*UNIVERSIDAD LAICA "ELOY ALFARO DE MANABÍ"*

*FACULTAD DE INGENIERÍA*

*CARRERA DE INGENIERÍA ELÉCTRICA*

***Trabajo de Titulación en la modalidad "Proyecto de Investigación"***

*Que se presenta como requisito para optar por el título de:*

***INGENIERO ELÉCTRICO***

***Tema***

# *Diseño de un Sistema de Domótica con Arduino Mega 2560 y Arduino Ethernet Shield, conectado y controlado remotamente desde un Servidor Web, para ser implementado en el sector Residencial de la Ciudad de Manta.*

***Autor***

*Quijije Marín Sandro Javier*

***Director de tesis***

*Ing. Carlo Cano*

***Enero, 2016***

*Manta – Manabí – Ecuador*

# AGRADECIMIENTO

Sin duda Agradezco a Dios por permitirme la vida y haber logrado esta meta, a mis padres por estar siempre pendiente de mí en todas las áreas de mi vida, porque gracias a ellos, a su esfuerzo y constancia pudieron brindarme lo mejor desde que era un niño y enseñarme el verdadero valor de la educación, gracias a esto he alcanzado una meta que me la propuse desde muy pequeño y hoy en día me siento muy orgulloso de la persona que soy.

Agradezco a todos los docentes que me acompañaron en este largo camino de mi carrera estudiantil, agradezco a mi tutor el Ing. Carlo Cano que me oriento en la realización de este proyecto que significa el último escalón de mi carrera.

Agradezco a todas aquellas personas que contribuyeron de manera directa e indirecta a la realización de este proyecto.

# DEDICATORIA

Dedico este trabajo de Titulación a Dios, por acompañarme en cada paso de mi vida, por haberme guiado por el buen camino y con mucho sacrificio y esfuerzo haber alcanzado esta meta propuesta.

A mi madre Martina, por darme la vida, por creer en mí, por estar siempre pendiente y apoyándome.

A mi padre Gerardo, por brindarme su apoyo, por sus sabios consejos y sus grandes conocimientos impartidos.

A mi novia Estefanía, por estar presente en los momentos más difíciles y ser fuente de fortaleza.

A todas aquellas personas que fueron parte de mi vida durante este proceso, amigos, compañeros, profesores, conocidos, a todos aquellos que creyeron en mí hasta el final.

Sandro Javier Quijije Marín.

# TABLA DE CONTENIDO

# ÍNDICE DE TABLAS

# ÍNDICE DE FIGURAS

# ÍNDICE DE ILUSTRACIONES DE DISEÑO

# ÍNDICE DE ANEXOS

# RESUMEN

El presente proyecto consiste en el diseño de un sistema de domótica utilizando hardware y software libre de Arduino. Para su efecto se realizó la construcción de un prototipo, para la demostración del caso, el mismo está compuesto por sensores que permitirán obtener información real y precisa de acuerdo a su funcionamiento, además integra módulos que le agrega características importantes al sistema.

Los sensores y módulos están conectados directamente a las dos placas principales seleccionadas para el proyecto, el Arduino Mega 2560 encargado de obtener información y ejecutar las acciones que se haya indicado en la programación, y el Arduino Ethernet Shield encargado de conectar el sistema a una red mediante servidor Web. En este caso para verificar su funcionamiento se puso en marcha a través de una red local utilizando un router, y a partir de este momento tan solo se necesitara un dispositivo con características para conectarse al router y con acceso a un explorador en el cual ubicando la IP configurada nos mostrara la página del servidor Web, desde donde podremos visualizar información de los sensores y encender o apagar salidas, además las salidas se pueden activar manualmente mediante pulsadores o también mediante un mando a distancia IR.

Al sistema de domótica podemos brindarle más características agregándole nuevos sensores o módulos o bien incrementándole los existentes. Bajo esta premisa, este Proyecto presenta los sensores y módulos posibles por utilizar y los utilizados, de los cuales se obtuvo resultados satisfactorios en la operación e implementación de prototipo.

# ABSTRACT

This project involves the design of a home automation system using free software and hardware Arduino. To effect the construction of a prototype was made, to show case the same is composed of sensors that allow real information and accurate according to their performance, integrated modules also adds important features to the system.

Sensors and modules are connected directly to the two main plates selected for the project, the Arduino Mega 2560 charge of information and execute actions that you have specified in the schedule, and the Arduino Ethernet Shield responsible for connecting the system to a network by Web server. In this case to verify proper operation was launched through a local network using a router, and from this time only needed a device with features to connect to the router and access to a browser in which placing the IP show us configured Web server page, where we can view information from the sensors and turn on or off outputs, the outputs can also be activated manually using buttons or also by an IR remote control.

Home automation system we can provide more features adding new sensors or modules or existing incrementándole. Under this premise, this project has the potential sensors and modules for use and used, which was obtained satisfactory results in the operation and implementation of prototype.

## TEMA

***Diseño de un Sistema de Domótica con Arduino Mega 2560 y Arduino Ethernet Shield, conectado y controlado remotamente desde un Servidor Web, para ser implementado en el sector Residencial de la Ciudad de Manta.***

# CAPITULO 1. INTRODUCCIÓN Y GENERALIDADES

## 1.1. Introducción

Con el pasar de los años la tecnología ha ido creciendo a pasos agigantados, en la vida diaria se ve como la tecnología reemplaza el trabajo a las personas. Vivimos en una era en la que la domótica, de la mano de la automatización está ganando terreno, aunque el ser humano todavía no está arraigado a las propiedades que ofrece la domótica es un hecho que en un futuro estará instalada en cualquier lugar, en cualquier vivienda. Las comunicaciones se han automatizado y han avanzado, las personas ya no necesitan movilizarse para obtener cosas, ahora simplemente lo pueden lograr haciendo un clic.

El cambio de los lugares tradicionales a lugares inteligentes, a partir de la incorporación de herramientas de domótica, trasciende los límites de la comodidad hacia la eficiencia y la sostenibilidad ambiental.

El presente trabajo tiene como propósito principal diseñar un sistema de domótica, mediante la aplicación de un conjunto de tecnologías para el control y una automatización inteligente, gracias a esto permitir una gestión eficiente del uso de la energía además de aportar una mayor seguridad, confort, multitareas y comunicación entre el usuario y el sistema. Para conseguir todas las características comentadas anteriormente es necesario que el sistema recoja información de su entorno mediante sensores que dispongan de la lógica para actuar en consecuencia utilizando diferentes actuadores para la respectiva ejecución de la tarea automatizada.

En este proyecto utilizaremos una placa electrónica Arduino Mega 2560 junto a una Arduino Ethernet Shield en la que nos apoyaremos con otros dispositivos para

poder construir un sistema domótica sencillo y muy eficiente. Arduino es una herramienta que pueden detectar y controlar más del mundo físico que un equipo o computador de escritorio; es una plataforma de computación física de código abierto basado en una placa electrónica simple, y un entorno de desarrollo para escribir el software. La programación del mismo sistema se dará mediante el software de Arduino que se basa en el entorno de programación multimedia Processing.

En este tipo de tecnologías, el uso de la computadora puede desaparecer completamente, debido a que el mismo sistema es controlado mediante un servidor Web y para ingresar a él no es necesario tener una computadora ya que podemos acceder desde una Tablet o ya sea desde nuestro Smartphone, teniendo absolutamente todo el control del sistema en la palma de nuestra mano.

Sin embargo, llevar a cabo la automatización de un hogar no es tarea fácil. Es un sistema complejo con una gran variedad de elementos conectados entre sí. Es imprescindible una organización rigurosa del sistema para que en su conjunto pueda funcionar correctamente.

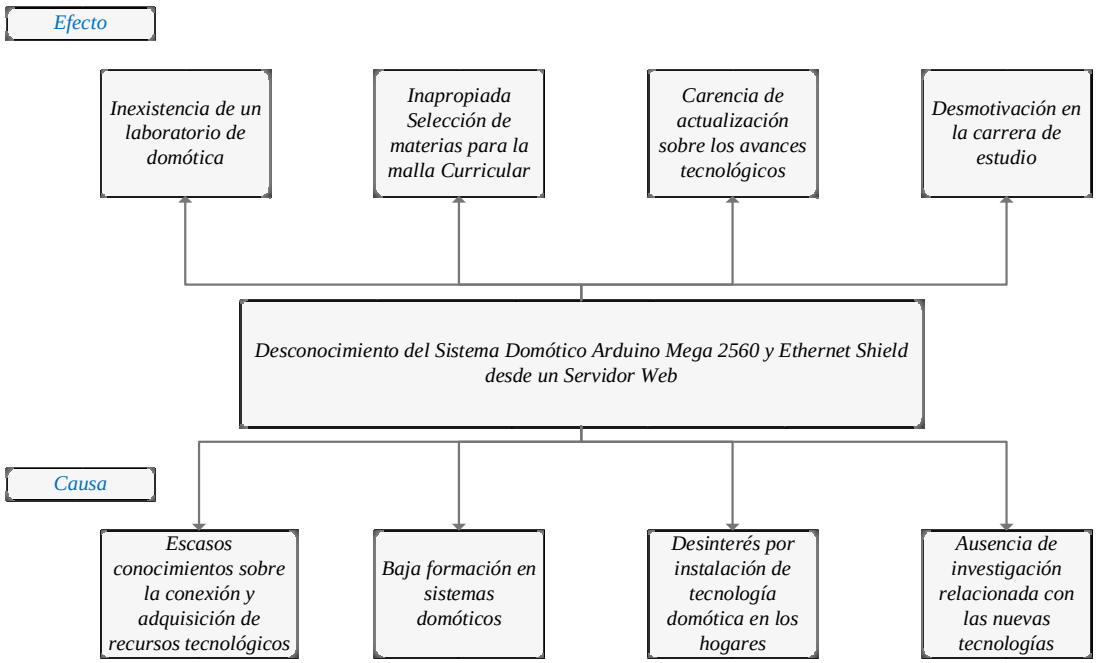
## 1.2. Planteamiento del Problema

En el Ecuador se habla mucho sobre los avances de ciencia y tecnología, pero muchos de ellos están dedicados a un nivel que va más allá del sector residencial, a pesar que actualmente se promueve la evolución de la tecnología en nuestros medios, también se debería hacer énfasis en promover hogares inteligentes y por ende utilizar tecnologías muy al alcance de cualquier bolsillo, un gran ejemplo de esto es Arduino, una plataforma de desarrollo a través de software y hardware libre que evoluciona día que bien nos permitiría desarrollar un sistema de domótica para una vivienda obteniendo un hogar confortable y seguro en vista de los robos y la inseguridad que existe actualmente en el país.

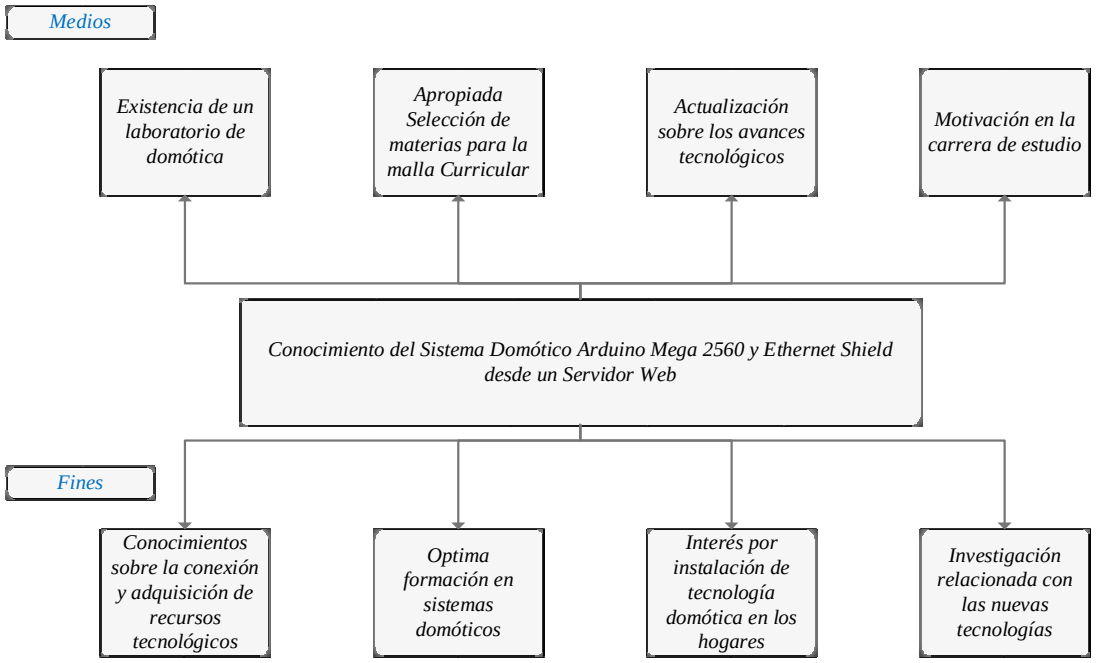
El avance tecnológico en los hogares manabitas evoluciona muy lentamente, este problema no es un tema que se viene tratando desde ahora, sino que siempre ha estado presente, debido muchas veces a los pocos conocimientos de tecnologías como Arduino, cuya plataforma se puede utilizar para desarrollar un sistema de domótica el mismo que competiría al mismo nivel que otros sistemas de marcas conocidas con costos muy elevados.

En la ciudad de Manta se detecta poco conocimiento acerca de la domótica y a la vez de Arduino, debido a la inexistencia de una adecuada formación en estos temas y al desconocimiento de los avances tecnológicos en el sector residencial impidiendo el desarrollo del mismo y el avance de la ciudad como tal.

## 1.3. Árbol del Problema




## 1.4. Árbol de Objetivos



## 1.5. Objetivos

### 1.5.1. Objetivo General

- Diseñar un Sistema de Domótica realizado con Arduino Mega 2560 y Arduino Ethernet Shield, conectado y controlado remotamente desde un Servidor Web, para ser implementado en el sector Residencial de la Ciudad de Manta ofreciendo confiabilidad y bajo costo.

### 1.5.2. Objetivos específicos

- Presentar conocimientos sobre la conexión y adquisición de recursos tecnológicos para promover la existencia de un laboratorio de domótica que fomente el avance tecnológico mediante el uso de plataformas como Arduino.
- Contribuir a la formación en sistemas domóticos mediante el presente proyecto dando a notar que una apropiada selección de materias para la malla curricular influiría en la expansión de la Carrera de Ingeniería Eléctrica en otros campos.
- Fomentar el interés por la instalación de tecnología domótica en los hogares dando a conocer una actualización sobre los avances tecnológicos que promuevan el desarrollo del sector residencial.
- Realizar una investigación relacionada con las nuevas tecnologías que contribuyan a la motivación en la carrera de estudio y el interés de los estudiantes por la investigación.

## 1.6. Justificación

La aplicabilidad de este proyecto, en los hogares está orientada a facilitar actividades domésticas asociados al control de acceso y seguridad de los espacios contando con un bajo costo, alto nivel de calidad, confiabilidad, seguridad y beneficiando en general la forma como las personas interactúa con su hábitat.

El presente trabajo gira en torno a la automatización de los espacios, no solamente como factor indicador de progreso, sino también como una necesidad claramente identificada en las nuevas tendencias, el mismo que propone el uso de Arduino basado en la utilización de hardware libre y software de código abierto, para el control de motores, luces, contactos, y finalizar con el control de diversos dispositivos a través de algo novedoso, como lo es el control en la domótica.

Este proyecto beneficia a la comunidad estudiantil, fortaleciendo los conocimientos en el área de Electrónica, Electricidad, Controles, Automatización, vinculando la parte teórica con la práctica, reforzando así el proceso de enseñanza-aprendizaje en la formación profesional de los estudiantes de la Carrera de Ingeniería Eléctrica, esto por mencionar algunos puntos importantes, ya que la fortaleza de este trabajo radica en el mejor uso de los materiales a nuestra disposición, para desarrollar la capacidad intelectual y creativa de cada uno según sus conocimientos.

Se espera que en las próximas décadas la incorporación de Sistemas de domótica en los hogares gane más terreno, gracias a la incorporación de herramientas que ofrezcan una alta calidad y confiabilidad a menor costo.

## 1.7. Hipótesis

El sistema de domótica con Arduino Mega 2560 y Arduino Ethernet Shield aporta significativamente al control y seguridad mediante la monitorización desde un servidor Web al ser implementado en el sector residencial de la ciudad de Manta.

## 1.8. Variables

**Variable Independiente**

Sistema de domótica con Arduino Mega y Arduino Ethernet Shield.

**Variable Dependiente**

Servidor Web

## 1.9. Metodología de la investigación

Es una tesis de estudio Explicativo, mediante el cual se analiza, diseña y evalúa la tecnología y plataforma de Arduino. Además se da a conocer un sinnúmero de módulos y sensores de tipo electrónico que de la mano de Arduino se pueden emplear en un proyecto de Domótica con la posibilidad de ser implementado en una vivienda de la ciudad de Manta.

**Paradigma:** Empírico - Analítico

**Enfoque:** Cuantitativo

**Diseño de la Investigación**

No Experimental Transversal – en el siguiente proyecto no se realizó operaciones indebidas que perturben o cambien las características de las variables del estudio, se procedió a la observación directa de los acontecimientos físicos electrónicos y como se desenvuelven en su ámbito natural, para concluir analizando las debidas recomendaciones del caso.

**Validez**

Dadas las cualidades del proyecto y con los debidos resultados obtenidos mediantes las pruebas realizadas al prototipo, se pudo determinar con franqueza que cumple con las pautas de la validez interna de una investigación científica.

# CAPITULO 2. FUNDAMENTOS TEÓRICOS

## 2.1. Domótica


*Figura 1 - Domótica*

*Fuente: El Autor*

La domótica es el conjunto de tecnologías aplicadas al control y la automatización inteligente de la vivienda, que permite una gestión eficiente del uso de la energía, aportando seguridad y confort, además de comunicación entre el usuario y el sistema.

Un sistema domótico *(Figura 1)* es capaz de recoger información proveniente de sensores o entradas (digitales o analógicas), procesarla y emitir órdenes a los actuadores o salidas (digitales o analógicas). El sistema puede acceder a redes exteriores de comunicación o información. La domótica permite dar respuesta a los

requerimientos que plantean estos cambios sociales y las nuevas tendencias de nuestra forma de vida (CEDOM, 2014).

Los sistemas domóticos incorporan a las viviendas las últimas tecnologías informáticas y de comunicaciones, proporcionando cuantiosas ventajas a sus usuarios. Estas tecnologías ayudan a reducir consumos energéticos y a aumentar la seguridad, las posibilidades de comunicaciones y sobre todo, la comodidad (Sáez, 2012).

## 2.2. Beneficios de la domótica

El hogar inteligente suele concentrar las acciones en cuatro ámbitos diferenciados: confort, ahorro energético, comunicaciones y seguridad, cada uno de ellos con una buena variedad de equipamientos y servicios a tu disposición (aunque quizá no tan al alcance del bolsillo). La domótica trae consigo una variedad de beneficios entre los que pueden destacar:

- Ahorro: Ahorrarás tanto a nivel energético como económico, y es que ambas cosas están fuertemente vinculada. En este sentido, la gestión tarifaria de sistemas y conjuntos podrá ser controlada de modo que la factura no sea tan escalofriante a fin de mes.
- Bienestar: Garantiza algunas cuestiones que sin duda nos harán vivir más tranquilos.
- Variedad: La oferta domótica se amplía cada día que pasa ofreciendo productos de más calidad, más fáciles de utilizar y a unos precios más al alcance que en años anteriores.

Además de lo anterior, la domótica permite que todos los sistemas inteligentes puedan ser controlados por medio de redes desde dentro y fuera del hogar (RIBELLES, 2015).

## 2.3. Aplicaciones domóticas

Los servicios que ofrece la domótica se pueden agrupar en aspectos o ámbitos como se detallan a continuación (CEDOM, 2014):

- **Confort:** abrir, cerrar, apagar, encender, regular… dispositivos y actividades domésticas (iluminación, climatización, persianas, toldos, cortinas, puertas, ventanas, cerraduras, riego, electrodomésticos, suministro de agua, gas, electricidad…).
- **Gestión energética:** conexión de dispositivos de calefacción y aire acondicionado según criterios de ahorro y confort, control de toldos, persianas, cortinas y ventanas para aprovechamiento de las energías naturales, control de alumbrados, racionalización de cargas eléctricas.
- **Seguridad:** vigilancia automática de personas y bienes, e incidencias y averías, así como alarmas de intrusión, cierre automático de todas las aberturas, simulación dinámica de presencia, fachadas dinámicas, cámaras de vigilancia, alarmas personales, alarmas técnicas de incendio, humo, agua, gas, fallo del suministro eléctrico.
- **Comunicaciones:** control y supervisión remoto de la vivienda a través de su teléfono, PDA, PC…, transmisión de voz y datos, incluyendo textos, imágenes, sonidos (multimedia) con redes locales (LAN) y compartiendo acceso a Internet; recursos e intercambio entre todos los dispositivos, acceso a nuevos servicios de telefonía IP, televisión digital, por cable, diagnóstico remoto, videoconferencias, etc..

## 2.4. Arduino

Arduino es una plataforma de prototipos de código abierto basado en hardware y software fácil de usar. Las Placas de Arduino son capaces de leer los insumos - la luz en un sensor, un dedo en un botón, o un mensaje de Twitter, y lo convierten en una salida; la activación de un motor, encender un LED, publicar algo en línea.

A través de los años Arduino ha sido el cerebro de miles de proyectos, a partir de objetos cotidianos a los instrumentos científicos complejos. Una comunidad mundial de los fabricantes, estudiantes, aficionados, artistas, programadores y profesionales, ha reunido alrededor de esta plataforma de código abierto, sus contribuciones han añadido hasta una cantidad increíble de conocimiento accesible que puede ser de gran ayuda para los principiantes como para expertos.

Arduino nació en INTERACTION DESIGN INSTITUTE IVREA como una herramienta fácil para prototipado rápido, dirigido a estudiantes sin experiencia en electrónica y programación. Tan pronto como llegó a una comunidad más amplia, la placa Arduino comenzó a cambiar para adaptarse a las nuevas necesidades y desafíos, diferenciar su oferta de tablas simples de 8 bits, la impresión 3D portátil y entornos integrados. Todas las placas Arduino son completamente de código abierto, permitiendo a los usuarios crear de forma independiente y, finalmente, adaptarlos a sus necesidades particulares. El software también es de código abierto, y está creciendo a través de las aportaciones de los usuarios en todo el mundo (ARDUINO, Introducción a Arduino, 2013).

### 2.4.1. ¿Por qué Arduino?

Gracias a su sencilla y accesible experiencia de usuario, Arduino se ha utilizado en miles de proyectos diferentes y aplicaciones. El software de Arduino es

fácil de usar para los principiantes, pero lo suficientemente flexible para los usuarios avanzados. Se ejecuta en Mac, Windows y Linux. Los profesores y los estudiantes lo utilizan para construir los instrumentos científicos de bajo coste, para demostrar los principios de química y física, o para empezar con la programación y la robótica. Cualquier persona, niños, aficionados, artistas, programadores, puede comenzar a juguetear simplemente siguiendo paso a paso las instrucciones de un kit, o compartir ideas en línea con otros miembros de la comunidad de Arduino (ARDUINO, Introducción a Arduino, 2013).

Arduino también simplifica el proceso de trabajar con microcontroladores, pero ofrece algunas ventajas para los profesores, estudiantes y aficionados interesados sobre otros sistemas:

- **Asequible:** placas Arduino son relativamente baratos en comparación con otras plataformas de microcontroladores. La versión menos costosa del módulo Arduino puede ser montado a mano, e incluso los módulos de Arduino premontados cuestan menos de $ 50.
- **Multiplataforma:** El software de Arduino (IDE) se ejecuta en los sistemas operativos Windows, Macintosh OSX y Linux.
- **Entorno de programación simple:** El software de Arduino (IDE) es fácil de usar para los principiantes, pero lo suficientemente flexible para los usuarios avanzados. Para los profesores, se basa convenientemente en el entorno de programación Processing, por lo que los estudiantes que aprenden a programar en ese entorno estarán familiarizados con cómo funciona el Arduino IDE.
- **El código abierto y el software extensible**: El software de Arduino está publicado como herramientas de código abierto, disponible para la extensión por programadores experimentados.

- **Código abierto y el hardware extensible**: Los proyectos de las placas Arduino se publican bajo una licencia de Creative Commons, por lo que los diseñadores de circuitos experimentados pueden hacer su propia versión del módulo, ampliándolo y mejorándolo. Incluso los usuarios con poca experiencia pueden construir la versión del módulo con el fin de entender cómo funciona y ahorrar dinero.

## 2.5. Garantía de Productos Arduino

Los productos Arduino destinados a la venta y uso, en los mercados mundiales cumplen con los requisitos internacionales aplicables para la seguridad del producto, la compatibilidad electromagnética (EMC), la información de uso y seguridad esencial, WEEE, RoHS, calidad, y para su uso en lugares peligrosos (ARDUINO, Introducción a Arduino, 2013).

### 2.5.1. Fabricación

Todos los componentes y aleaciones de soldadura utilizados en este producto cumplen con la Directiva RoHS. La Directiva RoHS impide a todos los nuevos aparatos eléctricos y electrónicos puestos en el mercado Europeo de contener más de los niveles acordados de plomo, cadmio, mercurio, cromo hexavalente, bifenilos poli-bromado (PBB) y éteres difenil-poli bromados (PBDE ) (ARDUINO, Introducción a Arduino, 2013).

### 2.5.2. Seguridad

Todas las tarjetas están marcadas con el logotipo de la FCC y CE, que cumplan con las normas de compatibilidad electromagnética establecidos en sus respectivas

jurisdicciones. Productos Arduino cumplen los requisitos esenciales de la Directiva de la UE 2001/95 / CE (Directiva general sobre los productos de seguridad y de la Directiva 93/68 / CE) (ARDUINO, Introducción a Arduino, 2013).

### 2.5.3. Cumplimiento fcc

Este dispositivo cumple con la Parte 15 de las Reglas de la FCC. La operación está sujeta a las dos condiciones siguientes: (1) este dispositivo no puede causar interferencias perjudiciales y (2) este dispositivo debe aceptar cualquier interferencia recibida, incluyendo interferencias que puedan causar un funcionamiento no deseado (ARDUINO, Introducción a Arduino, 2013).

### 2.5.4. Observaciones y recomendaciones sobre las placas Arduino

Este equipo ha sido probado y cumple con los límites para un dispositivo digital de Clase B, de acuerdo con la parte 15 de las normas FCC. Estos límites están diseñados para proporcionar una protección razonable contra interferencias perjudiciales en una instalación residencial. Este equipo genera, utiliza y puede irradiar energía de radiofrecuencia y, si no se instala y utiliza de acuerdo con las instrucciones, puede causar interferencias perjudiciales en las comunicaciones de radio (ARDUINO, Introducción a Arduino, 2013).

## 2.6. Variedad de Placas Arduino

Existen muchas versiones de Arduino, en forma y tamaño. Esto nos ayuda en la variedad de aplicaciones que les podemos dar. Cada una de las placas o versiones de Arduino tiene sus ventajas y desventajas, esto dependerá de las necesidades del proyecto a realizar. En el ***Anexo 1*** se encuentran las variedades de placas Arduino existentes con sus respectivas características.

## 2.7. Microcontroladores

Un microcontrolador es un circuito integrado o chip que incluye en su interior las tres unidades funcionales de una computadora: CPU, Memoria y Unidades de E/S, es decir, se trata de una computadora completa en un solo circuito integrado programable y se destina a gobernar una sola tarea con el programa que reside en su memoria. Sus líneas de entrada/salida soportan el conexionado de los sensores y actuadores del dispositivo a controlar (Alberto, 2013).

## 2.8. Unidades funcionales de los microcontroladores de Arduino

Entre las unidades funcionales de Arduino se pueden destacar:

- Pines digitales: Se pueden configurar como entradas y salidas, siendo estos pines entradas por defecto y en modo de salida pueden proporcionar una corriente de hasta los 40mA a otros dispositivos o circuitos.
- Pines Análogos: Si bien la función principal de los pines analógicos para la mayoría de usuarios de Arduino es leer sensores analógicos, los pines analógicos también tienen toda la funcionalidad de propósito general pines de entrada/ salida.
- PWM: Es una técnica para obtener resultados análogos con medios digitales. El control digital se utiliza para crear una onda cuadrada, una señal cambia entre encendido y apagado.
- Memoria: Las Placas Arduino contienen de 3 partes de memoria: Memoria flash (espacio de programa), SRAM (donde el sketch crea y manipula las variables cuando se ejecuta), EEPROM (memoria para almacenar información a largo plazo).

## 2.9. Arduino MEGA 2560

Arduino MEGA 2560 *(Figura 2)* está diseñado para los proyectos más complejos. Con 54 pines digitales E/S, 16 entradas analógicas y un espacio más grande para su sketch, esta es la placa recomendada para las impresoras 3D y proyectos de robótica. Esto le da a sus proyectos con mucho espacio y oportunidades (ARDUINO, Arduino MEGA 2560, 2014).

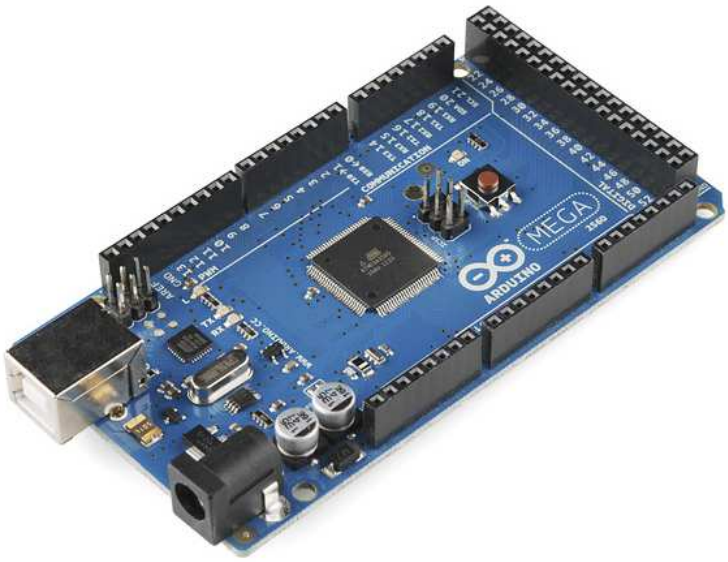



*Figura 2 - Arduino Mega 2560*

*Fuente: https://www.5hertz.com*

### 2.9.1. Información general

El Arduino Mega 2560 es una placa electrónica basada en el Atmega2560. Cuenta con 54 pines digitales de entrada / salida (***Anexo 3***), de los cuales 15 se pueden utilizar como salidas PWM, 16 entradas analógicas, 4 UARTs (puertos serie), un oscilador de cristal de 16 MHz, una conexión USB, un conector de alimentación, una cabecera ICSP, y un botón de reinicio. Contiene todo lo necesario para apoyar el microcontrolador; simplemente conectarlo a un ordenador con un cable USB o el poder con un adaptador de CA o la batería a CC para empezar. Arduino Mega 2560 es

compatible con la mayoría de los Shields diseñados para Arduino Uno y las antiguas placas Duemilanove o Diecimila (ARDUINO, Arduino MEGA 2560, 2014).

### 2.9.2. Ficha técnica

En la *Tabla 1* se muestran las especificaciones técnicas del Arduino Mega 2560, y en la *Figura 3* se muestran las dimensiones de la placa.

| | |
|---|---|
| Microcontroladores | Atmega2560 ***(Anexo 4)*** |
| Tensión de funcionamiento | 5V |
| Voltaje de entrada (recomendado) | 7-12V |
| Voltaje de entrada (límite) | 6-20V |
| Digital pines I / O | 54 (de las cuales 15 proporcionan PWM) |
| Pines de entrada analógica | 16 |
| Corriente DC por E / S Pin | 20 mA |
| Corriente DC de 3.3V Pin | 50 mA |
| Memoria flash | 256 KB y 8 KB de gestor de arranque |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Velocidad de reloj | 16 MHz |
| Largo | 101.52 mm |
| Ancho | 53,3 mm |
| Peso | 37 g |

*Tabla 1 - Ficha Técnica Arduino Mega 2560*

*Fuente: https://www.arduino.cc/en/Main/ArduinoBoardMega2560*

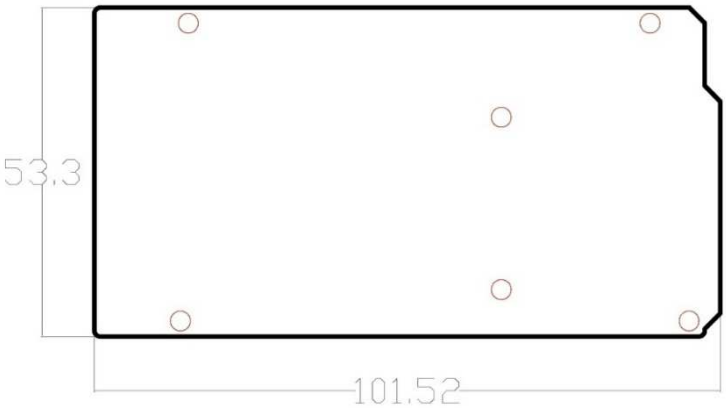



*Figura 3 - Dimensiones del Arduino Mega 2560 (medidas en milímetros).*

*Fuente: https://www.arduino.cc*

### 2.9.3. Programación

La placa Arduino Mega 2560 se puede programar con el software de Arduino. Los Atmega2560 en el Arduino Mega 2560 viene preprogramado con un gestor de arranque que le permite cargar nuevo código a él sin el uso de un programador de hardware externo.

### 2.9.4. Advertencias

El Arduino Mega 2560 tiene un polifusible reajustable que protege al puerto USB de la computadora de cortos y sobrecorriente. Si hay más de 500 mA aplicados al puerto USB, el fusible se rompe automáticamente interrumpiendo la conexión hasta que el corto o sobrecarga se elimina.

### 2.9.5. Alimentación

El Arduino Mega 2560 puede ser alimentado a través de la conexión USB o con una fuente de alimentación externa. Se recomienda que la placa trabaje con un rango de voltaje de 7 a 12V. La placa cuenta con unos pines de alimentación que se muestran en la *Figura 4* y son:

- **Vin.** A través de este pin es posible proporcionar alimentación a la placa.
- **5V.** Suministra un voltaje de 5V y una corriente de 40mA.
- **3V3.** Suministra un voltaje de 3.3V y una corriente máxima de 50mA.
- **GND.** Pines Ground (tierra)
- **IOREF.** Este pin proporciona la referencia de tensión con la que opera el microcontrolador.

