UNIVERSIDAD DE LA LAGUNA.

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA.

TRABAJO FIN DE GRADO:

MEDIDA DEL CONSUMO ELECTRÍCO DE UNA INSTALACIÓN DE BAJA TENSIÓN EN TIEMPO REAL MEDIANTE MICROCONTROLADOR.

Autores:

Rayco Viera García.

Sergio Daniel Febles Donate.

Tutores:

Alejandro José Ayala Alfonso.

Beatriz Rodríguez Mendoza.

Agradecimientos.

En primer lugar agradecer a Alejandro Ayala la oportunidad que nos ha brindado para realizar este proyecto y aprender de él. Queremos expresar nuestra más sincera admiración por la labor que ha realizado. Se ha convertido en mentor y referente para nosotros. Gracias por su apoyo, ayuda, constancia y dedicación.

A José Carlos Sanluis, siempre poniendo a nuestra disposición el valor incalculable de sus conocimientos. Sin él muchas cosas se habrían complicado demasiado. Gracias por la atención que nos ha dedicado.

Agradecer a los profesores del departamento que han contribuido en la realización del proyecto colaborando en el desarrollo del mismo, en especial, Beatriz Rodríguez, Oswaldo Bernabé, Francisco LLopis, Delfín y Fernando Rosa.

A nuestras parejas y familias por el apoyo prestado todos estos años y, finalmente, a nuestros compañeros y amigos de carrera por todos esos buenos momentos que hemos pasado juntos.

Índice.

Abstract.

The purpose of this project is to design a device to measure the electrical power of a house in real time. To this end, we have used the ATmega2560 and ATmega328 microcontrollers on an Arduino board.

The project is divided into two parts. The first one involved the design and test circuits for voltage signals and current. In these circuits we have used current sensors and optocouplers.

The second part was focused on the development of a program to translate the voltage and current values that the user can view. This system allows the measure of the electrical power in real time and different parameters such as electricity bill or power factor.

In addition, all data can be viewed on a smartphone by installing an Android application. The device can notify when the light is cut. It also alerts when we overcome some power and current values . Actually, the system is a low cost power meter.

CAPÍTULO I: Introducción general.

CAPITULO I. INTRODUCCIÓN GENERAL.

I.1. Introducción.

Tales de Mileto, hacia el año 600 a.C, realizó una serie de observaciones sobre electricidad estática, de las cuales concluyó que la fricción dotaba de magnetismo al ámbar. El caso contrario ocurría con minerales como la magnetita, que no necesitaban frotarse. Durante cientos de años no hubo cambios significativos en el fenómeno de la electricidad y el magnetismo, hasta que en 1790 Alessandro Volta, físico italiano, crea la primera pila capaz de producir corriente continua. En 1820 Faraday concluye mediante experimentos que la corriente eléctrica produce un campo magnético, enunciando el "Principio de la inducción electromagnética". Sin embargo, no fue hasta el año 1879 cuando Thomas Alva Edison crea la primera bombilla eléctrica, diseñada para lucir durante 40 horas, aproximadamente (*FiguraI.1).[1].*

Figura I.1. Primera bombilla.

La creación de la bombilla por parte Edison supuso un gran avance, pues se cambió la manera de entender la vida y el trabajo al liberar al hombre de los ciclos día/noche. En principio, estaba ideada para estar presente en lugares públicos, si bien más tarde fue ampliamente extendida.

En esta misma época, se produjo la llamada "guerra de las corrientes" [2], entre el propio Edison y un ingeniero de origen Serbio, Nikola Tesla. La electricidad era un negocio pionero que podía sustituir al vapor para hacer funcionar motores, lo que contribuyó enormemente al desarrollo de una segunda revolución industrial y pronto las centrales eléctricas comenzaron a emerger. A medida que aumentaba la demanda de electricidad, también lo hizo la necesidad de construir centrales mayores y de llevar la energía a distancias más lejanas.

El sistema de Edison utilizaba corriente continua, mientras Tesla defendía la corriente alterna .Tesla se basaba en dos argumentos para defenderla: el primero, las pérdidas en el cobre eran mucho menores que con corriente continua y, el segundo, la corriente alterna se podía elevar mediante un transformador, permitiendo ser transportada a grandes distancias con pocas pérdidas. Posteriormente el voltaje se puede reducir a niveles seguros para ser consumidos por el usuario. Finalmente, la corriente alterna de Tesla triunfó.

