar los pines 0 y 1 como entradas o salidas digitales.

Se Puede utilizar el monitor del puerto serie incorporado en el entorno Arduino para comunicarse con la placa Arduino. Accediendo a la opción de monitor de puerto serie en la barra de herramientas y seleccionando la misma velocidad en baudios que se haya especificado en el código.

La placa Arduino Mega tiene tres puertos adicionales de serie: Serial1 en los pines 19 (RX) y 18 (TX), Serial2 en los pines 17 (RX) y 16 (TX), Serial3 en los pines 15 (RX) y 14 (TX).

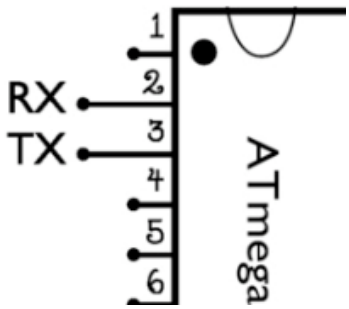


Fig 1.22 Pines encargados de la comunicación serial en los microcontroladores Atmega8, Atmega 168 y Atmega328. (Fuente: http://galaxi0.wordpress.com/el-puerto-serial)

1.3.3 COMUNICACIÓN SPI.

SPI (Serial Peripheral Interface) es básicamente un bus de comunicación a nivel de circuitos integrados. La transmisión de datos se realiza en forma serial.
El bus SPI se define mediante 4 pines:

-SCLK o SCK : Señal de reloj del bus. Esta señal rige la velocidad a la que se transmite cada bit.

-MISO(Master Input Slave Output): Transmisión de datos desde el Slave (Esclavo) hacia el dispositivo Maestro.

-MOSI(Master Output Slave Input): Transmisión de datos desde el Master (Maestro) hacia el dispositivo esclavo.

-SS o CS: Chip Select o Slave Select, habilita el dispositivo hacia el que se envían los datos. Esta señal es opcional y en algunos casos no se usa.

El bus SPI se comporta como un shift register donde a cada golpe de clock se captura un bit. En parte no es necesario hacer un direccionamiento de los chips ya que mediante la señal Chip select, se habilita al integrado al que queremos enviar los datos.

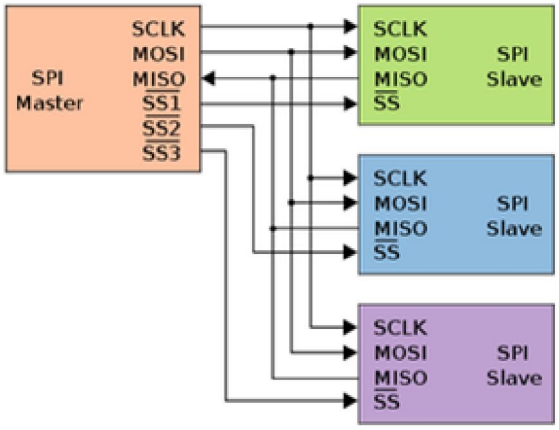


Fig 1.23 Imagen de la conexión SPI. (Fuente: http://es.wikipedia.org/wiki/Serial_Peripheral_Interface)

El funcionamiento para un envío de un Master es el siguiente:

-Se habilita el chip al que hay que enviar la información mediante el CS (Opcional).

-Se carga en el buffer de salida el byte a enviar.

-La línea de Clock empieza a generar la señal cuadrada donde normalmente por cada flanco de bajada se pone un bit en MOSI.

-El receptor normalmente en cada flanco de subida captura el bit de la línea MISO y lo incorpora en el buffer.

-Se repite el proceso 8 veces y se ha transmitido un byte. Si se ha terminado de transmitir se vuelve a poner la línea CS en reposo. Hay que tener en cuenta que a la vez que el Master está enviando un dato también lo recibe así que si el Slave ha depositado algún byte en el buffer de salida, este también será enviado y recibido por el Master.

CAPITULO 2: METODOLOGIA DE DISEÑO.

2.1 DEFINICIÓN DE LA ARQUITECTURA PRINCIPAL DEL SISTEMA.

Debido a la practicidad, modularidad y amplia bibliografía disponible para los dispositivos de la familia Arduino, esta es la tecnología elegida para construir el instrumento.

La complejidad del sistema y la cantidad de sensores requieren una gran cantidad de memoria, tanto de almacenamiento de código como de RAM operativa, por este motivo, se ha elegido el modulo Arduino Mega 2560 R3 como la placa principal del sistema. Este se encargara de la lectura de los sensores, el funcionamiento del servidor y cualquier otra función necesaria para la operación del instrumento.

2.2 ELECCIÓN DE SENSORES.

Para los parámetros a medir que se han planteado, se han elegido los siguientes sensores, teniendo en cuenta la disponibilidad de los mismos a través de diferentes distribuidores, así como la disponibilidad de librerías y/o drivers para su implementación. Estos sensores ya vienen calibrados de fábrica y generan señales digitales para facilitar su lectura a través de un microcontrolador, son:

-DHT22: Para obtener datos de humedad y temperatura.
-BMP05: Para obtener Presión Atmosférica.

Aparte, para registrar velocidad y dirección del viento, se utiliza:

-Anemómetro y Veleta Davis: De origen propietario, las funcionalidades de estos dispositivos se han portado a un entorno de hardware libre para poder obtener datos confiables.

2.3 OTROS DISPOSITIVOS.

El sistema utiliza otros dispositivos de apoyo para poder cumplir con todas sus funcionalidades:

-Modulo Ethernet R3: Provee la capacidad al sistema de operar como un servidor web, además incluye el modulo para poder conectar una memoria microSD para respaldo de datos.

-RTC: el reloj de tiempo real mantiene un registro constante del tiempo, el dispositivo utilizado es el DS1307, un modelo ampliamente difundido y con gran respaldo de controladores y mucha bibliografía.

-GPS: Para poder obtener de forma confiable la fecha y la hora, además de otros parámetros de localización. Ya Incorpora una antena cerámica, lo que evita el inconveniente de montar una antena externa, pero siempre queda la opción de poder conectar una en caso de ser necesario.

-LCD: Para control, configuración y visualización de datos. La pantalla utilizada es de 20 caracteres x 4 filas, El modulo LCD se conecta a través del bus I2C, pero también puede conectarse vía serial. Además cuenta con un controlador de teclado, evitándole esa función a la placa principal.

-Teclado: Modelo sencillo de 4 filas por 4 columnas.

La tabla 2.1 muestra un resumen de los elementos con los que se implementará el sistema, junto a sus principales características y limitaciones.

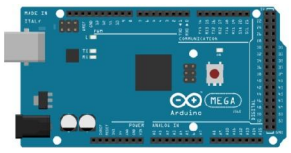
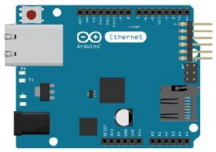
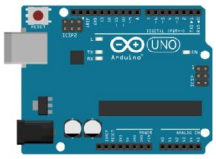
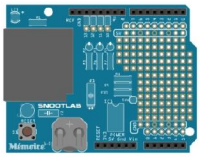
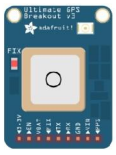
Elemento	Nombre	Función Principal	Características	Limita-ciones
	Arduino MEGA.	Modulo principal del sistema.	-Pines E/S digitales: 54. - Pines de entrada analógica: 16. -Memoria Flash: 256 KB. -Memoria RAM: 8KB.	No se observo.
	Ethernet Shield.	Servidor web del sistema y soporte de tarjeta microSD.	-Conexión SPI. -Incluye interfaz para microSD.	4 clientes como máximo.
	Arduino UNO.	Lectura del anemómetro y transmisión de la misma hacia el Arduino Mega.	-Pines E/S digitales: 14. - Pines de entrada analógica: 6. -Memoria Flash: 32 KB. -Memoria RAM: 2 KB.	Únicamen-te transmite datos enteros.
	RTC DS1307.	Proporciona fecha y hora.	-Conexión I2C. - Precisión: +/- 5%.	Únicamen-te se pueden programar fechas desde el año 2000
	Módulo GPS.	Proporciona fecha y hora para sincronización del RTC, además de coordenadas geográficas.	-Precisión: 2 a 10 m. -Hasta 22 satélites. - Frecuencia de actualización: 1 a 10 Hz.	Debe colocarse en una ubicación al aire libre.

Tabla 2.1 Elementos que conforman la estación meteorológica.

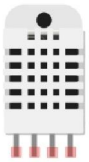
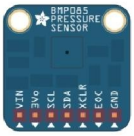
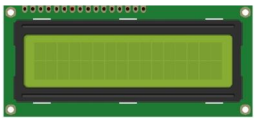
Elemento	Nombre	Función Principal	Características	Limita-ciones
	DHT22.	Sensor de Humedad y Temperatura.	Humedad: -Rango : 0 a 100 %. -Precisión: +/-2% hasta +/- 5% max. Temperatura: -Rango : -40°C a 80°C. -Precisión: +/- 0.5° C	Necesita 2 segundos entre cada lectura.
	BMP085.	Sensor de Presión Atmosférica.	-Rango de Operación: 300-1100 hPa. -Precisión : +/- 2.5 hPa.	No se observo.
	Anemó-metro y Veleta.	Sensor de Velocidad y Dirección del Viento.	Anemómetro: -Rango : 1 a 80 m/s. -Precisión : +/- 5%. Veleta: -Rango: 0° a 360°. -Precisión : +/- 7°.	Requiere 3 segundos entre cada lectura del anemóm etro.
	Pantall a LCD con controla -dor de teclado.	Medio para establecer interfaz de control del usuario.	-Conexión I2C/Serial. -Buffer 48 bytes. -Puerto para control de teclado. -4 filas x 20 col.	No se observo.
	Teclado .	Para ingreso de datos y configuración del instrumento.	4 Filas x 4 Columnas.	No se observo.

Tabla 2.1 (Cont.) Elementos que conforman la estación meteorológica.

La figura 2.1 muestra un esquema de la conexión completa de todos los componentes.

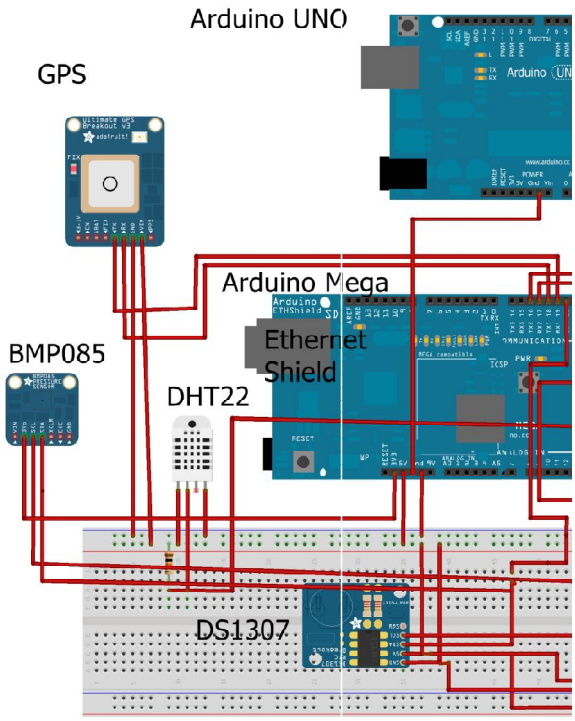


Figura 2.1 Esquema de conexiones de los elementos.

2.4 METODOS DE COMUNICACIÓN.

Para optimizar el uso de las terminales del microcontrolador, se ha tratado de utilizar los protocolos de comunicación que permiten hacer uso de varios dispositivos sobre un mismo bus de datos. También se ha hecho uso intensivo de los puertos seriales disponibles del Arduino Mega.

2.4.1 BUS SERIAL.

El Arduino Mega cuenta con 4 buses seriales, de los cuales se utilizan 3 de la siguiente manera:

- Puerto 0: Es el puerto por defecto utilizado en la interfaz del compilador, y se utiliza para efectos de control y debugging de parte del usuario.
- Puerto 1: Para recibir los datos del anemómetro.

- Puerto 3: Para recibir la información del GPS.

2.4.2 BUS I2C.

Este bus es utilizado por los siguientes dispositivos:

- Sensor de Presión Atmosférica BMP05.
- Modulo LCD/Teclado.
- RTC DS1307.

2.4.3 BUS SPI.

En este bus se conectan:
-Modulo Ethernet.
-Memoria microSD.

2.4.4 OTRAS CONEXIONES.

Entonces, los dispositivos restantes se conectan de la siguiente manera:

- Veleta, en una de las 16 entradas/salidas analógicas.
- Sensor de Humedad y Temperatura DHT22, en una de las 42 entradas/salidas digitales que quedan disponibles (Las restantes 12 son utilizadas por los buses de comunicación).

Por lo que queda un amplio margen para añadir más dispositivos en una futura expansión del sistema, rango que es aún mayor si se tienen en cuenta las facilidades de los buses I2C y SPI. Si esto no fuera suficiente, También es posible utilizar las entradas/salidas que quedan en el Arduino Uno del anemómetro, 6 analógicas y 11 digitales, teniendo en cuenta que cualquier operación programada se realizara cada 3 segundos.