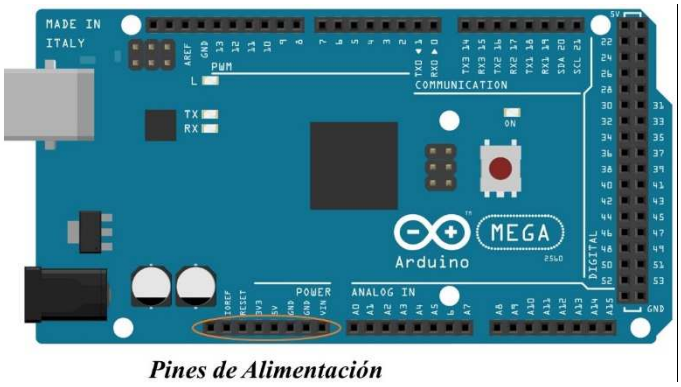



*Figura 4 - Pines de Alimentación del Arduino mega 2560*

*Fuente: El Autor*

### 2.9.6. Memoria

El Atmega2560 tiene 256 KB de memoria flash para almacenar el código (de los cuales 8 KB se utiliza para el cargador de arranque), 8 KB de SRAM y 4 KB de EEPROM (que puede ser leído y escrito con la biblioteca EEPROM).

### 2.9.7. Entradas y salidas

Cada uno de los 54 pines digitales en el Mega se puede utilizar como una entrada o salida, utilizando las funciones pinMode (), digitalWrite (), y digitalRead (). Cada pin opera a 5V y puede proporcionar o recibir 20 mA y como máximo 40mA. Arduino Mega 2560 tiene 16 entradas analógicas, cada uno de los cuales proporcionan 10 bits de resolución (es decir, 1024 valores diferentes). Además, algunos pines tienen funciones especializadas:

- **Serial:** Se utiliza para recibir y transmitir datos en serie.
- **Interrupciones externas:** Estos pines pueden configurarse para activar una interrupción en un nivel bajo, un flanco ascendente o descendente.
- **PWM (Pulse Width Modulation):** Proporcionar una salida PWM de 8 bits con la función analogWrite ().

- **SPI (Serial Peripheral Interface):** Estos pines soportan la comunicación SPI utilizando la biblioteca de SPI.
- **LED:** Hay un LED incorporado conectado al pin digital 13.
- **TWI (Two Wire Interface):** Soporta comunicación TWI utilizando la biblioteca Wire Library.
- **AREF (analogReference):** Tensión de referencia para las entradas analógicas.
- **Reset:** Colocar a 0V (Ground) para  para reiniciar el microcontrolador.

### 2.9.8. Asignación de los pines Atmega2560 – Arduino

En la *Figura 5* se muestra la asignación de pines para el Atmega2560, chip utilizado en la placa Arduino mega 2560. En *Anexo 5* se detalla la asignación de pines que le da el microcontrolador al Arduino.

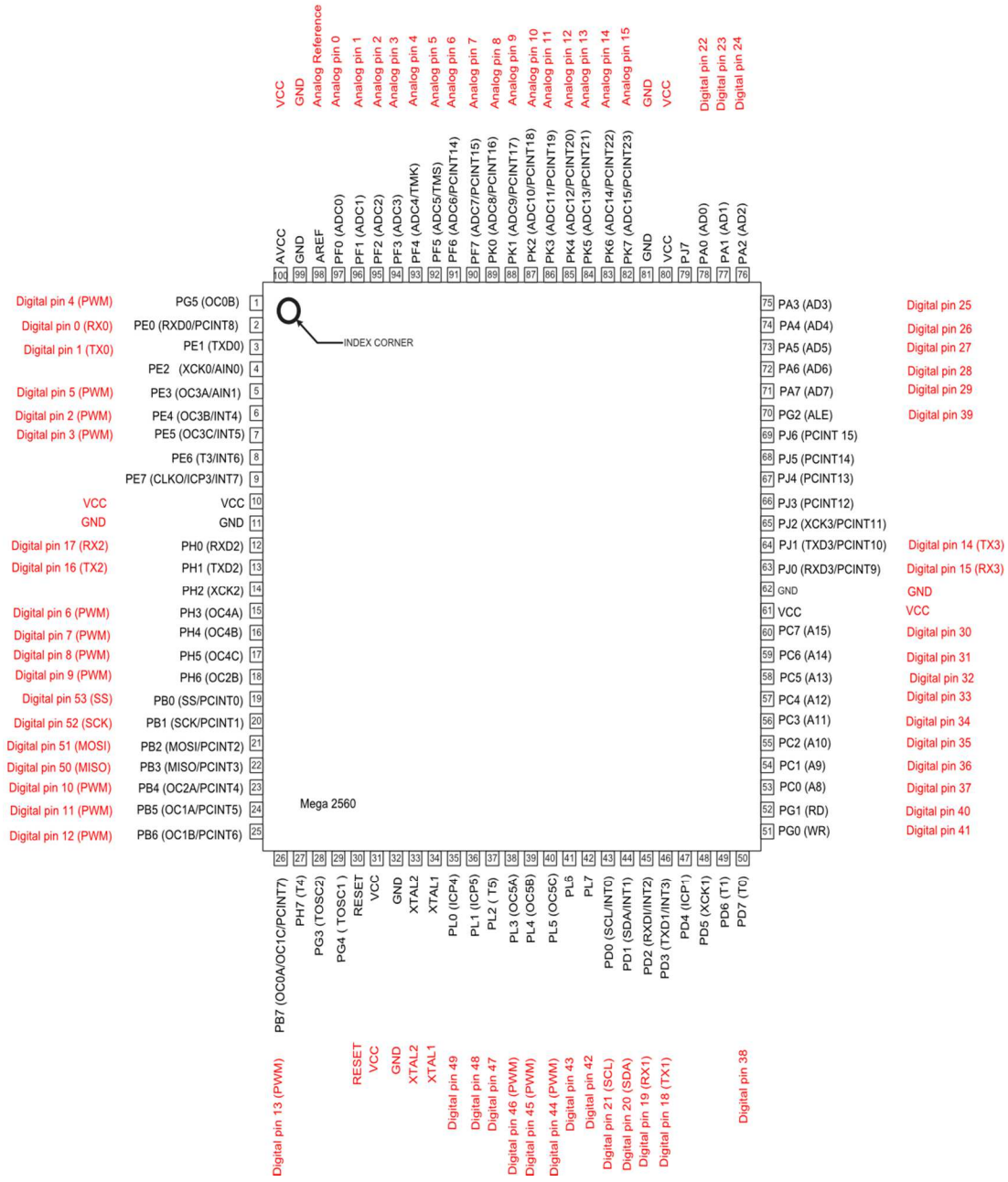



*Figura 5 - Distribución de los pines en el chip Atmega 2560*

*Fuente: https://www.arduino.cc*

### 2.9.9. Comunicación

La placa Arduino Mega 2560 tiene una serie de facilidades para la comunicación con un ordenador, con otra placa, u otros microcontroladores. El Atmega2560 provee de hardware UART para comunicación serie de TTL (5V). Un ATmega16U2 en la placa a través del USB, proporciona un puerto COM virtual para el software en el equipo.

El Arduino Mega 2560 también es compatible con la comunicación TWI y SPI. El software de Arduino (IDE) incluye una librería Wire para simplificar el uso del bus TWI (ARDUINO, Arduino MEGA 2560, 2014).

### 2.9.10. Compatibilidad de Shields

Arduino Mega 2560 está diseñado para ser compatible con la mayoría de los Shields[1] diseñados para Arduino Uno y las placas Arduino Diecimila o Duemilanove. A diferencia de las otras placas I2C en Arduino mega se encuentra ubicado en los pines 20 y 21.

### 2.9.11. Restablecimiento automático

En lugar de presionar físicamente el botón de reinicio antes de una carga del software, el Arduino Mega 2560 está diseñado de una manera que permite que sea restablecido por el software que se ejecuta en un ordenador conectado. Una de las líneas de control de flujo de hardware (DTR) de la ATmega8U2 está conectado a la línea de restablecimiento de los Atmega2560 través de un condensador de 100 nanofaradios, cuando esta línea se impone, la línea de reset cae lo suficientemente como para restablecer el chip (ARDUINO, Arduino MEGA 2560, 2014).

[1] Shield para Arduino: es una tarjeta de expansión que provee funcionalidades adicionales a la tarjeta Arduino.

## 2.10. Arduino Ethernet Shield

*Figura 6 - Arduino Ethernet Shield*

*Fuente: https://www.arduino.cc*

### 2.10.1. Información general

El Arduino Ethernet Shield *(Figura 6)* conecta tu Arduino a Internet en cuestión de minutos. Sólo tiene que conectar este módulo en una placa Arduino, conectarlo a la red mediante un cable con conector RJ45 y seguir algunas instrucciones sencillas para empezar a controlar su mundo a través de internet (ARDUINO, Arduino Ethernet Shield, 2012).

### 2.10.2. Datos técnicos

- Requiere una placa Arduino.
- Utiliza 5V para su funcionamiento (suministrado por la placa Arduino)
- Controlador Ethernet: W5100 con buffer interno 16K
- Velocidad de conexión: 10 / 100Mb
- Conexión con Arduino mediante el puerto SPI

### 2.10.3. Descripción

La placa Arduino Ethernet Shield se basa en el chip de Ethernet Wiznet W5100, el mismo que ofrece una red (IP) capaz de unir TCP y UDP. El Ethernet Shield tiene una conexión RJ-45 estándar, con un transformador de línea integrado y tiene habilitada la alimentación a través de internet. Tiene una ranura para tarjetas microSD, que se puede utilizar para almacenar archivos, que luego pueden ser utilizados a través de la red. El Shield contiene una serie de LEDs informativos *(Figura 7)*:

- **PWR:** indica que la placa y el Shield Ethernet están encendidos.
- **LINK:** indica la presencia de un enlace de red.
- **FULLD:** indica que la conexión de red es full dúplex.
- **100M:** indica la presencia de 100 Mb/s de conexión en la red.
- **RX:** Parpadea cuando el Shield Ethernet recibe datos.
- **TX:** parpadea cuando el Shield Ethernet envía datos.
- **COLL:** parpadea cuando se detectan colisiones de red.

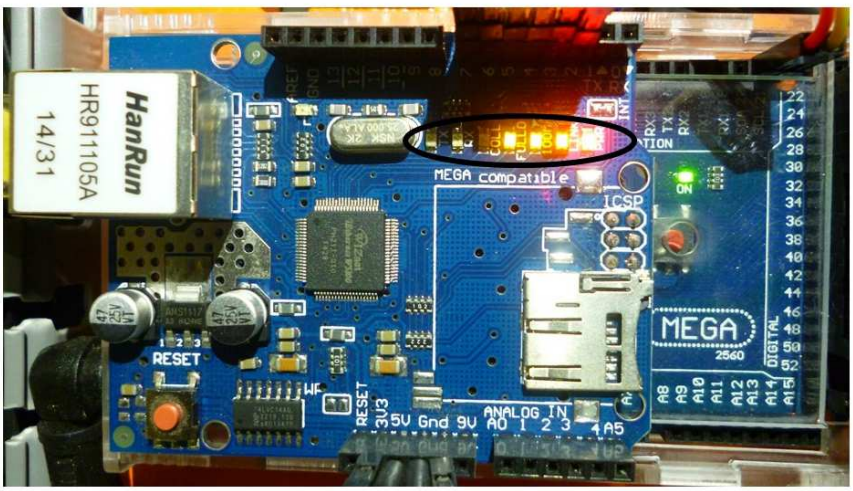



*Figura 7 - Leds Informativos del Arduino Ethernet Shield*

*Fuente: El Autor.*

### 2.10.4. Configuración de la red

Al Shield se le debe asignar una dirección MAC y una dirección IP fija mediante la función Ethernet.begin (). Es posible utilizar DHCP para asignar dinámicamente una dirección IP al Shield, opcionalmente, también puede especificar la máscara de subnet y una puerta de enlace (ARDUINO, Arduino Ethernet Shield, 2012).

### 2.10.5. Chip W5100


*Figura 8 - Chip W1500*
*Fuente: http://www.wiznet.co.kr*

El chip W5100 *(Figura 8)* es un controlador de Ethernet programado para TCP/IP que permite la fácil conexión a Internet para sistemas embebidos. El W5100 se adapta a los usuarios que necesitan una conectividad estable a Internet, utilizando un solo chip para implementar TCP / IP, 10/100 Ethernet MAC y PHY. W5. En el *Anexo 11* podemos observar el diagrama de bloques del chip W5100 y en el *Anexo 12* la asignación de pines del chip (WIZnet Co., 2013).

#### *2.10.5.1. Características principales.*

- Protocolos TCP / IP: TCP, UDP, ICMP, IPv4, ARP, IGMP, PPPoE.
- Interfaz Host: Directo / Indirecto Autobuses y Serial Peripheral Interface (SPI).
- 4 Toma de hardware independiente.
- Memoria Interna 16Kbytes para el procesamiento de paquetes TCP / IP.
- Soporte Auto-MDIX.
- 3.3V Operación con la tolerancia de la señal de E / S 5V I.
- Salidas LED (TX, RX, full/half duplex, Collision, link, speed).
- Paquete sin plomo 80LQFP (10x10mm) (Electronics, 2012).

## 2.11. Módulos para Arduino

Un módulo de Arduino consiste en una placa electrónica compuesta por una serie de elementos electrónicos, dicha placa, elaborada, realizada o fabricada para realizar una función específica, a su vez la misma puede ser montada encima de los pines de la placa Arduino (siempre y cuando el modulo este fabricado para ser montado encima de la placa) o sino conectada al Arduino mediante cables adecuados, como los de la *Figura 9*.

*Figura 9 - Cables Tipo Dupont para Arduino*

*Fuente: https://altronics.cl*

### 2.11.1. Módulo de relé de 8 canales

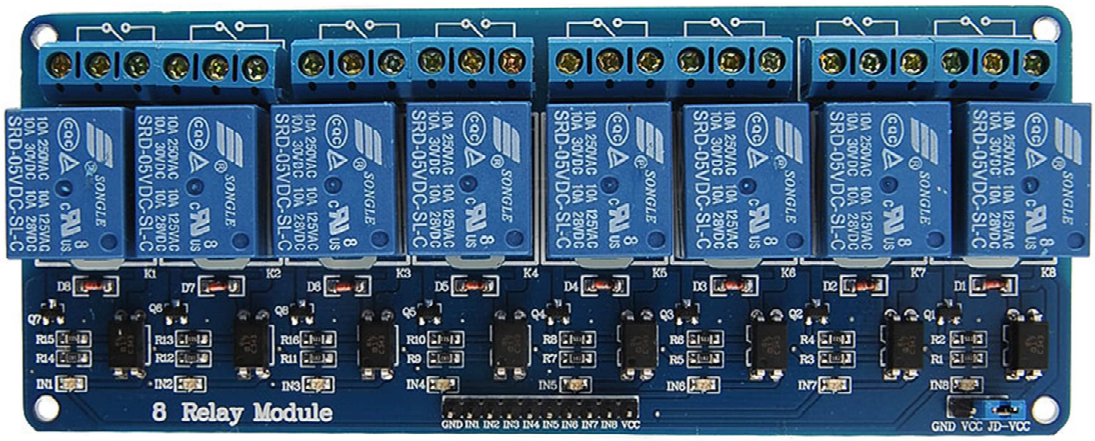



*Figura 10 - Modulo de Relé de 8 Canales*

*Fuente: http://forum.arduino.cc*

Modulo compuesto por 8 relés *(Figura 10)* que puede ser controlado por una amplia gama de microcontroladores como: Arduino, Raspberry Pi, 8051, AVR, PIC, ARM, ARM, MSP430, DSP, entre otros. Trabaja con 5V DC y es capaz de controlar cualquier aparato eléctrico, cada relé tiene la capacidad de controlar hasta 250V AC, 10Amp.  y hasta 30V DC, 10 Amp.

Cuenta con leds indicadores del estado de cada canal, en la parte del centro se encuentran localizados 10 pines de los cuales 8 son los destinados a los 8 canales y los otros dos restantes para alimentar la placa a VCC (5V) y GND respectivamente, también cuenta con unos pines extras (GND, VCC y JD VCC) los cuales solos se usaran para alimentar el modulo cuando este sea ubicado a una mayor distancia desde donde se encontraría el microcontrolador que lo esté controlando, en caso de no ser así estos pines estarían puenteados (VCC con JD VCC). Las dimensiones del módulo son 13.4 x 5.3 x 1.7 cm respectivamente, y un peso de 116 gramos.

### 2.11.2. Módulo Display LCD (Liquid Crystal Display) 1604A


*Figura 11 - LCD 1604A*

*Fuente: http://es.aliexpress.com*

Modulo destinado de salida destinado a mostrar la información que provea el microcontrolador, cuenta con una pantalla LCD (Liquid Crystal Display) *(Figura 11)* que cuenta con 4 líneas y en cada línea puede mostrar hasta 16 caracteres. El modulo tiene 16 pines los cuales se detallan a continuación:

- VSS: Corresponde al Pin GND (0V) o pin negativo.
- VDD: Pin positivo 5V que alimenta todo el modulo.
- V0: es el contraste del display, se conecta a GND.
- RS: corresponde al selector de registro, controlado por el microcontrolador.
- RW: pin que ejecuta la lectura o escritura.
- E: (Enable) Habilita la pantalla respectivamente para recibir información.
- D0 – D7: Líneas de comunicación por donde se transfieren los datos.
- LED A: Pin positivo del Backlight (Fondo de Pantalla). Se conecta a 3.3V.
- LED K: Pin Negativo del Backlight. Se conecta a GND.

### 2.11.3. Módulo de pantalla LCD Gráfico Nokia 5110

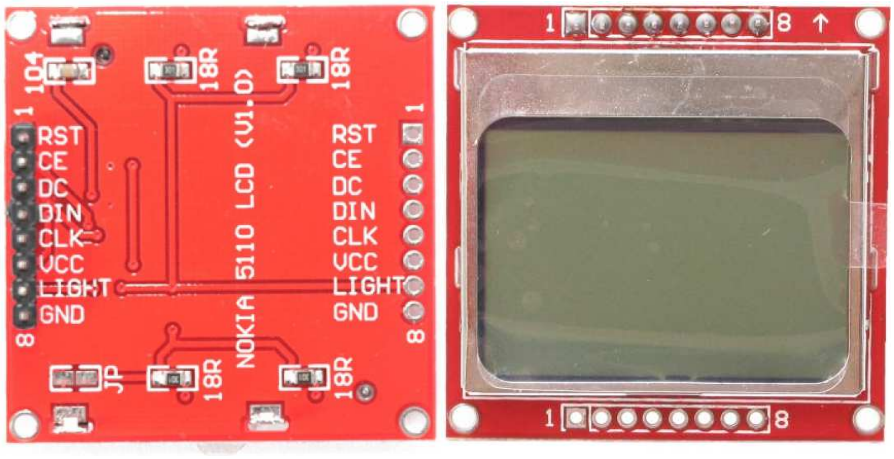



*Figura 12 - LCD Gráfico Nokia 5110*

*Fuente: http://electronilab.co*

El LCD Gráfico de Nokia 5110 *(Figura 12)*, es una pequeña pantalla gráfica LCD montada sobre un PCB de 45mm x 45mm, que tiene una resolución de 84 x 48 píxeles sobre los que podemos dibujar gráficos o textos. La pantalla utiliza el chip controlador PCD8544 de Philips que fue utilizado en el Nokia 3310 y el 5110. Este chip está diseñado para funcionar sólo a 3.3V y tienen niveles de comunicación de 3V, por lo que para los microcontroladores de 5V se requiere un conversor de nivel lógico o resistencias de limitación de corriente. (Ardumanía, 2013)

Existen varias librerías que funcionan con este LCD, sin embargo, como existen varios modelos, los pines pueden cambiar de nombre y posición y por tanto su uso puede resultar complicado. Los pines del LCD se muestran en la *Tabla 2* y son los siguientes:

| Pin | Nombre del Pin | Función del Pin |
|---|---|---|
| 1 | RST | Reset |
| 2 | CE | Selección de Chip |
| 3 | DC | Data/Commands choice |
| 4 | DIN | Serial data in |
| 5 | CLK | Serial Clock |
| 6 | VCC | Alimentación Positiva (Conectar a 3.3V) |
| 7 | LIGHT | Alimentación de Backlight (Conectar a GND 0V) |
| 8 | GND | Ground (Conectar a GND 0V) |

*Tabla 2 - Pines del módulo LCD Grafico Nokia 5110*

*Fuente: http://electronilab.co*

### 2.11.4. Módulo TinyRTC DS1307

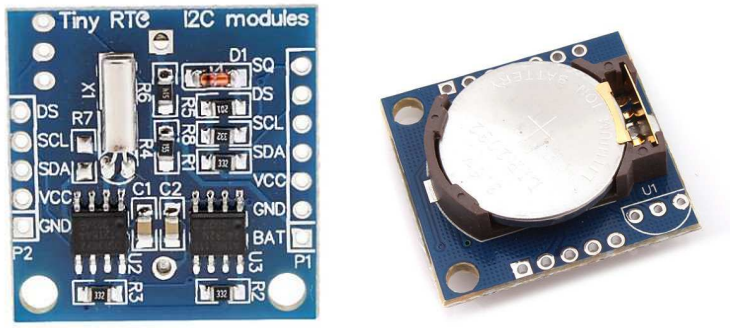



*Figura 13 - Módulo RTC DS1307*

*Fuente: http://www.taringa.net*

El RTC o Reloj en Tiempo Real *(Figura 13)* es un circuito electrónico especializado cuya función es mantener la hora y fecha actual en un sistema informático (ya sea con microcontrolador u otro tipo de CPU). Se caracteriza por tener un bajo consumo de energía y también normalmente su propia fuente de alimentación auxiliar. Normalmente al recurrir a este tipo de circuitos integrados obtenemos una mejor precisión del tiempo. Un ejemplo de dispositivos que incluyen relojes en tiempo real son las computadoras personales (PC) (Geekfactory, 2014).

## 2.12. Sensores para Arduino

Los sensores son dispositivos que nos permiten recibir información del entorno o exterior y a su vez interactuar con ella, son capaces de detectar magnitudes físicas o químicas y transformarlas en magnitudes eléctricas.

### 2.12.1. Sensor de Temperatura y Humedad DHT11

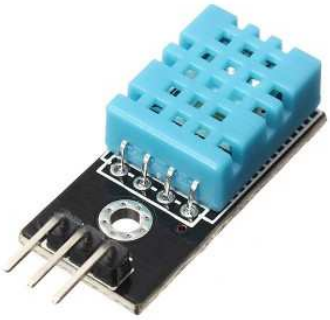


*Figura 14 - Sensor DHT11*

*Fuente: http://www.nextiafenix.com*

El DHT11 *(Figura 14)* es un sensor de temperatura y humedad digital de bajo costo. Utiliza un sensor capacitivo de humedad y un termistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos. En la *Tabla 3* se muestran las características del sensor (Electronilab, 2013).

| **Características** | |
|---|---|
| Alimentación: | 3Vdc ≤ Vcc ≤ 5Vdc. |
| Precisión de medición de temperatura: | ±2.0 °C. |
| Resolución Temperatura: | 0.1°C. |
| Rango de medición de humedad: | 20% a 90% RH. |
| Precisión de medición de humedad: | 4% RH. |
| Resolución Humedad: | 1% RH. |
| Tiempo de censado: | 1 seg. |

*Tabla 3 - Características del Sensor DHT11*

*Fuente: http://electronilab.co*

### 2.12.2. Sensor de movimiento PIR HC-SR501

*Figura 15 - Sensor PIR HC-SR501*

*Fuente: http://www.satkit.com*

El módulo HC-SR501 (*Figura 15*) tiene 3 pines de conexión +5v, OUT (3,3v) y GND, y dos resistencias variables de calibración (Ch1 y RL2). En la *Tabla 4* se muestran las características del sensor.

- Ch1: Con esta resistencia podemos establecer el tiempo que se va a mantener activa la salida del sensor.
- RL2: Esta resistencia variable nos permite establecer la distancia de detección que puede variar entre 3-7m (Electronilab, 2013).

| **Características** | |
|---|---|
| Rango de detección: | 3 m a 7 m, ajustable mediante trimmer (Sx). |
| Salida activa alta | 3.3 V. |
| Consumo de corriente en reposo: | < 50 µA. |
| Voltaje de alimentación: | 4.5 VDC a 20 VDC. |

*Tabla 4 - Características del Sensor PIR HC-SR501*

*Fuente: http://electronilab.co*

### 2.12.3. Sensor Receptor Infrarrojo IR

*Figura 16 - Sensor IR*

*Fuente: http://www.davidmiguel.com*

El Receptor IR *(Figura 16)* es un receptor miniaturizado para sistemas de control a distancia por infrarrojos. Un diodo PIN y un preamplificador están montados en un bastidor de conductores mientras que el paquete epoxi actúa como un filtro de IR. La señal de salida demodulada puede ser decodificada directamente por un microprocesador. La mayoría de los Receptores IR son compatibles con todos los formatos comunes de datos infrarrojos de mando a distancia (Electronilab, 2013).

### 2.12.4. Sensor de Distancia de Ultrasonido HC-SR04

El HC-SR04 *(Figura 17)* es un sensor de distancias por ultrasonidos capaz de detectar objetos y calcular la distancia a la que se encuentra en un rango de 2 a 450 cm. El sensor funciona por ultrasonidos y contiene toda la electrónica encargada de hacer la medición. Su uso es tan sencillo como enviar el pulso de arranque y medir la anchura del pulso de retorno. La fórmula con la que calcula la que trabaja es la siguiente: Distancia = {(Tiempo entre Trig y el Echo) * (V.Sonido 340 m/s)}/2. En la *Tabla 5* se muestran las características del sensor (Electronilab, 2013).


*Figura 17 - Sensor HC-SR04*

*Fuente: http://electronilab.co*

| **Características** | |
|---|---|
| Dimensiones del circuito: | 43 x 20 x 17 mm. |
| Tensión de alimentación: | 5 Vcc. |
| Frecuencia de trabajo: | 40 KHz. |
| Rango máximo: | 4.5 m. |
| Rango mínimo: | 1.7 cm. |
| Duración mínima del pulso de disparo (nivel TTL): | 10 µS. |
| Duración del pulso eco de salida (nivel TTL): | 100-25000 µS. |
| Tiempo mínimo de espera entre una medida y el inicio de otra | 20 mS. |

*Tabla 5 - Características del Sensor de Distancia de Ultrasonido HC-SR04*

*Fuente: http://electronilab.co*

### 2.12.5. Sensor de humedad de suelo – Higrómetro

*Figura 18 - Sensor de Humedad del suelo*

*Fuente: http://electronilab.co*

Este sensor de humedad *(Figura 18)* puede leer la cantidad de humedad presente en el suelo que lo rodea. Es un sensor de baja tecnología, pero es ideal para el seguimiento de un jardín urbano. En la *Tabla 6* se muestran las características del sensor (Electronilab, 2013).

| **Características** | |
|---|---|
| Voltaje de operación: | 3.3V ~ 5V. |
| Dimensiones PCB: | 30mm * 16mm. |
| Dimensiones de sonda: | 60mm * 30mm. |
| Amplificador Operacional | LM393. |

*Tabla 6 - Características del Sensor de humedad de suelo*

*Fuente: http://electronilab.co*

### 2.12.6. Sensor Detector De Lluvia o Gotas de Agua

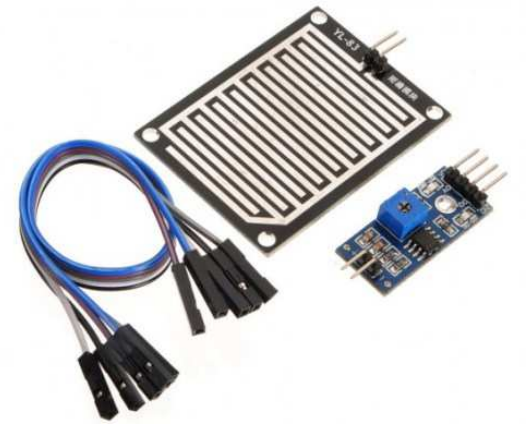


*Figura 19 - Sensor de Lluvia o gotas de agua*

*Fuente: http://electronilab.co*

Este Sensor *(Figura 19)* permite detectar gotas de lluvia, como un sensor de lluvia, y seguimiento de humedad y se puede utilizar para una variedad de condiciones climáticas. Convierte en números la señal de referencia de salida output AO. La salida analógica puede ser conectada al puerto AD de un microcontrolador para detectar la intensidad de la humedad y la precipitación. En la *Tabla 7* se muestran las características del sensor (Electronilab, 2013).

| **Características** | |
|---|---|
| Voltaje de Operación: | 3.3V-5V |
| Tamaño de PCB: | 3.2cm x 1.4cm |
| Tamaño de celda: | 5cm x 4cm |
| Chip Comparador: | LM393 |

*Tabla 7 - Características del Sensor Detector De Lluvia o Gotas de Agua*

*Fuente: http://electronilab.co*

### 2.12.7. Sensor de Gas Metano (Gas Natural) – MQ-4

*Figura 20 - Sensor de gas metano*

*Fuente: http://electronilab.co*

Este es un sensor *(Figura 20)* para detectar Gas Metano (Gas Natural) en el aire, el MQ-4 puede detectar concentraciones desde las 300 hasta las 10000 ppm. Este sensor tiene una alta sensibilidad y un tiempo de respuesta rápido. La salida del sensor tiene una resistencia analógica. En la *Tabla 8* se muestran las características del sensor (Electronilab, 2013).

| **Características** | |
|---|---|
| Alimentación | 5V DC. |
| Temperatura de funcionamiento: | -10 a 50 °C. |
| Consumo de potencia: | Menos de 900 mW. |
| Concentración: | 300 hasta las 10000 ppm. |

*Tabla 8 - Características del Sensor de gas metano*

*Fuente: http://electronilab.co*

### 2.12.8. Sensor de alcohol Etanol- MQ-3

*Figura 21 - Sensor de Alcohol Etanol*

*Fuente: http://electronilab.co*

Este sensor *(Figura 21)* detecta la concentración de alcohol en aire. Simplemente se conecta a una entrada analógica de un microcontrolador como Arduino y podremos medir la concentración de alcohol. En la *Tabla 9* se muestran las características del sensor (Electronilab, 2013).

| **Características** | |
|---|---|
| Alimentación: | 5Vdc. |
| Integrado amplificador | LM393 |
| Salida analógica | de 0 ~ 5 V |
| Condiciones de trabajo: | Temperatura ambiente -10℃ to 65℃ |

*Tabla 9 - Características del Sensor de Alcohol Etanol*

*Fuente: http://electronilab.co*

### 2.12.9. Sensor de inclinación – AT407

*Figura 22 - Sensor de Inclinación*

*Fuente: http://electronilab.co*

Este sensor *(Figura 22)* de inclinación básica AT407 puede ser fácilmente utilizado para detectar la orientación. En el interior del cilindro hay un par de bolas que hacen contacto con los pines cuando el sensor está en posición vertical. Incline el sensor y las bolas no se tocan, por lo que no hacen contacto y no existe conexión (Electronilab, 2013).

### 2.12.10. Módulo sensor de corriente ACS712 30 A

Este módulo *(Figura 23)* basado en el circuito integrado ACS712 de Allegro MicroSystems permite medir la cantidad de corriente que fluye a través de un circuito de corriente alterna (AC) o corriente directa (DC). El método de censado es a través de un sensor de efecto hall que provee un voltaje de salida proporcional a la corriente que fluye en el circuito. En la *Tabla 10* se muestran las características del sensor (Electronilab, 2013).

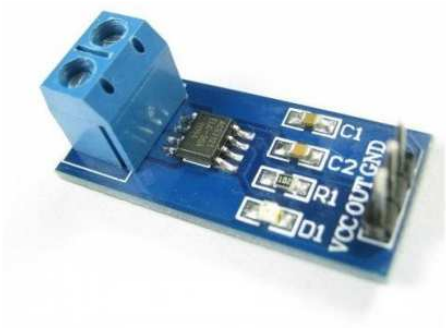



*Figura 23 - Sensor de Corriente*
*Fuente: http://electronilab.co*

| Características | |
|---|---|
| Voltaje de salida: | Analog output 66mV / A. |
| Voltaje de operación: | 4.5V ~ 5.5V. |
| Salida de voltaje sin corriente: | VCC / 2. |
| Dimensiones PCB: | 31 (mm) x14 (mm). |
| Tiempo de respuesta al paso de corriente de entrada | 5 μs. |
| Resistencia interna: | 1.2 mΩ. |
| Sensibilidad de salida: | 66 to 185 mV/A. |

*Tabla 10 - Características del Sensor de Corriente*

*Fuente: http://electronilab.co*

### 2.12.11. Sensor De Flujo De Agua G1/2 1 a 30L/min

*Figura 24 - Sensor de Flujo de Agua*

*Fuente: http://electronilab.co*

Este sensor básico *(Figura 24)* se conecta al tubo o manguera de agua, y utiliza un sensor de molinete para medir la cantidad de líquido que ha pasado a través de él. El molinillo tiene un pequeño imán atado, y hay un sensor magnético de efecto Hall en el otro lado del tubo de plástico que puede medir la cantidad de vueltas que el molinillo ha hecho a través del plástico. Este método permite que el sensor permanezca seguro y seco (Electronilab, 2013).