Conforme el negocio de la electricidad se fue extendiendo, era necesario idear sistemas para la medida del consumo eléctrico. Actualmente estos sistemas son parte de la instalación eléctrica de cualquier vivienda o industria. Asimismo, existen diversos equipos para la medida de parámetros propios de este tipo de señales tales como voltímetros, amperímetros, vatímetros, etc. (*Figura I.2).*

Figura I.2. Vatímetro comercial.

I.2. Objetivo del proyecto.

En el presente proyecto, se ha llevado a cabo el diseño y construcción de un sistema para la medida de la potencia eléctrica en tiempo real. Todo ello se ha realizado mediante el uso del entorno Arduino [3]. La supervisión de las diferentes tareas, tanto de adquisición como de control y comunicación, es efectuada a través de microcontroladores ATmega2560 y ATmega328. El proyecto cuenta con dos partes bien diferenciadas. En la primera, tiene lugar el acondicionamiento de la señal a medir mediante el diseño e implementación de circuitos de este tipo, mientras la segunda, se centra en el desarrollo del software para leer e interpretar la información remitida por los circuitos anteriores.

La *Figura I.3* muestra el esquema general de bloques del dispositivo. Su funcionamiento está controlado por un microcontrolador ATmega2560 de Atmel presente en una tarjeta Arduino Mega 2560 que será el encargado de procesar la información que le llega desde dos microcontroladores ATmega328 (Arduinos Nano, N1 y N2) que aportan, respectivamente, los valores instantáneos de la corriente y la tensión, permitiendo calcular los valores eficaces de la intensidad de corriente (I), tensión (V) y potencia (P). Para ello, sendos circuitos acondicionadores de señal para la tensión y corriente, proceden a adaptarlas para que operen en el rango de 0 a 5 V en el que trabaja el conversor analógico-digital (CAD) de los ATmega328. El usuario, en todo momento, podrá disponer de la información que desee a través del display, vía SMS o Internet (*Figura I.4.*).

Para la medida de la intensidad de corriente se ha hecho uso de dos sensores, uno de tipo inductivo y otro de Efecto Hall (HWCT004 y ACS712, respectivamente), que generan una tensión proporcional a la corriente que circula por los mismos, con un límite de 25A. Para aumentar la sensibilidad, la escala de corriente se ha dividido en tres tramos: 0/0,5A, 0,5/3,3A y 3,3/25A. Para las dos primeras se utiliza el sensor inductivo, mientras el tercero hace uso del ACS712, siendo el ATmega2560 el encargado de seleccionar cada escala mediante un multiplexor analógico.

Un proceso similar se lleva a cabo con la tensión. Sin embargo, en este caso partimos de valores de 230V eficaces que se han de llevar al mismo rango de 0 a 5V.

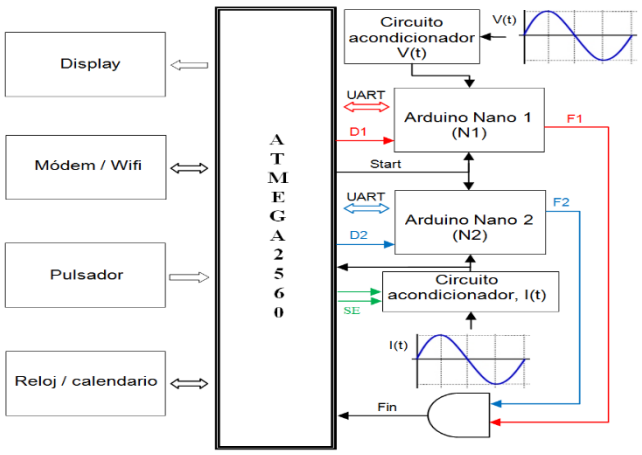


Figura I.3. Esquema general de bloques del dispositivo.

El sistema posibilita la medida en tiempo real, tanto de la potencia consumida en cualquier vivienda donde sea instalado, como los diferentes parámetros que se relacionan con esta magnitud.

Todos los datos recopilados son almacenados para posteriores consultas. Asimismo, el dispositivo permite ajustar distintos tipos de aviso tales como picos de corriente, corte del suministro eléctrico o estimar la factura de la luz. En la *Figura I.4* podemos observar el diagrama de bloques de las interfaces que permiten consultar al usuario la información proporcionada por el sistema.