2.5 ACONDICIONAMIENTO DE SEÑALES.

El anemómetro y la Veleta son los únicos dispositivos que no cuentan con una salida digital. Para obtener las señales respectivas se utiliza el circuito mostrado en la figura 2.2.

Internamente, el anemómetro es un interruptor magnético, la resistencia de 4.7K Ohms es para mantener el interruptor en estado alto ("pull up"). Cada vez que las copas giran y completan una vuelta el interruptor se cierra, ese evento se registra en una de las entradas digitales del Arduino Uno. El capacitor de 22nF es para evitar falsas lecturas por el rebote cuando se cierra el interruptor.

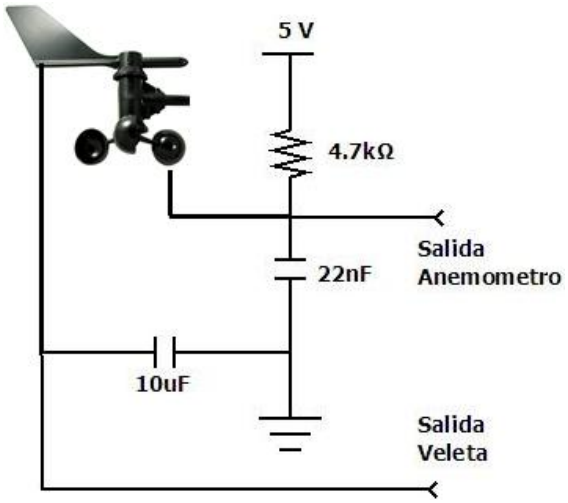


Figura 2.2 Circuito para acondicionamiento de las señales del Anemómetro/Veleta.

La Veleta es un potenciómetro cuyo valor actual (Dirección del viento) se lee en una entrada analógica del Arduino Mega. El Capacitor de 10uF es para mejorar la definición de la señal de salida de la Veleta.

La otra pieza de circuitería necesaria es una resistencia de 10K Ohms, que se utiliza para mantener la salida digital del sensor DHT22 en nivel alto.

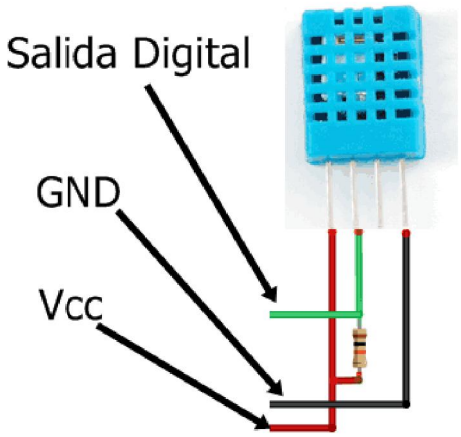


Figura 2.3 Conexión del DHT22.

2.6 HERRAMIENTAS DE DESARROLLO.

Para desarrollar el código se ha utilizado el entorno de compilación oficial del proyecto Arduino, específicamente la versión 1.0.3.

Este software tiene acceso al puerto serial principal de la placa que se encuentre conectada a la computadora, lo que permite facilitar el proceso de debugging y llevar un control sobre el flujo del programa.

También facilita la carga de códigos en las placas, utilizando el puerto USB para dicha función.

Los programas (O sketches, en el argot utilizado por esta comunidad de desarrolladores) se componen de 2 secciones principales:

- Setup, en el que comúnmente se inicializan los objetos y cualquier otro procedimiento necesario para el desempeño del programa, y es la primera función que se ejecuta cuando se inicia el flujo del programa.
- Loop, que contiene la sección de código que se ejecutara continuamente hasta que el microcontrolador sea reiniciado.

Queda a discreción del programador si es necesario añadir otras funciones, pero estas 2 siempre deben incluirse en un sketch para dispositivos Arduino.

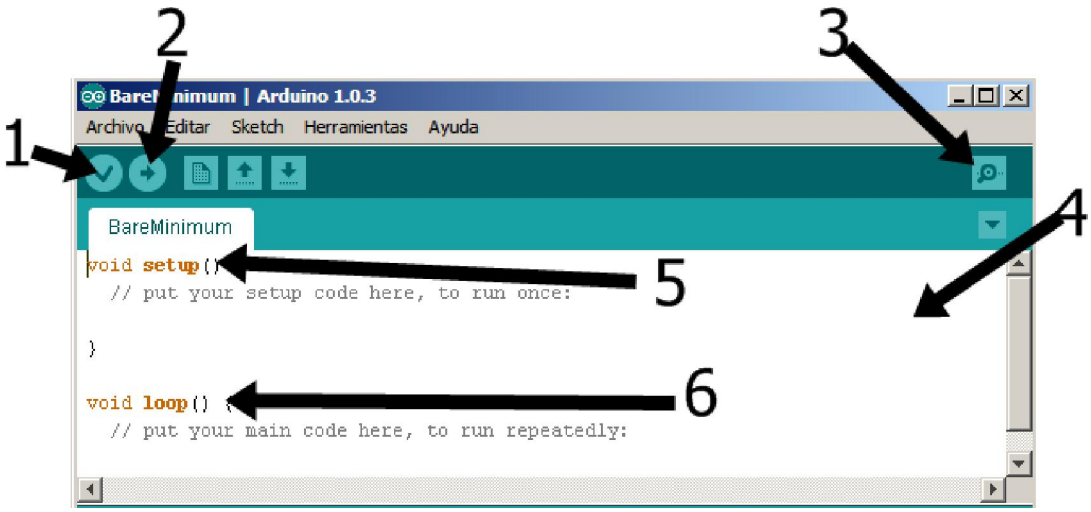


Figura 2.4 Entorno de desarrollo Arduino.

La figura 2.4 muestra la ubicación de las características y opciones más importantes del entorno de programación, en donde:

1-Compila el código actual.
2-Compila el código, y si no arroja ningún error, lo carga en la placa que esté conectada en ese momento a la computadora.
3-Acceso al puerto serial de la placa actualmente en uso.
4-Area para escribir el código.
5-Función Setup del código actual.
6-Función Void del código actual.

Las funciones básicas como el manejo sencillo de strings, comunicación serial, algunos procedimientos matemáticos, etc. se incluyen por defecto, y no es necesario declarar ninguna librería. Sin embargo, de ser necesario, el compilador permite el uso de las librerías C/C++ soportadas por AVR.

2.7 CONSIDERACIONES DE PROGRAMACIÓN Y UTILIZACIÓN DE HARDWARE.

2.7.1 COMUNICACIÓN CON PERIFÉRICOS.

La mayoría de los dispositivos del instrumento utilizan algún protocolo de comunicación con el Arduino Mega, en lugar de conectarse y ser accesados directamente. Para realizar estos procesos, se utilizan las siguientes librerías:

2.7.1.1 LIBRERÍA WIRE.H

Para utilizar el bus I2C. Debe declararse e inicializarse dentro del Setup.

```
#include <Wire.h>
Wire.begin(); /// Dentro del Setup
```

2.7.1.2 LIBRERÍA SPI.H.

Para utilizar el bus SPI.

```
#include <SPI.h>
```

Después de la inclusión de la librería, las conexiones del bus se asignan automáticamente a los pines del Arduino Mega: 50(MISO), 51(MOSI), 52(SCK) y 53(SS). El Shield Ethernet R3 accede a estos pines a través del conector ICSP. Para utilizar también la memoria SD, simplemente se declara el pin SS como salida, dentro del setup.

```
pinMode(SS, OUTPUT);
```

2.7.1.3 LIBRERÍA SOFTWARESERIAL.H .

Para definir uno o más puertos seriales diferentes a los ya existentes, no se utilizara y se incluye únicamente por dependencias del GPS.

2.7.1.4 LIBRERÍA EASYTRANSFER.H

La comunicación Serial con el Arduino UNO conectado al anemómetro se realiza a través de esta librería.

Después de incluir la librería, hay que declarar un objeto de la misma, y definir una estructura que contendrá las variables que se transmitirán/recibirán a través del bus Serial, en este caso será únicamente una variable, "vel", que contendrá el valor de la velocidad del viento. Luego hay que asignarle un nombre a esta estructura, en este caso será "mydata".

```
#include <EasyTransfer.h>
EasyTransfer ET;
struct SEND_DATA_STRUCTURE{
  int vel;
};
SEND_DATA_STRUCTURE mydata;
```

La definición de las variables dentro de la estructura debe ser igual en los 2 dispositivos para que la comunicación funcione.
En el transmisor, el Arduino UNO, el objeto ET debe ser inicializado y asignado al puerto Serial, dentro del setup.

```
ET.begin(details(mydata), &Serial);
```

Luego en el loop, se le asigna el valor a transmitir a la variable vel, y se envía a través del puerto Serial con el comando sendData.

```
mydata.vel = int(RPM);
ET.sendData();
```

En el lado del receptor, el Arduino Mega monitorea el puerto serial asignado al Arduino UNO mediante la función ET.receiveData(), de haber un dato disponible, se asigna a una variable para poder ser utilizado por el programa según convenga.

```
if(ET.receiveData()){
v=mydata.vel;
}
```

2.7.1.5 BUS SERIAL.

No es necesario incluir ninguna librería extra, pero debe inicializarse dentro del setup, especificando el número del puerto y la velocidad a la que trabajará.

```
Serial.begin(9600); // Debugging y flujo del programa.
Serial1.begin(9600); // Arduino UNO.
Serial3.begin(115200); // GPS
```

2.7.2 SENSORES.

El uso de las respectivas librerías facilita el proceso de obtención de datos en los dispositivos de salida digital. Para poder utilizar los dispositivos, primero se deben declarar las librerías al inicio del fichero.

```
#include <Adafruit_BMP085.h>
#include "DHT.h"
```

Luego hay que crear el objeto con el que se trabajaran las variables de cada dispositivo:

```
Adafruit_BMP085 bmp;
#define DHTTYPE DHT22
#define DHTPIN 48
DHT dht(DHTPIN, DHTTYPE);
```

En la parte del setup, estos objetos deben inicializarse:

```
bmp.begin();
dht.begin();
```

Finalmente, en el loop principal, basta con acceder a las funciones predefinidas de la librería para obtener los datos:

```
t = dht.readTemperature();
h = dht.readHumidity();
p = bmp.readPressure();
```

El sensor de presión es capaz de entregar lecturas con intervalos de tiempo en el orden de los milisegundos, pero el de humedad/temperatura es considerablemente más lento, necesitando un intervalo de 2 segundos para poder entregar cada lectura.

La Veleta y el Anemómetro no cuentan con librerías propias de carácter libre, por lo que se monitorea el estado de los sensores mediante las entradas del Arduino.

El cálculo de la velocidad del viento se realiza mediante la siguiente formula, que se obtiene de la datasheet del anemómetro, que es proporcionada por el fabricante:

$$V = P\left(\frac{2.25}{T}\right)$$

Donde:

V=Velocidad del viento, en millas por hora.

P=Conteo de vueltas.

T=Intervalo de tiempo sobre el que se cuentan las vueltas.

Tradicionalmente, el tiempo usado en los anemómetros para realizar esta medición es de 3 segundos, para garantizar que la cantidad de muestras verdaderamente sea representativa de la velocidad a determinar.

La salida del anemómetro se conecta al pin 2 (Usado para interrupciones externas) de un Arduino UNO destinado únicamente a obtener la velocidad del viento, la operación primaria de este Arduino consiste en que, en cada loop, el Microcontrolador habilitara las interrupciones durante 3 segundos, con la función sei().

Para definir la interrupción 0, asignada al pin 2 del Arduino, dentro del setup se declara el pin digital 2 como entrada, y se utiliza la función attachInterrupt, la cual asocia la ejecución de una función ("rpm") a la interrupción producida cuando el pin 2 cambia de estado, en este caso la interrupción se producirá durante el paso de nivel digital alto hacia el nivel bajo.

```
pinMode(RPMsensor, INPUT);
attachInterrupt(0, rpm, FALLING);
```

El anemómetro consiste básicamente en un interruptor magnético que se mantiene por defecto en estado alto (Pull Up). Cuando una vuelta de las copas se completa, el interruptor se cierra, ese cambio de estado de Alto a Bajo genera una interrupción en el Arduino, y se incrementa la variable que lleva el conteo de las vueltas. Después de 3 segundos, las interrupciones se inhabilitan con la función cli(), y la variable respectiva (RPMTops) contiene el número total de vueltas obtenidas. Este conteo de vueltas se pone a cero en cada ciclo que cumple el loop, para obtener un nuevo valor de velocidad.

```
RPMTops = 0;
sei();
delay (3000);
cli();
```

El valor obtenido, originalmente un flotante, se trunca mediante una conversión a valor entero, y se envía al Arduino Mega por medio del bus Serial. Esta conversión se efectúa debido a que, para transmitir el dato, este debe ser un valor entero, ya que esa es la limitación impuesta por la librería utilizada para efectuar la comunicación entre los módulos Uno y Mega, que sin embargo provee otras ventajas, ya que evita el uso explicito de cualquier protocolo de handshaking, dejando esa función en manos de la librería. Además se asegura que el dato se transmitirá y recibirá correctamente, debido a que incorpora un método de checksum para verificar la integridad del dato. Si el dato recibido pasa la prueba, queda disponible para que el

programa pueda acceder a él mediante una estructura previamente definida. Finalmente, el dato recibido en el Mega se convierte a las unidades respectivas (m/s).