El sensor viene con tres cables: rojo (potencia 5 -24VDC), negro (a tierra) y amarillo (salida de impulsos de efecto Hall). Al contar los pulsos de la salida del sensor, puede seguir fácilmente el movimiento del fluido: cada pulso es de aproximadamente 2,25 mililitros. En la *Tabla 11*se muestran las características del sensor.

| **Características** | |
|---|---|
| Voltaje de funcionamiento: | de 5 a 18 VCC. |
| Rango de Trabajo: | de 1 a 30 litros / minuto. |
| Temperatura de funcionamiento: | -25 a 80 ° C. |
| Rango de Humedad de trabajo: | 35%-80% RH |
| Presión máxima del agua: | 2,0 MPa. |
| Diámetro externo: | 1.9cm. |
| Diámetro interior: | 1.1cm. |
| Longitud del cable: | 10.5cm. |
| Tamaño: | 3.3cm x 3.5cm x 6.0cm. |
| Roscas externas: | ½ pulgada. |

*Tabla 11 - Características del Sensor de Flujo de Agua*

*Fuente: http://electronilab.co*

### 2.12.12. Electroválvula – Válvula Selenoide Agua 12 VDC – 1/2″

*Figura 25 - Electroválvula de Agua*

*Fuente: http://electronilab.co*

Controlar el flujo de fluido utilizando esta válvula *(Figura 25)* es fácil. Esta válvula tiene roscas de 1/2”. Normalmente, la válvula está cerrada. Cuando se aplica 12VDC a los dos terminales, la válvula se abre y el agua puede pasar a través. Se realizó la prueba del solenoide con diversos voltajes DC y se encontró que se pudo accionar el solenoide a 6 VCC (aunque era un poco más lento para abrir). En la *Tabla 12* se muestran las características del sensor (Electronilab, 2013).

| **Características** | |
|---|---|
| Presión de trabajo: | 0.02 Mpa – 0.8 Mpa. |
| Temperatura de trabajo: | 1 ℃ – 75 ℃. |
| Tiempo de respuesta (open): | ≤ 0.15 sec. |
| Tiempo de respuesta (close): | ≤ 0.3 sec. |
| Voltaje de actuación: | 12VDC |
| Vida útil: | ≥ 50 millones de ciclos. |
| Peso: | 4.3 oz. |
| Dimensiones: | 3″ x 2.25″ x 2″. |

*Tabla 12 - Características de Electroválvula de Agua*

*Fuente: http://electronilab.co*

### 2.12.13. Sensor de Sonido


*Figura 26 - Sensor de Sonido*

*Fuente: http://www.electronicaestudio.com*

Es un pequeño sensor *(Figura 26)* basado en el LM393 y un micrófono muy sensible. Para proyectos de automatización y domótica funciona perfecto, puedes controlar luces, alarmas, incluso un pequeño robot seguidor de sonidos Tiene un potenciómetro por lo que es posible configurar el volumen sin problema. En la *Tabla 13* se muestran las características del sensor (ELECTRONICAESTUDIO, 2013).

| **Características** | |
|---|---|
| Alimentación | 5V |
| Interfaz | 3 pines. |

*Tabla 13 - Características del Sensor de Sonido*

*Fuente: http://www.electronicaestudio.com*

# CAPÍTULO 3. DISEÑO DEL SISTEMA DE DOMÓTICA

## 3.1. Descripción del Sistema de Domótica

El sistema de domótica desarrollado en el presente proyecto consiste en una serie de elementos acoplados junto a una placa de desarrollo Arduino MEGA 2560, en dicha placa además se monta un Shield de Ethernet el mismo que nos permite establecer comunicación con una red (enviar datos, recibir datos, etc.).

Existe una gran variedad de elementos que podemos acoplar a nuestro Arduino ya sea desde módulos, hasta una gran variedad de sensores existentes en el mercado, al ser, el Arduino MEGA 2560 una de las placas de desarrollo de Arduino que cuenta con la mayor cantidad de pines digitales (54), dicha placa se convierte en la ideal para este proyecto, los sensores y módulos a implementar varían de acuerdo a las necesidades que se desean satisfacer, ya que sin duda alguna no todos tenemos las mismas necesidades.

Además de cumplir con nuestras necesidades, este sistema nos ofrece comodidad y confort, porque desde donde quiera que nos encontremos solamente con tener acceso a internet, ingresamos a una IP correspondiente y podremos acceder al servidor WEB del sistema.

El servidor WEB *(Figura 27)* con el que trabaja el sistema es un servidor amigable y muy sencillo de utilizar, el mismo que nos proporciona información acerca del estado de nuestro sistema: Estado de sensores, Fecha y hora actual, Estado de salidas de actuadores, Relés etc. Además de eso desde el mismo servidor tenemos el control actuadores, relés, etc. Y con tan solo un clic podremos cambiar el estado del mismo.

Implementando un módulo de relés tenemos acceso al control de muchos elementos eléctricos y a través de nuestro servidor podemos lograr activarlo o desactivarlo, y además de eso si usamos la memoria EEPROM de nuestro Arduino podemos guardar el estado de salidas de relé o salidas digitales, y en caso que haya ausencia de energía cuando retorne la misma las salidas regresaran al estado que quedo guardado.


*Figura 27 - Interfaz de servidor Web*

*Fuente: El Autor*

## 3.2. Diagrama de flujo del sistema

Para tener un mejor detalle del proceso y funcionamiento del Sistema de domótica en las ilustraciones de diseño que se muestran a continuación se encuentra un diagrama de flujo que permitirá una mejor comprensión del sistema de domótica.

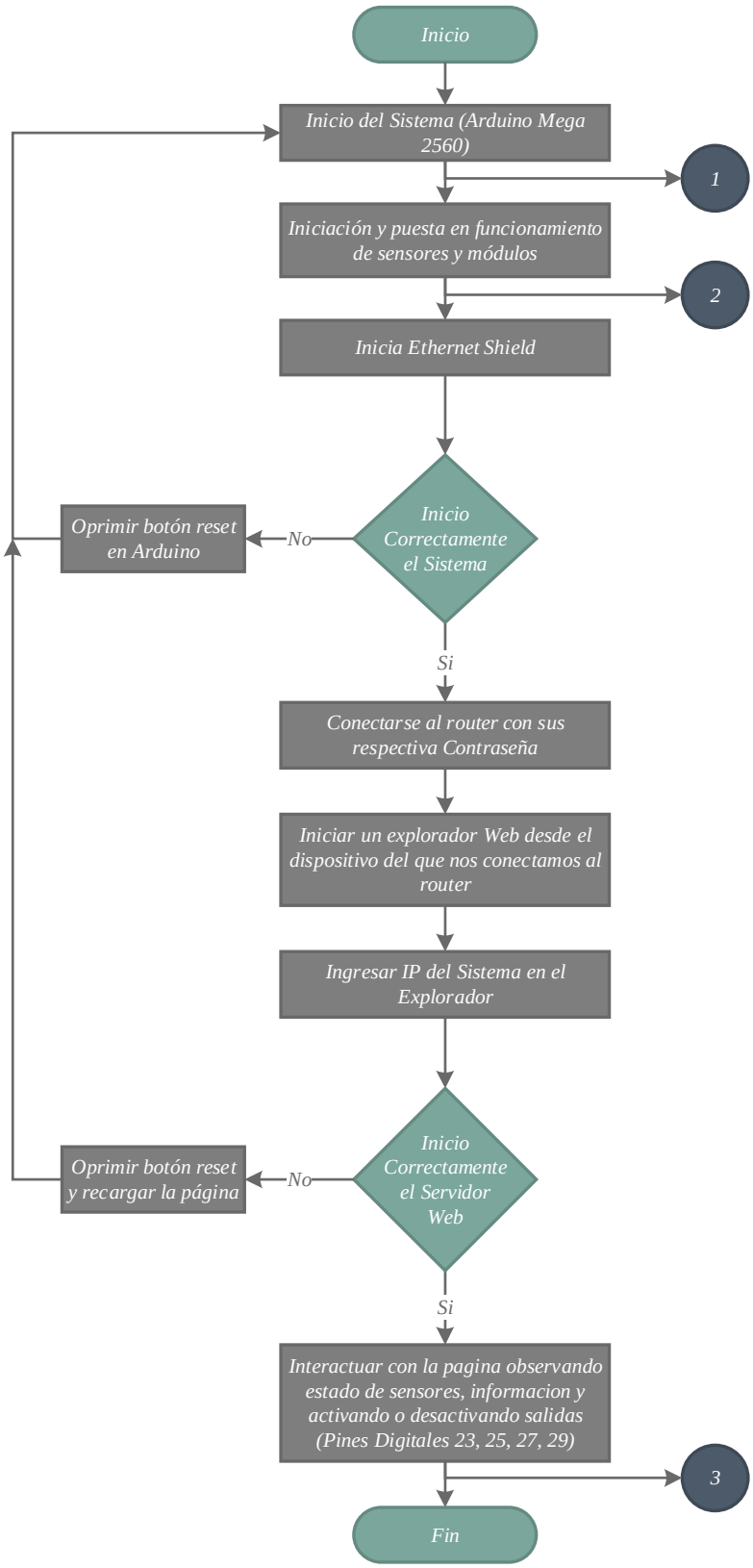



*Ilustración de Diseño 1- Diagrama de flujo del funcionamiento y proceso del Sistema de Domótica – Parte 1*

*Fuente: El Autor*

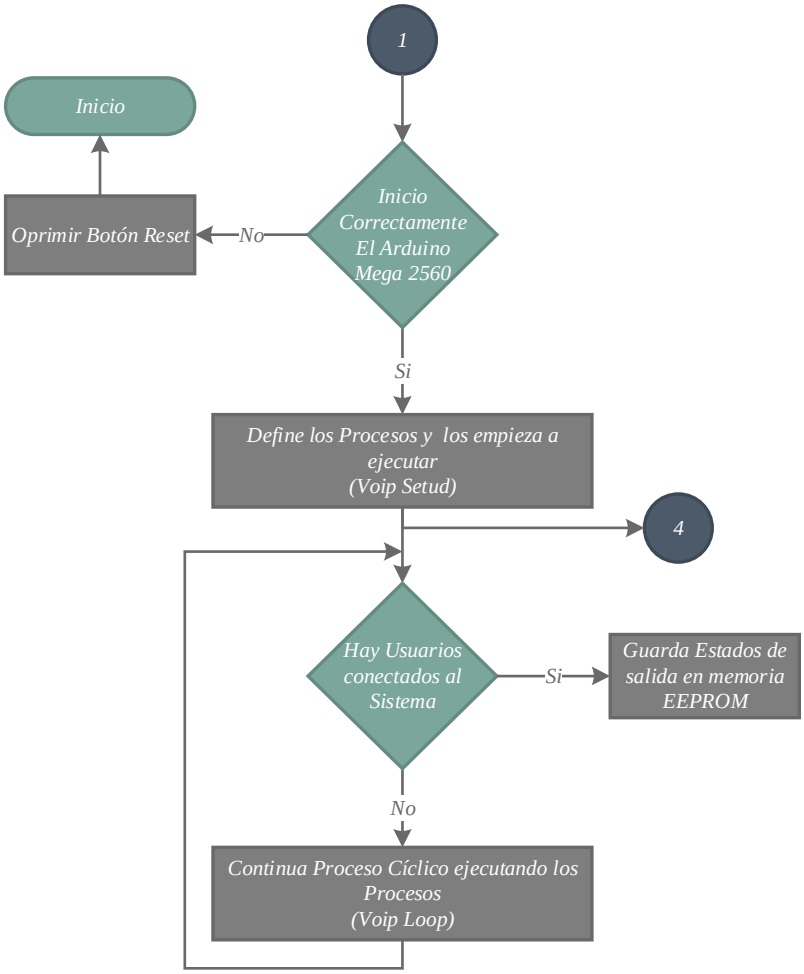



*Ilustración de Diseño 2 - Diagrama de flujo del funcionamiento y proceso del Sistema de Domótica – Parte 2*

*Fuente: El Autor*

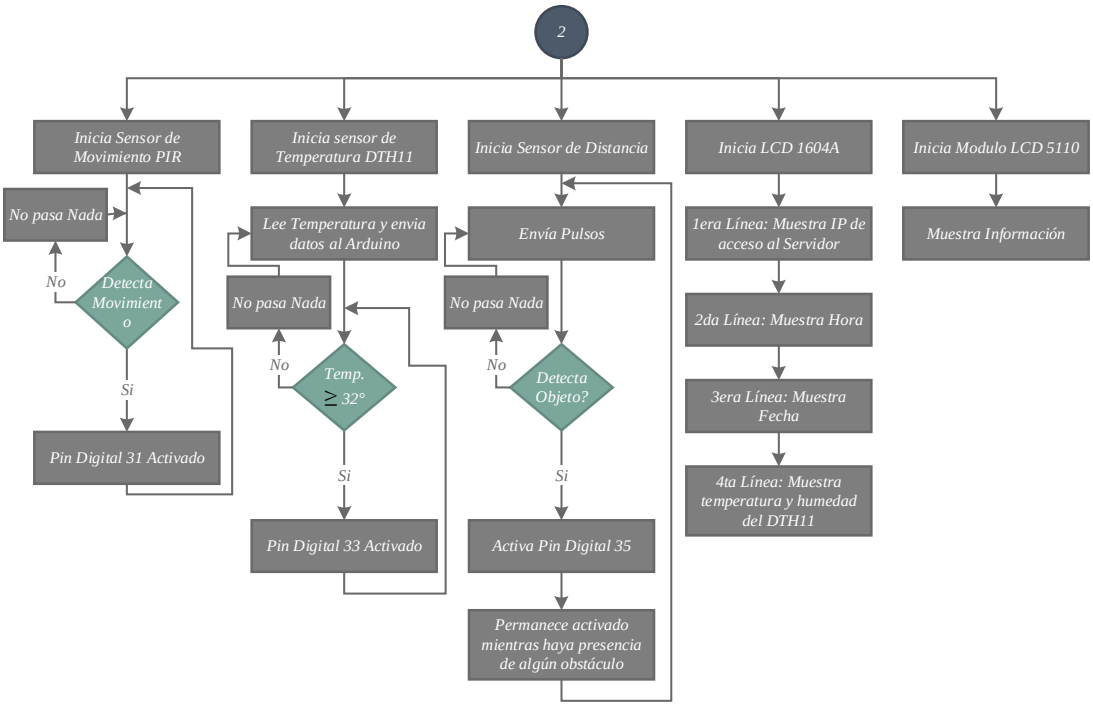



*Ilustración de Diseño 3 - Diagrama de flujo del funcionamiento y proceso del Sistema de Domótica – Parte 3*

*Fuente: El Autor*

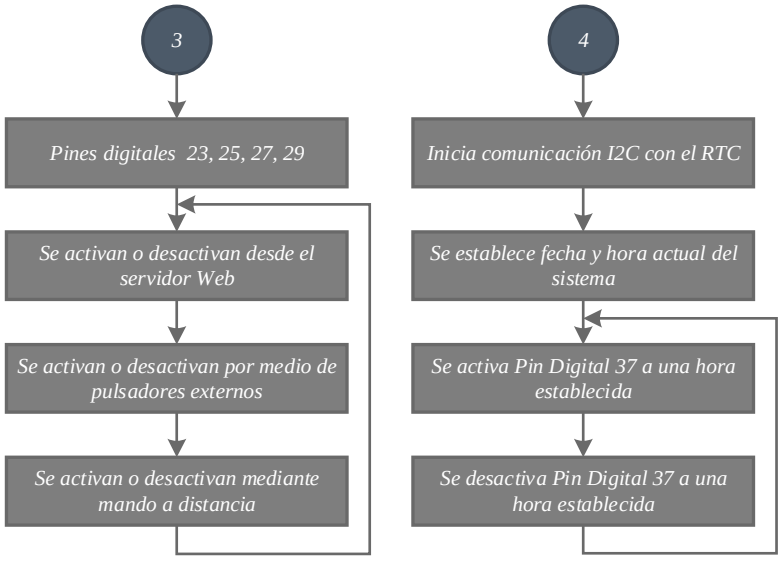



*Ilustración de Diseño 4 - Diagrama de flujo del funcionamiento y proceso del Sistema de Domótica – Parte 4*

*Fuente: El Autor*

## 3.3. Diseño de hardware del Tablero del Prototipo

Un prototipo es una versión reducida de un invento o alguna otra cosa para su demostración antes de producir una versión completa o una versión mejorada. Para demostrar el funcionamiento y comprobar los alcances del Sistema de Domótica se realizó un prototipo *(Figura 28)* en un tablero metálico con dimensiones de 60X40X20 CM. En el cual se incluyó los elementos más esenciales para que trabaje el sistema.

*Figura 28 - Prototipo del Sistema de Domótica*

*Fuente: El Autor*

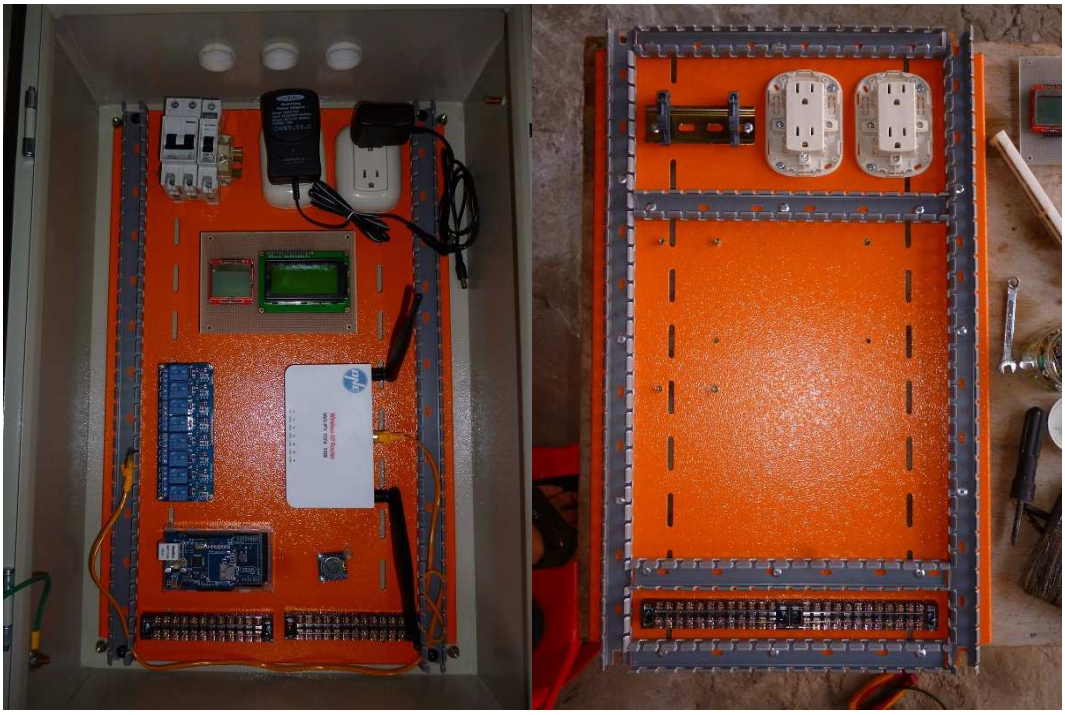


*Ilustración de Diseño 5 - Diseño del Hardware Parte 1*

*Fuente: El Autor*

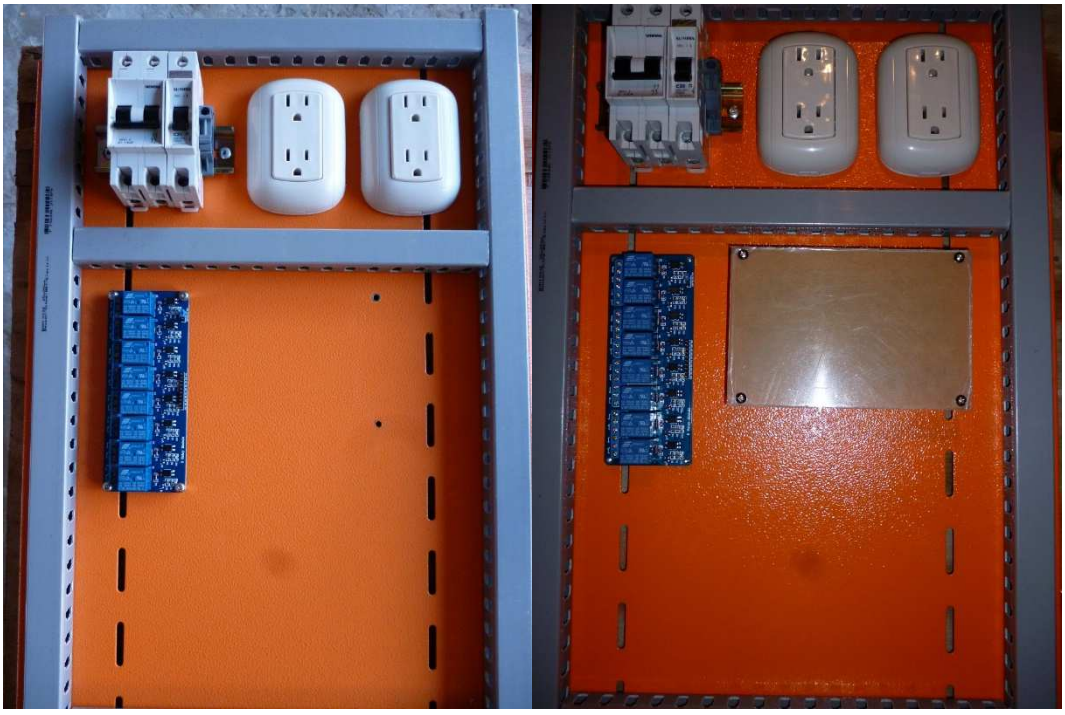


*Ilustración de Diseño 6 - Diseño del Hardware Parte 2*

*Fuente: El Autor*

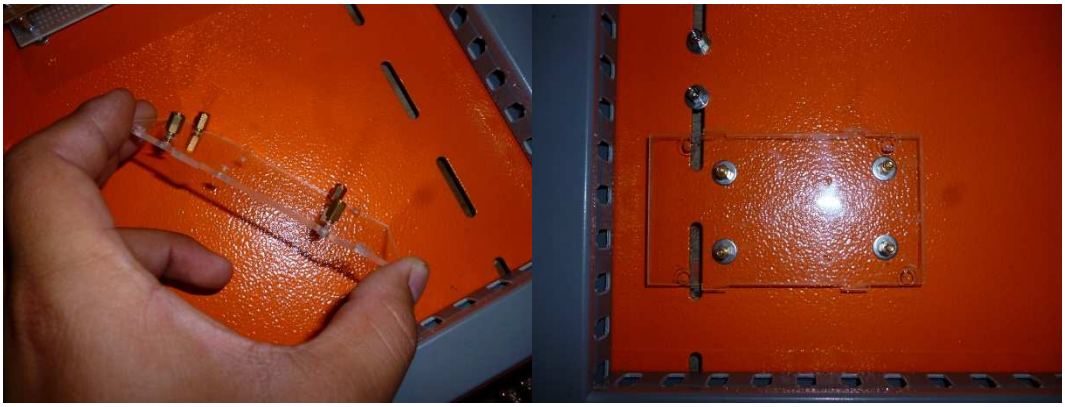


*Ilustración de Diseño 7 - Diseño del Hardware Parte 3*

*Fuente: El Autor*

*Ilustración de Diseño 8 - Diseño del Hardware Parte 4*

*Fuente: El Autor*

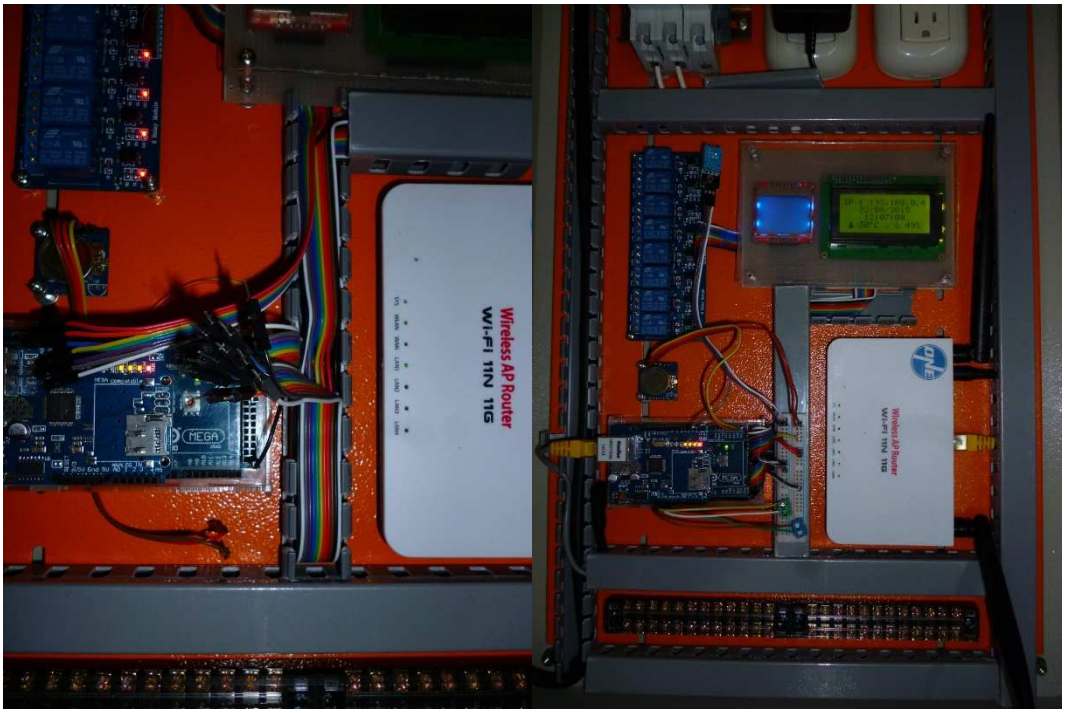



*Ilustración de Diseño 9 - Diseño del Hardware Parte 5*

*Fuente: El Autor*

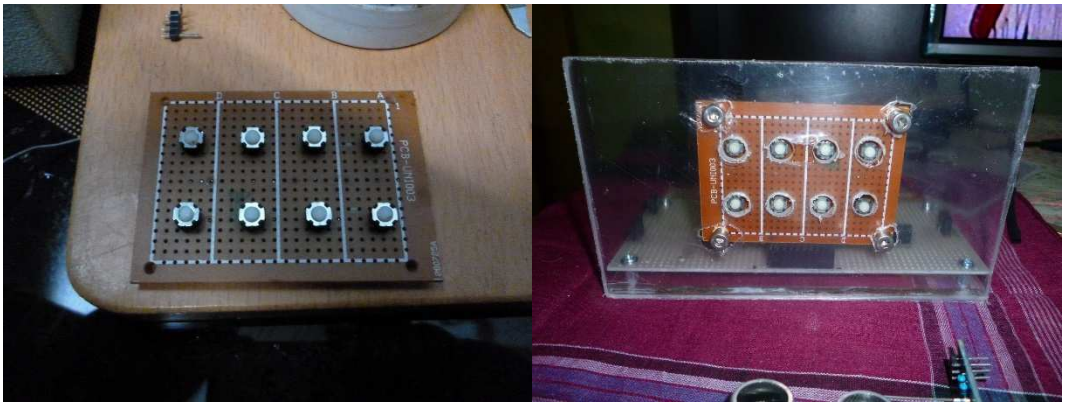


*Ilustración de Diseño 10 - Diseño del Hardware Parte 6*

*Fuente: El Autor*

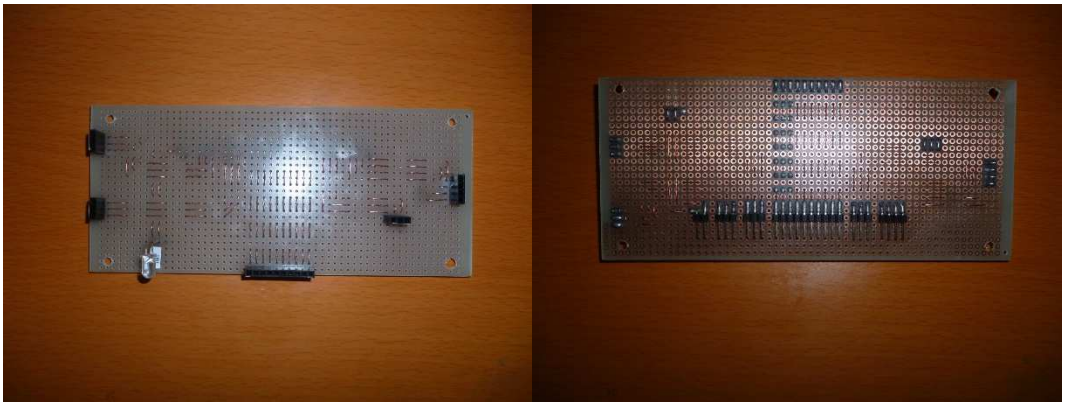


*Ilustración de Diseño 11 - Diseño del Hardware Parte 7*

*Fuente: El Autor*

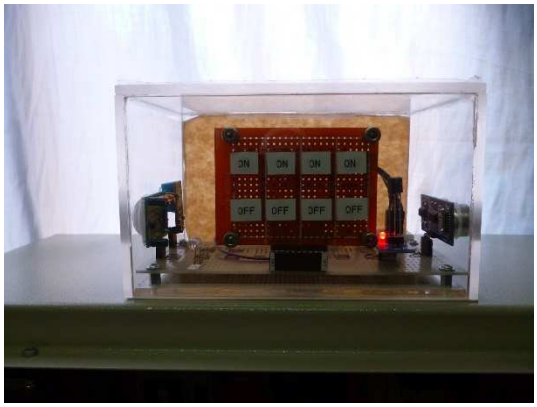
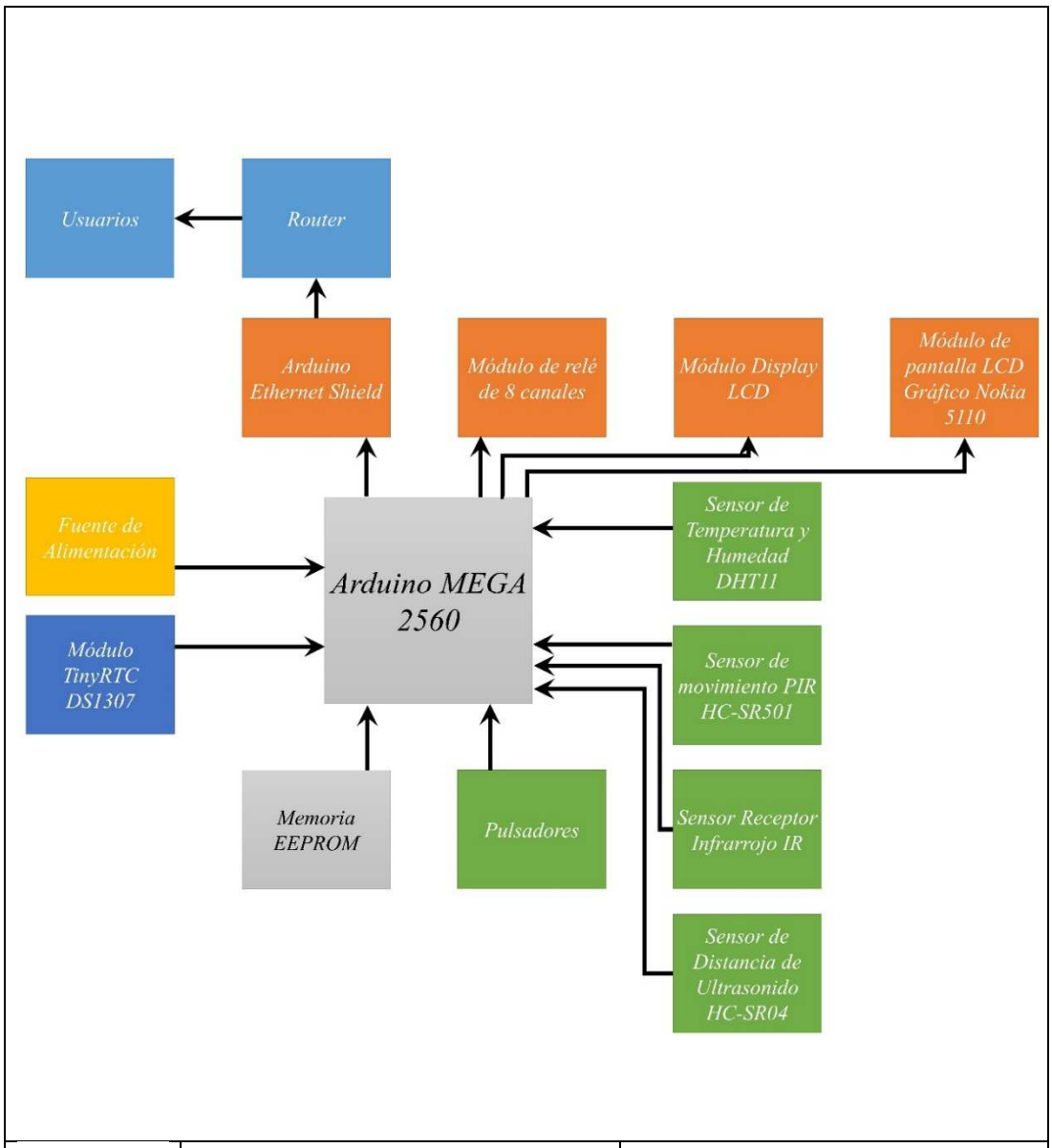



*Ilustración de Diseño 12 - Diseño del Hardware Parte 8*

*Fuente: El Autor*

Para ver como se encuentra distribuido el hardware utilizado en el prototipo, en la *Ilustración de Diseño 13* encontramos un diagrama de bloques y en el ***Anexo 6*** se puede visualizar un diagrama de conexiones del prototipo.




| | Diagrama de Bloques del Prototipo del Sistema de domótica | **Uleam**<br>**Carrera de Ing. Eléctrica** | |
|---|---|---|---|
| | | ***Elaborado por:***<br>*Sandro Quijije Marín* | ***Fecha:***<br>*Noviembre 2015* |

*Ilustración de Diseño 13 - Diagrama de Bloques del Prototipo*

*Fuente: El Autor*

### 3.3.1. Estructura

El prototipo está compuesto por una serie de elementos cada uno cumpliendo una función específica y brindándole nuevas características al sistema, los elementos se detallan a continuación.

- **Fuente de Alimentación:** encargada de suministrar los 5V a la placa Arduino.
- **Arduino MEGA 2560:** fue la placa seleccionada para el prototipo, debido a que es la placa Arduino que cuenta con las características apropiadas que se necesitan el proyecto.
- **Memoria EEPROM:** memoria borrable que guardara los datos de las salidas cuando no haya usuarios conectados por más de 10 minutos.
- **Módulo RTC DS1307:** se encarga de proveer la hora y fecha actuales al Arduino y gracias a que cuenta con una pila no se desprograma su hora y fecha preestablecidos en caso de cortar la alimentación a la placa.
- **Arduino Ethernet Shield:** permite a la placa Arduino conectarse a una red local o internet en conjunto a una programación previamente establecida.
- **Router Inalámbrico:** está conectado a la tarjeta Ethernet Shield mediante un cable de red con conector RJ45, y permite a los usuarios el ingreso a la interfaz Web una vez conectado al router con el ingreso de la IP correspondiente.
- **Usuarios:** son los elementos conectados al router vía inalámbrica con acceso a la interfaz Web entre ellos: Smartphone, Tablet, laptop, etc.
- **Módulo de relé de 8 canales:** aquel que va a controlar grandes corrientes
- **Módulo de Display LCD:** permite mostrar información proporcionada por los sensores.