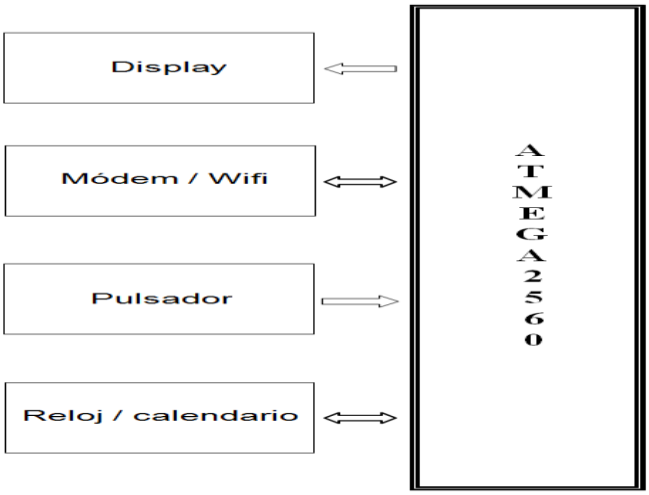


Figura I.4. Interfaces para consulta del usuario.

I.3. Estructura general del trabajo.

En la presente memoria, la descripción de los diferentes módulos que conforman el sistema implementado así como su funcionamiento, se realiza mediante el desarrollo de ocho capítulos. En el primero de los mismos, se pretende ofrecer una visión rápida y general de los objetivos alcanzados con el proyecto de forma que su lectura nos oriente para el resto de la memoria.

El Capítulo II centra su objetivo en la descripción de las características generales de los componentes, en torno a los cuales, se desarrolla todo el trabajo. Nos referimos a los microcontroladores Atmega2560 y Atmega328 de Atmel, encargados de gestionar todo el sistema, incluyendo las interfaces de usuario, procesado de datos, etc.

Una de las características más importante del sistema diseñado, es la posibilidad de consultar datos en el propio dispositivo, pero también de forma remota. Los bloques que permiten este proceso son presentados durante el desarrollo del Capítulo III.

El Capítulo IV aborda el desarrollo de los circuitos acondicionadores de señal y el funcionamiento de los mismos, además de mostrar cómo ha sido el proceso de adquisición de los datos necesarios para el funcionamiento del sistema. Por otro lado, el Capítulo V muestra como estos datos son procesados por el software necesario para el correcto funcionamiento del sistema y que conforma una parte fundamental del mismo.

La descripción del funcionamiento del sistema es presentada durante el Capítulo VI, en el cual el usuario puede obtener una visión general de la configuración del mismo.

El Capítulo VII se ha dedicado a presentar los resultados experimentales obtenidos con desarrollo del presente trabajo, donde se muestran diferentes imágenes del dispositivo con los datos obtenidos, así como la comparación con otros similares de tipo comercial.

La memoria finaliza con el Capítulo VIII, dedicado a presentar el presupuesto económico del proyecto y donde se han incluido gastos de material, mano de obra y costes de ejecución.

Capítulo II: Microcontroladores.

CAPITULO II. MICROCONTROLADORES.

En este Capítulo se hace referencia a los microcontroladores Atmega2560 y Atmega328 de Atmel, componentes principales para la realización del dispositivo, desde el punto de vista de su utilización en el diseño desarrollado, es decir, se analizarán sus prestaciones y características para la implementación del sistema.

II.1. Microcontroladores.

Gracias a los continuos avances en la electrónica, especialmente en la tecnología de los semiconductores y en el campo de la microtecnología, se ha conseguido desarrollar un tipo de ordenador potente, pequeño y de reducidos costes económicos, llamado microcontrolador (*Figura II.1*). Los microcontroladores constituyen un ordenador en un único circuito integrado, su reducido tamaño les permite estar en el interior de dispositivos que gobiernan, haciendo los productos finales más versátiles y potenciándolos sin apenas alterar su tamaño y coste. Actualmente los microcontroladores son usados en todos los sectores de la industria, siendo los de la automoción, la informática y las comunicaciones los que más uso hacen de este tipo de integrados.

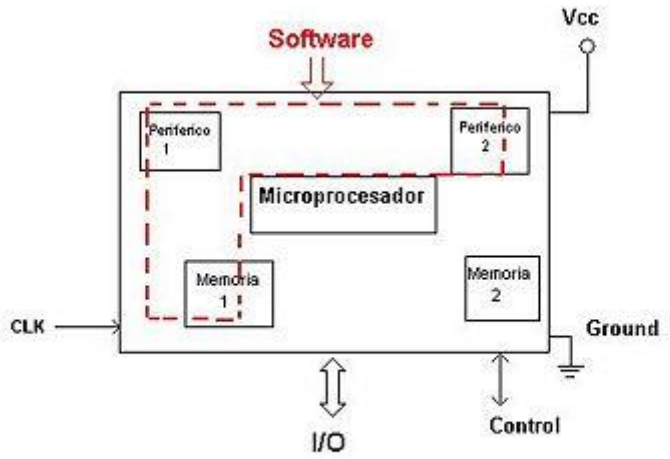


Figura II.1 Esquema general de un microcontrolador.