Para obtener la dirección del viento, la salida de la Veleta se conecta a una entrada analógica en el Arduino Mega, estas entradas son convertidores analógico-digitales con 10 bits de resolución. Entonces, se lee el estado del pin conectado a la Veleta.

Esencialmente, la Veleta es un potenciómetro. El rango de este potenciómetro se mapea desde 0 hasta 1023, ya que ese es el numero de valores posibles que provee la resolución del ADC interno del Arduino. El valor cero se utiliza como referencia para ubicar el Norte. A partir de ahí, los demás valores se mapean y se les asigna el valor de ubicación respectivo según el desplazamiento del potenciómetro, incrementándose estos valores en sentido horario.

```
PotValue = analogRead(PotPin);
Direction = map(PotValue, 0, 1023, 0, 359);
Direction = Direction + DirCorr + 3;
```

2.7.3 GPS.

Las librerías a incluir son las siguientes:

```
#include <Adafruit_GPS.h>

#include <SoftwareSerial.h>
```

Luego se declara el objeto GPS, especificando el puerto serial en el que será leído, también se define un valor booleano que se utiliza en las funciones de lectura propias del GPS.

```
Adafruit_GPS GPS(&Serial1);
#define GPSECHO  true
```

Dentro del setup, se inicializa el objeto, y además se especifican parámetros del comportamiento del dispositivo.

```
GPS.begin(9600);
GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);
GPS.sendCommand(PGCMD_ANTENNA);
```

El proceso de lectura del GPS es un poco más complejo y se ha enmascarado en 2 funciones, LlamarGPS y LeerGPS.

```
if (GPS.newNMEAreceived()) {
    if (!GPS.parse(GPS.lastNMEA()))
      return;
  }
```

Las líneas anteriores se incluyen debido a que, para que el GPS funcione adecuadamente, debe ser leído a alta velocidad y de forma dedicada. Una vez que el GPS obtiene una trama de datos, el Arduino Mega debe interpretarlos correctamente ("parse"), si el proceso no se realiza suficientemente rápido, los datos se pierden. Combinar la lectura del GPS con procesos muy pesados ralentiza la obtención de los parámetros y se vuelve difícil que el dispositivo entregue datos coherentes, por este motivo, el GPS únicamente se lee en cada reinicio del instrumento, y en cada cambio de día al crear un nuevo archivo.

Si es posible obtener una trama de datos, e interpretarlos correctamente, entonces acceder a los mismos se reduce a interrogar directamente al modulo por los datos que se requieren.

```
Serial.print(GPS.hour, DEC);
Serial.print(GPS.minute, DEC);
Serial.print(GPS.seconds, DEC);
Serial.println(GPS.milliseconds);
Serial.print(GPS.day, DEC);
Serial.print(GPS.month, DEC);
Serial.println(GPS.year, DEC);
Serial.print(GPS.latitude, 4);
Serial.print(GPS.lat);
Serial.print(GPS.longitude, 4);
Serial.println(GPS.lon);
```

La operación del GPS puede monitorearse a través de la conexión serial en el entorno de programación. En el peor de los casos, todos los datos que se obtengan serán cero, situación que indica que la lectura no es constante, está siendo ralentizada o las conexiones son erróneas.

En operación normal y al aire libre, el dispositivo obtiene prácticamente de inmediato la hora UTC, y después la fecha actual, obtener la trama de las coordenadas y demás datos de localización, es un proceso que puede tardar de 1 a 5 minutos. Una vez que se obtienen satisfactoriamente, el GPS cambia a Verdadero el valor de su variable booleana "fix". Este hecho es el que se ha aprovechado en el código para garantizar que los datos proporcionados sean reales, coherentes y acordes a lo que se espera.

Una vez que el "fix" se cumple, la rutina de lectura es interrumpida y con los datos obtenidos, se programa el reloj y las coordenadas geográficas, de ser necesario.

Aparte de los parámetros anteriores, el modulo GPS también proporciona la Altitud actual, en metros sobre el nivel del mar.

2.7.4 MODULO ETHERNET.

El dispositivo Ethernet R3 se utiliza como servidor web, para poder visualizar los datos obtenidos en tiempo real según el intervalo definido para las mediciones, y también para acceder a los datos recolectados sin necesidad de retirar la memoria microSD del instrumento. Se accede a través de la interfaz SPI, utilizando el pin 10 para selección del dispositivo.

El código básico utilizado para el servidor es una modificación del código que oficialmente es provisto por el proyecto Arduino, añadiéndole las funcionalidades necesarias para la publicación de una tabla de datos, los cuales se van actualizando a medida que se realizan las mediciones.

```
#include <Ethernet.h>
#include <SPI.h>
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
byte ip[] = { 192, 168, 0, 177 };
byte gateway[] = {10, 0, 0, 1};
EthernetClient client;
EthernetServer server(80);
```

Después de declarar las librerías necesarias, Ethernet y SPI (para comunicación con el Mega) se le asignan al modulo una dirección MAC, una dirección IP, un Gateway, se declara un objeto cliente y se establece el servidor en el puerto 80.

Luego, dentro del setup, hay que iniciar el modulo, asociándolo con las MAC e IP definidas previamente, y después iniciar el modo servidor.

```
Ethernet.begin(mac, ip, gateway);
server.begin();
```

En el loop, el modulo esperara hasta que haya un cliente solicitando conexión con el servidor. Cuando este evento se produce, el Modulo comienza a enviar líneas de código HTML hacia el web browser del cliente.

```
EthernetClient client = server.available();
  if (client) {
    boolean current_line_is_blank = true;
    index = 0;
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        if (c != '\n' && c != '\r') {
          clientline[index] = c;
          index++;
          if (index >= BUFSIZ)
            index = BUFSIZ -1;
                  continue;
        }
clientline[index] = 0;
Serial.println(clientline);
if (strstr(clientline, "GET / ") != 0) {
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println();
client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.println("<meta                http-equiv=\"refresh\"
content=\"5\">");
```

Al final de esas líneas, el modulo le indica al browser que se mantenga efectuando una nueva conexión cada 5 segundos.

La tabla de datos se genera con simples tags HTML enviados hacia el browser del cliente, junto a las variables medidas, las cuales se actualizan cuando se cumple el periodo de medición que se ha definido.

Por ejemplo, para mostrar el encabezado, y la fecha, tenemos:

```
client.println("<br />");
client.println("<b>");
client.println("<big>");
client.println("<big>");
client.println("Estación Meteorológica");
client.println("</big>");
client.println("</big>");
client.println("</b>");
client.println("<br />");
client.println("<table border='10'>");
client.println("<tr>");
client.println("<td>");
client.println("Fecha");
client.println("</td>");
client.println("<td>");
client.println(now.year(), DEC);
client.print('/');
client.print(now.month(), DEC);
client.print('/');
client.print(now.day(), DEC);
client.println("</td>");
client.println("</tr>");
```

Aparte de mostrar los valores de los parámetros que se están sensando, también se ha incorporado la capacidad de presentar los archivos que se encuentran en la memoria microSD, y poder acceder a ellos para presentarlos a través del navegador de internet.

Finalmente, se ha agregado la capacidad de cambiar la dirección IP por la que sea necesaria según los requerimientos de la red que se utilice.

2.7.5 MEMORIA MICRO SD.

El modulo Ethernet incorpora un slot para poder utilizar una memoria microSD, la cual se accede a través del mismo bus SPI utilizado para el modulo Ethernet.

Las memorias microSD utilizan el sistema de archivos FAT32. Para el instrumento, la consecuencia más importante de esto es que los nombres de archivos quedan limitados al formato 8.3, es decir, un nombre de hasta 8 caracteres, y una extensión de 3 caracteres como máximo.

Debido a que la librería oficialmente soportada por el proyecto Arduino para la interacción con memorias microSD (SD.h) presenta algunas limitaciones (En cuanto a bugs en la interacción con otros dispositivos, en algunas funcionalidades y en la velocidad de operación de la microSD), se ha decidido utilizar en su lugar la librería SdFat, siendo esta la librería original de la cual se deriva la SD. De esta manera se evitan las limitaciones anteriores y se aprovechan nuevas funcionalidades, con el precio de incrementar un poco la complejidad del código.

Para utilizar la librería, primero hay que hacer las declaraciones respectivas:

```
#include <SdFat.h>
#include <SdFatUtil.h>
Después, declarar los objetos que se utilizaran para
acceder a la microSD.
Sd2Card card;
SdVolume volume;
SdFile root;
SdFile file;
```

Dentro del setup, hay que inicializar estos objetos. La inicialización de la tarjeta indica que se operara a velocidad media, y se utilizara el pin 4 para selección del dispositivo.

```
card.init(SPI_HALF_SPEED, 4);
volume.init(&card);
root.openRoot(&volume);
file.open(&root, nombre, O_CREATE | O_APPEND | O_WRITE);
```

La última declaración especifica primero, que en el directorio raíz de la tarjeta, se creara un archivo que llevara por nombre lo que sea que este contenido en la variable "nombre", de tipo cadena de caracteres, siempre y cuando se cumpla con el standard 8.3, de no ser así, automáticamente la librería recorta el nombre, convirtiendo a mayúsculas, añadiendo una virgulilla y numerando el archivo al final.

Para evitar cualquier problema de este tipo, el formato de nombre utilizado en el instrumento consiste en la fecha actual, en forma numérica, en el formato: año-mes-día, de manera que el nombre nunca va a sobrepasar los 8 caracteres. La extensión utilizada para el archivo de salida es .CSV, ya que el formato para guardar los datos será el de valores separados por comas.

Los 3 comandos al final de la última declaración indican que: Si el archivo no existe, se creara uno nuevo. Si ya existe, se continuara su uso desde el final, y además se abrirá en forma de escritura, para guardar datos.
Una vez definidos estos parámetros, el archivo puede ser manipulado a través del objeto "file", y las operaciones se reducen a sentencias simples, por ejemplo:

```
file.print("Fecha");
file.println();
file.close();
```

Para escribir la palabra "Fecha", para indicar fin de línea y pasar a la siguiente, y para cerrar el archivo, respectivamente. A pesar de que se invoque el comando para escribir, los datos se guardan de manera física únicamente cuando se ha alcanzado la capacidad del cluster definido para el sistema de archivos FAT32 (512 bytes), o cuando específicamente se cierra el archivo. Por tal razón, el proceso a seguir es obtener y/o calcular los datos, una vez listos, abrir el archivo, escribir en él y cerrar inmediatamente. Este procedimiento contribuye a mantener la integridad del archivo y evitar que

haya pérdidas de datos. El proceso se repite al cumplirse cada intervalo de medición de sensores.

La librería SdFat también proporciona una función bastante útil para el control de flujo del programa. La sentencia PgmPrint("mensaje"), envía al puerto serial cualquier mensaje que se escriba enmarcado por las comillas, básicamente igual que la función Serial.print("mensaje"), pero con la diferencia que PgmPrint guarda el mensaje automáticamente en memoria flash, evitando el consumo de RAM del mismo.

Finalmente, la librería permite manipular la memoria microSD usando comandos similares a los que se utilizan en el sistema operativo Linux para el manejo de archivos. Esto se ha aprovechado para realizar diversas funcionalidades como listar los archivos disponibles, volcar su contenido para ser accedido a través del navegador web e incluso poder eliminarlos para liberar el espacio de la memoria SD.

2.7.6 RTC DS1307.

La forma de utilizar este dispositivo sigue el proceso ya familiar, declarar la librería, crear un objeto, inicializarlo y acceder a través de él, a las variables disponibles.

```
#include "RTClib.h"
RTC_DS1307 RTC;
RTC.begin(); //// Dentro del setup
```

Para acceder a la hora y fecha, hay que declarar un objeto denominado DateTime que contendrá los datos actuales:

```
DateTime now = RTC.now();
```

Entonces, esos datos podrán ser accedidos fácilmente:

```
now.day();
now.month();
now.year();
now.hour();
now.minute();
now.second();
```

Respectivamente para obtener día, mes, año, hora, minutos y segundos.
Para programar el reloj con una fecha y hora específicas, se utilizan 2 cadenas, una para la fecha y otra para la hora, las cuales siguen el formato del ejemplo siguiente:

"Dec 26 2009" "12:34:56"

2.7.7 PANTALLA LCD/TECLADO.

Este dispositivo cuenta con su propia librería:

```
#include <LCDI2Cw.h>
```

Es necesario asignarle una dirección en el bus I2C, y después declarar un objeto LCD con el número de columnas (20), filas (4) y la dirección I2C.

```
unsigned char i2cAddress = 0x4C;
LCDI2Cw lcd(20, 4, i2cAddress);
```

Luego se inicializa en el setup.

```
lcd.begin();
```

Y se pueden utilizar los siguientes comandos para imprimir en pantalla, para posicionar el cursor (de 0 a 19 columnas y de 0 a 3 filas) y para limpiar la pantalla, respectivamente, entre otras funciones.

```
lcd.print("mensaje");
lcd.setCursor(0, 1); //Columna cero, fila uno.
lcd.clear();
```

Para obtener datos del teclado, se utiliza la función keypad, que lee el buffer correspondiente al teclado y lo almacena en la variable dataByte.

```
dataByte = lcd.keypad();
delay(200);
```

El delay es para evitar falsas lecturas por el rebote de la tecla.