- **Módulo de pantalla LCD Grafica Nokia 5110:** permite mostrar información precargada al sketch.
- **Pulsadores:** permiten activar las salidas de los relés de forma manual.
- **Sensor de temperatura y humedad DHT11:** encargado de suministrar la temperatura y humedad al Arduino.
- **Sensor de movimiento PIR:** detecta el movimiento y envía los datos a la placa Arduino para de inmediato activar una salida.
- **Sensor receptor Infrarrojo:** detecta los comandos infrarrojos enviados por un control remoto y envía los datos a la placa Arduino para de inmediato activar una salida.
- **Sensor de distancia por ultrasonido:** mide la distancia consecutivamente y envía los datos a la placa Arduino.

Además de estos elementos que componen el control electrónico el prototipo cuenta con la debida alimentación eléctrica AC, protección eléctrica y una fuente de alimentación o convertidor de AC a DC, en la *Ilustración de Diseño 14* se detalla un pequeño diagrama unifilar del prototipo donde detalla lo mencionado anteriormente.

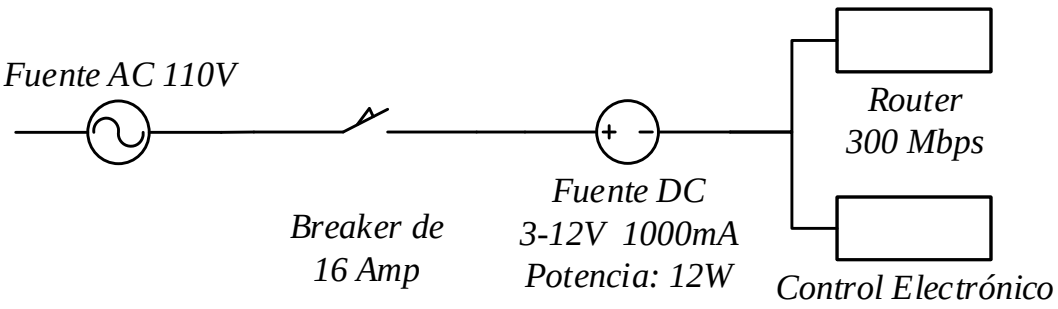



*Ilustración de Diseño 14 - Diagrama Unifilar*

*Fuente: El Autor*

### 3.3.2. Conexiones

Todos los elementos del prototipo se encuentran conectados a la placa Arduino mediante cables de colores tipo Dupont, y con su respectiva alimentación de voltaje por lo general casi en su totalidad los elementos trabajan con 5V a excepción del módulo RTC DS1307 y la pantalla LCD Grafica Nokia 5110 que trabajan con 3.3V, ambos voltajes son suministrados por nuestra placa Arduino.

## 3.4. Diseño del software

### 3.4.1. Software Arduino (IDE)

El entorno de desarrollo integrado Arduino o Software Arduino (IDE) contiene un editor de texto para escribir código, un área de mensajes, una consola de texto, una barra de herramientas con botones para funciones comunes y una serie de menús. Se conecta al hardware Arduino para cargar programas y comunicarse con ellos (Web-Robótica, 2013).

### 3.4.2. Entorno del Software Arduino

Los programas escritos utilizando Software Arduino (IDE) se llaman Sketch. Estos se escriben en el editor de texto y se guardan con la extensión de archivo .ino. El editor *(Figura 29)* tiene funciones para cortar, pegar, para buscar y reemplazar texto. El área de mensajes proporciona información mientras se realiza la exportación y también muestra los errores. La consola muestra la salida de texto por el software de Arduino (IDE), incluidos los mensajes de error y otra información. En la esquina inferior derecha se muestra la placa conectada y el puerto serie. Los botones de la barra *(Tabla 14)* le permiten verificar y cargar programas, crear, abrir y guardar dibujos, y abrir el monitor de serie.

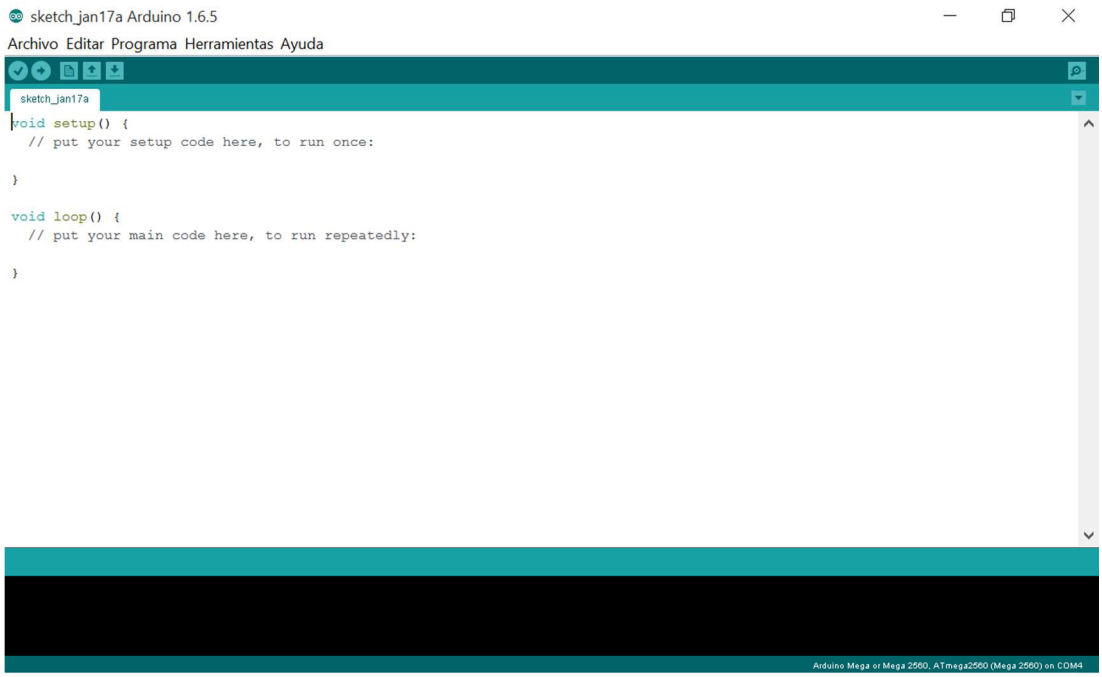



*Figura 29 - Entorno del Software Arduino*

*Fuente: El Autor*

### *3.4.2.1. Comandos del Software Arduino.*

| Icono | Descripción |
|---|---|
|  | Verificar: Chequea el código que no tenga errores. |
|  | Subir: Compila el código y carga el Sketch a la placa configurada. |
|  | Nuevo: Crea un nuevo Sketch. |
|  | Abrir: Abre una ventana con todos los sketch realizados. |
|  | Guardar: Guarda el sketch. |
|  | Monitor Serial: Abre del monitor serial. |

*Tabla 14 - Comandos del Software*

*Fuente: El Autor*

Comandos adicionales se encuentran dentro de los cinco menús: Archivo, Editar, Programa, Herramientas, Ayuda. Los menús son sensibles al contexto, lo que significa que sólo aquellos elementos pertinentes a la labor que está llevando a cabo están disponibles.

### 3.4.3. Programación en Arduino

El microcontrolador en las placas Arduino (ATmega328) se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo de Arduino (basado en Processing) (BURUTEK, 2013).

- Wiring: Es un lenguaje de programación de código abierto para microcontroladores que permite escribir software multiplataforma para controlar dispositivos ligados a una amplia gama de microcontroladores.
- Processing: Es un lenguaje de programación de código abierto y el entorno para las personas que desean crear imágenes, animaciones e interacciones.

El lenguaje de programación de las tarjetas Arduino es de código abierto basado en la flexibilidad y en el uso simple tanto del software como del hardware.

El software de Arduino consiste en un entorno de desarrollo (IDE) y las librerías centrales. El IDE está escrito en Java y basado en el entorno de Processing. Las librerías centrales están escritas en C y C++; y, compilado con avr-gcc y AVR Libc. El código fuente para Arduino está alojado en GitHub (GitHub, 2014).

Los programas desarrollados con Arduino se dividen en tres partes principales mostrados en el ***Anexo 7***: estructura, valores (variables y constantes), y funciones. El lenguaje de programación Arduino se basa en C/C++. Estos proyectos se ejecutan sin la necesidad de conectarse con un ordenador, además pueden comunicarse con diferentes tipos de software (Flash, Processing, MaxMSP).

### 3.4.4. Estructura de un programa

La estructura básica del lenguaje de programación de Arduino es bastante simple y se compone de al menos dos partes. Estas dos partes necesarias, o funciones, encierran bloques que contienen declaraciones, estamentos o instrucciones como lo muestra la *Figura 30*.

En donde setup() es la parte encargada de recoger la configuración y loop() es la que contiene el programa que se ejecutará cíclicamente (de ahí el término loop-bucle-). Ambas funciones son necesarias para que el programa trabaje.

La función de configuración (setup) debe contener la declaración de las variables. Es la primera función a ejecutar en el programa, se ejecuta sólo una vez, y se utiliza para configurar o inicializar pinMode (modo de trabajo de las E/S), configuración de la comunicación en serie y otras. Debe ser incluido en un programa aunque no haya declaración que ejecutar. Así mismo se puede utilizar para establecer el estado inicial de las salidas de la placa.

La función bucle (loop) siguiente contiene el código que se ejecutara continuamente (lectura de entradas, activación de salidas, etc.) Esta función es el núcleo de todos los programas de Arduino y la que realiza la mayor parte del trabajo, lo que posibilita que el programa esté respondiendo continuamente ante los eventos que se produzcan en la placa (Martínez, 2014).


*Figura 30 - Estructura del Software Arduino*

*Fuente: El Autor*

### 3.4.5. Librerías

Las Librerías proporcionan funcionalidad adicional para uso en nuestro sketch, por ejemplo, trabajar con el hardware o la manipulación de los datos. Para utilizar una biblioteca en un sketch, selecciónelo en el Sketch> Import Library. Algunas bibliotecas se incluyen con el software de Arduino. Otros pueden ser descargados desde una variedad de fuentes o a través del Administrador de bibliotecas. (ARDUINO, Libraries, 2012).

#### *3.4.5.1. ¿Cómo instalar una Librería?.*

Para instalar una nueva librería en el Arduino IDE *(Figura 31)* se puede utilizar el Administrador de Librería (disponible en IDE versión). Se abre el IDE y haciendo clic en el menú "Programa" y luego Incluir Librería> Administrar Librerías.

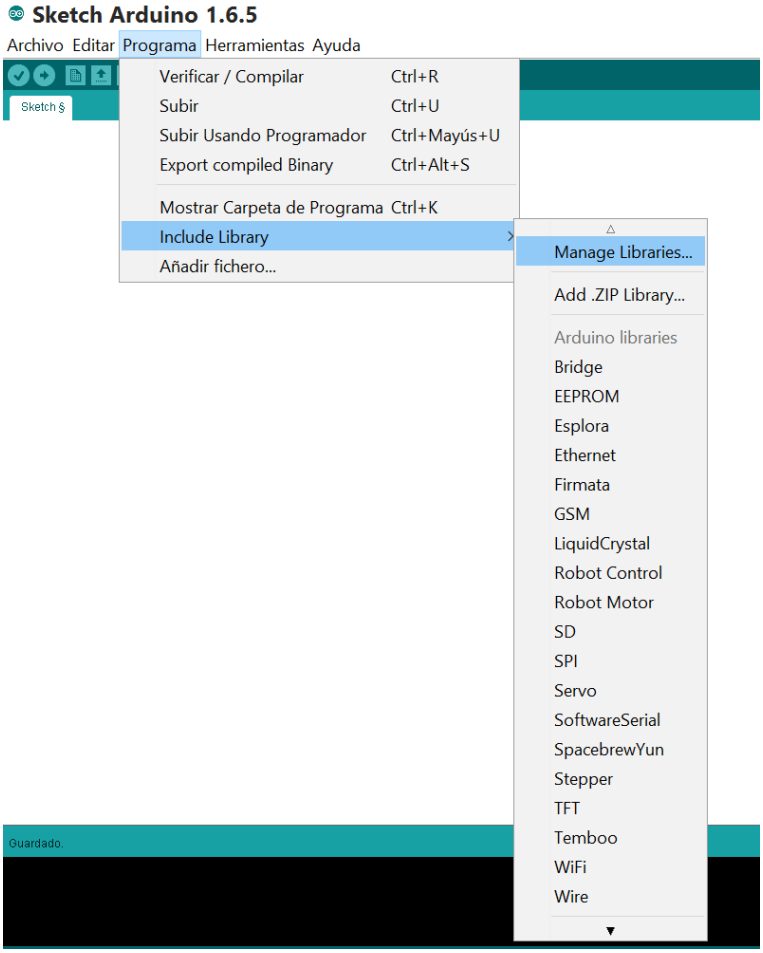



*Figura 31 - Menú para Incluir Librería por Administrador de librería*

*Fuente: El Autor*

Luego de esto, se nos abrirá el administrador de librería *(Figura 32)* con una serie de librerías listas para descargar e instalar.

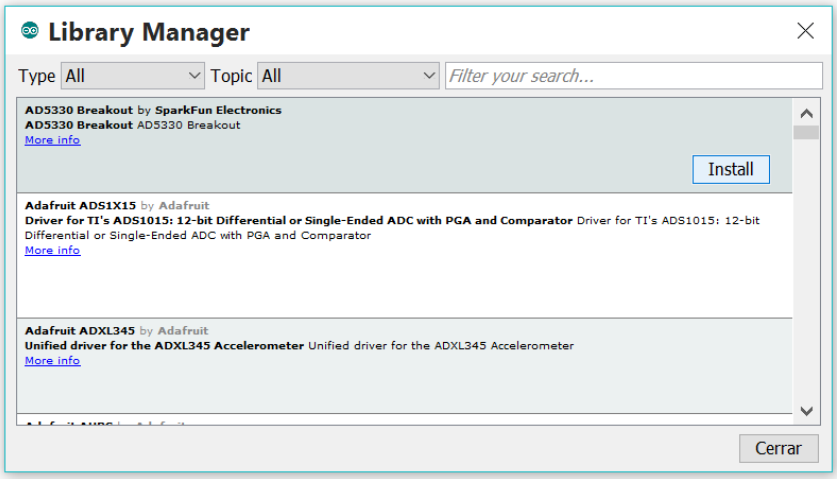



*Figura 32 - Administrador de Librería del Software de Arduino*

*Fuente: El Autor*

En caso que necesitemos instalar una biblioteca que no se encuentre en el administrador de librerías procederemos a descargarla y luego se abre el IDE de Arduino y haciendo clic en el menú "Programa" y luego Incluir Librería> Agregar librería .ZIP *(Figura 33)* (el archivo debe estar comprimido en formato .ZIP), de inmediato se nos abrirá una ventana en donde buscaremos la ubicación del archivo y lo abrimos hasta que cargue la librería.

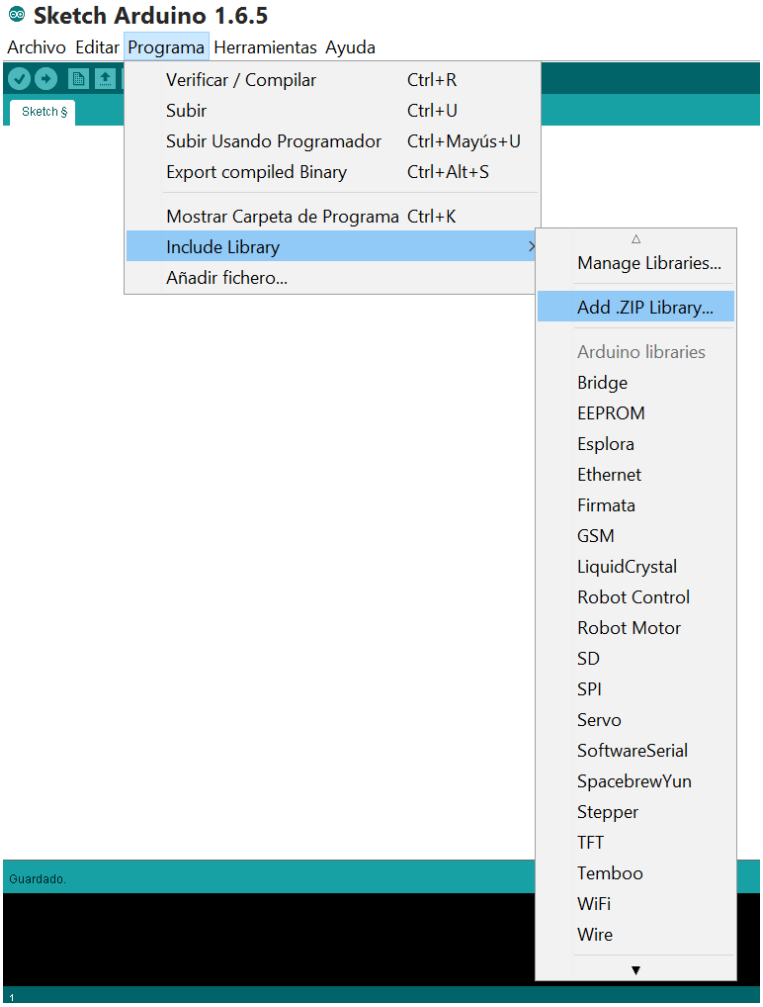



*Figura 33 - Menú para Incluir Librería por Archivo*

*Fuente: El Autor*

### 3.4.6. Librería Ethernet

Con el Arduino Ethernet Shield, esta librería permite a la placa Arduino conectarse a internet. Puede funcionar tanto como servidor capaz de aceptar conexiones entrantes o como cliente permitiendo realizar conexiones de salida. La librería permite hasta cuatro conexiones simultáneas (conexiones entrantes, salientes, o una combinación de ambas).

Arduino se comunica con el escudo mediante el bus SPI. Esto es en los pines digitales 11, 12, y 13 en el Uno y los pines 50, 51, y 52 en los Mega *(Figura 34)*. En ambas Placas, el pin 10 se utiliza como SS (ARDUINO, Libraries, 2012).

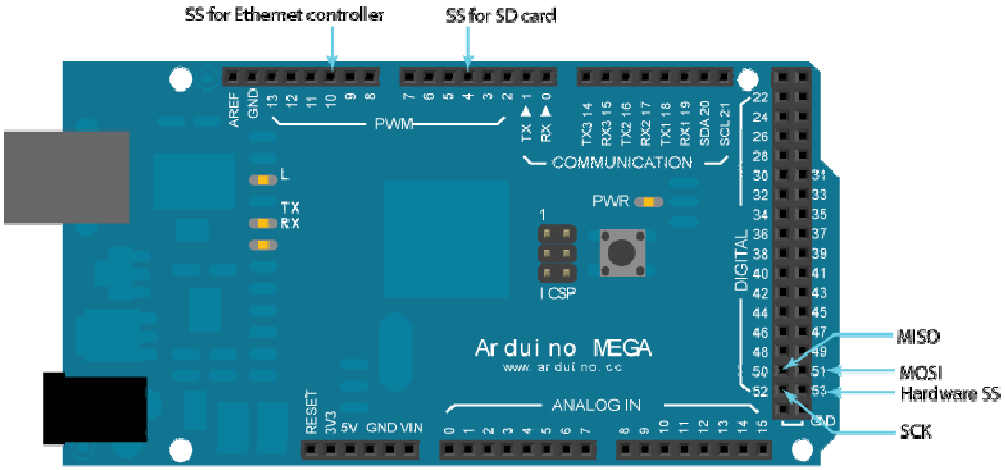



*Figura 34 - Pines que utiliza el Ethernet en Arduino Mega 2560*

*Fuente: https://www.arduino.cc*

### 3.4.7. Librería SPI

Esta biblioteca le permite comunicarse con los dispositivos SPI. El Serial Peripheral Interface Bus o SPI bus es una conexión estándar de datos seriales sincronizados llamado por Motorola que opera en modo completo doble. Lo dispositivos se comunican en modo maestro/esclavo donde el dispositivo maestro inicia el marco de datos. Varios dispositivos esclavo son permitidos con una línea individual esclava (chip seleccionado) usando un pin para cada dispositivo. Hay una amplia disponibilidad de sensores que se comunican usando SPI como protocolo. Incluyendo esta librería automáticamente se define constantes para cada pin involucrado: SS, SCK, MOSI, MISO (Wiring, 2012).

### 3.4.8. Librería EEPROM

El microcontrolador en la placa Arduino basado en AVR tiene EEPROM: memoria cuyos valores se mantienen cuando el tablero está apagado (como un pequeño disco duro). Esta biblioteca le permite leer y escribir los bytes.

Los microcontroladores compatibles en los diferentes placas Arduino tienen diferentes cantidades de EEPROM: 1024 bytes en el ATmega328, 512 bytes en el ATmega168 y ATmega8, 4 KB (4096 bytes) en el ATmega1280 y Atmega2560 (ARDUINO, Libraries, 2012).

### 3.4.9. Librería DHT

Es una librería diseñada para la utilización de los sensores de temperatura y humedad DHT11 y DHT22, con esta librería nos será muy fácil tomar los datos de temperatura y humedad relativa leídos por el sensor. El modelo del sensor debe ser definido al incluir esta librería en nuestro sketch.

### 3.4.10. Librería Wire

Esta librería permite comunicarse con I2C / dispositivos TWI. En las placas Arduino con el diseño R3 (1.0 pinout), la SDA (línea de datos) y SCL (línea de reloj) están en los conectores macho cerca del pin AREF. El Arduino Due cuenta con dos I2C / TWI interfaces de SDA1 y SCL1 están cerca del pin AREF y el adicional es en los pines 20 y 21 (ARDUINO, Libraries, 2012).

### 3.4.11. Librería RTClib

Esta librería permite a una placa Arduino a controlar el uso de un RTC (Real Time Clock). Un reloj en tiempo real es un reloj que hace un seguimiento de la hora actual y que puede ser utilizado con el fin de programar acciones en un tiempo

determinado. La mayoría de los RTC usan un oscilador de cristal cuya frecuencia es 32,768 kHz (la misma frecuencia utilizada en relojes de cuarzo y relojes) (ARDUINO, Libraries, 2012).

### 3.4.12. Librería LiquidCrystal

La librería LiquidCrystal.h permite que una tarjeta Arduino pueda controlar pantallas LCD que sean compatibles con el chipset Hitachi HD44780, que se encuentra en la mayaría de las LCD basadas en texto. La librería trabaja ya sea utilizando 4 u 8 bits (es decir, utilizando 4 u 8 líneas de datos, además de RS, Enable y opcionalmente RW) (ARDUINO, Libraries, 2012).

### 3.4.13. Librería LCD5110_Graph

La librería LCD5110_Graph permite el uso del módulo LCD de Nokia 5110 como una pantalla de gráficos en un Arduino. Esta librería se ha hecho para que sea fácil de utilizar el modulo permitiéndonos agregar gráficos monocromáticos de una manera fácil e inclusive cambiar el tamaño de la fuente de letra a las que están incluidas en la librería (TinyFont y SmallFont).

### 3.4.14. Librería IRremote

Esta librería IRremote le permite enviar y recibir códigos remotos IR en múltiples protocolos. Es compatible con NEC, Sony SIRC, Philips RC5, RC6 Philips, y otros protocolos abiertos. Esta librería consta de dos partes IRsend e IRrecv. IRsend permite transmitir paquetes de datos mediante el uso de un LED infrarrojo conectado al pin 3 de nuestro Arduino, mientras que, IRrecv permite recibir y decodificar un paquete de datos desde cualquier control remoto utilizando un detector infrarrojo.

### 3.4.15. Librería Ultrasonic

La librería Ultrasonic nos ayuda a medir distancias mediante ultrasonidos. Para esto debemos definir los dos pines digitales del sensor, echo y trigger, en lo cual el primer pin (trigger) recibe pulsos desde el microcontrolador para indicar al sensor en uso que inicie la debida medición de distancia, y en el segundo pin (echo) el sensor muestra al microcontrolador el tiempo que tarda el sonido en viajar desde el sensor al obstáculo más cercano que se encuentre *(Figura 35)*.

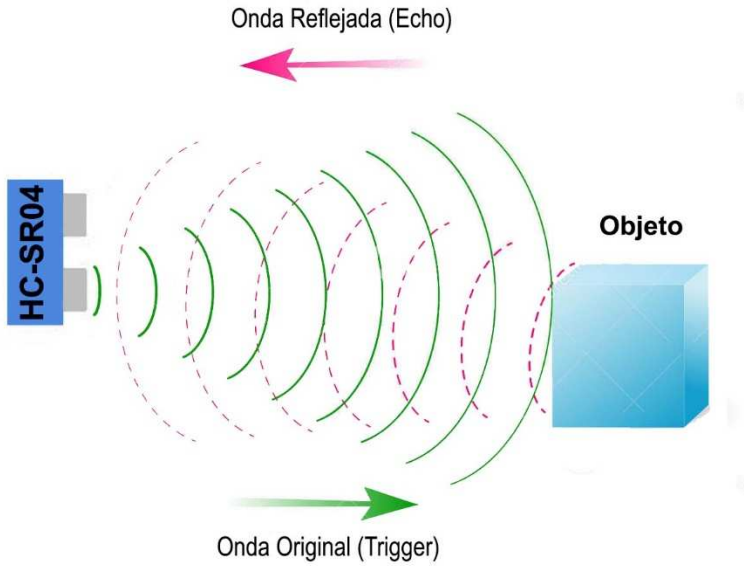



*Figura 35 - Funcionamiento del Sensor de Distancia*

*Fuente: http://www.zonamaker.com*

### 3.4.16. Configuración de la Ethernet

Para realizar la debida configuración de la Ethernet en nuestro sketch lo primero que debemos realizar es acceder al Router al que estará conectado nuestro Arduino para de esta manera poder revisar su configuración y rango de IP con la que trabaja. De la configuración de nuestro router depende mucho la IP que le asignemos

a nuestro Arduino en el sketch. Si no contamos con la IP para acceder a nuestro router podemos hacerlo desde una computadora con Windows de la siguiente manera:

a) Presionamos el botón de inicio.
b) Escribimos CMD o Símbolo de sistema y abrimos *(Figura 36).*
c) De inmediato se nos abrirá una ventana y en la posición que se encuentra el cursor escribimos "ipconfig" (sin las comillas).
d) Luego se nos desplegara una información y buscaremos donde diga Puerta de enlace predeterminada como lo indica la figura y allí encontraremos nuestra IP de acceso al router.

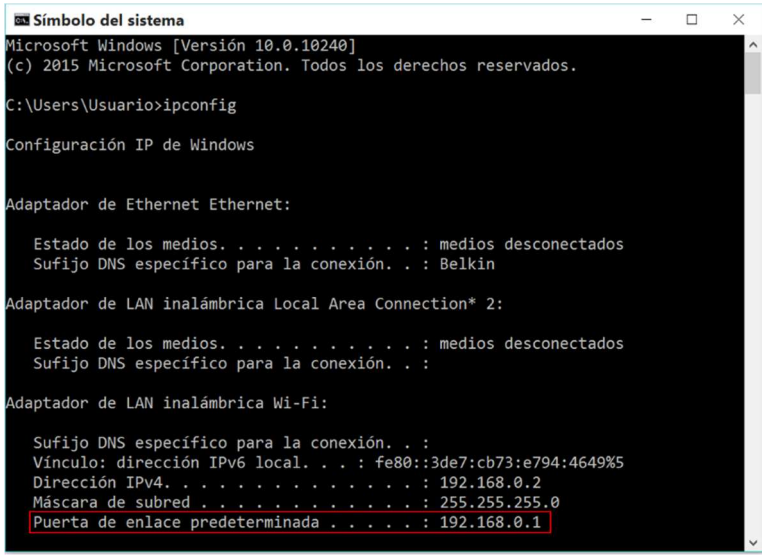



*Figura 36 - IP del Router*
*Fuente: El Autor*

Una vez que obtenemos la IP abrimos nuestro explorador favorito e ingresamos la IP y se nos abrirá la configuración del router *(Figura 37)* (debemos asegurarnos que el router no tenga usuario ni clave de acceso), en este caso como solo se realizara el sistema de domótica a nivel de red local solo se accederá a la configuración LAN del router para revisar el rango de las IP como lo indica la figura. En el rango lo único que

debemos cambiar es la última numeración Como el rango de las IP va desde 192.168.0.2 hasta 254 escogemos 192.168.0.4 para nuestro sketch.

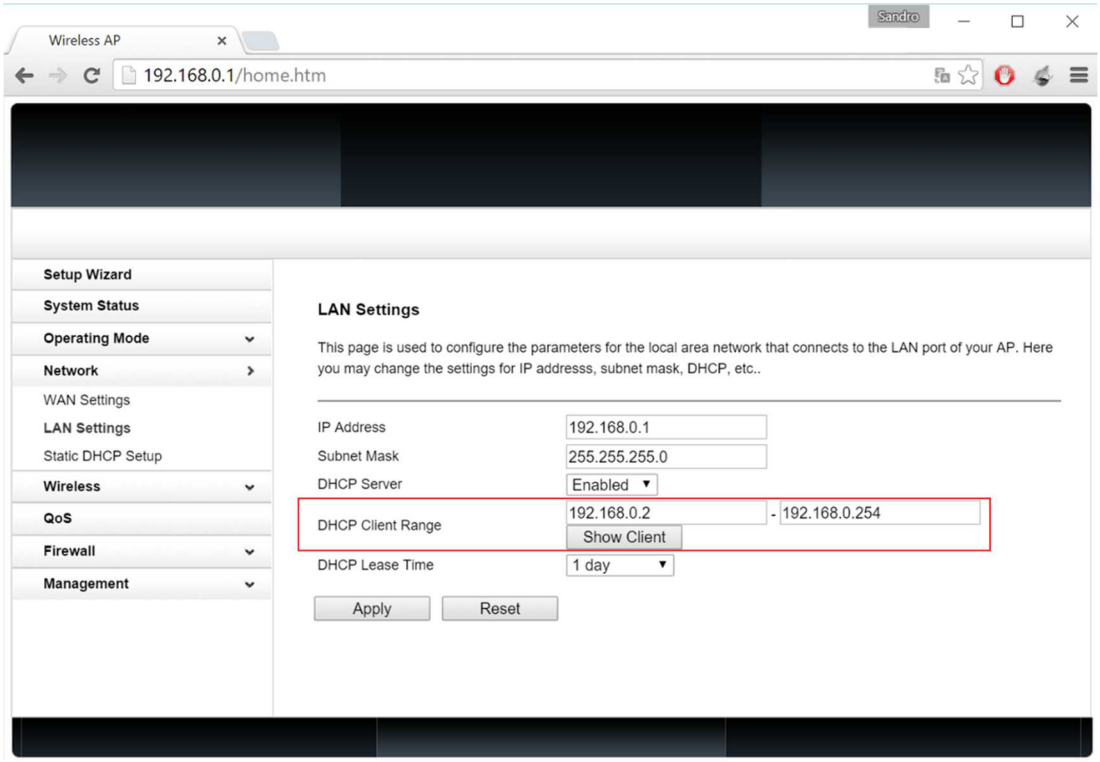



*Figura 37 - Configuración del Router*

*Fuente: El Autor*

En el sketch debemos realizar la configuración de red manualmente una vez que ya tengamos la IP a asignar. Luego d esto se introducirá los valores correspondientes a MAC, IP local, puerta de enlace, mascara de subred y puerto html, la configuración designada fue el siguiente código:

```
//////////////////////////////////////////////////////////////////
//Configuración y Ajustes del Ethernet Shield
//////////////////////////////////////////////////////////////////

//Configuración de la IP
byte ip[] = {
  192, 168, 0, 4
};   //IP
byte gateway[] = {
  192, 168, 0, 2
};   //Puerta de enlace
byte subnet[] = {
  255, 255, 255, 0
}; //Máscara de Subred

//Dirección MAC de la Ethernet Shield
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

//Puerto Ethernet
EthernetServer server = EthernetServer(80); //Por defecto el puerto
html 80
```

### 3.4.17. Programación de la placa Arduino MEGA 2560

La programación del Arduino MEGA 2560 para el sistema de domótica se realizó en el software de Arduino, para realizar estar programación se tomó como código base el ejemplo: "Web Server Ethernet Switching Versión 4.06" [2]realizado por Claudio Vella, cabe recalcar que este ejemplo más bien sirvió para formar el servidor web, aun así el aspecto de servidor web fue modificado. El código del sketch desarrollado para el sistema de domótica se encuentra con sus instrucciones debidamente detalladas a la vez el código está apto para implementarle más sensores y más salidas a relés para controlar voltajes mayores (110V-220V), el código se detalla en el ***Anexo 8***.

[2] (ClaudioVella, 2012)

# CAPÍTULO 4. PRUEBAS Y RESULTADOS

## 4.1. Generalidades

En el presente capitulo se realizó el análisis del Funcionamiento práctico del sistema de Domótica, basándonos en el prototipo desarrollado, lo cual constituye una herramienta muy importante para demostrar el funcionamiento del mismo, ya que como se trató en el capítulo anterior el prototipo está compuesto de elementos básicos que darán a entender muy claramente cómo funciona nuestro sistema.

Las pruebas realizadas al prototipo sirven para tener conocimiento y a la vez corregir los posibles errores que se presenten y de esta manera poder corregir, mejorar y porque no perfeccionar el sistema, y a su vez obteniendo propias conclusiones y posibles recomendaciones.

## 4.2. Descripción del funcionamiento del Sistema de Domótica basado en el Prototipo

El prototipo cuenta con su debido breaker de encendido y apagado y que a su vez contribuye en la protección eléctrica; encendiendo el breaker alimentamos todo el prototipo que cuenta con dos fuentes de alimentación una que alimenta con 9V el router y la otra que alimenta con 6V la placa Arduino, la placa Arduino tiene montado un Shield Ethernet mediante los pines de Arduino el mismo que se encuentra conectado por un cable de red con conector RJ45 a uno de los puertos LAN del Router, el router se encuentra configurado para que trabaje como un router repetidor, lo cual significa que sin necesidad de cables el router tiene acceso a internet.

A la placa Arduino se encuentra conectada un módulo RTC DS1307 el mismo que suministra la hora y fecha sin desconfigurarse, también se encuentran conectado

un módulo de relés de 8 canales de los cuales los 4 primeros canales se pueden activar o desactivar por 3 maneras diferentes como se detalla:

- La primera: la más importante mediante el servidor Web, accediendo al mismo tenemos al opción de activar o desactivar dicha salida escogida y además podemos visualizar un pequeño icono que cambiara de color al activar o desactivar una de las salidas.
- La segunda: mediante el uso de un mando a distancia que mediante un sensor infrarrojo establece comunicación con la placa Arduino dándole ordenes, en este caso activando o desactivando las salidas.
- La tercera: no siempre podemos tener un dispositivo móvil a la mano o mando a distancia y debido a esto es conveniente tener la opción de un encendido o apagado manual en este caso se realizara la activación y desactivación mediante el uso de pulsadores, estos al momento de ser presionados envían un 1 lógico a una entrada digital que será leída por la placa activando inmediatamente la salida correspondiente.