En este proyecto se ha optado por utilizar dos modelos de microcontroladores del fabricante de componentes electrónicos Atmel [4]. Se puede decir que ambos son elementos centrales del dispositivo, puesto que los demás componentes giran en torno a ellos.

II.2. ATmega2560.

El ATmega2560 es un microcontrolador que emplea los últimos avances en el campo de los sistemas integrados con procesadores, obteniéndose a un dispositivo potente a la vez que versátil [5]. Hace uso de arquitectura de tipo AVR (basado en la arquitectura Harvard), concretamente pertenece a la subfamilia “tinyAVR”. En la *Figura II.2* podemos observar el aspecto de dicho microcontrolador.

Figura II.2 Microcontrolador ATmega2560.

Seguidamente se precede a enumerar aquellas características más relevantes del ATmega2560.

II.2.1. Patillas.

Las patillas (llamadas también “pines”) de entrada/ salida del microcontrolador son necesarias para comunicar al mismo con el mundo exterior. A continuación, en la *Figura II.3* se muestra el patillaje del microcontrolador ATmega2560. Observando la imagen se puede saber que patilla es el que recibe la alimentación eléctrica (señalado como “VCC”), que pines están conectados a tierra (los señalados como “GND”), que patillas son las de E/S, etc.

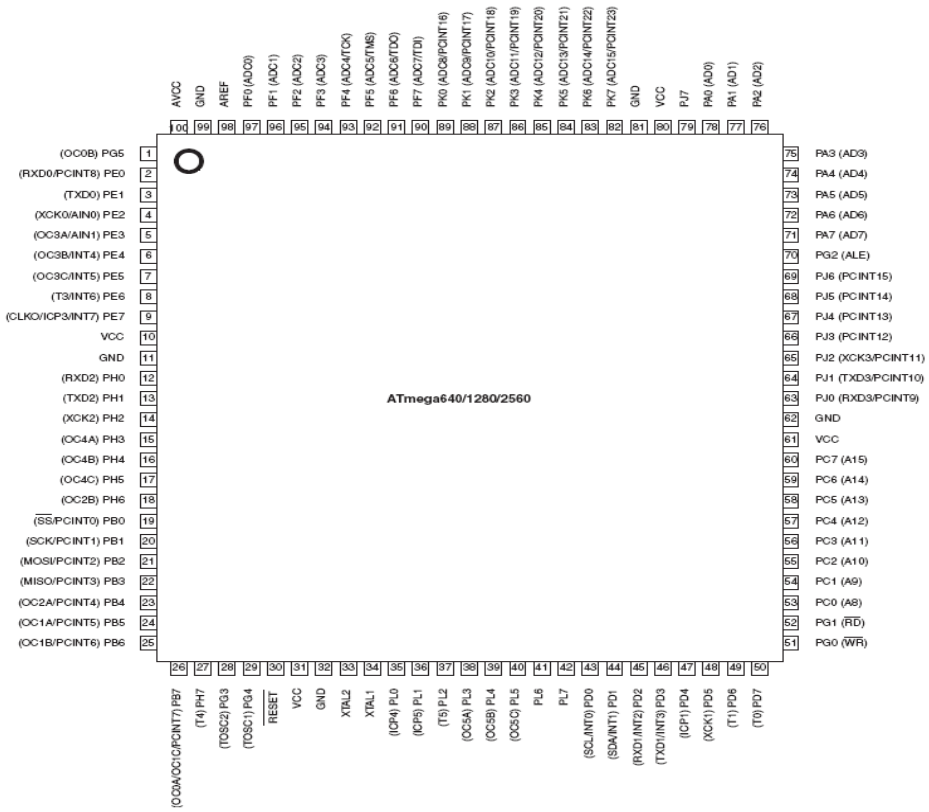


Figura II.3 Patillaje del Microcontrolador ATmega2560.

II.2.2. Memorias del microcontrolador.