El teclado devuelve números del 1 al 16. El método lógico para facilitar su uso es desarrollar un controlador que asigne caracteres específicos a los

números obtenidos para poder ingresar datos desde el teclado. Sin embargo, este acercamiento produjo algunos conflictos en el código, específicamente la ralentización del instrumento y el desbordamiento de buffers de datos, debido seguramente a algún bug en el dispositivo y/o en la interacción del mismo con el sistema en general.

Por tal motivo, se opto por definir específicamente el comportamiento requerido para cada tecla cuando así sea necesario, situación que no arrojo ningún problema. Para minimizar la cantidad de código necesaria, se ha conectado el teclado en el ordenamiento original del controlador en el LCD, de forma que los números quedan en orden secuencial de izquierda a derecha, para mayor comprensión del usuario, y en filas de 4. De esta forma, se leen directamente los números que devuelve el módulo (Del 1 al 9), y solo es necesario definir 4 teclas, una para obtener el carácter 0, una tecla enter, una esc y una exit, que se utilizaran en el menú del instrumento, quedando 3 teclas que no se utilizarán.

2.7.8 DESARROLLO DE LA INTERFAZ DE CONTROL.

En cada reinicio del instrumento, se ha programado el acceso a un menú para poder definir algunos parámetros a través del teclado y la LCD.

La estructura básica del menú principal y los submenús siguientes, consiste en un lazo while controlado por alguna variable cuyo valor (0) se ha definido previamente en las declaraciones del programa. Mientras esta variable de control no cambie, el while es infinito y queda monitoreando permanentemente el comportamiento del teclado. Cuando una tecla es presionada, el programa compara el valor obtenido para revisar si tiene alguna acción asignada, de ser así, la ejecuta, pudiendo ser esta el despliegue de un nuevo menú, la confirmación de una acción/valor/configuración, la eliminación de un dato o la salida hacia un menú superior.

Según el flujo de las acciones del menú, la tecla con el símbolo exit sirve para abandonar todos los menús internos, excepto el principal. Esto simplemente consiste en que esta tecla hará que la variable de control en turno cambie de valor (asignándosele un 1). Luego todas las variables internas de control son puestas a cero de nuevo en el menú principal (que básicamente continúa siendo un while infinito) esto se hace con el objetivo de poder acceder una vez más a los submenús en caso de ser necesario.

Para abandonar el menú principal, se debe utilizar la opción definida como "Iniciar", con la cual cambiara la variable de control principal, y el setup seguirá su curso. De haber algún cambio en la configuración del equipo, se harán las modificaciones necesarias en el programa.

Las modificaciones posibles que se han definido son:

-Tipo de actualización del RTC (Manual o Automático).
-Elección del Intervalo de medición (En segundos, minutos u horas).
-Calibración de la referencia de la Veleta.
-Definición de la zona UTC.
-Dirección IP.
-Comportamiento de la pantalla LCD.
-Tipo de actualización de coordenadas (Manual o automático).
-Eliminación de archivos contenidos en la memoria microSD.

Lógicamente, una configuración personalizada debería ser permanente, aun cuando se reinicie el instrumento. Por este motivo, los parámetros de configuración se guardan en la memoria EEPROM del Arduino Mega, de acuerdo a la tabla 2.2, correspondiente a las ubicaciones en memoria, y a los posibles valores que pueden tomar.

Debido a que la memoria EEPROM solo acepta bytes, los valores posibles oscilan como máximo de 0 a 255, siendo estos números enteros sin signo. Esta limitación constituye la principal causa de la necesidad de utilizar más de una posición en memoria para definir una variable.

Dirección EEPROM	Configuración correspondiente	Valores posibles.	Acción Asociada o Valor resultante
1	Calibración de la Veleta	Corrección de 0 a 255.	Mapeo de la posición actual del potenciómetro.
2	Calibración de la Veleta	Corrección de 0 a 255.	Mapeo de la posición actual del potenciómetro.
3	Intervalo de medición	1 a 255	Cantidad de unidades de tiempo.
4	Tipo de Intervalo.	1, 2, 3	Tipo de unidad de tiempo: 1-Segundos. 2-Minutos. 3-Horas.
5	Zona UTC.	1 a 14, valores enteros.	Zona Horaria actual.
6	Alteración de la Hora UTC.	1, 2	La hora actual será: 1-Hora-UTC. 2-Hora+UTC.
7	Modificación del RTC	1	RTC programado manualmente.
8	Comportamiento de pantalla LCD.	0, 1	0-LCD Encendida. 1-LCD Apagada.
9	Modificación de IP.	0, 1	0-IP por defecto. 1-Una nueva Dirección IP ha sido programada.
10	Primer octeto de la IP.	0-255	Primer Octeto.
11	Segundo octeto de la IP.	0-255	Segundo Octeto.
12	Tercer octeto de la IP.	0-255	Tercer Octeto.
13	Cuarto octeto de la IP.	0-255	Cuarto Octeto.

Tabla 2.2 Estructura de configuración en EEPROM.

Dirección EEPROM	Configuración correspondiente	Valores posibles.	Acción Asociada o Valor resultante
14	Modo de actualización de coordenadas.	0, 1	0-Mediante GPS. 1-Manual. Usar coordenadas en memoria.
15	Latitud	0 a 90	Parte entera de la Latitud.
16	Latitud	0 a 255	Parte decimal de la Latitud.
17	Latitud Norte o Sur	0, 1	0-Norte. 1-Sur.
18	Longitud	0 a 180	Parte entera de la Longitud.
19	Longitud	0 a 255	Parte decimal de la Longitud.
20	Longitud Este u Oeste	0, 1	0-Este. 1-Oeste.
21	Modificación de Zona UTC.	0, 1	0-Zona UTC por defecto. 1-UTC manual.

Tabla 2.2 (Cont.) Estructura de configuración en EEPROM.

2.7.9 CONSIDERACIONES DE MEMORIA OPERATIVA Y DESEMPEÑO DEL SISTEMA.

Dada la cantidad de dispositivos y funciones de la Estación Meteorológica, es vital asegurar que su rendimiento no se vea mermado por limitaciones en la memoria RAM, sobre todo si se considera que uno de los objetivos principales del sistema es poder funcionar de forma autónoma durante una gran cantidad de tiempo, sin asistencia externa del usuario.

Si el instrumento llegara al límite de su memoria RAM, se producirían resultados inesperados y/o aleatorios, por lo que se vuelve necesario aplicar algunas medidas para optimizar el uso de memoria.

La función Pgmprint proporciona ayuda en este sentido, pero se limita al monitoreo por medio del bus serial. Todas las demás variables y Strings del programa se cargan directamente en memoria RAM. La cantidad de Strings

necesarias para la interfaz del usuario hacen que este proceso se vuelva critico, por lo tanto, para optimizar el rendimiento del programa, estas Strings se guardarán en la memoria Flash del Arduino Mega, que comúnmente se utiliza únicamente para almacenar el código de operación.

Para este proceso, es necesario incluir la siguiente librería de AVR:

```
#include <avr/pgmspace.h>
```

Después hay que definir las Strings propiamente como un tipo de dato especial para indicar que se guardarán en memoria:

```
prog_char string_0[] PROGMEM = "Enter  Del  Exit";
prog_char string_1[] PROGMEM = "Estación";
prog_char string_2[] PROGMEM = "Meteorológica";
prog_char string_3[] PROGMEM = "Pulsar numero";
……
```

Finalmente, es necesario definir una tabla que se utilizará para acceder a las Strings guardadas en memoria:

```
PROGMEM const char *string_table[] =
{
  string_0,
  string_1,
  string_2,
  string_3,
…… };
```

De esa forma las Strings quedan en la memoria Flash del Arduino Mega. Para poder utilizarlas, se hace uso de la siguiente línea de código:

```
strcpy_P(buffer, (char*)pgm_read_word(&(string_table[i])));
```

En donde "i" es el número correlativo de la String en cuestión y buffer es una cadena de caracteres previamente definida, que idealmente debe ser lo suficientemente grande para contener la totalidad de la String en memoria.

Una vez invocada esta línea, la String correspondiente a la ubicación en memoria "i", queda en el contenido de la cadena "buffer", para poder ser utilizada según convenga.

2.7.10 ENVÍO DE DATOS A TRAVÉS DE INTERNET.

La estación meteorológica puede enviar datos a través de internet, mediante el modulo Ethernet, esta es la razón de que se hayan declarado tanto un objeto Servidor como un objeto Cliente en la inicialización del Ethernet, además de la definición de un gateway de salida. Mediante su operación como cliente, el modulo enviara datos hacia el sitio Xively.com, que se especializa en darle apoyo a este tipo de proyectos. Para efectuar esta operación son necesarias las siguientes librerías:

```
#include <HttpClient.h>
#include <Cosm.h>
```

Luego se definen los parámetros necesarios para que el servidor en Xively identifique a que cuenta se ha asignado el dispositivo que está realizando la petición de conexión, estos parámetros son Api Key y Feed.

```
#define                                                    API_KEY
"pVHnk6A8mqhBxygoLLFoPBwlqEOSAKxnemhFUGtlUENydz0g"
#define FEED_ID 127547
```

Después se definen los nombres que se le asignarán a cada canal de datos.

```
char sensorId[] = "Temperatura";
char humidityId[] = "Humedad";
char pressureId[] = "Presión";
char windspeedId[] = "VientoVel";
```

Y se organizan en un arreglo de canales de datos que se enviarán hacia el sitio.

```
CosmDatastream datastreams[] = {
CosmDatastream(sensorId,                    strlen(sensorId),
DATASTREAM_FLOAT),
CosmDatastream(humidityId,strlen(humidityId),
DATASTREAM_FLOAT),
CosmDatastream(pressureId,strlen(pressureId),
DATASTREAM_FLOAT),
CosmDatastream(windspeedId,strlen(windspeedId),DATASTREAM_F
LOAT)
};
```

Luego se inicializa el objeto feed, pasándole como parámetros el identificador del flujo de datos, el arreglo antes definido y el número de canales.

```
CosmFeed feed(FEED_ID, datastreams, 4 );
```

Finalmente, se crea el objeto cliente del sitio web.

```
CosmClient cosmclient(client);
```

Entonces, una vez iniciado el loop de operación, cuando el instrumento se dispone a enviar los canales, se incluye cada dato medido, de tipo flotante, en el canal respectivo de cada parámetro.

```
datastreams[0].setFloat(temperature);
datastreams[1].setFloat(humidity);
datastreams[2].setFloat(pressure);
datastreams[3].setFloat(windspeed);
```

Y se envían los datos mediante la función put, pasándole el objeto feed y la Api Key. Además el servidor del sitio responde con un código de confirmación que queda guardado en la variable ret, pudiéndose monitorear serialmente el resultado de cada transmisión.

```
int ret = cosmclient.put(feed, API_KEY);
Serial.println(ret);
```

2.7.11 PARÁMETROS Y FUNCIONES EXTRAS.

Con las variables que se están monitoreando, es posible calcular algunos parámetros adicionales.

2.7.11.1 PUNTO DE ROCÍO.

Se define como la temperatura a la cual inicia la condensación del vapor de agua que contiene el aire, de manera que se produce rocío, neblina, y en casos más extremos, escarcha. Una temperatura baja indica condiciones secas, y una temperatura de rocío alta es propia de un clima húmedo.

Para el cálculo del Punto de Rocío, se utiliza la formula de Magnus, una conocida y ampliamente utilizada aproximación del cálculo exacto:

$$\gamma(T,RH) = ln\left(\frac{RH}{100}e^{\left(\frac{bT}{c+T}\right)}\right) = ln\left(\frac{RH}{100}\right) + \frac{bT}{c+T}$$

$$T_{dp} = \frac{c\gamma(T,RH)}{b-\gamma(T,RH)}$$

En donde:

Tdp=Temperatura de Punto de Rocío.
T: Temperatura en Grados Centígrados.
RH: Humedad Relativa, en porcentaje.
b = 17.271.
c = 237.7.

2.7.11.2 PUESTA Y SALIDA DEL SOL.

Utilizando los parámetros de latitud, longitud, fecha, hora y zona horaria es posible calcular de manera bastante precisa las horas de puesta y salida del sol.
La altitud solar α es la distancia angular del sol sobre el horizonte, y viene dada por:

$$\sin \alpha = \sin \varphi \sin \delta + \cos \varphi \cos \delta \cos h$$

donde:

φ: Latitud
δ: ángulo de declinación del sol.
h: ángulo horario.