Los canales 5, 6, y 7 están configurados para que trabajen en conjunto con los sensores esto quiere decir que dichos canales solo pueden ser activados por los sensores, y el canal 8 trabaja con los datos del módulo RTC,

- El quinto canal trabaja con un sensor de movimiento PIR, los sensores PIR poseen elementos fabricados de un material cristalino que genera una carga eléctrica cuando se expone a la radiación infrarroja. Los cambios en la cantidad de radiación producen cambios de voltaje los cuales son medidos por un amplificador. El PIR contiene unos filtros especiales llamados lentes de Fresnel que enfocan las señales infrarrojas sobre el elemento sensor. Cuando

las señales infrarrojas del ambiente donde se encuentra el sensor cambian rápidamente, el amplificador activa la salida (canal 5) para indicar movimiento. Esta salida permanece activa durante algunos segundos permitiendo al microcontrolador saber si hubo movimiento dicha activación de la salida se ve reflejada en el servidor web.

- El sexto canal trabaja en conjunto con un sensor de temperatura, este previo a un rango de temperatura establecida en la programación se activa cuando la temperatura sea mayor o igual a 32 grados centígrados. El sensor de temperatura se conecta al Arduino mediante uno de sus pines digitales a diferencia de otros sensores que trabajan con pines análogos.
- El séptimo canal trabaja en conjunto con un sensor de distancia HC-SR04 que bien detecta una distancia previamente establecida y se activa. El HC-SR04 lleva dos transductores (dos cilindros de color gris, que parecen micrófonos). El sensor envía un pulso de ultrasonidos a través de un transductor cuándo el pin “Trig” está a HIGH. El pulso avanza hasta que choca con un obstáculo y rebota, volviendo al sensor. El segundo transductor detecta la señal de este “eco”. El sensor mide el tiempo que ha tardado la señal en rebotar, estos datos son enviados al Arduino que multiplica los datos por una constante para obtener la distancia en centímetros, luego esta distancia es comparada con la que establecimos y si es igual o menor mandara a activar la salida respectiva.
- El ultimo canal (8) del módulo de relés trabaja con la hora del sistema se le configura una hora de encendido y otra de apagado, para esto se utilizan los datos que proporciona el módulo RTC.

El sistema se puede monitorear a través del servidor web donde podremos constatar datos de hora, fecha, temperatura, estado de los sensores y estados de las salidas del módulo de relés.

## 4.3. Análisis y pruebas realizadas a los sensores

Para efectos de investigación, se realizó pruebas a dos sensores para comprobar su correcto funcionamiento, se les realizo pruebas a:

- El sensor de Temperatura: se realizó una toma de temperatura en diferentes horas del día y se compararon los valores leídos con valores que nos ofrecen fuentes en internet, los valores de temperatura constan con sus respectivas horas para demostrar que esos valores obtenidos fueron tomados en el mismo momento de diferentes fuentes como los muestran las figuras siguientes.

**Toma de datos 1**


*Figura 38 - Toma de datos 1 - Sensor del Prototipo*

*Fuente: El Autor*


*Figura 39 - Toma de datos 1- AccuWeather.com*

*Fuente: AccuWeather.com*


*Figura 40 - Toma de datos 1 - weather*

*Fuente: weather.com*

**Toma de datos 2**


*Figura 41 - Toma de datos 2 - Sensor del Prototipo*

*Fuente: El Autor*


*Figura 42 - Toma de datos 2 - AccuWeather.com*

*Fuente: AccuWeather.com*


*Figura 43 - Toma de datos 2 - weather*

*Fuente: weather.com*

**Toma de datos 3**


*Figura 44 - Toma de datos 3 - Sensor del Prototipo*

*Fuente: El Autor*


*Figura 45 - Toma de datos 3 - AccuWeather.com*

*Fuente: AccuWeather.com*


*Figura 46 - Toma de datos 3 - weather*

*Fuente: weather.com*

Se realizaron 3 pruebas en diferentes horarios del día una en la noche otra en la mañana y la última al medio día. Cada foto indica la toma de pruebas de fuentes diferentes, teniendo como fuente principal el sensor DHT11 del prototipo, luego tenemos las demás fuentes; cómo podemos observar la temperatura del sensor del prototipo coincide con la de las demás fuentes que se encuentran en internet, cada imagen contiene la hora y fecha en que se toma para una mayor veracidad del caso.

- Sensor de distancia HC-SR04: a este sensor se lo configuro para que active una salida cuando detecte un objeto a menos de 10 cm de distancia. En la *Figura 47* se muestra la distancia físicamente y se coloca un objeto a 10 cm y en la *Figura 48* se puede observar cómo se activa el canal del relé.

*Figura 47 - Sensor de distancia*

*Fuente: el Autor*

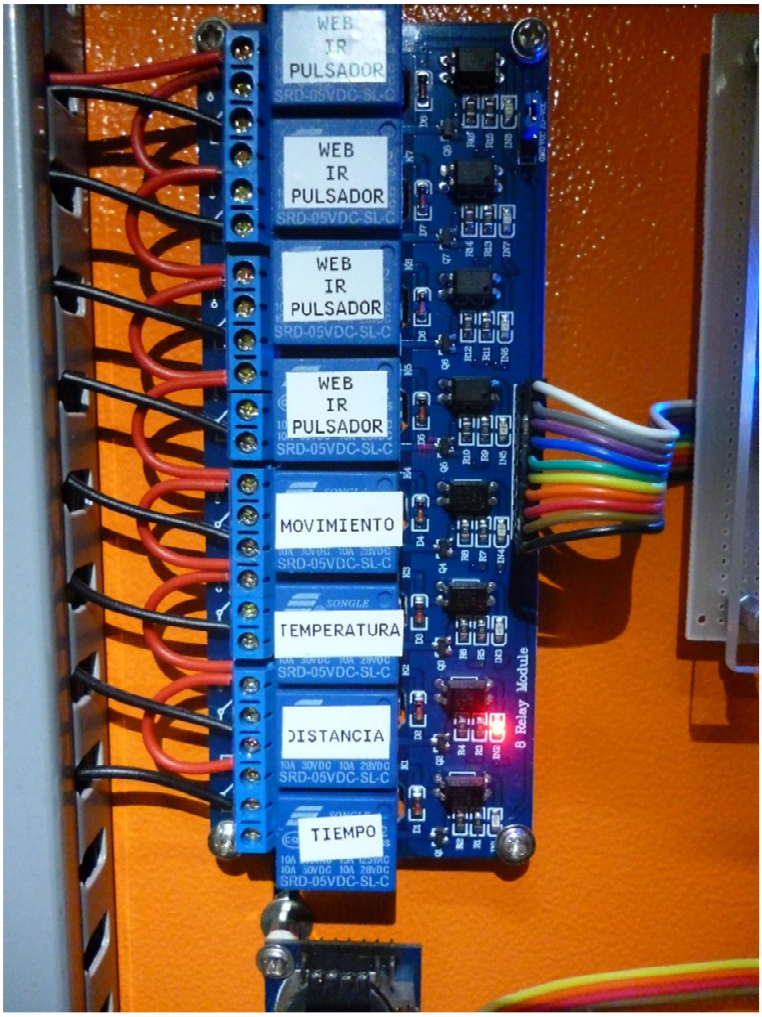



*Figura 48 - Relé del sensor de distancia*

*Fuente: El Autor*

## 4.4. Presupuesto del Sistema de domótica

La *Tabla 15* detalla los gastos de materiales utilizados en la realización del prototipo, entre ellos tablero metálico, accesorios eléctricos, router sensores, módulos, placas, entre otros. Cabe recalcar que este estimado de $ 335,33 solo corresponde a gastos en puro material para la implementación del prototipo que demuestra el efectivo funcionamiento del Sistema de domótica.

| *Cantidad* | *Descripción* | *Valor unitario* | *Valor total* |
|---|---|---|---|
| *1* | *Arduino Mega 2560* | *$ 29,46* | *$ 29,46* |
| *1* | *Arduino Ethernet* | *$ 25,00* | *$ 25,00* |
| *1* | *Modulo DS1307* | *$ 7,10* | *$ 7,10* |
| *1* | *DHT11* | *$ 9,02* | *$ 9,02* |
| *1* | *Modulo de relé 8 canales* | *$ 16,96* | *$ 16,96* |
| *1* | *Receptor infrarrojo* | *$ 1,12* | *$ 1,12* |
| *3* | *Juegos de cables Dupont (40 cables C/U)* | *$ 4,50* | *$ 13,50* |
| *1* | *Sensor HC-SR04* | *$ 6,70* | *$ 6,70* |
| *1* | *Sensor PIR* | *$ 6,03* | *$ 6,03* |
| *1* | *Fuente DC variable* | *$ 8,88* | *$ 8,88* |
| *1* | *Case para placa Arduino Mega* | *$ 8,48* | *$ 8,48* |
| *1* | *Modulo Kit Ir Ideal Para Arduino* | *$ 7,00* | *$ 7,00* |
| *1* | *Modulo Display Lcd Nokia 5110* | *$ 9,00* | *$ 9,00* |
| *1* | *Display Lcd 16x4 Luz De Fondo Amarillo* | *$ 21,00* | *$ 21,00* |
| *2* | *Placa de baquelita Grande* | *$ 2,50* | *$ 5,00* |
| *1* | *Placa de baquelita Pequeña* | *$ 1,00* | *$ 1,00* |
| *8* | *Pulsadores* | *$ 0,25* | *$ 2,00* |
| *8* | *Resistencias 10k* | *$ 0,10* | *$ 0,80* |
| *2* | *Borneras de 2 vías* | *$ 0,50* | *$ 1,00* |
| *3* | *Juegos de pines para soldar Macho* | *$ 0,50* | *$ 1,50* |
| *3* | *Juegos de pines para soldar Hembra* | *$ 0,50* | *$ 1,50* |
| *2* | *Tomacorriente Bticino Polarizado* | *$ 2,23* | *$ 4,46* |
| *2* | *Canaleta Gris ranurada 25X25MM* | *$ 4,24* | *$ 8,48* |
| *2* | *Borneras en bloques #14AWG* | *$ 1,87* | *$ 3,74* |
| *1* | *Tablero Metálico 60X40X20CM* | *$ 61,77* | *$ 61,77* |
| *1* | *Breaker Siemens C16A 2 vías* | *$ 15,00* | *$ 15,00* |
| *1* | *Breaker Siemens C20 C 1 vía* | *$ 16,00* | *$ 16,00* |
| *1* | *Microswitch Rodillo Largo* | *$ 3,10* | *$ 3,10* |
| *2* | *Tope para bornera Aplicable* | *$ 2,07* | *$ 4,14* |
| *1* | *Router Marca One Wireless 300mbp* | *$ 26,99* | *$ 26,99* |
| *3* | *3 metros de cable concéntrico 3 en uno* | *$ 2,40* | *$ 7,20* |
| *1* | *Enchufe 110v* | *$ 1,50* | *$ 1,50* |
| *1* | *Prensa Estopa 1/2"* | *$ 0,90* | *$ 0,90* |
| | | *Total* | *$ 335,33* |

*Tabla 15 - Presupuesto para la realización del prototipo.*

*Fuente: EL Autor*

Si bien queremos implementar este proyecto en campo de una manera real en una vivienda, se incluirían gastos adicionales no solo por el número de sensores que aumentarían, sino también otros valores más de elementos utilizados en una implementación real, estos también pueden variar de acuerdo a las necesidades del usuario que solicite el sistema de domótica. Para esto a continuación se van a plantear 3 escenarios distintos con valores reales para podernos hacer la idea, cuanto exactamente nos constaría implementar un sistema de este tipo en nuestro domicilio.

El primer escenario será una vivienda básica de una planta en donde solo se implementara un Sistema para controlar el encendido y apagado mediante el uso del Servidor Web, de 8 puntos eléctricos que se encuentran distribuidos alrededor de la vivienda cada punto Eléctrico contara con su respectivo pulsador de encendido y apagado para su respectiva acción manual.

En la *Tabla 16* se detalla una lista de los gastos respetivos en donde se puede apreciar una lista de materiales con sus respectivos precios, la mano de obra de implementación y el costo por la realización del software teniendo un costo total de $ 713,45. Adicional a estos costos de agrega un pago mensual de $ 25,60 por la contratación de una IP publica, dicho pago correría por cuenta del usuario, el uso de esta IP publica se daría en el caso que el usuario desearía controlar el sistema desde cualquier lugar del mundo, en caso de no ser así se podría dejar trabajando el Sistema de manera local hasta donde la señal del router tenga alcance.

Cabe mencionar que en este primer escenario solo se podrá contar con las opciones de apagar y encender 8 puntos diferentes ya sea d manera manual o mediante el servidor web, además el sistema contara con su respectiva fecha y hora gracias al módulo RTC.

| Cantidad | Descripción | Valor unitario | Valor total |
|---|---|---|---|
| 1 | Arduino Mega 2560 | $ 29,46 | $ 29,46 |
| 1 | Arduino Ethernet | $ 25,00 | $ 25,00 |
| 1 | Modulo DS1307 | $ 7,10 | $ 7,10 |
| 2 | Modulo de relé 8 canales | $ 16,96 | $ 33,92 |
| 2 | Juegos de cables Dupont (40 cables C/U) | $ 4,50 | $ 9,00 |
| 1 | Fuente DC variable | $ 8,88 | $ 8,88 |
| 1 | Case para placa Arduino Mega | $ 8,48 | $ 8,48 |
| 1 | Display Lcd 16x4 Luz De Fondo Amarillo | $ 21,00 | $ 21,00 |
| 1 | Placa de baquelita Grande | $ 2,50 | $ 2,50 |
| 1 | Placa de baquelita Pequeña | $ 1,00 | $ 1,00 |
| 16 | Pulsadores | $ 0,25 | $ 4,00 |
| 16 | Resistencias 10k | $ 0,10 | $ 1,60 |
| 4 | Borneras de 2 vías | $ 0,50 | $ 2,00 |
| 2 | Juegos de pines para soldar Macho | $ 0,50 | $ 1,00 |
| 2 | Juegos de pines para soldar Hembra | $ 0,50 | $ 1,00 |
| 2 | Tomacorriente Bticino Polarizado | $ 2,23 | $ 4,46 |
| 2 | Canaleta Gris ranurada 25X25MM | $ 4,24 | $ 8,48 |
| 2 | Borneras en bloques #14AWG | $ 1,87 | $ 3,74 |
| 1 | Tablero Metálico 50X30X20CM | $ 45,60 | $ 45,60 |
| 1 | Breaker Siemens C16A 2 vías | $ 15,00 | $ 15,00 |
| 1 | Breaker Siemens C32 | $ 20,00 | $ 20,00 |
| 1 | Microswitch Rodillo Largo | $ 3,10 | $ 3,10 |
| 2 | Tope para bornera Aplicable | $ 2,07 | $ 4,14 |
| 1 | Router Marca One Wireless 300mbp | $ 26,99 | $ 26,99 |
| 30 | Cable UTP Categoria 5 30m. | $ 0,50 | $ 15,00 |
| 20 | Cable #12 Super Flex 20m. | $ 0,55 | $ 11,00 |
| 1 | Implementacion e instalacion del Sistema | $ 300,00 | $ 300,00 |
| 1 | Diseño del software para cargar en el Arduin | $ 100,00 | $ 100,00 |
| | | Total | $ 713,45 |

*Gastos Adicionales*

| Cantidad | Descripción | Valor unitario | Valor total |
|---|---|---|---|
| 1 | IP Publica de pago mensual | $ 25,60 | $ 25,60 |
| | | Total | $ 739,05 |

*Tabla 16 - Gastos del escenario 1*

*Fuente: El Autor*

El segundo escenario será una vivienda de una planta pero la misma ya no con un sistema simple sino ahora con un sistema más completo, que permita controlar hasta 12 puntos mediante el servidor web, sensor IR o acción manual 4 salidas más adicionales mediante sensores, contara con 2 sensores para medir temperatura en 2 sitios estratégicos de la casa, un sensor de movimiento para poder ser ubicado en el patio de la vivienda.

De la misma manera que el escenario anterior habrá un costo adicional por la contratación de la IP publica dichos costos de este escenario lo podremos visualizar en la *Tabla 17* donde podemos ver que el valor de implementación del sistema y diseño de software aumento debido a la utilización de sensores que le brindaran nuevas características al sistema en comparación del escenario 1.

El tercer escenario consiste en una vivienda de dos plantas con un Sistema mejorado con nuevas características el cual contara con sensores de movimiento, temperatura, distancia para poder ser ubicado en un estacionamiento, sensores infrarrojos para controlar puntos eléctricos mediante el mando a distancia y pulsadores para la acción manual.

En la *Tabla 18* se pueden visualizar en detalle la lista de materiales e incluso podemos observar como aumento el costo por implantación y diseño de software ya que en esta ocasión se aumentaron sensores y al ser una vivienda de 2 plantas aumentaría el espacio de trabajo, la implementación incluye la instalación completa del sistema a la vivienda y su adaptación correspondiente a las instalaciones eléctricas ya realizadas, en diseño de software corresponde a toda la programación que se cargara a la placa Arduino.

| Cantidad | Descripción | Valor unitario | Valor total |
|---|---|---|---|
| 1 | Arduino Mega 2560 | $ 29,46 | $ 29,46 |
| 1 | Arduino Ethernet | $ 25,00 | $ 25,00 |
| 1 | Modulo DS1307 | $ 7,10 | $ 7,10 |
| 2 | DHT11 | $ 9,02 | $ 18,04 |
| 2 | Modulo de relé 8 canales | $ 16,96 | $ 33,92 |
| 4 | Receptor infrarrojo | $ 1,12 | $ 4,48 |
| 4 | Juegos de cables Dupont (40 cables C/U) | $ 4,50 | $ 18,00 |
| 1 | Sensor PIR | $ 6,03 | $ 6,03 |
| 1 | Fuente DC variable | $ 8,88 | $ 8,88 |
| 1 | Case para placa Arduino Mega | $ 8,48 | $ 8,48 |
| 1 | Modulo Kit Ir Ideal Para Arduino | $ 7,00 | $ 7,00 |
| 1 | Display Lcd 16x4 Luz De Fondo Amarillo | $ 21,00 | $ 21,00 |
| 2 | Placa de baquelita Grande | $ 2,50 | $ 5,00 |
| 1 | Placa de baquelita Pequeña | $ 1,00 | $ 1,00 |
| 24 | Pulsadores | $ 0,25 | $ 6,00 |
| 24 | Resistencias 10k | $ 0,10 | $ 2,40 |
| 4 | Borneras de 2 vías | $ 0,50 | $ 2,00 |
| 2 | Juegos de pines para soldar Macho | $ 0,50 | $ 1,00 |
| 2 | Juegos de pines para soldar Hembra | $ 0,50 | $ 1,00 |
| 2 | Tomacorriente Bticino Polarizado | $ 2,23 | $ 4,46 |
| 2 | Canaleta Gris ranurada 25X25MM | $ 4,24 | $ 8,48 |
| 3 | Borneras en bloques #14AWG | $ 1,87 | $ 5,61 |
| 1 | Tablero Metálico 60X40X20CM | $ 61,77 | $ 61,77 |
| 1 | Breaker Siemens C16A 2 vías | $ 15,00 | $ 15,00 |
| 2 | Breaker Siemens C20 C 1 vía | $ 16,00 | $ 32,00 |
| 1 | Microswitch Rodillo Largo | $ 3,10 | $ 3,10 |
| 2 | Tope para bornera Aplicable | $ 2,07 | $ 4,14 |
| 1 | Router Marca One Wireless 300mbp | $ 26,99 | $ 26,99 |
| 50 | Cable UTP Categoria 5 50m. | $ 0,50 | $ 25,00 |
| 30 | Cable super flex #12 30m | $ 0,55 | $ 16,50 |
| 10 | Cable #10 10m | $ 0,80 | $ 8,00 |
| 1 | Implementacion e instalacion del Sistema | $ 400,00 | $ 400,00 |
| 1 | Diseño del software para cargar en el Ardu | $ 200,00 | $ 200,00 |
| | | Total | $ 1.016,84 |

*Gastos Adicionales*

| Cantidad | Descripción | Valor unitario | Valor total |
|---|---|---|---|
| 1 | IP Publica de pago mensual | $ 25,60 | $ 25,60 |
| | | Total | $ 1.042,44 |

*Tabla 17 - Gastos del Escenario 2*

*Fuente: El autor*

| Cantidad | Descripción | Valor unitario | Valor total |
|---|---|---|---|
| 2 | Arduino Mega 2560 | $ 29,46 | $ 58,92 |
| 1 | Arduino Ethernet | $ 25,00 | $ 25,00 |
| 1 | Modulo DS1307 | $ 7,10 | $ 7,10 |
| 3 | DHT11 | $ 9,02 | $ 27,06 |
| 3 | Modulo de relé 8 canales | $ 16,96 | $ 50,88 |
| 5 | Receptor infrarrojo | $ 1,12 | $ 5,60 |
| 4 | Juegos de cables Dupont (40 cables C/U) | $ 4,50 | $ 18,00 |
| 1 | Sensor HC-SR04 | $ 6,70 | $ 6,70 |
| 2 | Sensor PIR | $ 6,03 | $ 12,06 |
| 1 | Sensor de gas | $ 9,00 | $ 9,00 |
| 1 | Fuente DC variable | $ 8,88 | $ 8,88 |
| 1 | Case para placa Arduino Mega | $ 8,48 | $ 8,48 |
| 1 | Modulo Kit Ir Ideal Para Arduino | $ 7,00 | $ 7,00 |
| 1 | Display Lcd 16x4 Luz De Fondo Amarillo | $ 21,00 | $ 21,00 |
| 3 | Placa de baquelita Grande | $ 2,50 | $ 7,50 |
| 2 | Placa de baquelita Pequeña | $ 1,00 | $ 2,00 |
| 34 | Pulsadores | $ 0,25 | $ 8,50 |
| 34 | Resistencias 10k | $ 0,10 | $ 3,40 |
| 4 | Borneras de 2 vías | $ 0,50 | $ 2,00 |
| 3 | Juegos de pines para soldar Macho | $ 0,50 | $ 1,50 |
| 3 | Juegos de pines para soldar Hembra | $ 0,50 | $ 1,50 |
| 2 | Tomacorriente Bticino Polarizado | $ 2,23 | $ 4,46 |
| 2 | Canaleta Gris ranurada 25X25MM | $ 4,24 | $ 8,48 |
| 4 | Borneras en bloques #14AWG | $ 1,87 | $ 7,48 |
| 1 | Tablero Metálico 60X40X20CM | $ 61,77 | $ 61,77 |
| 1 | Breaker Siemens C16A 2 vías | $ 15,00 | $ 15,00 |
| 3 | Breaker Siemens C20 C 1 vía | $ 16,00 | $ 48,00 |
| 1 | Microswitch Rodillo Largo | $ 3,10 | $ 3,10 |
| 2 | Tope para bornera Aplicable | $ 2,07 | $ 4,14 |
| 1 | Router Marca One Wireless 300mbp | $ 26,99 | $ 26,99 |
| 80 | Cable UTP Categoria 5 50m. | $ 0,50 | $ 40,00 |
| 40 | Cable super flex #12 30m | $ 0,55 | $ 22,00 |
| 20 | Cable #10 10m | $ 0,80 | $ 16,00 |
| 1 | Implementacion e instalacion del Sistema | $ 600,00 | $ 600,00 |
| 1 | Diseño del software para cargar en el Ardu | $ 300,00 | $ 300,00 |
| | | Total | $ 1.449,50 |

*Gastos Adicionales*

| Cantidad | Descripción | Valor unitario | Valor total |
|---|---|---|---|
| 1 | IP Publica de pago mensual | $ 25,60 | $ 25,60 |
| | | Total | $ 1.475,10 |

*Tabla 18 - Gastos del Escenario 3*

*Fuente: El Autor*

Los costos ubicados en la *Tabla 19* corresponden en el caso que el usuario no cuente con un teléfono celular o una computadora, siempre y cuando el mismo lo requiera. Para poder controlar el sistema desde cualquier lugar, en cualquiera de los 3 escenarios hemos planteado la opción de una IP pública, pero otra alternativa y tal vez más económica, es la contratación de un servicio DDNS lo cual también tiene un pago mensual que correría por cuenta del usuario.

| ***Cantidad*** | ***Descripción*** | ***Valor unitario*** | ***Valor total*** |
|---|---|---|---|
| *1* | *Celular Samsung para tener acceso al siste* | *$ 190,00* | *$ 190,00* |
| *1* | *Computador basico con windows 10* | *$ 450,00* | *$ 450,00* |
| *1* | *IP publica Pago mensual* | *$ 25,60* | *$ 25,60* |

*Tabla 19 - Costos de Elementos extras*

*Fuente: El Autor*

## 4.5. Análisis comparativo de costos del Sistema de domótica con Arduino en relación con Sistemas Existentes.

En el siguiente análisis vamos a tomar como ejemplo de comparación el escenario 2 descrito anteriormente como podemos observar en la *Tabla 17* los gastos para la implementación del Sistema de domótica con Arduino en este escenario rodean los $ 1016,84 fuera del pago mensual de la IP publica que correría por cuenta del usuario. En este escenario el sistema cuenta con las características de monitorización y control del sistema mediante el servidor web, 12 relés para aparatos electrónicos controlados mediante servidor web, comunicación IR o modo manual, dos relés para ser controlados mediante temperatura, uno por movimiento y uno por tiempo y además cuenta con una pantalla donde se podrá visualizar hora, fecha y temperatura actuales.

Tomando como punto de partida las características descritas anteriormente se tomó en cuenta dos sistemas de domótica de marcas reconocidas, cotizando el valor de los equipos que se necesiten para que asemejen las características con las que cuenta el escenario 2 y el costo por instalación.

El primer sistema de domótica que se escogió para realizar la comparación de costos es el Sistema de domótica Fibaro con Home Center 2, el costo de este sistema con sus respectivos elementos se puede apreciar en la *Tabla 20*. El segundo sistema que se escogió es el Sistema de domótica de Bticino con My Home en la *Tabla 21* se puede observar el costo total de la instalación que incluye los elementos a instalar.

| ***Cantidad*** | ***Descripción*** | ***Valor unitario*** | ***Valor total*** |
|---|---|---|---|
| *1* | *Controlador Home Center 2* | *$ 902,50* | *$ 902,50* |
| *8* | *Relay Switch 2x1,5kW* | *$ 66,50* | *$ 532,00* |
| *1* | *Smoke Sensor* | *$ 57,34* | *$ 57,34* |
| *1* | *Motion Sensor* | *$ 64,50* | *$ 64,50* |
| *1* | *Instalacion y configuracion del sistema* | *$ 500,00* | *$ 500,00* |
| | | *Total* | *$ 2.056,34* |

*Tabla 20 - Costos de Sistema de domótica con Home Center 2*

*Fuente: SmartHome - Soluciones Inteligentes*

| ***Descripción*** | ***Valor total*** |
|---|---|
| *Instalación del sistema de domótica My Home para una vivienda básica con los módulos de control de iluminación, servidor web, modulo de memoria para el estado de los actuadorescon sus respectiva fuente de alimentación.* | *$ 3.142,18* |
| *Instalación de 3 sensores de temperatura, 2 sensores de movimiento, mando infrarrojo y detector de humo.* | *$ 1.346,81* |
| *Total* | *$ 4.488,99* |

*Tabla 21 - Costos del Sistema de domótica con My Home*

*Fuente: SmartHome – PROHOME Hogares Inteligentes*

En las tablas anteriores se detallaron los costos por instalación y elementos de cada sistema, como podemos visualizar el sistema de domótica con Arduino tiene un costo muy inferior en la relación con los otros dos sistemas el mismo se asemeja a las características que ofrecen los otros sistemas de domótica, con este análisis podemos decir que este sistema de domótica con Arduino puede llegar a competir en el mercado ya que ofrece las mismas o mejores características que otros y a más bajo costo.

# CAPÍTULO 5. CONCLUSIONES Y RECOMENDACIONES

## 5.1. Conclusiones

- La existencia de un laboratorio de domótica fomentará el desarrollo y avance tecnológico, promoviendo el uso de plataformas libres para el mismo se abarataran costos y a la vez puede existir una mayor adquisición de recursos tecnológicos.
- Incluir nuevas materias en la malla curricular, apoyaría en la expansión de la carrera de Ingeniería Eléctrica en otros campos, teniendo en cuenta que temas que apunten hacia el desarrollo tecnológico como la domótica influirían en el perfil del profesional.
- Dando a conocer un Sistema de domótica con Arduino, que presente características y prestaciones importantes a bajo costo llamaremos la atención de personas que mostraran interés por la instalación de este tipo de tecnologías en sus hogares.
- Con el pasar de los días se crean nuevas tendencias en tecnología, incluso se crean nuevas placas Arduino que nos ofrecen nuevas características, teniendo mayores beneficios al utilizar esta plataforma para domótica, esta investigación sirve como apoyo si en un futuro se desee implementar el proyecto de manera real o algún estudiante se interese por mejorar el Sistema.

## 5.2. Recomendaciones

- Se considera importante promover la existencia de un laboratorio de Domótica, que cuente con plataformas de acceso libre y muchos recursos tecnológicos que promuevan la investigación, mejoren los conocimientos y motiven al desarrollo de nuevas tendencias.

- Sería importante que al momento de reformar la malla curricular de la carrera de Ingeniería Eléctrica, se tome en cuenta temas como la enseñanza de la plataforma Arduino que ya se enseña en otras universidades del país.

- Implementando un sistema de domótica con Arduino en nuestra vivienda, se puede lograr ya sea desde obtener la temperatura ambiente hasta encender un foco mediante nuestro Smartphone, se recomienda aplicarlo debido a que mejoraría considerablemente nuestra forma de vida y a más de eso es un Sistema que lo encontramos a más bajo costo en comparación con otros existentes.

- Motivar a los estudiantes que muestren más interés por la investigación de la plataforma Arduino, ya que esta no solo se puede utilizar para proyectos de domótica sino también para un sinnúmero de proyectos y aplicaciones inimaginables.

# REFERENCIAS


Alberto, S. G. (2013). *MICROCONTROLADORES*. Obtenido de http://losmicrocontroladores.blogspot.com/

ARDUINO. (2012). *Arduino Ethernet Shield*. Obtenido de Arduino: https://www.arduino.cc/en/Main/ArduinoEthernetShield

ARDUINO. (2012). *Libraries*. Obtenido de https://www.arduino.cc/en/Reference/Libraries

ARDUINO. (2013). *Introducción a Arduino*. Obtenido de https://www.arduino.cc/en/Guide/Introduction

ARDUINO. (2014). *Arduino MEGA 2560*. Obtenido de Arduino: https://www.arduino.cc/en/Main/ArduinoBoardMega2560

Ardumanía. (2013). *LCD gráfico*. Obtenido de http://www.ardumania.es/lcd-grafico-de-84x48/

Atmel Corporation. (2012). *Microcontroller*. Obtenido de Atmel: http://www.atmel.com/products/microcontrollers/default.aspx

BURUTEK. (2013). *BURUTEK*. Obtenido de http://burutek.org/es/arduino/

CEDOM. (2014). *Domótica*. Obtenido de http://www.cedom.es/sobre-domotica/que-es-domotica

ClaudioVella. (10 de 2012). *CONTROL DE TU CASA DESDE INTERNET*. Obtenido de Prometec: http://www.prometec.net/etherswitch/

ELECTRONICAESTUDIO. (2013). *Sensores*. Obtenido de electronicaestudio.com: http://www.electronicaestudio.com/sensores.htm

Electronics, S. (2012). *WIZnet W5100*. Obtenido de https://www.sparkfun.com/categories

Electronilab. (2013). *Sensores*. Obtenido de http://electronilab.co/categoria-producto/sensores/

Geekfactory. (2014). *DS1307 con Arduino*. Obtenido de Geekfactory.mx: http://www.geekfactory.mx/tutoriales/tutoriales-arduino/ds1307-en-tinyrtc-con-arduino/

GitHub. (2014). *GitHub*. Obtenido de GitHub: https://github.com/

Martínez, F. (2014). *OpenWebinars*. Obtenido de https://openwebinars.net

RIBELLES, T. (2015). *Beneficios de la domótica en el hogar*. Obtenido de http://interiorismos.com/beneficios-de-la-domotica-en-el-hogar/

Sáez, D. (2012). *Domótica.* Obtenido de Domótica: http://web.iti.upv.es

Web-Robótica. (2013). *Web-Robótica*. Obtenido de http://www.web-robotica.com/arduino

Wiring. (2012). *Libraries\Wiring*. Obtenido de http://wiring.org.co/reference/es/libraries/

WIZnet Co., L. (2013). *Chip W5100*. Obtenido de WIZnet: http://www.wiznet.co.kr/product-item/w5100/

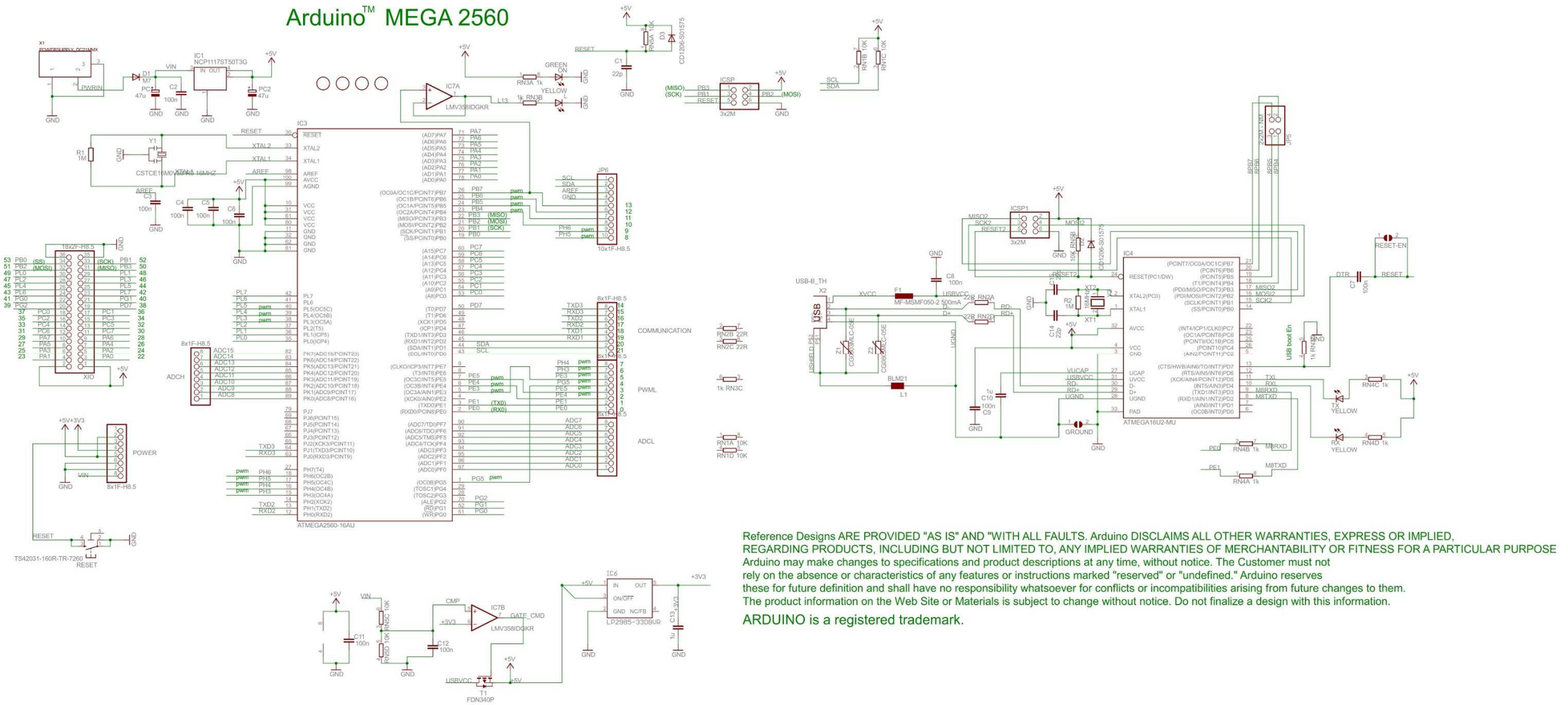
# ANEXOS

## *Anexo 1 - Placas Arduino Existentes*

| Arduino | Procesador | Voltaje Operativo / Entrada | CPU Speed | Analog In/Out | Digital IO/PWM | EEPROM [KB] | SRAM [KB] | Flash [KB] | USB | UART |
|---|---|---|---|---|---|---|---|---|---|---|
| **BT** | ATmega328P | 5 V / 2.5-12 V | 16 MHz | 6/0 | 14/6 | 1 | 2 | 32 | - | 1 |
| **Due** | ATSAM3X8E | 3.3 V / 7-12 V | 84 MHz | 12/2 | 54/12 | - | 96 | 512 | 2 Micro | 4 |
| **Esplora** | ATmega32U4 | 5 V / 7-12 V | 16 MHz | - | - | 1 | 2.5 | 32 | Micro | - |
| **Ethernet** | ATmega328P | 5 V / 7-12 V | 16 MHz | 6/0 | 14/4 | 1 | 2 | 32 | Regular | - |
| **Fio** | ATmega328P | 3.3 V / 3.7-7 V | 8 MHz | 8/0 | 14/6 | 1 | 2 | 32 | Mini | 1 |
| **Gemma** | ATtiny85 | 3.3 V / 4-16 V | 8 MHz | 1/0 | 3/2 | 0.5 | 0.5 | 8 | Micro | 0 |
| **Leonardo** | ATmega32U4 | 5 V / 7-12 V | 16 MHz | 12/0 | 20/7 | 1 | 2.5 | 32 | Micro | 1 |
| **LilyPad** | ATmega168V ATmega328P | 2.7-5.5 V / 2.7-5.5 V | 8MHz | 6/0 | 14/6 | 0.512 | 1 | 16 | - | - |
| **LilyPad SimpleSnap** | ATmega328P | 2.7-5.5 V / 2.7-5.5 V | 8 MHz | 4/0 | 9/4 | 1 | 2 | 32 | - | - |
| **LilyPad USB** | ATmega32U4 | 3.3 V / 3.8-5 V | 8 MHz | 4/0 | 9/4 | 1 | 2.5 | 32 | Micro | - |
| **Mega 2560** | ATmega2560 | 5 V / 7-12 V | 16 MHz | 16/0 | 54/15 | 4 | 8 | 256 | Regular | 4 |
| **Mega ADK** | ATmega2560 | 5 V / 7-12 V | 16 MHz | 16/0 | 54/15 | 4 | 8 | 256 | Regular | 4 |
| **Micro** | ATmega32U4 | 5 V / 7-12 V | 16 MHz | 12/0 | 20/7 | 1 | 2.5 | 32 | Micro | 1 |
| **Mini** | ATmega328P | 5 V / 7-9 V | 16 MHz | 8/0 | 14/6 | 1 | 2 | 32 | - | - |
| **Nano** | ATmega168 ATmega328P | 5 V / 7-9 V | 16 MHz | 8/0 | 14/6 | 0.5121 | 12 | 1632 | Mini | 1 |
| **Pro** | ATmega168 ATmega328P | 3.3 V / 3.35-12 V 5 V / 5-12 V | 8 MHz 16 MHz | 6/0 | 14/6 | 0.5121 | 12 | 1632 | - | 1 |
| **Pro Mini** | ATmega328P | 3.3 V / 3.35-12 V 5 V / 5-12 V | 8 MHz 16 MHz | 6/0 | 14/6 | 0.512 | 1 | 16 | - | 1 |
| **Uno** | ATmega328P | 5 V / 7-12 V | 16 MHz | 6/0 | 14/6 | 1 | 2 | 32 | Regular | 1 |
| **Yùn** | ATmega32U4 AR9331 Linux | 5 V | 16 MHz 400MHz | 12/0 | 20/7 | 1 | 2.5 16MB | 32 64MB | Micro | 1 |
| **Zero** | ATSAMD21G18 | 3.3 V / 7-12 V | 48 MHz | 6/1 | 14/10 | - | 32 | 256 | 2 Micro | 2 |

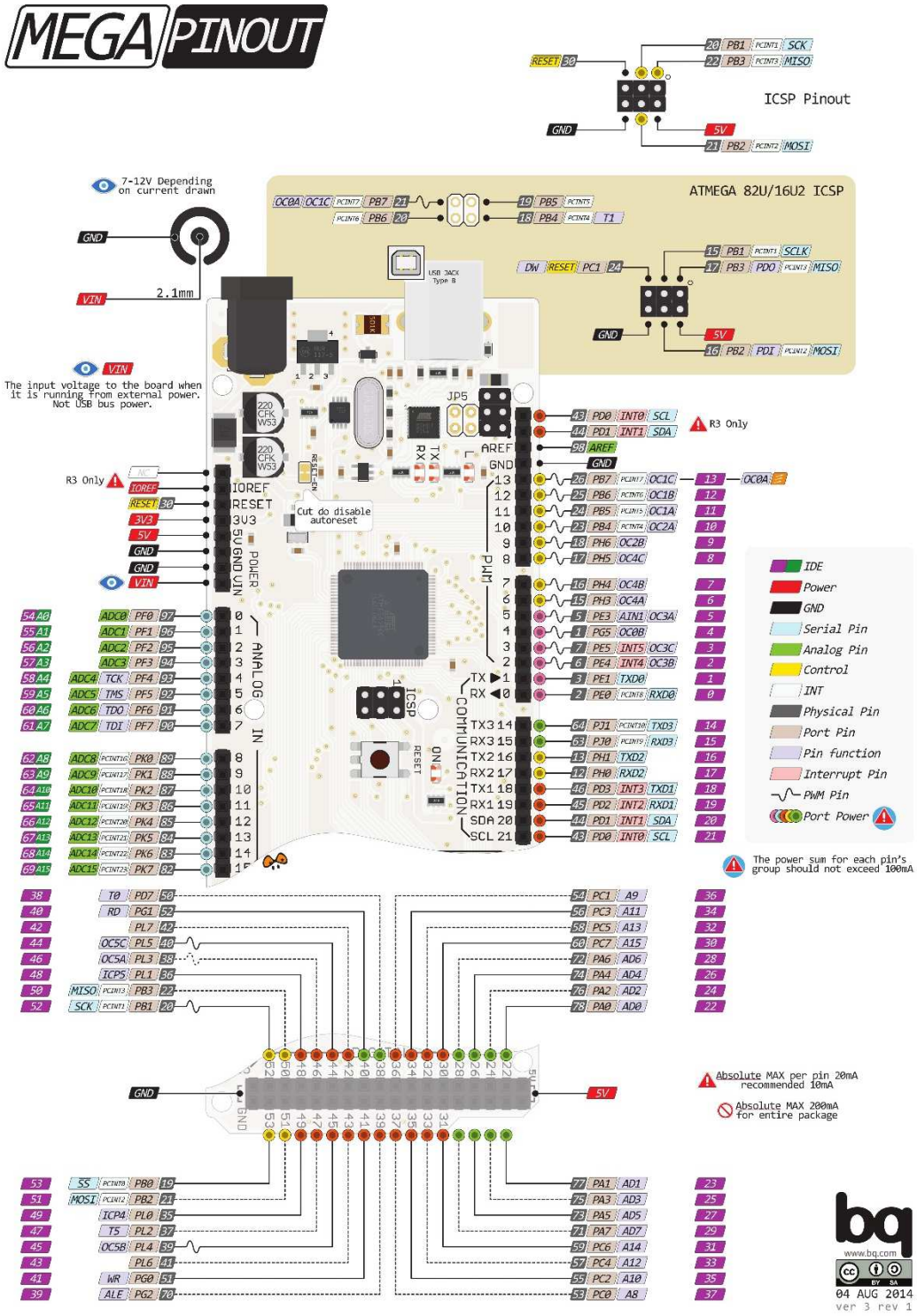
*Fuente: https://www.arduino.cc*

***Anexo 2 - Diagrama del Arduino Mega 2560***



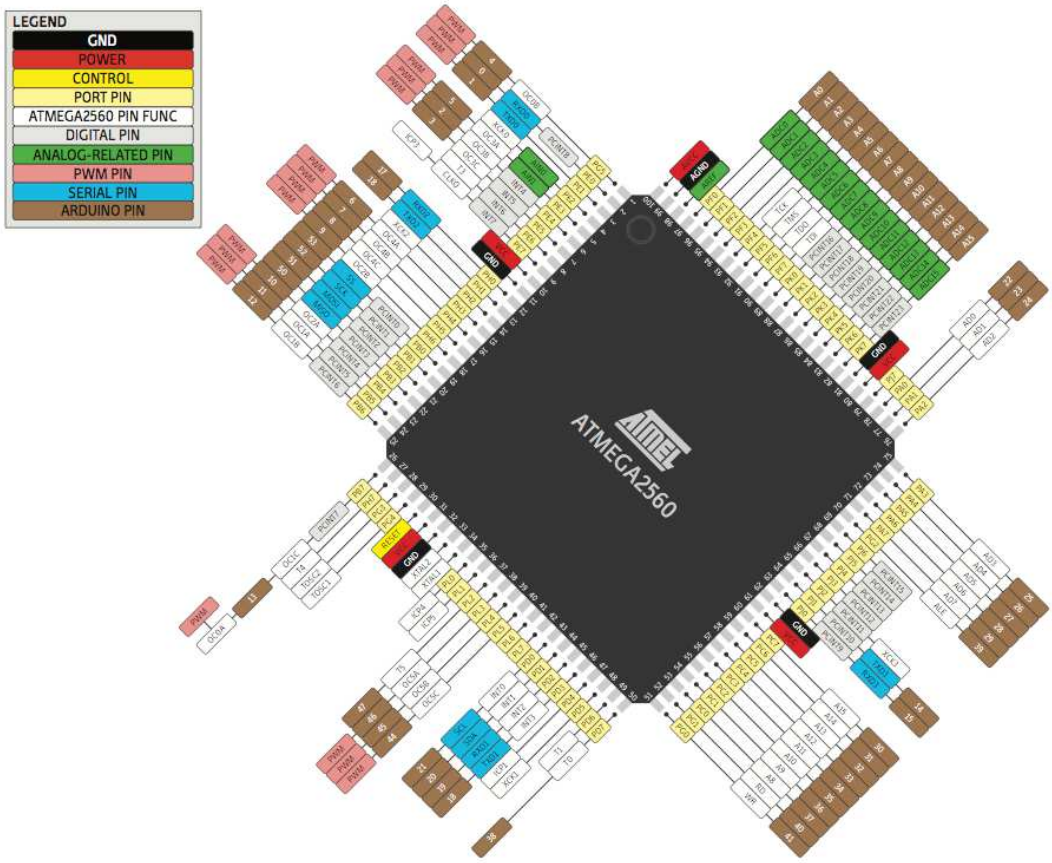

*Fuente: https://www.arduino.cc*

## *Anexo 3 - Pines del Arduino Mega 2560*




*Fuente: https://www.arduino.cc/en/Main/ArduinoBoardMega2560*

## *Anexo 4 - Pines Atmega 2560*




*Fuente: https://www.arduino.cc/en/Main/ArduinoBoardMega2560*

***Anexo 5 - Asignación de los pines Atmega2560 – Arduino***

| *Numero PIN* | *Nombre Pin* | *Asignado Nombre Pin* |
|---|---|---|
| *1* | *PG5 (OC0B)* | *Pin digital 4 (PWM)* |
| *2* | *PE0 (RXD0 / PCINT8)* | *Pin digital 0 (RX0)* |
| *3* | *PE1 (TXD0)* | *Pin digital 1 (TX0)* |
| *4* | *PE2 (XCK0 / AIN0)* | |
| *5* | *PE3 (OC3A / AIN1)* | *Pin digital 5 (PWM)* |
| *6* | *PE4 (OC3B / INT4)* | *Pin digital 2 (PWM)* |
| *7* | *PE5 (OC3c / INT5)* | *Pin Digital 3 (PWM)* |
| *8* | *PE6 (T3 / INT6)* | |
| *9* | *PE7 (CLKO / ICP3 / INT7)* | |
| *10* | *VCC* | *VCC* |
| *11* | *GND* | *GND* |
| *12* | *PH0 (RXD2)* | *Pin digital 17 (RX2)* |
| *13* | *PH1 (TXD2)* | *Pin digital 16 (TX2)* |
| *14* | *PH2 (XCK2)* | |
| *15* | *PH3 (OC4A)* | *Pin digital 6 (PWM)* |
| *16* | *PH4 (OC4B)* | *Pin digital 7 (PWM)* |
| *17* | *PH5 (OC4C)* | *Pin digital 8 (PWM)* |
| *18* | *PH6 (OC2B)* | *Pin digital 9 (PWM)* |
| *19* | *PB0 (SS / PCINT0)* | *Pin digital 53 (SS)* |
| *20* | *PB1 (SCK / PCINT1)* | *Pin digital 52 (SCK)* |
| *21* | *PB2 (MOSI / PCINT2)* | *Pin digital 51 (MOSI)* |
| *22* | *PB3 (MISO / PCINT3)* | *Pin digital 50 (MISO)* |

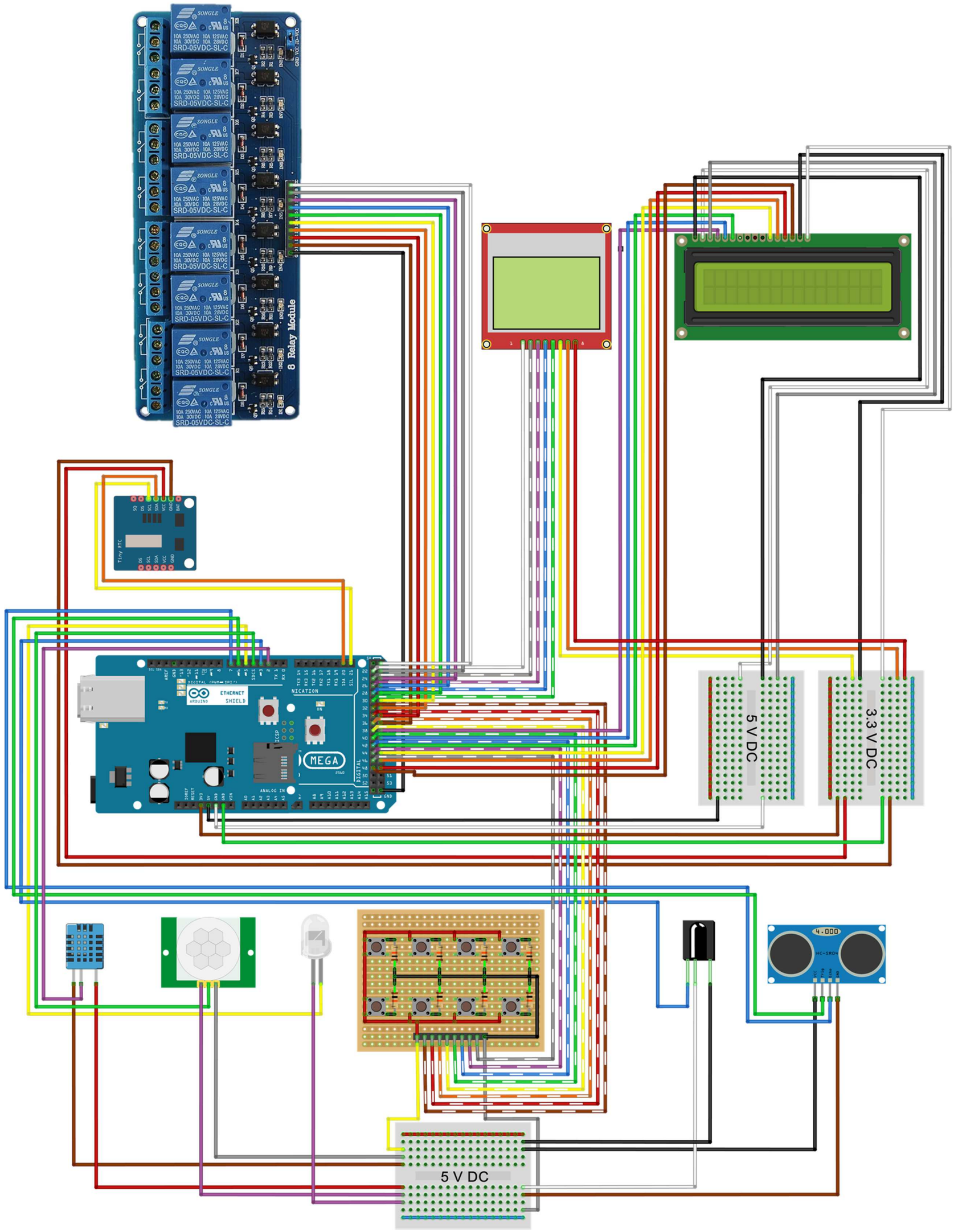
| | | |
|---|---|---|
| *23* | *PB4 (OC2A / PCINT4)* | *Pin digital 10 (PWM)* |
| *24* | *PB5 (OC1A / PCINT5)* | *Pin digital 11 (PWM)* |
| *25* | *PB6 (OC1B / PCINT6)* | *Pin digital 12 (PWM)* |
| *26* | *PB7 (OC0A / OC1C / PCINT7)* | *Pin digital 13 (PWM)* |
| *27* | *PH7 (T4)* | |
| *28* | *PG3 (TOSC2)* | |
| *29* | *PG4 (TOSC1)* | |
| *30* | *REINICIO* | *REINICIO* |
| *31* | *VCC* | *VCC* |
| *32* | *GND* | *GND* |
| *33* | *XTAL2* | *XTAL2* |
| *34* | *XTAL1* | *XTAL1* |
| *35* | *PL0 (ICP4)* | *Pin digital 49* |
| *36* | *PL1 (ICP5)* | *Pin digital 48* |
| *37* | *PL2 (T5)* | *Pin digital 47* |
| *38* | *PL3 (OC5A)* | *Pin digital 46 (PWM)* |
| *39* | *PL4 (OC5B)* | *Pin digital 45 (PWM)* |
| *40* | *PL5 (OC5C)* | *Pin digital 44 (PWM)* |
| *41* | *PL6* | *Pin digital 43* |
| *42* | *PL7* | *Pin digital 42* |
| *43* | *PD0 (SCL / INT0)* | *Pin digital 21 (SCL)* |
| *44* | *PD1 (SDA / INT1)* | *Pin digital 20 (SDA)* |
| *45* | *PD2 (RXDI / INT2)* | *Pin digital 19 (RX1)* |
| *46* | *PD3 (TXD1 / INT3)* | *Pin digital 18 (TX1)* |
| *47* | *PD4 (ICP1)* | |
| *48* | *PD5 (XCK1)* | |
| *49* | *PD6 (T1)* | |

| | | |
|---|---|---|
| *50* | *PD7 (T0)* | *Pin digital 38* |
| *51* | *PG0 (WR)* | *Pin digital 41* |
| *52* | *PG1 (RD)* | *Pin digital 40* |
| *53* | *PC0 (A8)* | *Pin digital 37* |
| *54* | *PC1 (A9)* | *Pin digital 36* |
| *55* | *PC2 (A10)* | *Pin digital 35* |
| *56* | *PC3 (A11)* | *Pin digital 34* |
| *57* | *PC4 (A12)* | *Pin digital 33* |
| *58* | *PC5 (A13)* | *Pin digital 32* |
| *59* | *PC6 (A14)* | *Pin digital 31* |
| *60* | *PC7 (A15)* | *Pin digital 30* |
| *61* | *VCC* | *VCC* |
| *62* | *GND* | *GND* |
| *63* | *PJ0 (RXD3 / PCINT9)* | *Pin digital 15 (RX3)* |
| *64* | *PJ1 (TXD3 / PCINT10)* | *Pin digital 14 (TX3)* |
| *65* | *PJ2 (XCK3 / PCINT11)* | |
| *66* | *PJ3 (PCINT12)* | |
| *67* | *PJ4 (PCINT13)* | |
| *68* | *PJ5 (PCINT14)* | |
| *69* | *PJ6 (PCInt 15)* | |
| *70* | *PG2 (ALE)* | *Pin digital 39* |
| *71* | *PA7 (AD7)* | *Pin digital 29* |
| *72* | *PA6 (AD6)* | *Pin digital 28* |
| *73* | *PA5 (AD 5)* | *Pin digital 27* |
| *74* | *PA4 (AD4)* | *Pin digital 26* |
| *75* | *PA3 (AD3)* | *Pin digital 25* |
| *76* | *PA2 (AD2)* | *Pin digital 24* |

| | | |
|---|---|---|
| *77* | *PA1 (AD1)* | *Pin digital 23* |
| *78* | *PA0 (Ad0)* | *Pin digital 22* |
| *79* | *PJ7* | |
| *80* | *VCC* | *VCC* |
| *81* | *GND* | *GND* |
| *82* | *PK7 (ADC15 / PCINT23)* | *Pin analógico 15* |
| *83* | *PK6 (ADC14 / PCINT22)* | *Pin analógico 14* |
| *84* | *PK5 (ADC13 / PCINT21)* | *Pin analógico 13* |
| *85* | *PK4 (ADC12 / PCINT20)* | *Pin analógico 12* |
| *86* | *PK3 (ADC11 / PCINT19)* | *Pin analógico 11* |
| *87* | *PK2 (ADC10 / PCINT18)* | *Pin analógico 10* |
| *88* | *PK1 (ADC9 / PCINT17)* | *Pin analógico 9* |
| *89* | *PK0 (ADC8 / PCINT16)* | *Pin analógico 8* |
| *90* | *PF7 (ADC7)* | *Pin analógico 7* |
| *91* | *PF6 (ADC6)* | *Pin analógico 6* |
| *92* | *PF5 (ADC5 / TMS)* | *Pin analógico 5* |
| *93* | *PF4 (ADC4 / TMK)* | *Pin analógico 4* |
| *94* | *PF3 (ADC3)* | *Pin analógico 3* |
| *95* | *PF2 (ADC2)* | *Pin analógico 2* |
| *96* | *PF1 (ADC1)* | *Pin analógico 1* |
| *97* | *PF0 (ADC0)* | *Pin analógico 0* |
| *98* | *AREF* | *Referencia analógica* |
| *99* | *GND* | *GND* |
| *100* | *AVCC* | *VCC* |

*Fuente: https://www.arduino.cc/en/Hacking/PinMapping2560*

## *Anexo 6 - Diagrama de conexiones del Prototipo*




| | | | |
|---|---|---|---|
| | ***Diagrama de conexiones del Prototipo*** | ***Uleam*** ***Carrera de Ing. Eléctrica*** | |
| | | ***Elaborado por:*** *Sandro Quijije Marín* | ***Fecha:*** *Noviembre 2015* |

*Anexo 7 - Estructura, Variables y funciones*

**Estructura**

- setup ()
- loop ()

**Estructuras de Control**

- if
- if...else
- for
- while
- do... while
- continue

**Sintaxis**

- ; (Punto y coma)
- {} (Llaves)
- // (línea de comentario)
- / * * / (Multi-línea de comentario)
- #define
- #include

**Operadores de comparación**

- == (Igual a)
- ! = (Distinto de)
- <(Menor que)
- > (Mayor que)
- <= (Menor o igual a)
- > = (Mayor que o igual a)

**Operadores booleanos**

- && (Y)
- || (O)
- ! (No)

**Variables**

**Constantes**

- HIGH | LOW
- INPUT | OUTPUT | INPUT_PULLUP
- LED_BUILTIN
- true | false
- integer constants

**Tipos de datos**

- void
- boolean
- char
- unsigned char
- byte
- int
- unsigned int
- long
- unsigned long
- short
- float
- double

**Conversión**

- char()
- byte()
- int()
- word()
- long()
- float()

**Variable y Calificadores**

- variable scope
- static
- const

**Funciones**

**I/O Digital**

- pinMode()
- digitalWrite()
- digitalRead()

**I/O Analógicas**

- analogReference()
- analogRead()
- analogWrite() – PWM

**I/O Advanzadas**

- tone()
- shiftOut()
- shiftIn()
- pulseIn()

**Tiempo**

- millis()
- micros()
- delay()
- delayMicroseconds()

**Bits y Bytes**

- lowByte()
- highByte()
- bitRead()
- bitWrite()
- bitSet()
- bitClear()