El ángulo de declinación δ es la latitud sobre la tierra que se encuentra directamente bajo el Sol. Y se define como:

$$\delta = \sin^{-1}(\sin T \sin \varepsilon)$$

Donde:

T = L + C

L = (280.46646 + 0.98564736*días)

C = ((1.914602-0.00000013188*días)*sin M + (0.019993-0.000000002765*días)*sin 2M)

ε = 23.43929°

M = (357.52911 + 0.985600281*días)

Días: Es el número de días transcurridos desde el 1 de Enero del año 2000 a las 12:00 UTC.

T: Es la distancia angular entre el punto donde la órbita de la Tierra se encuentra más cerca del Sol, y la posición de la órbita actual.

El ángulo horario solar h es el distancia angular no negativa, Este u Oeste, hasta el punto que está directamente bajo el Sol. Entonces:

$$\cos(h) = \frac{\sin \propto - \sin \varphi \sin \delta}{\cos \varphi \cos \delta}$$

Para utilizar estas ecuaciones, se hace una primera iteración asumiendo un primer valor de Salida a las 6:00 A.M y para Puesta a las 6:00 P.M. y estas horas se convierten a tiempo UTC. Se calcula la declinación y luego el ángulo horario, este ultimo esta en coordenadas geométricas, que se pueden convertir a tiempo UTC. Este tiempo UTC se utiliza para recalcular la declinación y subsecuentemente, un nuevo ángulo horario, que de nuevo puede convertirse a Tiempo UTC. Este resultado está en tiempo Solar promedio local.

Hay que convertir este tiempo solar local a tiempo actual:

Tiempo Solar Actual= Tiempo Solar local – E.

Donde:

E = y sin 2L − 2e sin M + 4ey sin M cos 2L

e = 0.016708634 - 0.0000000011509*días

y = tan2 (ε / 2)

Y para convertir el tiempo solar actual a tiempo civil local (La convención del tiempo usada por la gente en una localidad dada), se debe tomar en cuenta que tan lejos se encuentra la localidad en cuestión del meridiano standard, definiéndose este como:

$$\text{Meridiano Standard} = |(\text{UTC Offset})| * 15$$

Entonces, se puede determinar el offset del Meridiano Standard en horas:

$$\text{Offset Local} = (\text{Standard Meridian} - \text{Longitud}) / 15$$

Y la formula del Tiempo Local Civil queda:

$$\text{Tiempo Local Civil} = \text{Tiempo Solar} - E + \text{Local Offset}$$

Y con respecto al tiempo UTC:

$$\text{UTC} = \text{Tiempo Local Civil} - \text{Offset Local}$$

2.7.11.3 FASE DE LA LUNA.

Utilizando los datos conocidos de Latitud, Longitud, Fecha, Hora, y Zona horaria, es posible deducir la fase actual de la Luna.

El ángulo de fase de la Luna i, se puede obtener con bastante precisión con la siguiente fórmula:

$$i = 180° - D - 6.289° \sin M' + 2.1° \sin M - 1.274° \sin (2D - M') - 0.658° \sin 2D$$

Donde:

D: Es la elongación promedio de la Luna (Distancia angular máxima entre la Tierra y la Luna).

M': Es la anomalía promedio (distancia angular, medida desde el punto donde la Luna se encuentra más cercana a la Tierra).

M: Es la anomalía promedio del Sol (Distancia angular medida desde el punto donde la Tierra se encuentra más cercana al Sol).

Entonces:

$$D = 297.8501921 + 12.19074911*\text{días}$$

$$M' = 134.9633964 + 13.06499295*\text{días}$$

$$M = 357.52911 + 0.985600281*\text{días}$$

Donde días, es el número de días transcurridos desde el 1 de Enero del año 2000 a las 12:00 UTC.

Entonces, k es un número entre 0 y 1 que indica la fracción del disco lunar que está iluminada.

$$k = (1 + \cos i)/ 2$$

Que puede interpretarse de la siguiente manera:

k:

0.00 = Luna Nueva. 0.50 = Luna Llena.

0.25 = Cuarto Creciente. 0.75 = Cuarto Menguante.

2.8 DISEÑO Y CONSTRUCCIÓN DE CIRCUITOS.

Se diseñaron 3 placas para organizar la ubicación física de los módulos y facilitar el proceso de conexionado de los mismos.

2.8.1 PLACA DEL ARDUINO UNO/ANEMÓMETRO.

Esta placa, mostrada en la figura 2.5, está diseñada como un modulo o shield que se conecta directamente al Arduino UNO, e implementa el circuito de acondicionamiento de señales que se muestra en la sección 2.5. La señal de salida de la Veleta y el puerto serial del UNO se envían hacia la placa principal de sensores.

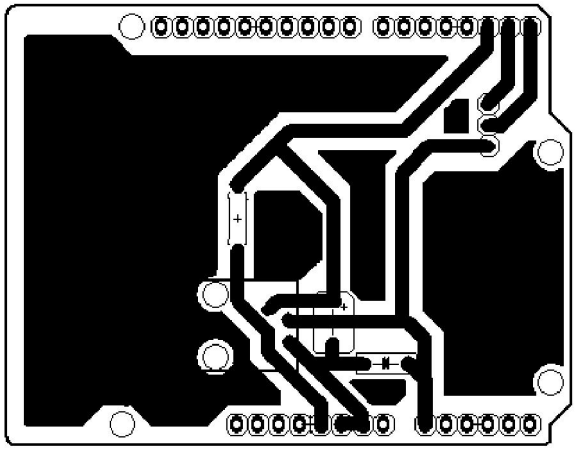


Figura 2.5, placa del Arduino UNO.

2.8.2 PLACA PRINCIPAL DE SENSORES.

Este circuito tiene como objetivo ser el punto de concentración de los módulos y dispositivos periféricos que conforman la estación meteorológica. Aquí se conectan directamente los sensores de Humedad Relativa, Temperatura y Presión Atmosférica, además del GPS. También cuenta con un puerto para conectar la pantalla LCD y un puerto para recibir las señales del Arduino UNO y la Veleta.

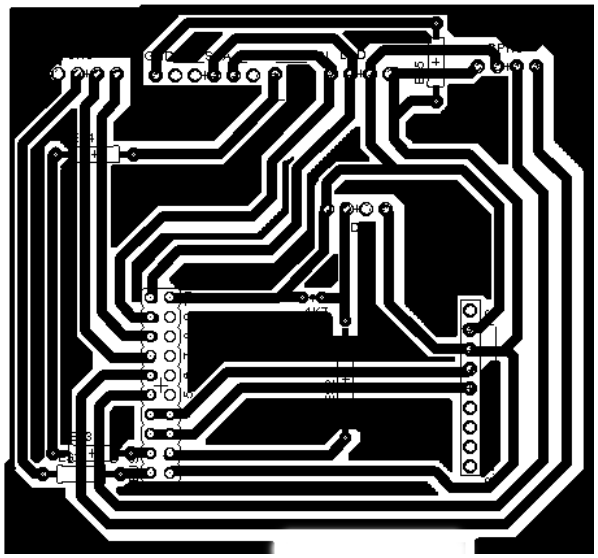


Figura 2.6 Placa principal de sensores.

Todas las señales de los diferentes dispositivos se envían hacia el Arduino Mega a través de un puerto de salida que funge como bus de datos principal del sistema. El diseño se muestra en la figura 2.6

2.8.3 PLACA DE RECEPCIÓN DE DATOS DEL ARDUINO MEGA.

Este circuito se ha diseñado como un shield personalizado para el Arduino Mega, construido de forma que pueda ser utilizado al mismo tiempo que los otros shields que se encuentran ya conectados al Mega (Ethernet y RTC), y consiste en la sección derecha de la figura 2.7.

Este módulo simplemente recibe las señales proporcionadas por la placa principal de sensores y las distribuye adecuadamente entre los pines del Mega, según el procesamiento necesario al que deben ser sometidas para obtener los parámetros de interés.

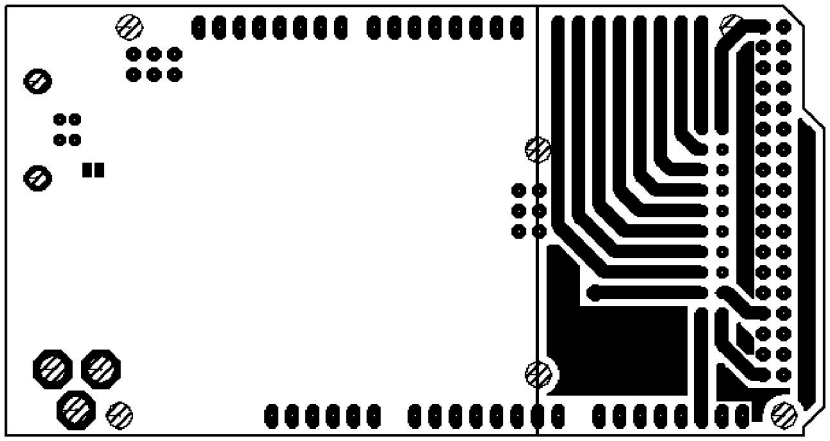


Figura 2.7 Placa de recepción de datos, mostrada en la sección derecha de un Arduino Mega, según el esquema de conexión.

CAPITULO 3: RESULTADOS DEL PROYECTO.

3.1 CARACTERISTICAS GENERALES.

El resultado final consiste en una Estación de monitoreo meteorológico, capaz de obtener, presentar y registrar los siguientes parámetros.

Fecha	2013 /8/9
Hora	14:58:25
Latitud:	13.87 N
Longitud:	-89.63 W
Altitud (m):	719.7
Temperatura (Grados C)	28.50
Humedad Relativa (%)	62.50
Presion Atmosferica (HPa)	932.00
Punto de Rocio (Grados C):	20.63
Velocidad del Viento (m/s):	0
Direccion:	S
Grados:	195
Salida del Sol	5:44
Puesta del Sol	18:23
Fase de la Luna	Nueva

Figura 3.1, Tabla de datos del Servidor de la Estación Meteorológica.

Además el instrumento es programable, siendo el usuario capaz de definir:

- Fecha.
-Hora.
-Intervalo de Medición.
-Latitud.
-Longitud.
-Comportamiento de la LCD durante mediciones.
-Zona UTC.
-Dirección IP.

Finalmente, aparte de albergar su propio servidor y memoria de registro de datos, la estación meteorológica construida también puede respaldar sus

datos mediante la transmisión de los mismos a un servidor externo en Internet.

3.2 IMPLEMENTACIÓN.

En la figura 3.2 muestra la ubicación de todos los módulos y dispositivos que constituyen la Estación Meteorológica ya construida, la cual se ha ubicado dentro de una caja impermeable, para poder soportar las inclemencias del tiempo.

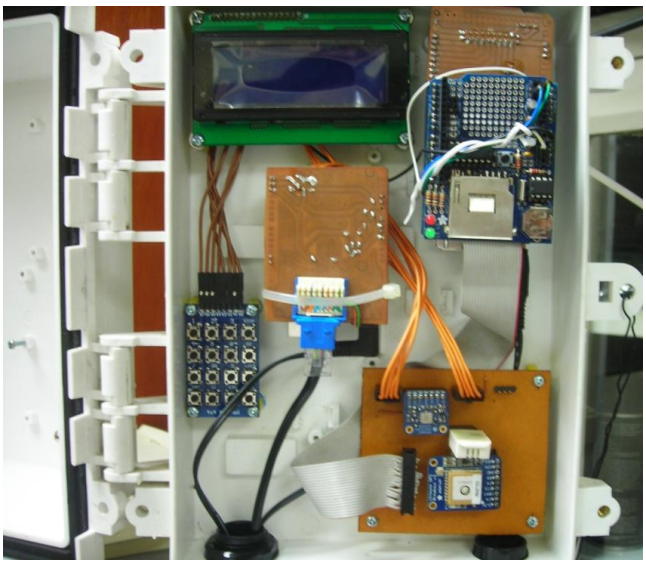


Figura 3.2 Implementación final de la Estación Meteorológica.

3.3 PRUEBAS EFECTUADAS Y FUNCIONAMIENTO.

El servidor de la Estación muestra los parámetros obtenidos en la tabla de la figura 3.1, la cual se actualiza con cada nueva medición.

La memoria microSD contiene los datos medidos, que se van guardando según el formato mostrado por la tabla 3.1.

Figura 3.3 Estación Meteorológica.

Estación Meteorológica							
Fecha	09/08/2013						
Latitud	13.87						
Longitud	-89.63						
Altitud	719.7						
Salida del Sol:	05:44						
Puesta del Sol	18:23						
Fase de la Luna	Nueva						
Hora	Temperatura	Humedad	Presión	Punto de Rocío	Anemómetro	Dirección	Grados
14:06:09	29.5	58.9	932	20.6	0	S	188
14:08:51	29.7	58.2	932	20.59	0	S	185
14:09:21	29.7	58	932	20.54	0	SW	230
14:09:51	29.7	58	932	20.54	0	SW	235
14:10:21	29.7	58.2	932	20.59	0	SW	235
14:10:51	29.7	59.1	932	20.84	0	S	160
14:11:21	29.7	58.6	932	20.7	0	S	160
14:11:51	29.7	59.2	932	20.87	0	N	5
14:12:21	29.7	58.2	932	20.59	0	W	255
14:12:51	29.8	57.7	932	20.55	0	W	255
14:13:21	29.9	56.5	932	20.3	0	SW	230
14:13:51	30	56.9	932	20.51	0	SW	230

Tabla 3.1 Tabla de datos característica de los archivos de salida de la estación meteorológica.