**Comunicación**

- Serial
- Stream

*Fuente: https://www.arduino.cc*

## *Anexo 8 - Código de programación del Prototipo*

```
//////////////////////////////////////////////////////////////
// Diseño de un Sistema de Domótica con Arduino Mega 2560 y
// Arduino Ethernet Shield, conectado y controlado remotamente
// Desde un Servidor Web, para ser implementado en el sector
// Residencial de la Ciudad de Manta.
//////////////////////////////////////////////////////////////
//
// Descripción y Características
//  1. Salidas invertidas para activar los relés en modo HIGH.
//  2. Mostrar Datos de IP, Fecha, Hora, Temperatura y humedad
mediante LCD 16X4.
//  3. Mostrar Títulos Mediante LCD 84x48 - Nokia 5110.
//  4. Activa Pines 23, 25, 27, 29 desde la página Web.
//  5. Activa Pines 23, 25, 27, 29 mediante un control remoto.
//  6. Activa Pines 23, 25, 27, 29 mediante Pulsadores (Modo
Manual).
//  7. Activa Pin 31 mediante sensor de movimiento.
//  8. Activa Pin 33 Mediante Temperatura (rango mínimo 32).
//  9. Activa Pin 35 Mediante sensor de distancia (Aparcamiento).
// 10. Activa Pin 37 y lo desactiva en un tiempo preestablecido.
// 11. Guarda los estados de las salidas 23, 25, 27, 29 después de
un corte de energía.
// 12. Cualquier dispositivo inteligente (Smartphone, Tablet,
Laptop, etc.) con conexión Wifi Puede Conectarse al servidor Web.
// 13. Datos de la Red Wifi: Nombre de red: Arduino Clave de Red:
arduinomega.
//
// Descripción y Características de la página Web (Mediante servidor
web creado por la Placa Arduino)
//  1. Interfaz Web Amigable mostrando los estados de las salidas.
//  2. Muestra Datos leídos por los sensores de: Temperatura
//                                               Humedad Relativa
//                                               Hora
//                                               Fecha
//  3. Muestra los estados de los demás sensores.
//
// Observaciones a tomar en consideración
// Pin  10, 11, 12 y 13 en el Duemilanove se utilizan para el escudo
Ethernet, por lo tanto no se puede utilizar.
// Pin  10, 50, 51 y 52 y 53 en los Mega se utilizan para el escudo
Ethernet, por lo tanto no se puede utilizar.
// Pin  4, se utilizan para la tarjeta SD, por lo tanto no se puede
utilizar.
// Pin  2 se utiliza para la notificación de interrupción impulsada,
por tanto, no podría ser utilizado.

//////////////////////////////////////////////////////////////
// Librerías
//////////////////////////////////////////////////////////////

#include <Ethernet.h>     // Librería Ethernet
#include <SPI.h>          // Librería Serial Peripheral Interface
(SPI)
#include <EEPROM.h>       // Librería de Memoria EEPROM

#include "DHT.h"          // Librería Sensor Temperatura
```

```
#define DHTPIN 2          // Pin al que se conecta la señal del
sensor
#define DHTTYPE DHT11     // Tipo de sensor
DHT dht(DHTPIN, DHTTYPE); // Inicializa Sensor DHT

#include <Wire.h>         // Librería para comunicarse con
I2C  Mega2560  20 (SDA), 21 (SCL)
#include "RTClib.h"       // Librería Reloj RTC DS1307
RTC_DS1307 RTC;           // Inicializa Sensor Reloj

#include <LiquidCrystal.h> // Librería LCD para Display 16X4
LiquidCrystal lcd(39, 41, 43, 45, 47, 49, 48); // Se define los
pines

////////////////////////////////////////////////////////////////
//             -----------                 ----------
//            |  Arduino  |               |   LCD    |
//            |   MEGA    |               |          |
//            |        GND|-------------->|VSS       |
//            |        VCC|-------------->|VDD       |
//            |        GND|-------------->|VO        |
//            |     PIN 39|-------------->|RS        |
//            |     PIN 41|-------------->|RW        |
//            |     PIN 43|-------------->|E         |
//            |           |               |DB0       |
//            |           |               |DB1       |
//            |           |               |DB2       |
//            |           |               |DB3       |
//            |     PIN 45|-------------->|DB4       |
//            |     PIN 47|-------------->|DB5       |
//            |     PIN 49|-------------->|DB6       |
//            |     PIN 48|-------------->|DB7       |
//            |       3.3V|-------------->|LED A     |
//            |        GND|-------------->|LED K     |
//             -----------                 ----------
////////////////////////////////////////////////////////////////


#include <LCD5110_Graph.h> // Librería Graphic LCD 84x48 - Nokia
5110
LCD5110 myGLCD(30, 28, 26, 22, 24); // Se define los pines
extern unsigned char TinyFont[]; // Fuente de texto en LCD 84x48 -
Nokia 5110

//    Conexion de pines
//      SCK  - Pin 8
//      MOSI - Pin 9
//      DC   - Pin 10
//      RST  - Pin 11
//      CS   - Pin 12



#include <IRremote.h>     // Librería IR (Recibe o transmite códigos
de control remoto por infrarrojos.)
#define RECV_PIN 3        // Pin por el que recibimos los datos del
sensor IR
boolean encendido;        // Variable donde se comprueba si el led
esta encendido o no
IRrecv irrecv(RECV_PIN);  // Inicializa Sensor IR en modo Receiver
decode_results results;
```

```
#include <Ultrasonic.h>  // Librería para sensor Ultrasonico
Ultrasonic ultra1(6, 7); // Declarando los pines (Trig,Echo)


////////////////////////////////////////////////////////////////
//Configuración y Ajustes del Ethernet Shield
////////////////////////////////////////////////////////////////

//Configuración de la IP
byte ip[] = {
  192, 168, 0, 4
};   //IP
byte gateway[] = {
  192, 168, 0, 2
};   //Puerta de enlace
byte subnet[] = {
  255, 255, 255, 0
}; //Máscara de Subred

//Dirección MAC de la Ethernet Shield
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

//Puerto Ethernet
EthernetServer server = EthernetServer(80); //Por defecto el puerto
html 80

// Número de salidas
int outputQuantity = 4;  //no debe superar los 8
int outputQuantity2 = 4;  //no debe superar los 8

// Invertir las salidas
boolean outputInverted = true; //true or false (Verdadero o Falso)
boolean outputInverted2 = true; //true or false (Verdadero o Falso)
// Esto se hace en caso de que la placa de relé dispara el relé en
negativo, en lugar de sobre alimentación positiva

// Actualización de la página HTML
int refreshPage = 20; //20 segundos por default
//Tener en cuenta que si se hace refrescar demasiado rápido, la
página podría convertirse inaccesible.

// Mostrar u ocultar "Encender todos los Botones" en la parte
inferior de la página.
int switchOnAllPinsButton = true; //true or false (Verdadero o
Falso)

// Orden de Botones
//Iniciar de 0 a 7, como 0 se cuenta entonces son 8 salidas

// Pines de salida
int outputAddress[4] = { 23, 25, 27, 29,}; //Asignar 8 espacios y el
nombre de la dirección de pin de salida.
int outputAddress2[4] = {31, 33, 35, 37}; //Asignar 8 espacios y el
nombre de la dirección de pin de salida.


// Descripción del canal de salida
String buttonText[11] = {
  "Web+IR+Pulsador", "Web+IR+Pulsador", "Web+IR+Pulsador",
"Web+IR+Pulsador"
```

```
};

String buttonText2[11] = {
  "Sensor de Movimiento", "Sensor de Temperatura", "Sensor de
Distancia", "Encendido por Tiempo"
};

// Ajuste la salida de retener el último estado.
int retainOutputStatus[4] = {1, 1, 1, 1}; //1-retener el último
estado. 0-será apagado después de corte de energía.
int retainOutputStatus2[4] = {1, 1, 1, 1}; //1-retener el último
estado. 0-será apagado después de corte de energía.

////////////////////////////////////////////////////////////////
// Variables de Declaración Generales
////////////////////////////////////////////////////////////////

int outp = 0;
boolean printLastCommandOnce = false;
boolean printButtonMenuOnce = false;
boolean initialPrint = true;
String allOn = "";
String allOff = "";
boolean reading = false;
boolean outputStatus[4]; //Crear una matriz booleana para el Monto
máximo.
boolean outputStatus2[4]; //Crear una matriz booleana para el Monto
máximo.
unsigned long timeConnectedAt;
boolean writeToEeprom = false;
//Cliente EthernetClient;


////////////////////////////////////////////////////////////////
// Variables de Declaracion de Iconos para Display 16X4
////////////////////////////////////////////////////////////////

byte termometro[8] = //icon for termometer
{
  B00100,
  B01010,
  B01010,
  B01110,
  B01110,
  B11111,
  B11111,
  B01110
};

byte humedad[8] = //icon for water droplet
{
  B00100,
  B00100,
  B01010,
  B01010,
  B10001,
  B10001,
  B10001,
  B01110,
};
byte smiley[8] =
```

```
{
  0b00000,
  0b00000,
  0b01010,
  0b00000,
  0b00000,
  0b10001,
  0b01110,
  0b00000
};

//////////////////////////////////////////////////////////////////
// Variables de Declaracion de Pines a Usar con Pulsadores (Push-
Button)
//////////////////////////////////////////////////////////////////

int Pin23 = 23; // Pin 23
int Pin25 = 25; // Pin 25
int Pin27 = 27; // Pin 27
int Pin29 = 29; // Pin 29


int pulsador23on  = 32;  // Pulsador de Encendido del Pin23
int pulsador23off = 34;  // Pulsador de Apagado del Pin23

int pulsador25on  = 36;  // Pulsador de Encendido del Pin25
int pulsador25off = 38;  // Pulsador de Apagado del Pin25

int pulsador27on  = 40;  // Pulsador de Encendido del Pin27
int pulsador27off = 42;  // Pulsador de Apagado del Pin27

int pulsador29on  = 44;  // Pulsador de Encendido del Pin29
int pulsador29off = 46;  // Pulsador de Apagado del Pin29

//////////////////////////////////////////////////////////////////
// Variables de Declaracion de Encendido por Movimiento (PIR)
//////////////////////////////////////////////////////////////////

const int Pin31 = 31;  // Pin 31
const int inputPin4 = 4;  // Pin de datos del sensor PIR

//////////////////////////////////////////////////////////////////
// Variables de Declaracion de Encendido por Temperatura (DTH)
//////////////////////////////////////////////////////////////////

const int Pin33 = 33;  // Pin 33
int Temp = 32; // Parámetro de Temperatura para encendido del Pin 33

//////////////////////////////////////////////////////////////////
// Variables de Declaracion de Encendido por Distancia (HC-SR04)
//////////////////////////////////////////////////////////////////

int Pin35 = 35;   // Pin 35
int distancia;    // Declarando Variable Distancia

//////////////////////////////////////////////////////////////////
// Variables de Declaracion de Encendido por Tiempo
//////////////////////////////////////////////////////////////////

int Pin37 = 37;  // Pin 37
```

```
////////////////////////////////////////////////////////////////
// Inicio del Programa - Ejecutar una vez
////////////////////////////////////////////////////////////////

void setup() {

  Serial.begin(9600);  // Inicia Puerto Serial
  dht.begin();         // Inicia Sensor Dht

  initEepromValues();
  readEepromValues();

  //Grupo de Pines de Salida 1
  boolean currentState = false;
  for (int var = 0; var < outputQuantity; var++) {

    pinMode(outputAddress[var], OUTPUT);

    //Cambiar las salidas de inicio On o Off
    if (outputInverted == false) {
      //digitalWrite(outputAddress[var], HIGH);
      if (outputStatus[var] == 0) {
        currentState = true; //Comprobar estado de salida si esta
apagado, cambiar la salida
      } else {
        currentState = false;
      }
      digitalWrite(outputAddress[var], currentState);

    }
    else {

      //digitalWrite(outputAddress[var], LOW);
      if (outputStatus[var] == 0) {
        currentState = false; //Comprobar estado de salida si esta
apagado, cambiar la salida
      } else {
        currentState = true;
      }
      digitalWrite(outputAddress[var], currentState);
    }

  }

  //Grupo de Pines de Salida 2
  boolean currentState2 = false;
  for (int var = 0; var < outputQuantity2; var++) {

    pinMode(outputAddress2[var], OUTPUT);

    //Cambiar las salidas de inicio On o Off
    if (outputInverted2 == false) {
      //digitalWrite(outputAddress2[var], HIGH);
      if (outputStatus2[var] == 0) {
        currentState2 = true; //Comprobar estado de salida si esta
apagado, cambiar la salida
      } else {
        currentState2 = false;
      }
      digitalWrite(outputAddress2[var], currentState2);
```

```
    }
    else {

      //digitalWrite(outputAddress2[var], LOW);
      if (outputStatus2[var] == 0) {
        currentState2 = false; //Comprobar estado de salida si esta
apagado, cambiar la salida
      } else {
        currentState2 = true;
      }
      digitalWrite(outputAddress2[var], currentState2);
    }

  }

  //Configuración de la dirección IP. Comente el que usted no
necesita.
  //Ethernet.begin(mac); //Para Direccion DHCP. (Direccion sera
impresa al serial.)
  Ethernet.begin(mac, ip, gateway, subnet); //Configuración manual.
(Dirección es la configurada anteriormente.)


  server.begin();
  Serial.print("Servidor Conectado a ");
  Serial.println(Ethernet.localIP());

  ////////////////////////////////////////////////////////////////
  // Inicio del Programa - Reloj RTC
  ////////////////////////////////////////////////////////////////

  Wire.begin(); // Inicia el puerto I2C
  RTC.begin();  // Inicia la comunicación con el RTC
  //RTC.adjust(DateTime(__DATE__, __TIME__)); // Establece la fecha
y hora (Comentar una vez establecida la hora)

  ////////////////////////////////////////////////////////////////
  // Inicio del Programa - LCD para Display 16X4
  ////////////////////////////////////////////////////////////////

  {
    lcd.begin(16, 4);              // Columnas, filas. 16,4 utilizar
para un LCD 16x4.
    lcd.clear();                   // Comenzar con una pantalla en
blanco
    lcd.createChar(1, termometro); // Icono de temperatura
    lcd.createChar(2, humedad);    // Icono de Humedad
    lcd.createChar(3, smiley);     // Icono Carita Alegre
  }

  ////////////////////////////////////////////////////////////////
  // Inicio del Programa - LCD 84x48 - Nokia 5110
  ////////////////////////////////////////////////////////////////

  {
    myGLCD.InitLCD();         // Inicia LCD 84x48 - Nokia 5110
    myGLCD.setFont(TinyFont); // Selecciona Fuente de texto en LCD
84x48 - Nokia 5110
  }

  ////////////////////////////////////////////////////////////////
```

```
  // Inicio del Programa - IR
  ////////////////////////////////////////////////////////////////

  {
    pinMode(Pin23, OUTPUT); // Pin 23 como salida
    pinMode(Pin25, OUTPUT); // Pin 25 como salida
    pinMode(Pin27, OUTPUT); // Pin 27 como salida
    pinMode(Pin29, OUTPUT); // Pin 27 como salida

    encendido = 0;       // Indicamos estado inicial (apagado);
    irrecv.enableIRIn(); // Iniciamos la recepcion
  }

  ////////////////////////////////////////////////////////////////
  // Inicio del Programa - Encendido de los pines 23,25,27,29 por
Pulsadores (Push-Button)
  ////////////////////////////////////////////////////////////////

  {
    // Poner su código de configuración aquí, para ejecutar una vez:
    pinMode (Pin23, OUTPUT);          //Configurado como salida
    pinMode (pulsador23on, INPUT);    //Configurado de entrada
    pinMode (pulsador23off, INPUT);   //Configurado de entrada

    pinMode (Pin25, OUTPUT);          //Configurado como salida
    pinMode (pulsador25on, INPUT);    //Configurado de entrada
    pinMode (pulsador25off, INPUT);   //Configurado de entrada

    pinMode (Pin27, OUTPUT);          //Configurado como salida
    pinMode (pulsador27on, INPUT);    //Configurado de entrada
    pinMode (pulsador27off, INPUT);   //Configurado de entrada

    pinMode (Pin29, OUTPUT);          //Configurado como salida
    pinMode (pulsador29on, INPUT);    //Configurado de entrada
    pinMode (pulsador29off, INPUT);   //Configurado de entrada
  }

  ////////////////////////////////////////////////////////////////
  // Inicio del Programa - Encendido de pin 31 por Movimiento (PIR)
  ////////////////////////////////////////////////////////////////

  {
    pinMode(Pin31, OUTPUT);  // Pin 31
    pinMode(inputPin4, INPUT);  // Pin de datos del sensor PIR
  }

  ////////////////////////////////////////////////////////////////
  // Inicio del Programa - Encendido de pin 33 por Temperatura (DHT)
  ////////////////////////////////////////////////////////////////

  {
    pinMode(Pin33, OUTPUT); // Pin 33
  }

  ////////////////////////////////////////////////////////////////
  // Inicio del Programa - Encendido de pin 35 por Distancia (HC-
SR04)
  ////////////////////////////////////////////////////////////////

  {
    pinMode(Pin35, OUTPUT);  // Pin 35
```

```
  }

  ////////////////////////////////////////////////////////////////
  // Inicio del Programa - Encendido de pin 37 por Tiempo (RTC)
  ////////////////////////////////////////////////////////////////

  {
    pinMode(Pin37, OUTPUT); //Pin 37
  }

}// Finaliza void setud


////////////////////////////////////////////////////////////////
// Ejecutar ciclicamente una vez Iniciado
////////////////////////////////////////////////////////////////

void loop() {

  ////////////////////////////////////////////////////////////////
  // Datos a mostrar en LCD 16X4
  ////////////////////////////////////////////////////////////////

  // Lectura de temperatura o humedad tarda unos 250 milisegundos!
  // Lecturas de sensores también pueden ser 'antigua' (es un sensor
muy lento) hasta 2 segundos
  float h = dht.readHumidity();    // Lee la humedad
  float t = dht.readTemperature(); // Lee la temperatura en Celcius
  float f = dht.readTemperature(true);//Lee la temperatura en
Fahrenheit

  // Compruebe si cualquier lecturas es fallida (e intentarlo de
nuevo).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Error al leer desde el sensor DHT11");//Muestra
mensaje de error sino detecta el sensor DHT11 en el pin 2
    return;
  }

  {
    //tAhora = millis();
    //if ( tAhora - tAntes >= tEjecucion )
    // {
    // tAntes = tAhora;
    // }

    DateTime now = RTC.now();      // Inicia Reloj para mostrar en
LCD
    // lcd.clear();
    lcd.setCursor(0, 0);           // Linea en la que se escribe el
dato, columna 0, fila 0 (Primera Fila)
    lcd.print("IP : 192.168.0.4"); // Impresion de Dato de IP
mostrado en LCD
    lcd.setCursor(3, 1);           // Linea en la que se escribe el
dato, columna 3, fila 1 (Segunda Fila)
    if (now.day() < 10)            // Condicion para día < 10
    {
      lcd.print("0");
    }
    lcd.print(now.day(), DEC);     // Impresion Dato de Dia mostrado
en LCD
```

```
    lcd.print('/');                // Separador
    if (now.month() < 10)          // Condicion para mes < 10
    {
      lcd.print("0");
    }
    lcd.print(now.month(), DEC);   // Impresion Dato de Mes mostrado
en LCD
    lcd.print('/');                // Separador
    lcd.print(now.year(), DEC);    // Impresion Dato de Año mostrado
en LCD
    lcd.setCursor(0, 2);           // Linea en la que se escribe el
dato, columna 0, fila 2 (Tercera Fila)
    lcd.print("    ");
    if (now.hour() < 10)           // Condicion para hora < 10
    {
      lcd.print("0");
    }
    lcd.print(now.hour(), DEC);    // Impresion Dato de Hora
mostrado en LCD
    lcd.print(':');                // Separador
    if (now.minute() < 10)         // Condicion para minuto < 10
    {
      lcd.print("0");
    }
    lcd.print(now.minute(), DEC);  // Impresion Dato de Minutos
mostrado en LCD
    lcd.print(':');
    if (now.second() < 10)         // Condicion para segundo < 10
    {
      lcd.print("0");
    }
    lcd.print(now.second(), DEC);  // Impresion Dato de Segundos
mostrado en LCD
    lcd.setCursor(1, 3);//Inicio Fila 4
    lcd.write(1);
    lcd.setCursor(3, 3);
    lcd.print(dht.readTemperature(), 0); // Impresion Dato de
Temperatura mostrado en LCD
    lcd.setCursor(5, 3);
    lcd.print((char)223); // Impresion de Simbolo de Grado
    lcd.print("C");
    lcd.setCursor(8, 3);
    lcd.write(3);
    lcd.setCursor(10, 3);
    lcd.write(2);
    lcd.setCursor(12, 3);
    lcd.print(dht.readHumidity(), 0); // Impresion Dato de Humedad
mostrado en LCD
    lcd.print("%"); // Impresion de Simbolo de Porcentaje
    //delay(900);

  }

  ////////////////////////////////////////////////////////////////
  // Datos a mostrar en LCD 84x48 - Nokia 5110
  ////////////////////////////////////////////////////////////////

  {
    myGLCD.print("ULEAM", CENTER, 0);                    // Linea 1
    myGLCD.print("Carrera de ", CENTER, 6);              // Linea 2
    myGLCD.print("Ing. Electrica", CENTER, 12);          // Linea 3
```

```
    myGLCD.print("Tesis:", CENTER, 18);                  // Linea 4
    myGLCD.print("Sistema de Domotica", CENTER, 24);     // Linea 5
    myGLCD.print("con Arduino MEGA 2560", CENTER, 30);  // Linea 6
    myGLCD.print("Autor:Sandro Quijije", CENTER, 36);   // Linea 7
    myGLCD.print("Tutor:Ing. Carlo Cano", CENTER, 42);  // Linea 8
    myGLCD.update();
  }

  ////////////////////////////////////////////////////////////////
  // Encendido de los pines por IR
  ////////////////////////////////////////////////////////////////

  {
    // Si tenemos datos de lectura debido a que se pulsa una tecla
en el mando
    if (irrecv.decode(&results))
    {

      ////////////////////////////////
      //           Tecla 1          //
      ////////////////////////////////

      // Comprobamos si es la TECLA 1
      if (results.value == 0xFF30CF)
      {
        // Comprobamos si esta encendido el led, si lo esta lo
apagamos
        if (encendido == 1)
        {
          digitalWrite(Pin23, LOW); // Pin 23
          encendido = 0;
        }
        // Si no, lo encendemos
        else
        {
          digitalWrite(Pin23, HIGH); // Pin 23
          encendido = 1;
        }
      }

      ////////////////////////////////
      //           Tecla 2          //
      ////////////////////////////////

      // Comprobamos si es la TECLA 2
      if (results.value == 0xFF18E7)
      {
        // Comprobamos si esta encendido el led, si lo esta lo
apagamos
        if (encendido == 1)
        {
          digitalWrite(Pin25, LOW); // Pin 25
          encendido = 0;
        }
        // Si no, lo encendemos
        else
        {
          digitalWrite(Pin25, HIGH); // Pin 25
          encendido = 1;
        }
      }
```

```
      ////////////////////////////////
      //           Tecla 3            //
      ////////////////////////////////

      // Comprobamos si es la TECLA 3
      if (results.value == 0xFF7A85)
      {
        // Comprobamos si esta encendido el led, si lo esta lo
apagamos
        if (encendido == 1)
        {
          digitalWrite(Pin27, LOW); // Pin 27
          encendido = 0;
        }
        // Si no, lo encendemos
        else
        {
          digitalWrite(Pin27, HIGH); // Pin 27
          encendido = 1;
        }
      }

      ////////////////////////////////
      //           Tecla 4            //
      ////////////////////////////////

      // Comprobamos si es la TECLA 4
      if (results.value == 0xFF10EF)
      {
        // Comprobamos si esta encendido el led, si lo esta lo
apagamos
        if (encendido == 1)
        {
          digitalWrite(Pin29, LOW); // Pin 29
          encendido = 0;
        }
        // Si no, lo encendemos
        else
        {
          digitalWrite(Pin29, HIGH); // Pin 29
          encendido = 1;
        }
      }

      // delay(50); //retardo de 50 ms para evitar que el codigo se
lea dos veces en una pulsacion
      irrecv.resume(); // Recibimos el siguiente valor del sensor
    }
  } // Finaliza Encendido de los pines por IR

  ////////////////////////////////////////////////////////////////
  // Encendido de los pines por Pulsadores
  ////////////////////////////////////////////////////////////////

  { // Inicia Encendido de los pines por Pulsadores

    {
      if (digitalRead(pulsador23on) == HIGH) { //Si el pulsador23on
está en alto
        digitalWrite (Pin23, HIGH); //Encender el LED
```

```
      }
      else if (digitalRead(pulsador23off) == HIGH) { //Si el
pulsador23off está en alto
        digitalWrite (Pin23, LOW); // Apagar el LED
      }
    }
    {
      if (digitalRead(pulsador25on) == HIGH) { //Si el pulsador25on
está en alto
        digitalWrite (Pin25, HIGH); //Encender el LED
      }
      else if (digitalRead(pulsador25off) == HIGH) { //Si el
pulsador25off está en alto
        digitalWrite (Pin25, LOW); // Apagar el LED
      }
    }
    {
      if (digitalRead(pulsador27on) == HIGH) { //Si el pulsador27on
está en alto
        digitalWrite (Pin27, HIGH); //Encender el LED
      }
      else if (digitalRead(pulsador27off) == HIGH) { //Si el
pulsador27off está en alto
        digitalWrite (Pin27, LOW); // Apagar el LED
      }
    }
    {
      if (digitalRead(pulsador29on) == HIGH) { //Si el pulsador29on
está en alto
        digitalWrite (Pin29, HIGH); //Encender el LED
      }
      else if (digitalRead(pulsador29off) == HIGH) { //Si el
pulsador29off está en alto
        digitalWrite (Pin29, LOW); // Apagar el LED
      }
    }

  } // Finaliza Encendido de los pines por Pulsadores

  ////////////////////////////////////////////////////////////////
  // Encendido por Movimiento
  ////////////////////////////////////////////////////////////////

  {

    int value = digitalRead(inputPin4); // Lee Pin 4

    if (value == HIGH) // Si el valor es alto
    {
      digitalWrite(Pin31, LOW); // Pin 31 es Low
    }
    else
    {
      digitalWrite(Pin31, HIGH); // Pin 31 es High
    }

  }

  ////////////////////////////////////////////////////////////////
  // Encendido por Temperatura
  ////////////////////////////////////////////////////////////////
```

```
  {
    if (t >= Temp) {
          digitalWrite(Pin33, LOW); // Pin 33 es Low

    }
    if (t < Temp) {
          digitalWrite(Pin33, HIGH); // Pin 33 es High

    }
  }

  ////////////////////////////////////////////////////////////////
  // Encendido por Distancia
  ////////////////////////////////////////////////////////////////

  {
    distancia = ultra1.Ranging(CM);
    if (distancia < 10)          // Valor de distancia en cm.
    {
      digitalWrite(Pin35, LOW); // Pin 35 es Low
    }
    else
    {
      digitalWrite(Pin35, HIGH); // Pin 35 es High
    }

  }

  ////////////////////////////////////////////////////////////////
  // Encendido por tiempo
  ////////////////////////////////////////////////////////////////

  {

    DateTime now = RTC.now(); // Inicia Reloj
    if (now.hour() == 22 && now.minute() == 23) // Hora de Encendido
    {
      digitalWrite(Pin37, LOW); // Pin 37 es Low
    }
    if (now.hour() == 22 && now.minute() == 24) // Hora de Apagado
    {
      digitalWrite(Pin37, HIGH); // Pin 37 es High
    }
  }

  //Escuchar a los clientes entrantes y peticiones de proceso
  checkForClient();

}

////////////////////////////////////////////////////////////////
// Función checkForClient
////////////////////////////////////////////////////////////////

void checkForClient() { // Inicia void checkForClient

  EthernetClient client = server.available();

  if (client) {
```

```
    // Una petición http termina con una línea en blanco
    boolean currentLineIsBlank = true;
    boolean sentHeader = false;

    while (client.connected()) {
      if (client.available()) {

        // Si la cabecera no se estableció enviarlo

        // Leer la entrada del usuario
        char c = client.read();

        if (c == '*') {

          printHtmlHeader(client); //llamar para cabecera HTML y CSS
          printLoginTitle(client);
          printHtmlFooter(client);
          //sentHeader = true;
          break;
        }

        if (!sentHeader) {

          printHtmlHeader(client); //llamar para cabecera HTML y CSS
          printHtmlButtonTitle(client); //Imprimir el titulo del
botón
          //Esto es para el arduino para la construcción de la
página sobre la marcha.
          sentHeader = true;
        }

        // Leer entrada de usuario
        // char c = client.read();

        // Si esta leyendo, pero esta en blanco no se puede leer
        if (reading && c == ' ') {
          reading = false;
        }

        // Si hay una ? no había entrada de usuario
        if (c == '?') {
          reading = true; //encontrado la ?, comenzará la lectura de
la información
        }

        // Si hubiese entrada interrumpida cambiar la salida
relevante
        if (reading) {

          // Si la entrada de usuario es H grupo de salida es 1
          if (c == 'H') {
            outp = 1;
          }

          // Si la entrada de usuario es L grupo de salida es 0
          if (c == 'L') {
            outp = 0;
          }

          Serial.print(c);   // imprimir el valor de c para
comunicación a serial
```

```
          //Serial.print(outp);
          //Serial.print('\n');

          switch (c) { // Inicio del Interruptor

            case '0':
              //Agregar código aquí para encender 0
              triggerPin(outputAddress[0], client, outp);
              break;
            case '1':
              //Agregar código aquí para encender 1
              triggerPin(outputAddress[1], client, outp);
              break;
            case '2':
              //Agregar código aquí para encender 2
              triggerPin(outputAddress[2], client, outp);
              break;
            case '3':
              //Agregar código aquí para encender 3
              triggerPin(outputAddress[3], client, outp);
              break;

          } // Final del Interruptor

        }// Final del interruptor de cambiar la salida
correspondiente

        // Si la entrada del usuario esta en blanco
        if (c == '\n' && currentLineIsBlank) {
          printLastCommandOnce = true;
          printButtonMenuOnce = true;
          triggerPin(777, client, outp); //Llame para leer el menú
de entrada y de impresión. 777 se utiliza para no actualizar
cualquier salida
          break;
        }

      }
    }

    printHtmlFooter(client); //Imprime el pie de página html

  }

  else
  { //Si no hay ningún cliente

    //Y el tiempo de la última página que se sirve es más que un
minuto.
    if (millis() > (timeConnectedAt + 60000)) {

      if (writeToEeprom == true) {
        writeEepromValues();  //Escribir en la EEPROM los estados de
salida de corriente
        Serial.println("No hay clientes por más de un minuto -
Escribir estados de EEPROM.");
        writeToEeprom = false;
      }

    }
  }
```

```
} // Finaliza void checkForClient


//////////////////////////////////////////////////////////////////
// Función triggerPin
//////////////////////////////////////////////////////////////////

void triggerPin(int pin, EthernetClient client, int outp) { //
Inicia Función triggerPin
  // Encender o apagar salidas, lee las salidas e imprime los
botones

  // Configuración de salidas
  if (pin != 777) {

    if (outp == 1) {
      if (outputInverted == false) {
        digitalWrite(pin, HIGH);
      }
      else {
        digitalWrite(pin, LOW);
      }
    }
    if (outp == 0) {
      if (outputInverted == false) {
        digitalWrite(pin, LOW);
      }
      else {
        digitalWrite(pin, HIGH);
      }
    }
  }
  //Actualice la lectura de las salidas
  readOutputStatuses();

  // Configuración de salidas
  if (pin != 777) {

    if (outp == 1) {
      if (outputInverted2 == false) {
        digitalWrite(pin, HIGH);
      }
      else {
        digitalWrite(pin, LOW);
      }
    }
    if (outp == 0) {
      if (outputInverted2 == false) {
        digitalWrite(pin, LOW);
      }
      else {
        digitalWrite(pin, HIGH);
      }
    }
  }
  //Actualice la lectura de las salidas
  readOutputStatuses();
```

```
  //Imprime los botones
  if (printButtonMenuOnce == true) {
    printHtmlButtons(client);
    printButtonMenuOnce = false;
  }

} // Finaliza Función triggerPin

////////////////////////////////////////////////////////////////
// Función printHtmlButtons
////////////////////////////////////////////////////////////////

//Imprimir los botones del HTML para encender / apagar los canales

void printHtmlButtons(EthernetClient client) { // Inicia Función
printHtmlButtons

  ////////////////////////////////////////////////////////////////
  // Creacion de Tabla 1 (Fecha y hora)
  ////////////////////////////////////////////////////////////////

  client.println("");
  //client.println("<p>");
  client.println("<FORM>");
  client.println("<table border=\"0\" align=\"center\">");

  DateTime now = RTC.now(); // Obtiene la fecha y hora del RTC

  client.print("<tr>\n");//Abrimos la fila
  client.print("<td><h4>");//Abrimos Ingreso de datos
  client.println(' ');
  client.println(' ');
  if (now.hour() < 10)
  {
    client.print("0");
  }
  client.print(now.hour(), DEC);// Imprime Hora
  client.print(':');
  if (now.minute() < 10)
  {
    client.print("0");
  }
  client.print(now.minute(), DEC);// Imprime Minutos
  client.print(':');
  if (now.second() < 10)
  {
    client.print("0");
  }
  client.print(now.second(), DEC); // Imprime Segundos
  client.println(' ');
  client.print("</h3></td>\n");//Cerramos ingreso de datos
  client.print("</tr>");//Cerramos la fila

  client.print("<tr>\n");//Abrimos la fila
  client.print("<td><h4>");//Abrimos Ingreso de datos
  if (now.day() < 10)
  {
    client.print("0");
  }
  client.print(now.day(), DEC);// Imprime Dia
```

```
  client.print('/');
  if (now.month() < 10)
  {
    client.print("0");
  }
  client.print(now.month(), DEC);// Imprime Mes
  client.print('/');
  client.print(now.year(), DEC);// Imprime Año
  client.print("</h3></td>\n");//Cerramos ingreso de datos
  client.print("</tr>");//Cerramos la fila

  // Cerrando la tabla y la forma 1
  client.println("</table>");
  client.println("</FORM>");
  //client.println("</p>");

  ////////////////////////////////////////////////////////////////
  // Creacion de Tabla 2 (Humedad y Temperatura)
  ////////////////////////////////////////////////////////////////

  client.println("");
  //client.println("<p>");
  client.println("<FORM>");
  client.println("<table border=\"0\" align=\"center\">"); //
Centrar

  // Impresión de la Humedad Relativa
  client.print("<tr>\n");
  client.print("<td><h4>");
  client.print("Humedad Relativa:");
  client.print("</h4></td>\n");
  client.print("<td>");
  client.print("<h4>");
  client.print(dht.readHumidity()); // Impresión de la Humedad
  client.print(" %</h4></td>\n");
  client.print("</tr>");

  // Impresión de la Temperatura C'
  client.print("<tr>\n");
  client.print("<td><h4>");
  client.print("Temperatura °C: ");
  client.print("</h4></td>\n");
  client.print("<td>");
  client.print("<h4>");
  client.print(dht.readTemperature()); // Impresión de la
Temperatura
  client.print(" °C</h4></td>\n");
  client.print("</tr>");

  // Impresión de la Temperatura F'
  client.print("<tr>\n");
  client.print("<td><h4>");
  client.print("Temperatura °F:");
  client.print("</h4></td>\n");
  client.print("<td>");
  client.print("<h4>");
  client.print(dht.readTemperature(true)); // Impresión de la
Temperatura
  client.print(" °F</h4></td>\n");
  client.print("</tr>");
```

```
  //Cerrando la tabla y la forma 2
  client.println("</table>");
  client.println("</FORM>");
  //client.println("</p>");

  ////////////////////////////////////////////////////////////////////
  // Subtitulo
  ////////////////////////////////////////////////////////////////////

  client.println("\n<h3 align=\"center\"<h2><em>Estado de los
sensores</em></h2>");

  ////////////////////////////////////////////////////////////////////
  // Creacion de Tabla 4
  ////////////////////////////////////////////////////////////////////

  client.println("");
  //client.println("<p>");
  client.println("<FORM>");
  client.println("<table border=\"0\" align=\"center\">");

  // Inicie la impresión botón por botón
  for (int var = 0; var < outputQuantity2; var++)  {


    // Imprimir comienzo de la fila
    client.print("<tr>\n");

    // Imprime el botón Texto
    client.print("<td><h4>");
    client.print(buttonText2[var]);
    client.print("</h4></td>\n");



    //Imprimir primera parte de los círculos o los LED

    //Invertir la pantalla LED si la salida está invertida.

    if (outputStatus2[var] == true )
{                                                          // Si la
salida es ON
      if (outputInverted2 == false)
{                                                           // Y si
la salida no esta invertida
        client.print(" <td><div class='green-circle'><div
class='glare'></div></div></td>\n"); // Imprimir para html ON LED
      }
      else
{
                  // Despues la salida es Invertida
        client.print(" <td><div class='black-circle'><div
class='glare'></div></div></td>\n"); // Imprimir para html OFF LED
      }
    }
    else
                            // Si la salida es Off
    {
      if (outputInverted2 == false)
{                                                           // Y si
la salida no esta invertida
```

```
        client.print(" <td><div class='black-circle'><div
class='glare'></div></div></td>\n"); // Imprimir para html OFF LED
      }
      else
{
                  // Despues la salida es Invertida
        client.print(" <td><div class='green-circle'><div
class='glare'></div></div></td>\n"); // Imprimir para html ON LED
      }
    }

    // Imprimir final de la fila
    client.print("</tr>\n");
  }


  //Cerrando la tabla y la forma 3
  client.println("</table>");
  client.println("</FORM>");
  //client.println("</p>");

  ////////////////////////////////////////////////////////////////////
  // Subtitulo
  ////////////////////////////////////////////////////////////////////

  client.println("\n<h3 align=\"center\"<h2><em>Presione On para
Encender Off para Apagar</em></h2>");

  ////////////////////////////////////////////////////////////////////
  // Creacion de Tabla 4
  ////////////////////////////////////////////////////////////////////

  client.println("");
  //client.println("<p>");
  client.println("<FORM>");
  client.println("<table border=\"0\" align=\"center\">");

  // Inicie la impresión botón por botón
  for (int var = 0; var < outputQuantity; var++)  {

    // Conjunto de comandos para todo el encendido / apagado
    allOn += "H";
    allOn += outputAddress[var];
    allOff += "L";
    allOff += outputAddress[var];


    // Imprimir comienzo de la fila
    client.print("<tr>\n");

    // Imprime el botón Texto
    client.print("<td><h4>");
    client.print(buttonText[var]);
    client.print("</h4></td>\n");

    // Imprime los Botones ON
    client.print("<td>");
    //client.print(buttonText[var]);
    client.print("<INPUT TYPE=\"button\" VALUE=\"ON ");
    //client.print(buttonText[var]);
    client.print("\" onClick=\"parent.location='/?H");
```

```
    client.print(var);
    client.print("'\"></td>\n");

    // Imprime los Botones OFF
    client.print(" <td><INPUT TYPE=\"button\" VALUE=\"OFF");
    //client.print(var);
    client.print("\" onClick=\"parent.location='/?L");
    client.print(var);
    client.print("'\"></td>\n");


    //Imprimir primera parte de los círculos o los LED

    //Invertir la pantalla LED si la salida está invertida.

    if (outputStatus[var] == true )
{                                                                  // Si la
salida es ON
      if (outputInverted == false)
{                                                                   // Y si
la salida no esta invertida
        client.print(" <td><div class='green-circle'><div
class='glare'></div></div></td>\n"); // Imprimir para html ON LED
      }
      else
{
                  // Despues la salida es Invertida
        client.print(" <td><div class='black-circle'><div
class='glare'></div></div></td>\n"); // Imprimir para html OFF LED
      }
    }
    else
                            // Si la salida es Off
    {
      if (outputInverted == false)
{                                                                   // Y si
la salida no esta invertida
        client.print(" <td><div class='black-circle'><div
class='glare'></div></div></td>\n"); // Imprimir para html OFF LED
      }
      else
{
                  // Despues la salida es Invertida
        client.print(" <td><div class='green-circle'><div
class='glare'></div></div></td>\n"); // Imprimir para html ON LED
      }
    }

    // Imprimir final de la fila
    client.print("</tr>\n");
  }

  // Mostrar u ocultar la impresion del Botón Encender Todo
  if (switchOnAllPinsButton == false ) {

    // Imprimir Botón Encender Todo
    client.print("<tr>\n");
    client.print("</td>\n");
    client.print("<td>\n<td><INPUT TYPE=\"button\" VALUE=\"Encender
Todo");
    client.print("\" onClick=\"parent.location='/?H");
```

```
    client.print(allOn);
    client.print("'\"></td>\n");

    //Imprimir Botón Apagar Todo
    client.print("<td><INPUT TYPE=\"button\" VALUE=\"Apagar Todo");
    client.print("\" onClick=\"parent.location='/?L");
    client.print(allOff);
    client.print("'\"></td>\n<td></td>\n</tr>\n");
  }

  //Cerrando la tabla y la forma 4
  client.println("</table>");
  client.println("</FORM>");
  //client.println("</p>");



} // Final Función printHtmlButtons

////////////////////////////////////////////////////////////////////
// Función readOutputStatuses
////////////////////////////////////////////////////////////////////

// Leyendo Estado de las Salidas
void readOutputStatuses() {
  for (int var = 0; var < outputQuantity; var++)  {
    outputStatus[var] = digitalRead(outputAddress[var]);
    //Serial.print(outputStatus[var]);
  }
  // Leyendo Estado de las Salidas 2
  for (int var = 0; var < outputQuantity2; var++)  {
    outputStatus2[var] = digitalRead(outputAddress2[var]);
    //Serial.print(outputStatus2[var]);
  }
}




////////////////////////////////////////////////////////////////////
// Función readEepromValues
////////////////////////////////////////////////////////////////////

// Leer valores EEPROM y guardar en Estado de salida
void readEepromValues() {
  for (int adr = 0; adr < outputQuantity; adr++)  {
    outputStatus[adr] = EEPROM.read(adr);
  }
  for (int adr = 0; adr < outputQuantity2; adr++)  {
    outputStatus2[adr] = EEPROM.read(adr);
  }
}

////////////////////////////////////////////////////////////////////
// Función writeEepromValues
////////////////////////////////////////////////////////////////////

// Escribir valores EEprom
void writeEepromValues() {
  for (int adr = 0; adr < outputQuantity; adr++)  {
    EEPROM.write(adr, outputStatus[adr]);
```

```
  }
  for (int adr = 0; adr < outputQuantity2; adr++)  {
    EEPROM.write(adr, outputStatus2[adr]);
  }

}

////////////////////////////////////////////////////////////////
// Función initEepromValues
////////////////////////////////////////////////////////////////

// Inicializar Valores EEprom
// Si los valores de la EEPROM no son el formato correcto no es
igual a 0 o 1 (por lo tanto mayor que 1) inicializar poniendo 0
void initEepromValues() {
  for (int adr = 0; adr < outputQuantity; adr++) {
    if (EEPROM.read(adr) > 1) {
      EEPROM.write(adr, 0);
    }
  }
  for (int adr = 0; adr < outputQuantity2; adr++) {
    if (EEPROM.read(adr) > 1) {
      EEPROM.write(adr, 0);
    }
  }
}


////////////////////////////////////////////////////////////////
// Función htmlHeader
////////////////////////////////////////////////////////////////

// Imprimir encabezamiento html
void printHtmlHeader(EthernetClient client) {

  Serial.print("Serving html Headers at ms -");
  timeConnectedAt = millis(); //Anote la hora en que se sirvió la
última página.
  Serial.print(timeConnectedAt); //Tiempo de impresión para
propósitos de depuración
  writeToEeprom = true; //Página Cargada para la acción de escritura
en la eeprom

  // Enviar un encabezado de respuesta HTTP estándar
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println("Connnection: close");
  client.println();
  client.println("<!DOCTYPE HTML>");
  client.println("<head>");

  // Agregar título de la página
  client.println("<title>Sistema de Domótica con Arduino Mega 2560 y
Arduino Ethernet Shield</title>");
  client.println("<meta name=\"description\" content=\"Sistema de
Domótica con Arduino Mega 2560 y Arduino Ethernet Shield\"/>");

  // Añadir una etiqueta meta de actualización, por lo que el
navegador tira de nuevo cada X segundos:
  client.print("<meta http-equiv=\"refresh\" content=\"");
  client.print(refreshPage);
```

```
  client.println("; url=/\">");

  // Añadir otra configuración del navegador
  client.println("<meta name=\"apple-mobile-web-app-capable\"
content=\"yes\">");
  client.println("<meta name=\"apple-mobile-web-app-status-bar-
style\" content=\"default\">");
  client.println("<meta name=\"viewport\" content=\"width=device-
width, user-scalable=no\">");

  // La inserción de estilos de datos, por lo general se encuentran
en los archivos CSS.
  client.println("<style type=\"text/css\">");
  client.println("");

  // Esto establecerá cómo se verá la página de forma gráfica
  client.println("html { height:100%; }");

  client.println("  body {");
  client.println("    height: 100%;");
  client.println("    margin: 0;");
  client.println("    font:Times New Roman,sans-serif;"); // Fuente
de texto
  client.println("    -webkit-text-size-adjust: none;");
  client.println("   }");
  client.println("");
  client.println("body {");
  client.println("    -webkit-background-size: 100% 21px;");
  client.println("    background-color: #e3e3e3;");
  client.println("    -webkit-gradient(linear, left top, right
top,");
  client.println("    color-stop(.75, transparent),");
  client.println("    color-stop(.75, rgba(227,227,227,.1)) );");
  client.println("    -webkit-background-size: 21px;");
  client.println("   }");
  client.println("");
  client.println(".view {");
  client.println("    min-height: 100%;");
  client.println("    overflow: auto;");
  client.println("   }");
  client.println("");
  client.println(".header-wrapper {");
  client.println("    height: 44px;");
  client.println("    font-weight: bold;"); // Intensidad de negrita
normal o bold
  client.println("    text-shadow: rgba(0,0,0,0.7) 0 -1px 0;");
  client.println("    border-top: solid 1px
rgba(255,255,255,0.6);");
  client.println("    border-bottom: solid 1px rgba(0,0,0,0.6);");
  client.println("    color: #fff;");
  client.println("    background-color: #295f2d;");//Barra de titulo
  client.println("    -webkit-gradient(linear, left top, left
bottom,");
  client.println("    from(rgba(255,255,255,.4)),");
  client.println("    to(rgba(255,255,255,.05)) ),");
  client.println("    -webkit-gradient(linear, left top, left
bottom,");
  client.println("    from(transparent),");
  client.println("    to(rgba(0,0,64,.1)) );");
  client.println("    background-repeat: no-repeat;");
  client.println("    background-position: top left, bottom left;");
```

```
  client.println("    -webkit-background-size: 100% 21px, 100%
22px;");
  client.println("    -webkit-box-sizing: border-box;");
  client.println("   }");
  client.println("");
  client.println(".header-wrapper h1 {"); // Tiulo Principal 1
  client.println("    text-align: center;");
  client.println("    font-size: 25px;"); // Tamaño d fuente
  client.println("    line-height: 44px;");
  client.println("    font-weight: bold;"); // Intensidad de negrita
normal o bold
  client.println("    text-shadow: #aaa 1px 1px 3px;");
  client.println("    margin: 0;");
  client.println("   }");
  client.println("");
  client.println(".header-wrapper2 h1 {"); // Tiulo Principal 2
  client.println("    text-align: center;");
  client.println("    font-size: 20px;");
  client.println("    line-height: 44px;");
  client.println("    font-weight: bold;");
  client.println("    text-shadow: #aaa 1px 1px 3px;");
  client.println("    margin: 0;");
  client.println("   }");
  client.println("");
  client.println(".group-wrapper {");
  client.println("    margin: 9px;");
  client.println("    }");
  client.println("");
  client.println(".group-wrapper h2 {");
  client.println("    color: #FF0000;");
  client.println("    font-size: 17px;");
  client.println("    line-height: 0.8;");
  client.println("    font-weight: bold;");
  client.println("    text-shadow: #fff 0 1px 0;");
  client.println("    margin: 20px 10px 12px;");
  client.println("   }");
  client.println("");
  client.println(".group-wrapper h3 {"); // Texto Instruccion, valor
de temperatura y Autor
  client.println("    color: #000000;");
  client.println("    font-size: 14px;");
  client.println("    line-height: 1;");
  client.println("    font-weight: bold;");
  client.println("    text-shadow: #aaa 1px 1px 3px;");
  client.println("    margin: 20px 10px 12px;");
  client.println("   }");
  client.println("");
  client.println(".group-wrapper h4 {");  // Texto para la
descripción
  client.println("    color: #000000 ;");
  client.println("    font-size: 14px;");
  client.println("    line-height: 1;");
  client.println("    font-weight: bold;");
  client.println("    text-shadow: #aaa 1px 1px 3px;");
  client.println("    margin: 5px 5px 5px;");
  client.println("   }");
  client.println("");
  client.println(".group-wrapper table {");
  client.println("    background-color: #fff;");
  client.println("    -webkit-border-radius: 10px;");
```

```
  client.println("    -moz-border-radius: 10px;");
  client.println("    -khtml-border-radius: 10px;");
  client.println("    border-radius: 10px;");
  client.println("    font-size: 17px;");
  client.println("    line-height: 20px;");
  client.println("    margin: 9px 0 20px;");
  client.println("    border: solid 1px #a9abae;");
  client.println("    padding: 11px 3px 12px 3px;");
  client.println("    margin-left:auto;");
  client.println("    margin-right:auto;");

  client.println("    -moz-transform :scale(1);"); // Código para
Mozilla Firefox
  client.println("    -moz-transform-origin: 0 0;");

  client.println("   }");
  client.println("");


  // Se verá verde el LED si esta (ON)
  client.println(".green-circle {");
  client.println("    display: block;");
  client.println("    height: 23px;");
  client.println("    width: 23px;");
  client.println("    background-color: #0f0;");
  //client.println("    background-color: rgba(60, 132, 198,
0.8);");
  client.println("    -moz-border-radius: 11px;");
  client.println("    -webkit-border-radius: 11px;");
  client.println("    -khtml-border-radius: 11px;");
  client.println("    border-radius: 11px;");
  client.println("    margin-left: 1px;");

  client.println("    background-image: -webkit-gradient(linear, 0%
0%, 0% 90%, from(rgba(46, 184, 0, 0.8)), to(rgba(148, 255, 112,
.9)));@");
  client.println("    border: 2px solid #ccc;");
  client.println("    -webkit-box-shadow: rgba(11, 140, 27, 0.5) 0px
10px 16px;");
  client.println("    -moz-box-shadow: rgba(11, 140, 27, 0.5) 0px
10px 16px; /* FF 3.5+ */");
  client.println("    box-shadow: rgba(11, 140, 27, 0.5) 0px 10px
16px; /* FF 3.5+ */");

  client.println("    }");
  client.println("");

  // Se verá negro el LED si esta (OFF)
  client.println(".black-circle {");
  client.println("    display: block;");
  client.println("    height: 23px;");
  client.println("    width: 23px;");
  client.println("    background-color: #040;");
  client.println("    -moz-border-radius: 11px;");
  client.println("    -webkit-border-radius: 11px;");
  client.println("    -khtml-border-radius: 11px;");
  client.println("    border-radius: 11px;");
  client.println("    margin-left: 1px;");
  client.println("    -webkit-box-shadow: rgba(11, 140, 27, 0.5) 0px
10px 16px;");
```

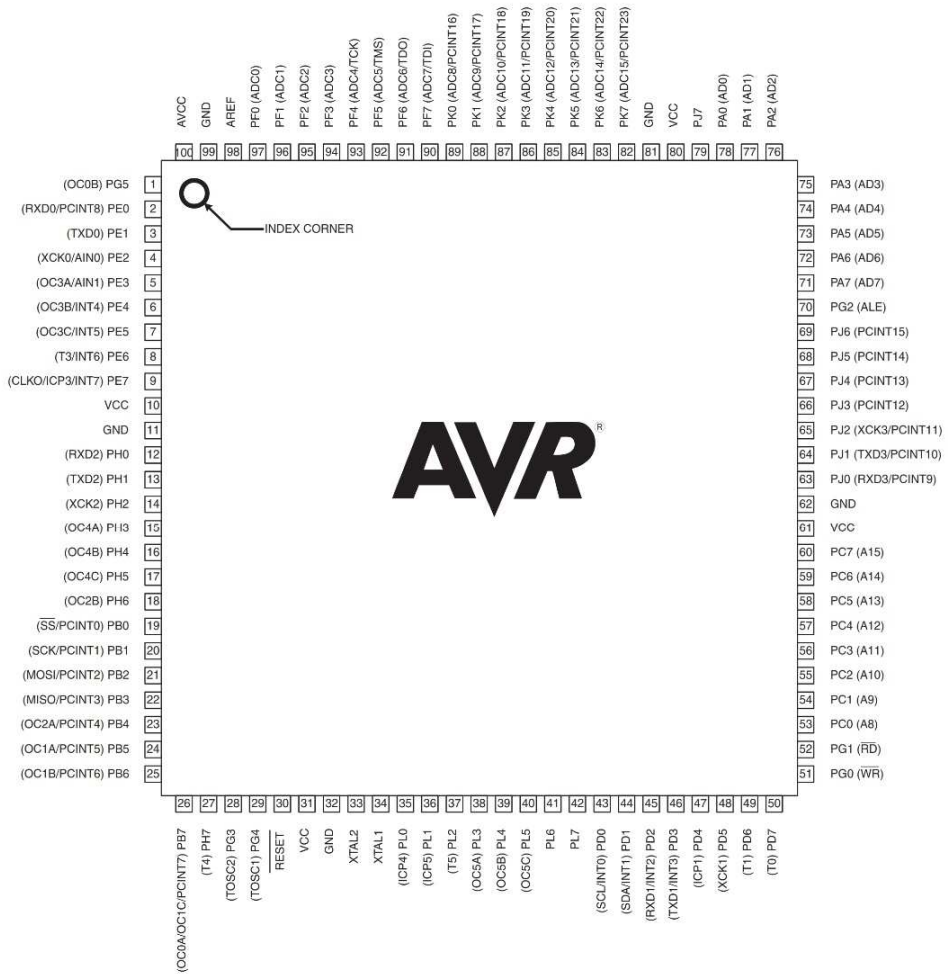
```
  client.println("    -moz-box-shadow: rgba(11, 140, 27, 0.5) 0px
10px 16px; /* FF 3.5+ */");
  client.println("    box-shadow: rgba(11, 140, 27, 0.5) 0px 10px
16px; /* FF 3.5+ */");
  client.println("    }");
  client.println("");

  // Esto añadirá el resplandor a los Leds
  client.println("   .glare {");
  client.println("      position: relative;");
  client.println("      top: 1;");
  client.println("      left: 5px;");
  client.println("      -webkit-border-radius: 10px;");
  client.println("      -moz-border-radius: 10px;");
  client.println("      -khtml-border-radius: 10px;");
  client.println("      border-radius: 10px;");
  client.println("      height: 1px;");
  client.println("      width: 13px;");
  client.println("      padding: 5px 0;");
  client.println("      background-color: rgba(200, 200, 200,
0.25);");
  client.println("      background-image: -webkit-gradient(linear,
0% 0%, 0% 95%, from(rgba(255, 255, 255, 0.7)), to(rgba(255, 255,
255, 0)));");
  client.println("    }");
  client.println("");

  // Y, finalmente, este es el final de los datos de estilo y
cabecera
  client.println("</style>");
  client.println("</head>");

  // Título
  client.println("<body>");
  client.println("<div class=\"view\">");
  client.println("<div class=\"header-wrapper\">");
  client.println("<h1><em>Sistema de Domótica con
Arduino</em></h1>");
  client.println("</div>");

  // Título
  client.println("<body>");
  client.println("<div class=\"view\">");
  client.println("<div class=\"header-wrapper2\">");
  client.println("<h1><em>Bienvenido/a</em></h1>");
  client.println("</div>");


} // Final de Cabecera html

////////////////////////////////////////////////////////////////
// Función htmlFooter
////////////////////////////////////////////////////////////////

// Imprimir Pie de página html
void printHtmlFooter(EthernetClient client) {

  // Establecer las variables antes de salir
  printLastCommandOnce = false;
  printButtonMenuOnce = false;
  allOn = "";
```

```
  allOff = "";

  // Impresión última parte del html
  client.println("\n<h3 align=\"center\"><em>Realizado por: Sandro
Quijije Marin</em>");
  client.println("\n<h3 align=\"center\"><em>Proyecto de Tesis de
Ingenieria Eléctrica</em>");
  client.println("\n<h3 align=\"center\"><em>Manta - 2015</em>");
  client.println("</h3></div>\n</div>\n</body>\n</html>");

  delay(1); //Dar tiempo al navegador web para recibir los datos

  client.stop(); //Cerrar la conexión:

  Serial.println(" - Hecho, Cerrando Coneccion.");

  delay (2); //Demora para que se le de tiempo para buffer de
cliente para borrar y no se repite varias páginas.

} // Final de pie de página html

////////////////////////////////////////////////////////////////
// Función printHtmlButtonTitle
////////////////////////////////////////////////////////////////

//Imprime botón con título  html
void printHtmlButtonTitle(EthernetClient client) {

  client.println("<div  class=\"group-wrapper\">");
  client.println("\n<h3 align=\"center\"<h2><em>Servidor Conectado a
192.168.0.4</em></h2>");
  client.println();
}

////////////////////////////////////////////////////////////////
//Función printLoginTitle
////////////////////////////////////////////////////////////////

// Imprime botón con título  html
void printLoginTitle(EthernetClient client) {
  // client.println("<div  class=\"group-wrapper\">");
  client.println("    <h2>Por favor ingrese los datos de usuario
para iniciar sesión.</h2>");
  client.println();
}
```

## *Anexo 9 – Pines del Shield Ethernet*

### 1. Pin Configurations

**Figure 1-1.** TQFP-pinout ATmega640/1280/2560






*Fuente: Datasheet Shield Ethernet - Atmel*

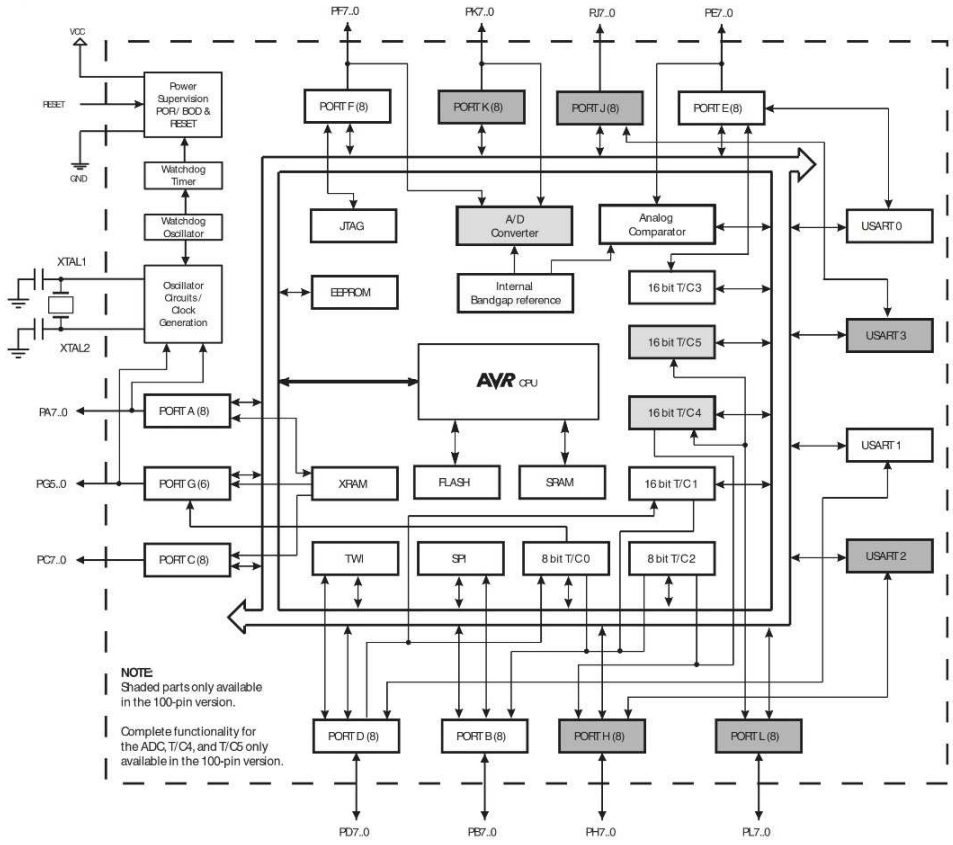
## *Anexo 10 - Diagrama de Bloques del Shield Ethernet*

## 2. Overview

The ATmega640/1280/1281/2560/2561 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega640/1280/1281/2560/2561 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

### 2.1 Block Diagram

**Figure 2-1.** Block Diagram




The Atmel® AVR® core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.



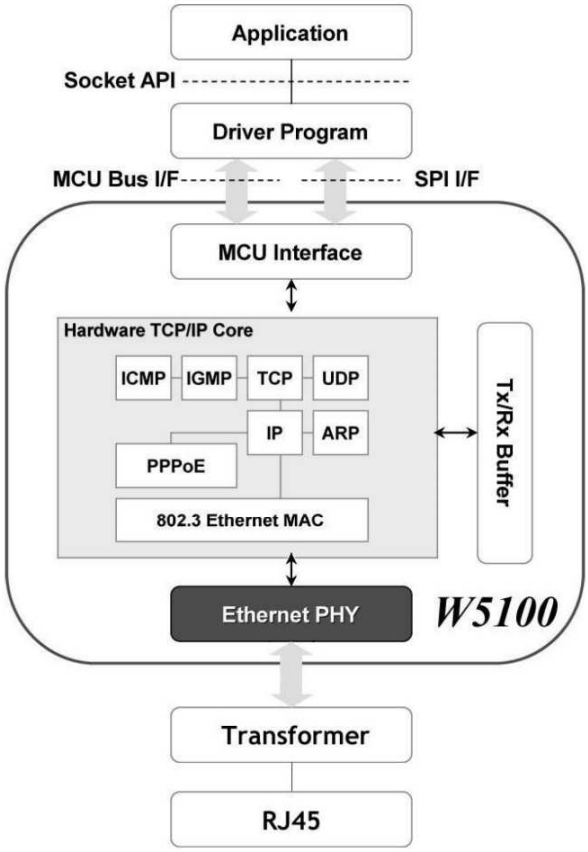
*Fuente: Datasheet Shield Ethernet – Atmel*

*Anexo 11 - Digrama de bloques del W5100*

WIZnet

Block Diagram









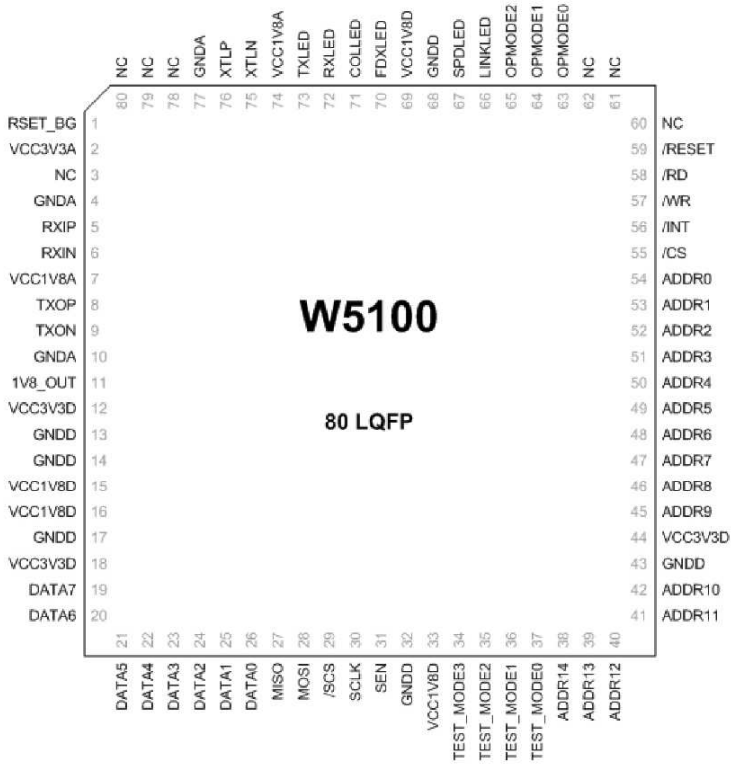

*Fuente: Datasheet WIZnet W5100 – WIZnet Co.*

## *Anexo 12 - Pines del W5100*



WIZnet

# 1. Pin Assignment




**Pinout W5100**





*Fuente: Datasheet WIZnet W5100 – WIZnet Co.*

*Anexo 13 - Hoja de datos del DS1307*

DALLAS SEMICONDUCTOR

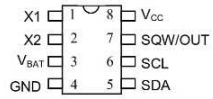
# DS1307
# 64 x 8 Serial Real-Time Clock

www.maxim-ic.com

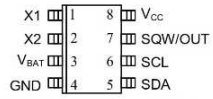
## FEATURES

- Real-time clock (RTC) counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap-year compensation valid up to 2100
- 56-byte, battery-backed, nonvolatile (NV) RAM for data storage
- Two-wire serial interface
- Programmable squarewave output signal
- Automatic power-fail detect and switch circuitry
- Consumes less than 500nA in battery backup mode with oscillator running
- Optional industrial temperature range: -40°C to +85°C
- Available in 8-pin DIP or SOIC
- Underwriters Laboratory (UL) recognized

## PIN ASSIGNMENT




DS1307 8-Pin DIP (300-mil)




DS1307 8-Pin SOIC (150-mil)

## ORDERING INFORMATION

| | |
|---|---|
| DS1307 | 8-Pin DIP (300-mil) |
| DS1307Z | 8-Pin SOIC (150-mil) |
| DS1307N | 8-Pin DIP (Industrial) |
| DS1307ZN | 8-Pin SOIC (Industrial) |

## PIN DESCRIPTION

| | |
|---|---|
| $V_{CC}$ | - Primary Power Supply |
| X1, X2 | - 32.768kHz Crystal Connection |
| $V_{BAT}$ | - +3V Battery Input |
| GND | - Ground |
| SDA | - Serial Data |
| SCL | - Serial Clock |
| SQW/OUT | - Square Wave/Output Driver |

## DESCRIPTION

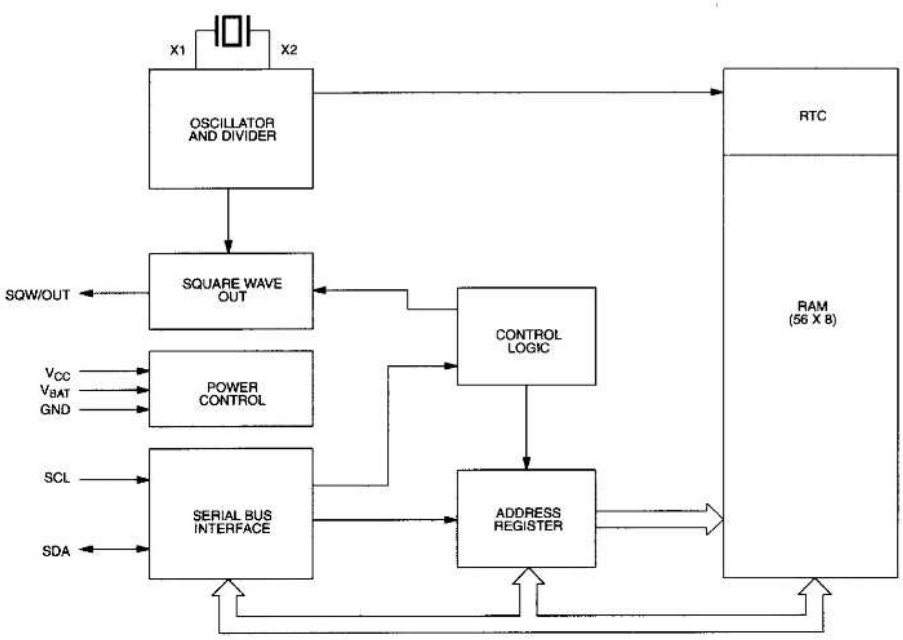
The DS1307 Serial Real-Time Clock is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially via a 2-wire, bi-directional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power sense circuit that detects power failures and automatically switches to the battery supply.



*Fuente: Datasheet DS1307 – Maxim Integrated*

***Anexo 14 - Diagrama de bloque del DS1307***

**DS1307 BLOCK DIAGRAM** Figure 1



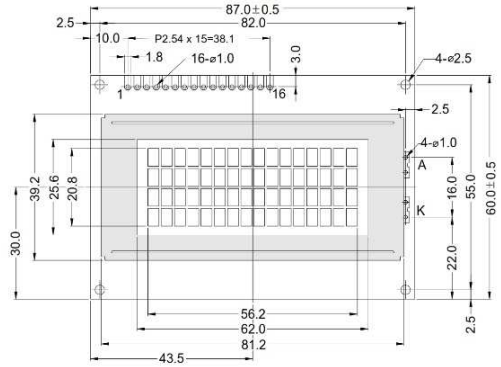

*Fuente: Datasheet DS1307 - Maxim Integrated*

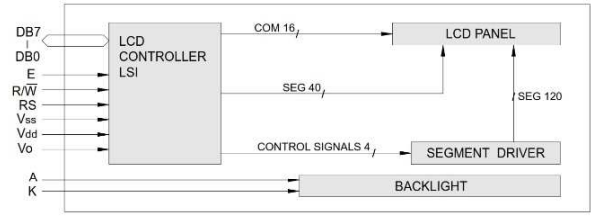
## *Anexo 15 - Hoja de datos LCD 1604A*

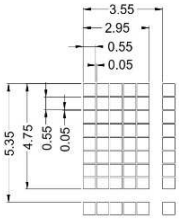
POWERTIP

**PC 1604-A**

**OUTLINE DIMENSION & BLOCK DIAGRAM**




The tolerance unless classified ±0.3mm

| MECHANICAL SPECIFICATION | | | |
|---|---|---|---|
| Overall Size | 87.0 x 60.0 | Module | H2 / H1 |
| View Area | 62.0 x 25.6 | W /O B/L | 5.1 / 9.7 |
| Dot Size | 0.55 x 0.55 | EL B/L | 5.1 / 9.7 |
| Dot Pitch | 0.60 x 0.60 | LED B/L | 8.9 / 13.5 |

| PIN ASSIGNMENT | | |
|---|---|---|
| Pin no. | Symbol | Function |
| 1 | Vss | Power supply(GND) |
| 2 | Vdd | Power supply(+) |
| 3 | Vo | Contrast Adjust |
| 4 | RS | Register select signal |
| 5 | R/W | Data read / write |
| 6 | E | Enable signal |
| 7 | DB0 | Data bus line |
| 8 | DB1 | Data bus line |
| 9 | DB2 | Data bus line |
| 10 | DB3 | Data bus line |
| 11 | DB4 | Data bus line |
| 12 | DB5 | Data bus line |
| 13 | DB6 | Data bus line |
| 14 | DB7 | Data bus line |

| ABSOLUTE MAXIMUM RATING | | | | | |
|---|---|---|---|---|---|
| Item | Symbol | Condition | Min. | Max. | Units |
| Supply for logic voltage | Vdd-Vss | 25°C | -0.3 | 7 | V |
| LCD driving supply voltage | Vdd-Vee | 25°C | -0.3 | 13 | V |
| Input voltage | Vin | 25°C | -0.3 | Vdd+0.3 | V |

| ELECTRICAL CHARACTERISTICS | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Item | Symbol | Condition | Min. | | Typical | | Max. | | Units |
| Power supply voltage | Vdd-Vss | 25°C | 2.7 | | – | | 5.5 | | V |
| LCD operation voltage | Vop | Top | N | W | N | W | N | W | V |
| | | -20°C | – | 7.1 | – | 7.5 | – | 7.9 | V |
| | | 0°C | 4.3 | – | 4.6 | – | 4.9 | – | V |
| | | 25°C | 3.9 | 6.1 | 4.2 | 6.4 | 4.5 | 6.7 | V |
| | | 50°C | 3.6 | – | 3.9 | – | 4.2 | – | V |
| | | 70°C | – | 5.7 | – | 6 | – | 6.3 | V |
| LCM current consumption (No B/L) | Idd | Vdd=5V | – | | 2 | | 3 | | mA |
| Backlight current consumption | LED/edge | VB/L=4.2V | – | | – | | – | | mA |
| | LED/array | VB/L=4.2V | – | | 220 | | – | | mA |

**REMARK**

LCD option: STN, TN, FSTN

Backlight Option: LED,EL Backlight feature, other Specs not available on catalog is under request.

*Fuente: Datasheet LCD 1604A - Powertip*

**5 BLOCK DIAGRAM**

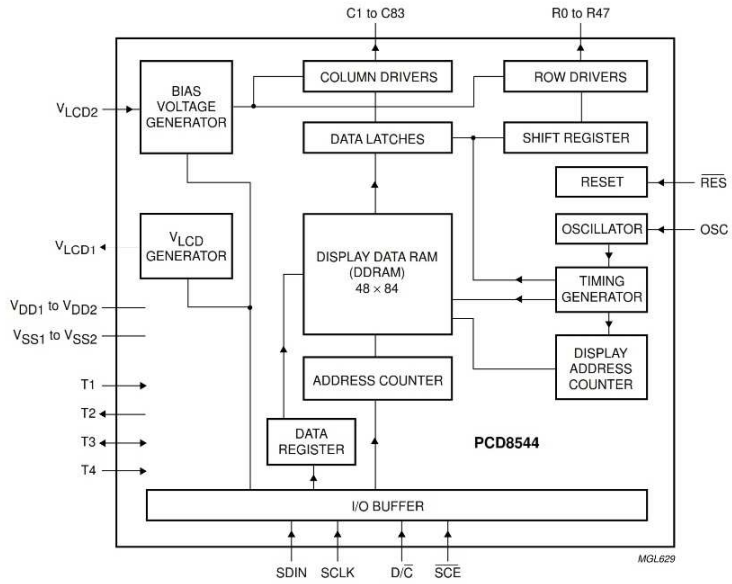



Fig.1 Block diagram.



*Fuente: Datasheet LCD 5110 – Philips Semiconductors*

## 6 PINNING

| SYMBOL | DESCRIPTION |
|---|---|
| R0 to R47 | LCD row driver outputs |
| C0 to C83 | LCD column driver outputs |
| $V_{SS1}$, $V_{SS2}$ | ground |
| $V_{DD1}$, $V_{DD2}$ | supply voltage |
| $V_{LCD1}$, $V_{LCD2}$ | LCD supply voltage |
| T1 | test 1 input |
| T2 | test 2 output |
| T3 | test 3 input/output |
| T4 | test 4 input |
| SDIN | serial data input |
| SCLK | serial clock input |
| D/$\overline{C}$ | data/$\overline{command}$ |
| $\overline{SCE}$ | chip enable |
| OSC | oscillator |
| $\overline{RES}$ | external reset input |
| dummy1, 2, 3, 4 | not connected |

**Note**

1. For further details, see Fig.18 and Table 7.

### 6.1 Pin functions

#### 6.1.1 R0 TO R47 ROW DRIVER OUTPUTS

These pads output the row signals.

#### 6.1.2 C0 TO C83 COLUMN DRIVER OUTPUTS

These pads output the column signals.

#### 6.1.3 $V_{SS1}$, $V_{SS2}$: NEGATIVE POWER SUPPLY RAILS

Supply rails $V_{SS1}$ and $V_{SS2}$ must be connected together.

#### 6.1.4 $V_{DD1}$, $V_{DD2}$: POSITIVE POWER SUPPLY RAILS

Supply rails $V_{DD1}$ and $V_{DD2}$ must be connected together.

#### 6.1.5 $V_{LCD1}$, $V_{LCD2}$: LCD POWER SUPPLY

Positive power supply for the liquid crystal display. Supply rails $V_{LCD1}$ and $V_{LCD2}$ must be connected together.

#### 6.1.6 T1, T2, T3 AND T4: TEST PADS

T1, T3 and T4 must be connected to $V_{SS}$, T2 is to be left open. Not accessible to user.

#### 6.1.7 SDIN: SERIAL DATA LINE

Input for the data line.

#### 6.1.8 SCLK: SERIAL CLOCK LINE

Input for the clock signal: 0.0 to 4.0 Mbits/s.

#### 6.1.9 D/$\overline{C}$: MODE SELECT

Input to select either command/address or data input.

#### 6.1.10 $\overline{SCE}$: CHIP ENABLE

The enable pin allows data to be clocked in. The signal is active LOW.

#### 6.1.11 OSC: OSCILLATOR

When the on-chip oscillator is used, this input must be connected to $V_{DD}$. An external clock signal, if used, is connected to this input. If the oscillator and external clock are both inhibited by connecting the OSC pin to $V_{SS}$, the display is not clocked and may be left in a DC state. To avoid this, the chip should always be put into Power-down mode before stopping the clock.

#### 6.1.12 $\overline{RES}$: RESET

This signal will reset the device and must be applied to properly initialize the chip. The signal is active LOW.



*Fuente: Datasheet LCD 5110 – Philips Semiconductors*