El Arduino Mega cuenta originalmente con 248 KB libres de memoria flash para albergar código de operación, de los cuales se utilizan casi 81 KB en el código operativo de la estación meteorológica, es decir que aún quedan 167 KB de memoria para desarrollar nuevas funcionalidades, lo que en realidad es un amplio margen teniendo en cuenta que varias librerías que se utilizan para diferentes propósitos ya se encuentran incluidas, y la inserción de nuevas librerías de funciones tampoco representa un gran consumo de memoria de código.

En cuanto a memoria RAM, el Mega tiene 8 KB, y para la ejecución del código del instrumento se utilizan aproximadamente 4 KB, de forma que la Estación tiene un desempeño de operación holgado y perfectamente puede soportar la añadidura de mejoras, dispositivos y nuevas funcionalidades.

La Estación Meteorológica también se comunica con el sitio web Xively.com, que va registrando los parámetros que se le envían, en los canales que se han definido previamente. El formato de salida en este sitio se muestra en la figura 3.4.

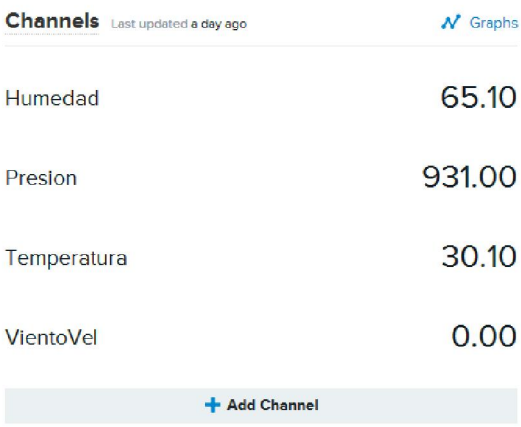


Figura 3.4, Canales de datos alojados en Xively.

Aparte de mostrar los datos enviados, el sitio también puede graficar todos los valores contenidos en los canales, en periodos de 5 minutos, 30 minutos o una hora. Para periodos de tiempo más grandes (desde 6 horas hasta 3 meses), el sitio grafica el promedio de los datos registrados.

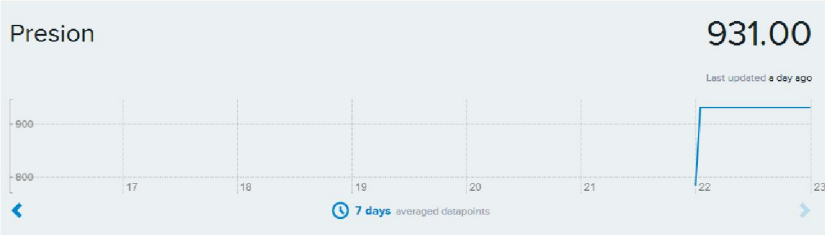


Figura 3.5 Grafica de salida generada por el servidor de Xively.

3.3 ANÁLISIS ECONÓMICO.

La tabla 3.2 muestra el presupuesto desglosado del proyecto. El costo total del proyecto es de 553.17 $.

Componente	Modelo/Tipo	Cantidad	Precio ($)
Microcontrolador	Arduino Mega	1	66.61
Microcontrolador	Arduino Uno	1	29.95
Modulo Web	Ethernet Shield	1	51.92
Sensor	Anemómetro/Veleta Davis	1	130
Sensor	BMP085	1	19.95
Sensor	DHT22	1	12.50
GPS	Ultimate GPS	1	39.95
Modulo RTC	DataLogger	1	19.95
LCD	LCD I2C & Serial	1	40
Teclado	4x4	1	5
Resistencia	10K	1	0.25
Resistencia	4.7K	1	0.25
Capacitor	10uF	1	0.49
Capacitor	22nF	1	0.35
Memoria MicroSD	4 GB	1	6
Costos de Envío		1	60
Caja	Caja Multipropósito Davis	1	50
Gastos varios		1	30
Total			553.17

Tabla 3.2 Presupuesto de la Estación Meteorológica.

Aunque es posible adquirir estaciones meteorológicas a un precio similar e incluso menor según tipo/calidad/fabricante, el costo total se justifica si se tienen en cuenta las consideraciones que se detallan a continuación.

-El instrumento es modular. Si uno o más de los dispositivos fallan, basta con ubicar donde se está dando el problema y sustituir el modulo correspondiente. Teniendo dispositivos específicos para cada parámetro a medir, el proceso de detectar un fallo se simplifica considerablemente. En el caso de un instrumento de origen propietario, un desperfecto implica que el usuario debe recurrir al fabricante, y si el periodo de garantía se ha vencido, esto generalmente incurre en un costo económico extra. La situación se agrava si el fabricante no cuenta con oficinas a nivel regional, en cuyo caso, si los desperfectos del instrumento son muy severos, significa que el dispositivo tendrá que ser desechado sin posibilidad de reparación.

-El instrumento no está sujeto a copyright, por lo que el usuario es libre de modificar a voluntad las características de la estación meteorológica, lo que incluye realizar expansiones del sistema, alterar el software, añadir o quitar sensores, o incluso sustituir los dispositivos cuando sea posible adquirir sensores de mayor rango y precisión. Es decir, que se puede efectuar una actualización y mejora del instrumento sin incurrir en costos significativos.

-El hecho de que los diseños y códigos sean de libre acceso también proporciona mayor seguridad contra la obsolescencia de los dispositivos, ya que existen varios fabricantes para los mismos módulos, que aunque con algunas diferencias, mantienen la compatibilidad de los dispositivos, y aunque alguna placa dejara de fabricarse, el usuario puede construir sus propios módulos de ser necesario.

Por lo tanto, aunque el costo es comparable con el de una estación meteorológica de origen propietario, dadas las características del instrumento, la inversión inicial implica que se evitaran mayores costos a futuro, proporcionando todas las ventajas antes mencionadas para el usuario final.

ANALISIS DE LEGALIDAD DEL PROYECTO.

El enfoque en la utilización de hardware libre permite evitar cualquier problema de tipo legal con respecto a patentes. El entorno de programación se puede descargar de forma gratuita desde el sitio web oficial del proyecto Arduino, y se mantiene siendo depurado, mejorado y actualizado de forma constante.

Los diferentes dispositivos y sensores utilizados cuentan con diferentes librerías desarrolladas de forma libre y gratuita por diferentes autores, siendo el usuario final quien decide qué código utilizar para implementar sus aplicaciones particulares, algunos usuarios incluso desarrollan ellos mismos sus propios drivers y librerías. Incluso, los diseños de los diferentes circuitos están disponibles al público, para que cualquier persona deseosa de construir ella misma las placas pueda hacerlo sin ningún tipo de restricción legal, siempre y cuando se mantengan los principios de la licencia bajo la que se distribuyen los módulos Arduino, Creative Commons, que garantiza libertad de distribución, fabricación y alteración del producto, mientras no se utilice el logo y el nombre distintivo Arduino para fines de lucro personal.

El proyecto Arduino cuenta con foros y tutoriales en los que hay amplia información disponible gratuitamente para el desarrollo de varias clases de proyectos. Los desarrolladores y los fabricantes de software/hardware también proporcionan amplia bibliografía junto a sus librerías y/o dispositivos y además generalmente brindan tutoría y orientación de forma gratuita a los usuarios para que estos puedan desarrollar sus proyectos de forma satisfactoria.

CONCLUSIONES.

- El Hardware libre es una opción viable para construir instrumentos de naturaleza compleja, mediante la implementación de módulos individuales prefabricados ha sido posible construir una estación meteorológica, capaz de obtener y registrar los parámetros más esenciales en este tipo de dispositivos: Temperatura, Humedad Relativa, Presión Atmosférica, Velocidad y Dirección del Viento. Además el instrumento realiza los cálculos necesarios para obtener el Punto de Rocío, la Salida y la Puesta del Sol, y la fase actual de la Luna.

- El instrumento cuenta con un servidor que muestra los datos a través de un navegador de Internet, también muestra los archivos contenidos en la memoria, su tamaño en bytes, y permite el acceso al contenido de los mismos, de forma que pueden ser descargados como archivos de formato CSV.

- Aparte del servidor, la estación meteorológica también es capaz de enviar información a través de la red Internet, hacia cualquier servidor externo que soporte la interacción con este tipo de dispositivos y cuente con el protocolo necesario para manejar los datos transmitidos. Específicamente, el instrumento envía los parámetros medidos hacia el sitio web Xively.com (Antes Cosm.com, y mejor conocido como Pachube), especializado en dar respaldo a proyectos e instrumentos que tienen que ver con la adquisición de datos. De esta forma, la estación cuenta con una segunda opción para respaldar los parámetros obtenidos.

- La estación meteorológica sincroniza la hora y fecha mediante el GPS de forma diaria, a las 0 horas de cada día y/o en cada reinicio. Esta es la opción por defecto y este enfoque evita el desfase horario que se produce en los RTC DS1307 cada cierto tiempo. Sin embargo, si el usuario así lo desea, se ha

dejado la opción de programar la fecha de forma arbitraria, en cuyo caso el GPS no se utilizara para sincronizar el RTC, hasta que el usuario configure de nuevo esta opción en el instrumento.

- El GPS también sincroniza la ubicación geográfica, proporcionando datos de Latitud, Longitud y Altitud (Sobre el nivel del mar). Al igual que con la fecha/hora, se tiene la posibilidad de que el usuario pueda definir las coordenadas geográficas, que se utilizaran hasta que se programe de nuevo la sincronización geográfica en modo automático.

- La estación meteorológica construida es programable, pudiendo el usuario definir a voluntad los siguientes parámetros: Fecha, Hora, Intervalo de Medición, Zona UTC, Tipo de sincronización horaria, Latitud, Longitud, Tipo de sincronización geográfica, y Dirección IP. También es posible calibrar la orientación de la Veleta, apagar/encender la pantalla LCD y borrar los archivos contenidos en la memoria microSD.

- El instrumento construido cuenta con varias vías de expansión, utilizando ya sea puertos seriales, el bus de datos I2C, la interfaz SPI, o usando directamente los pines que quedan disponibles en el Arduino Mega, 41 entradas/salidas digitales y 15 entradas analógicas, e incluso, teniendo en cuenta las limitaciones respectivas, se pueden utilizar los pines libres en el Arduino UNO, 11 entradas/salidas digitales y 6 entradas analógicas.

- Para obtener lecturas del anemómetro se utilizan interrupciones cada 3 segundos, eso ya impone una restricción de tiempo al sistema. Sin embargo, el verdadero problema es que estas interrupciones cortan las peticiones de los clientes que se conectan al servidor web. Y una de las prioridades es precisamente el acceso a los datos vía Internet. Este es el motivo principal de haber añadido un Arduino UNO para controlar el anemómetro, sin embargo, este módulo únicamente transmite valores enteros, como una

limitación de la librería utilizada para manejar la comunicación serial entre los microcontroladores.

- Es posible mejorar o cambiar los microcontroladores, sensores y otros dispositivos, ya que continuamente aparecen nuevas opciones disponibles. Más aun, fabricantes y desarrolladores siempre buscan mantener la compatibilidad de los diferentes dispositivos, lo que minimizaría cualquier problema a la hora de planear una actualización del sistema.

- La memoria actual del dispositivo para registro de datos es de 4 GB. El tamaño mínimo de cada archivo es de 320 bytes, y la cantidad máxima de bytes en cada lectura de sensores puede llegar a ser de hasta 47 bytes. Entonces, el sistema es capaz de registrar más de 90 millones de lecturas, lo que es comparable a realizar un registro de datos cada 10 segundos durante más de 25 años, superando ampliamente las expectativas iniciales. Además, es posible aumentar esta capacidad hasta donde lo permitan las características de las memorias microSD disponibles hasta la fecha.

- Algunos dispositivos, específicamente el modulo Ethernet, la interfaz microSD y la pantalla LCD, todavía poseen algunos bugs de programación que dificultaron su utilización.

- El uso de dispositivos prefabricados minimiza considerablemente el diseño y construcción de circuitos para acondicionamiento y tratamiento de las señales provenientes de los sensores, ya que la mayoría de los dispositivos proveen una salida digital se conectan directamente al microprocesador, y en caso de necesitar circuitería externa, no va más allá de algunos elementos comunes y de fácil obtención.

- La construcción del instrumento tiene un costo equiparable a algunos modelos de estaciones meteorológicas de origen propietario pero proporciona las siguientes ventajas:

1-Modularidad.

2-Simplificación del proceso de diagnostico/reparación.

3-Sin restricciones legales sobre el uso y modificación de hardware/software.

4-Capacidad de alteración del instrumento para adecuarlo a las necesidades del usuario, lo que incluye: Actualización del sistema, cambio de sensores, aumento o disminución del número de los mismos e inclusión de nuevos dispositivos.

5-Los costos de reparación, así como los de actualización y mejora del instrumento disminuyen considerablemente, comparados con los costos de reparación de dispositivos propietarios, y con los costos de adquisición de nuevos modelos no libres.

RECOMENDACIONES.

En una futura implementación de este trabajo, se podrían realizar las siguientes mejoras:

-Proveer al instrumento de una interfaz de control y configuración a través del servidor web.

-Añadir otros dispositivos sensores, como un piranómetro y un pluviómetro, y los respectivos parámetros que se pueden obtener de la combinación de datos de todos los sensores.

-Implementar un sistema de alimentación mediante paneles solares, de forma que no sea necesario contar con una instalación eléctrica en el sitio de medición.

-Añadir un modulo WIFI para evitar la necesidad de conectar físicamente la estación a una red LAN.

-Aprovechar la red celular para transmitir datos por medio de un módulo GPRS.

-Construir un circuito de control para el modulo GPS, para encenderlo únicamente cuando sea requerido.

-Utilizar el modulo SD extra como una segunda memoria, ya sea para aumentar la capacidad de almacenaje, o como segundo respaldo de datos.

-Transmitir valores flotantes correspondientes al anemómetro, desde el Arduino Uno hacia el Mega.

-Añadir la capacidad de verificar automáticamente el buen funcionamiento de los sensores.

-Incluir medidas de protección contra descargas atmosféricas y cortocircuitos que puedan dañar la Estación Meteorológica.

-Si se necesitara mayor resolución de los ADC, se pueden utilizar módulos externos con capacidad de hasta 16 bits de resolución, que están diseñados para conectarse al bus I2C.

REFERENCIAS.

- Gertz, E. & Di Justo, P. (2012). Environmental Monitoring with Arduino. O'Reilly Media Inc.

- Derived Weather Variables in Davis Weather Products, Application note 28. Davis Instruments.

- Getting Started with Arduino, recuperada el 4 de Septiembre de 2013 de http://arduino.cc/en/Guide/HomePage

- Arduino Language Reference, recuperada el 4 de Septiembre de 2013 de http://arduino.cc/en/Reference/HomePage

- Arduino Libraries, recuperada el 4 de Septiembre de 2013 de http://arduino.cc/en/Reference/Libraries

- Arduino Tutorials, recuperada el 4 de Septiembre de 2013 de http://arduino.cc/en/Tutorial/HomePage

- Arduino Mega, recuperada el 4 de Septiembre de 2013 de http://arduino.cc/en/Main/ArduinoBoardMega2560

- Arduino Uno, recuperada el 4 de Septiembre de 2013 de http://arduino.cc/en/Main/ArduinoBoardUno

- Arduino Ethernet Shield, recuperada el 4 de Septiembre de 2013 de http://arduino.cc/en/Main/ArduinoEthernetShield

- Davis Vantage Pro & Vantage Pro 2 Wind Sensor, recuperada el 4 de Septiembre de 2013, de http://www.lexingtonwx.com/anemometer/

- De Rey, J. ARDUINO readout of DAVIS Wind Sensor, recuperada el 4 de Septiembre de 2013 de http://www.qsl.net/on7eq/projects/arduino_davis.htm

- Fried, L. Adafruit Ultimate GPS, recuperada el 4 de Septiembre de 2013, de http://learn.adafruit.com/adafruit-ultimate-gps

- Fried, L. Bosch BMP085 Breakout Board, recuperada el 4 de Septiembre de 2013, de http://learn.adafruit.com/bmp085

- Fried, L. DHTxx Sensors, recuperada el 4 de Septiembre de 2013, de http://learn.adafruit.com/dht

- Earl, B. Adafruit Data Logger Shield, recuperada el 4 de Septiembre de 2013, de http://learn.adafruit.com/adafruit-data-logger-shield

- LCD Module with I2C/Serial Interface and Keypad Control «LCD I2C/Serial» User's Guide, recuperada el 4 de Septiembre de 2013, de http://www.web4robot.com/files/SerialLCDCtrN.pdf

- Porter, B. (2011) EasyTransfer Arduino Library, recuperada el 4 de Septiembre de 2013, de http://www.billporter.info/2011/05/30/easytransfer-arduino-library/

- Greiman, W. (2009) Arduino SdFat Library, recuperada el 4 de Septiembre de 2013, de http://www.arconlab.com/lab/Arduino/Library/SD%20Reader%20-%20Fat32/html/

- Cosm-Arduino, recuperada el 4 de Septiembre de 2013, de http://arduino-info.wikispaces.com/Cosm-Arduino

- Arduino WiFi. Connect an Arduino to Xively using the official Arduino WiFi shield (2013), recuperada el 4 de Septiembre de 2013, de https://xively.com/dev/tutorials/arduino_wi-fi/

- SD-Cards (2012), recuperada el 4 de Septiembre de 2013, de http://arduino-info.wikispaces.com/SD-Cards

- TimeLord Arduino Library, recuperada el 4 de Septiembre de 2013, de http://www.swfltek.com/arduino/timelord.html

- Dew Point, recuperada el 4 de Septiembre de 2013, de http://en.wikipedia.org/wiki/Dew_point

- Lazaridis, G. (2010) How a Key Matrix Work, recuperada el 4 de Septiembre de 2013, de http://pcbheaven.com/wikipages/How_Key_Matrices_Works/

- El Módulo TWI de los AVR, recuperada el 4 de Septiembre de 2013, de http://www.cursomicros.com/avr/bus-i2c/bus-i2c-en-avr-twi.html

- National Instruments. (2012) Comunicación Serial: Conceptos Generales, recuperada el 4 de Septiembre de 2013, de http://digital.ni.com/public.nsf/allkb/

- Larocca, S. Los instrumentos meteorológicos, recuperada el 4 de Septiembre de 2013, de http://www.tutiempo.net/silvia_larocca/Temas/instrumentos.htm

- I²C , recuperada el 4 de Septiembre de 2013, de http://es.wikipedia.org/wiki/I%C2%B2C

- Carletti, J. (2012) Comunicación - Bus I2C Descripción y funcionamiento, recuperada el 4 de Septiembre de 2013, de http://robots-argentina.com.ar/Comunicacion_busI2C.htm

- Protocolo I2C/TWI, recuperada el 4 de Septiembre de 2013, de http://www.quadruino.com/guia-2/sensores/protocolo-i2c-twi

- Serial Peripheral Interface, recuperada el 4 de Septiembre de 2013, de http://es.wikipedia.org/wiki/Serial_Peripheral_Interface

- Aurtenetxea, S. Formato de comunicación, recuperada el 4 de Septiembre de 2013, de http://www.sc.ehu.es/sbweb/webcentro/automatica/web_8051/Contenido/tutor8051_52/Capitulo%206/formato_comunicacion_PS.htm

- Universal asynchronous receiver/transmitter, recuperada el 4 de Septiembre de 2013, de http://en.wikipedia.org/wiki/UART#Parity_error

- Fitz, A., Mejia, N., Aguirre, E. Protocolos Seriales, recuperada el 4 de Septiembre de 2013, de http://es.scribd.com/doc/33743961/Protocolos-Seriales

- Meteorología, recuperada el 4 de Septiembre de 2013, de http://es.wikipedia.org/wiki/Meteorolog%C3%ADa

- Estación meteorológica, recuperada el 4 de Septiembre de 2013, de http://es.wikipedia.org/wiki/Estaci%C3%B3n_meteorol%C3%B3gica

ANEXO A: MANUAL DEL USUARIO.

A.1 INDICACIONES GENERALES.

Al iniciar su operación, el instrumento muestra un mensaje de bienvenida, y luego una indicación para utilizar el menú, que se despliega a continuación. Cada función del menú tiene un número correlativo. Para acceder a una opción debe presionarse el número correspondiente, la disposición del teclado es la que se muestra en la figura A.1

1	2	3	4
5	6	7	8
9	0		
Enter		Esc	Exit

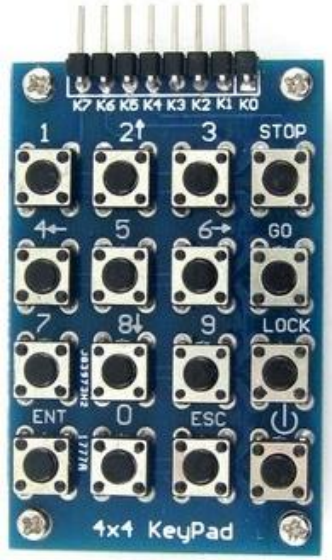


Figura A.1, Disposición del teclado. (Fuente: http://www.wvshare.com/product/4x4-Keypad.htm)

La tecla Ent (Enter) es para confirmar el ingreso de datos o la ejecución de una acción, la tecla Esc se utiliza para borrar datos en situaciones que así lo requieran. La tecla con el conocido símbolo Exit se utiliza para abandonar un menú, sin realizar ningún cambio y regresar al menú superior.

Al desplegarse el primer menú, se activa un temporizador, de manera que si no se presiona ninguna tecla en 20 segundos, el instrumento se inicializa con los parámetros correspondientes a la última configuración realizada.

A.2 MENÚ PRINCIPAL.

Figura A.2, Menú Principal.

A.2.1 INICIAR.

Al elegir esta opción, el instrumento inicia su operación con la última configuración realizada.

A.2.2 CONFIGURACION STANDARD.

Esta opción define parámetros de operación que se consideran típicos. La configuración predefinida consiste en:

-Intervalo de 10 segundos para obtener lecturas de los sensores.
-Sincronización del RTC mediante lectura del GPS cada día a las 00:00:00 horas, y en cada reseteo.
-Ajuste de la hora obtenida mediante el GPS de acuerdo a la zona UTC-6, que corresponde a la zona horaria definida para El Salvador.
-IP predefinida (192.168.0.177).
-Pantalla LCD apagada durante operación de lectura de sensores.
-Coordenadas geográficas predefinidas en memoria (Ciudad de San Salvador).

A.2.3 CONFIGURAR.

Esta opción permite configurar algunos parámetros que se muestran en un submenú.

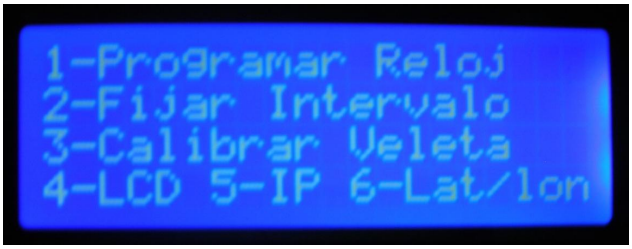


Figura A.3, Opciones del Menú Configurar.

A.2.3.1 PROGRAMAR RELOJ.

Figura A.4, Opciones de configuración del RTC.

El instrumento puede programar la fecha y hora del RTC de 2 formas:

A.2.3.1.1 PROGRAMACIÓN MANUAL DEL RELOJ.

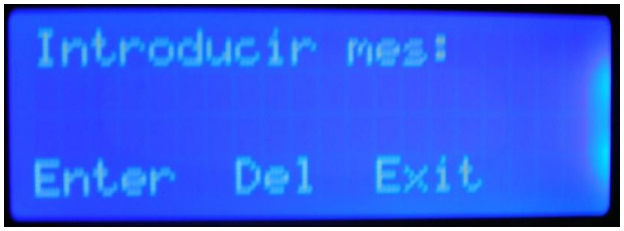


Figura A.5, Parte del proceso de programación manual de la fecha.

Si se desea una fecha y hora especificas y/o diferentes al tiempo standard, se puede elegir esta opción, inmediatamente aparecerá una pantalla solicitando especificar el año. Una vez que se ha definido el valor del año, al presionar la tecla enter se guardara el valor y se mostrara la siguiente pantalla para ingresar el valor correspondiente al mes, una vez que se ha tecleado el valor, de nuevo se presiona la tecla enter y se muestra la siguiente pantalla para ingresar el valor del día, y se repite el mismo

proceso ingresando los valores correspondientes a la hora, minutos y segundos. El usuario puede interrumpir el proceso en cualquier momento regresando al menú anterior, sin alterar la programación actual del RTC.

La hora debe estar en formato de 24 horas, y los números de un solo digito deben ingresarse con un cero inicial.

Una vez que se ha completado todo el proceso, después de la pantalla de los segundos, se mostrara la fecha y hora que se han ingresado, para que el usuario este seguro de los valores definidos y para que pueda programar la hora en el momento especifico que se desee. Presionar la tecla enter ingresara los datos en el RTC y un mensaje de confirmación se mostrara en pantalla.
Si la hora se programa de forma manual, el GPS no será utilizado para sincronizar el RTC, hasta que se defina específicamente que así se realice.

A.2.3.1.2 RELOJ EN MODO AUTOMÁTICO.

Es la opción por defecto. Inicialmente el usuario debe especificar la Zona UTC. En una primera pantalla deberá ingresar el valor de la zona UTC deseada, y luego en una segunda pantalla deberá especificar si esta magnitud se debe sumar o restar del tiempo UTC standard que provee el GPS.

Al ingresar estos valores, se iniciara el proceso de lectura del GPS, y una vez que se obtenga satisfactoriamente la fecha y la hora, el instrumento usara estos datos para programar el RTC y mostrara un mensaje de confirmación en pantalla, antes de regresar al menú anterior.

Una vez que se elije esta opción, El GPS se utilizara para programar el RTC cada día a las 00:00:00 horas, así como en cada reinicio del instrumento.

A.2.3.2 FIJAR INTERVALO.

El usuario puede definir el intervalo de tiempo que transcurre entre las lecturas de los sensores. Este intervalo puede ir desde 10 segundos hasta 23 horas, cualquier intervalo igual o mayor a 24 horas producirá un único registro de datos en el inicio de operación del instrumento.

Inicialmente, una pantalla solicita que el usuario ingrese el valor del intervalo deseado, este puede ser un número entero desde 1 a 255.

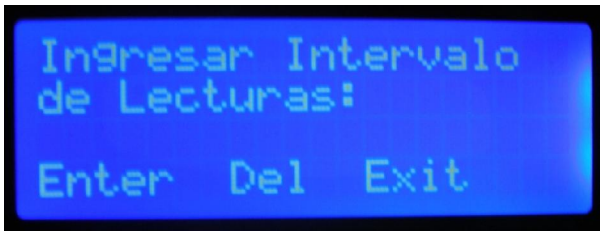


Figura A.6, Pantalla de ingreso del intervalo de tiempo.

Una vez definida esta cantidad, una nueva pantalla solicita que se defina el tipo de intervalo, este puede estar en segundos, minutos u horas.

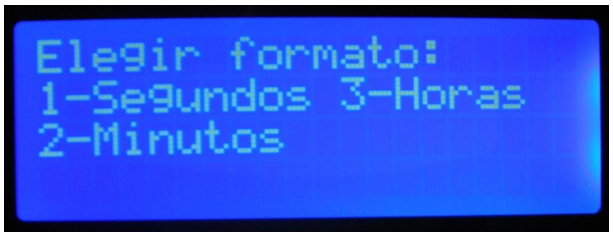


Figura A.7, Definición del tipo de formato del intervalo.

Cuando se haya seleccionado el valor y el tipo de intervalo, este se desplegara en la pantalla, y luego el instrumento regresara al menú anterior.

A.2.3.3 CALIBRAR VELETA.

Idealmente, la veleta correctamente calibrada indica 0° cuando apunta al Norte. Para calibrar, el usuario debe mantener la veleta apuntando hacia el norte, y elegir la opción "Calibrar Veleta", un mensaje indicara cuando el proceso se haya completado.

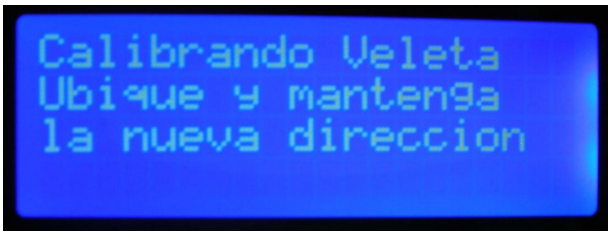


Figura A.8, Calibración de la Veleta.

A.2.3.4 APAGAR LCD.

Permite definir si la pantalla LCD se mantendrá encendida o apagada durante la operación de lectura de los sensores.

Si se elige apagar la pantalla, esta puede ser activada de nuevo mientras se registran los datos.

A.2.3.5 MODIFICAR DIRECCIÓN IP.

Figura A.9, Programación de la Dirección IP.

La dirección IP puede ser modificada en este submenú. En la primera pantalla mostrada, el usuario debe digitar el primer octeto de la dirección IP, y presionar la tecla enter. Este proceso se repite sucesivamente para los octetos siguientes de la dirección. Una vez que la dirección ha sido

modificada, el servidor la utilizara cada vez que haya un reinicio del instrumento.

A.2.3.6 LATITUD/LONGITUD.

Permite configurar las coordenadas geográficas. El usuario puede elegir entre 2 modos de actualización de coordenadas: Manual o Automático.

Figura A.10, Opciones de sincronización de coordenadas geográficas.

En el modo manual, el usuario debe ingresar los datos en formato de grados, iniciando con la latitud. Primero se debe digitar la parte entera, luego la parte decimal, y después especificar el hemisferio, Norte o Sur.

Figura A.11, Primera pantalla de programación manual de la Latitud.

Figura A.12, Segunda pantalla de programación manual de la Latitud.

Figura A.13, Tercera pantalla de programación manual de la Latitud.

A continuación, se inicia la secuencia de configuración de la Longitud, y se sigue el mismo procedimiento para definir este parámetro.

Una vez que se han configurado las coordenadas de forma manual, estos valores se mostraran en el sitio web hasta que se defina la actualización de coordenadas de forma automática.

En el modo Automático, se utiliza el GPS para obtener las coordenadas geográficas cada día, y cada vez que se reinicia el instrumento.

A.2.4 BORRAR SD.

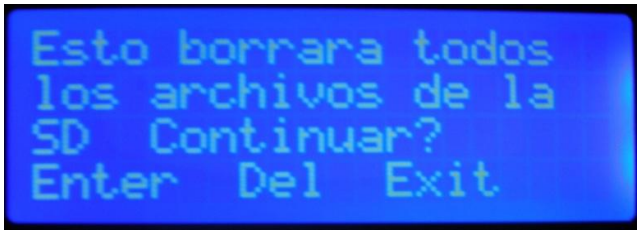


Figura A.14, Mensaje de Advertencia antes de borrar archivos de la microSD.

Esta opción permite borrar los archivos creados y así liberar espacio de la memoria SD. Un mensaje de advertencia de borrado se mostrara en pantalla. Si el usuario presiona la tecla enter, un segundo mensaje de advertencia es desplegado para confirmar el borrado de los archivos. Si se presiona enter de nuevo, todos los archivos contenidos en la memoria SD son eliminados.

A.2.5 AYUDA.

Despliega una serie de mensajes con la descripción básica de la operación y configuración del instrumento.

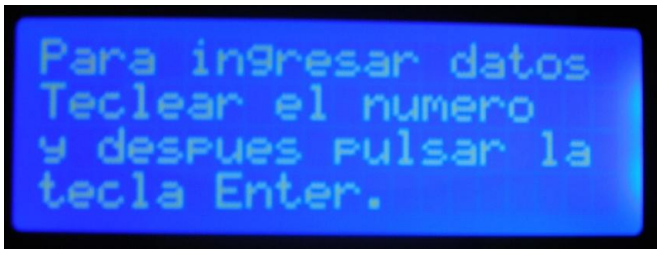


Figura A.15, Parte de los mensajes de Ayuda.

A.3 PROCESO DE REGISTRO DE DATOS.

A las 00:00:00 horas de cada día, el instrumento creara un archivo .CSV, de igual forma en cada inicio de operación, si es que el archivo no existe ya, en cuyo caso lo abrirá de nuevo y escribirá al final del mismo.

El nombre del archivo consiste en la fecha del día actual según los parámetros del RTC, con el formato año-mes-día, ese archivo será utilizado para guardar las lecturas de los sensores con el intervalo que se haya definido en la configuración del instrumento. Si se produce un reinicio del instrumento, los encabezados se imprimirán de nuevo en el final del archivo continuando con la operación normal de registro de datos.
A las 00:00:00 horas de cada día, el instrumento cerrara el archivo que hasta entonces ha estado en uso, y creara un nuevo archivo.

Si se presiona una tecla mientras el instrumento está registrando datos, se despliega un menú, con las siguientes opciones.

1-Apagar LCD: Permite apagar la pantalla LCD.
2-Encender LCD: Activa la pantalla LCD para poder visualizar los datos que se están registrando

3-Ir al Menú: Detiene la operación de lectura de sensores y servidor web, e inicia el menú de configuración del instrumento.

Figura A.16, Pantalla de control del LCD y acceso al Menú Principal durante operación de la Estación Meteorológica.

A.4 SERVIDOR WEB.

El instrumento cuenta con su propio servidor web, como se puede observar en la figura A.18. Se despliega en una tabla, la fecha, la hora y los valores de los sensores, que se van actualizando en tiempo real, según se haya definido el intervalo de lecturas en la configuración.

Además el servidor muestra una lista de los archivos existentes en la memoria SD, de forma que se puede acceder a los datos registrados haciendo click sobre el archivo que se desea obtener. Esto desplegara el contenido del archivo en la pantalla del navegador, y si el usuario lo desea, puede guardar los datos mostrados con la opción correspondiente en el navegador web. El archivo en cuestión es guardado con el formato CSV por defecto, pudiendo ser accedido después con cualquier visor de hojas de cálculo.

A.5 TRANSMISIÓN DE DATOS A SERVIDOR EXTERNO.

Mientras la estación meteorológica se encuentra en operación, estará enviando una serie de de datos, organizados en "canales" hacia el sitio web Xively.com. La transmisión de los parámetros se realiza cada 3 minutos, y los canales monitoreados de esta forma son: Temperatura, Humedad Relativa, Presión Atmosférica y Velocidad del Viento.

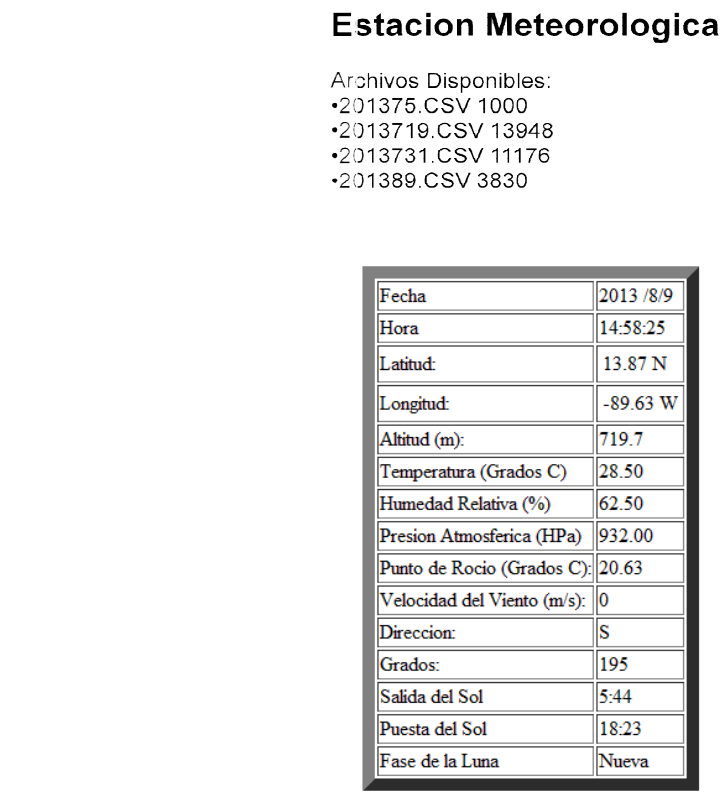


Figura A.18, Presentación del Servidor Web de la Estación Meteorológica.

Los datos transmitidos se acceden desde internet en la siguiente dirección: https://xively.com/feeds/127547

El sitio muestra los canales activos y los últimos datos recibidos.

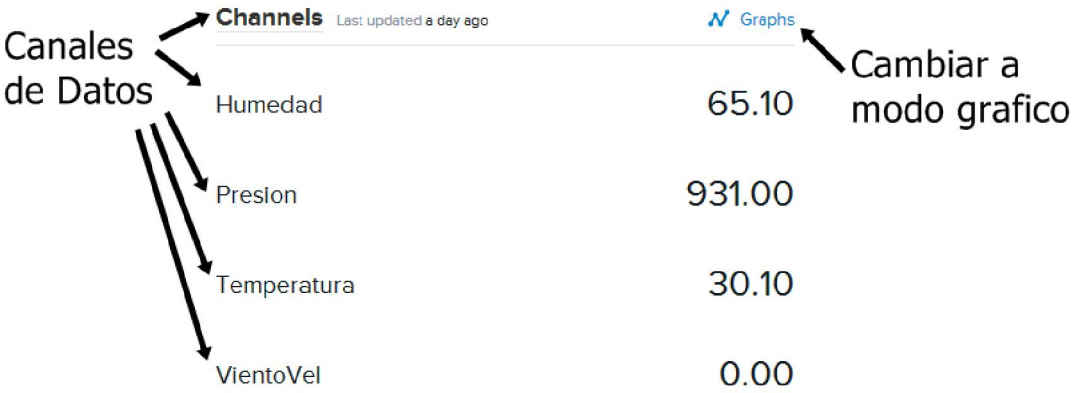


Figura A.19, Presentación de los canales de datos en el sitio web Xively.com

Este sitio web también puede mostrar graficas de los datos que se han transmitido. Se pueden graficar todos los puntos obtenidos en el lapso de 5 minutos, 30 minutos y 1 hora.

Figura A.20, Opciones de graficado del sitio web Xively.com.

Si se desean periodos de tiempo mayores, el sitio promedia los datos obtenidos y los grafica. Estos periodos son de 6 horas, 1 día, 7 días, 1 mes y 3 meses.