Hay diferentes tipos y tamaños de memoria alojadas dentro de los microcontroladores. En este caso:

- Memoria Flash: memoria persistente donde se almacena permanentemente el programa que ejecuta el microcontrolador (hasta una nueva reescritura si es necesario). En el Atmega2560 esta memoria es de 256KB.
- Memoria SRAM: memoria volátil donde se alojan los datos que en cada instante el programa (grabado separadamente en la memoria Flash) necesita crear o manipular para su correcto funcionamiento. Estos datos suelen tener un contenido variable a lo largo del tiempo de ejecución del programa y cada uno es de un tipo concreto. Independientemente del tipo de dato, su valor siempre será eliminado cuando se deje de alimentar eléctricamente al microcontrolador. En el caso del ATmega2560 la capacidad de la memoria es de 4KB. Si necesitáramos ampliar la cantidad de memoria SRAM disponible, siempre podríamos adquirir memorias SRAM independientes y conectarlas al microcontrolador utilizando algún protocolo de comunicación.
- Memoria EEPROM: memoria persistente donde se almacenan datos que se desea que permanezcan grabados una vez apagado el microcontrolador para poderlo usar posteriormente en siguientes reinicios. En el caso del Atmega2560 esta memoria tiene una capacidad de 4K. Al igual que la SRAM, la EEPROM también se puede ampliar adquiriendo módulos independientes y conectándolos al microcontrolador utilizando algún protocolo de comunicación.

II.2.3. Registros del microcontrolador.

Los registros son espacios de memoria existentes dentro de la propia CPU de un microcontrolador. Son muy importantes porque tienen varias funciones imprescindibles: sirven para albergar los datos (cargados previamente desde la memoria SRAM o EEPROM) necesarios para la ejecución de las instrucciones previstas próximamente (y así tenerlos perfectamente disponibles en el momento adecuado); sirven también para almacenar temporalmente los resultados de las instrucciones recientemente ejecutadas (por si se necesitan en algún instante posterior) y sirven además para alojar las propias instrucciones que en ese mismo momento estén ejecutándose. Los registros del Atmega2560 son de 8 bits.

II.2.4. Protocolos de comunicación.

Cuando se desea transmitir un conjunto de datos desde un componente electrónico a otro, se puede hacer de múltiples formas. Una de ellas es estableciendo un comunicación “serie”; en este tipo de comunicación la información es transmitida bit a bit por un único canal, enviando por tanto un solo bit en cada instante. Otra manera de transferir datos es mediante la llamada comunicación “paralela”, en la cual se envían varios bits simultáneamente, cada uno por un canal separado y sincronizado con el resto.

El microcontrolador, a través de sus pines E/S, utiliza el sistema de comunicación serie para transmitir y recibir órdenes y datos hacia/desde otros componentes electrónicos. Esto es debido a que en la comunicación serie solo necesitamos un único canal, mientras que en comunicación paralelo se necesitan varios cables, con el correspondiente incremento de complejidad, tamaño y coste.

No obstante, no podemos hablar de un solo tipo de comunicación serie. Existen muchos protocolos y estándares diferentes basados todos ellos en la transferencia de

información en serie, cada uno con sus características específicas. Los protocolos de comunicación utilizados por el microcontrolador son:

- I2C (Inter-IntegratedCircuit): sistema muy utilizado en la industria que utiliza dos líneas para transmitir la información ("SDA" y "SCL"). La velocidad de transferencia de datos es de 100kbits por segundo.
- SPI (Serial Peripheral Interface): el bus de comunicación SPI es un estándar que permite controlar (a cortas distancias) casi cualquier dispositivo electrónico digital que acepte un flujo de bits serie sincronizado). Tiene la desventaja de utilizar más pines que el bus I2C.

II.3. ATmega328.

Este microcontrolador también tiene una arquitectura AVR, arquitectura desarrollada por Atmel y en cierta medida "competencia" directa de otras como la PIC del fabricante Microchip. Pertenece a la subfamilia de microcontroladores "megaAVR". La *Figura II.4* muestra el aspecto de este microcontrolador [6].

Figura II.4 Microcontrolador ATmega328.

Aunque sus prestaciones no son tan potentes como las del ATmega2560, este microcontrolador también es muy eficaz y permite realizar una gran cantidad de tareas.

II.3.1. Patillas.

Como hemos señalado anteriormente, es necesario conocer su disposición concreta en cada microcontrolador. Observando la *Figura II.5* se puede saber que pin es el que recibe la tensión de alimentación eléctrica (señalado como "VCC"), cuáles están conectados a tierra (señalados como "GND"), quienes son los de E/S (PBx, PCx o PDx) y la existencia de otros más específicos como el AVCC (la alimentación suplementaria para el convertidor analógico digital interno del chip) o el AREF (referencia analógica para dicho convertidor). También se puede observar que junto al nombre de los pines de E/S se indica entre paréntesis las funciones especializadas que cada uno de ellos tiene en particular (además de su función genérica de entrada/salida).

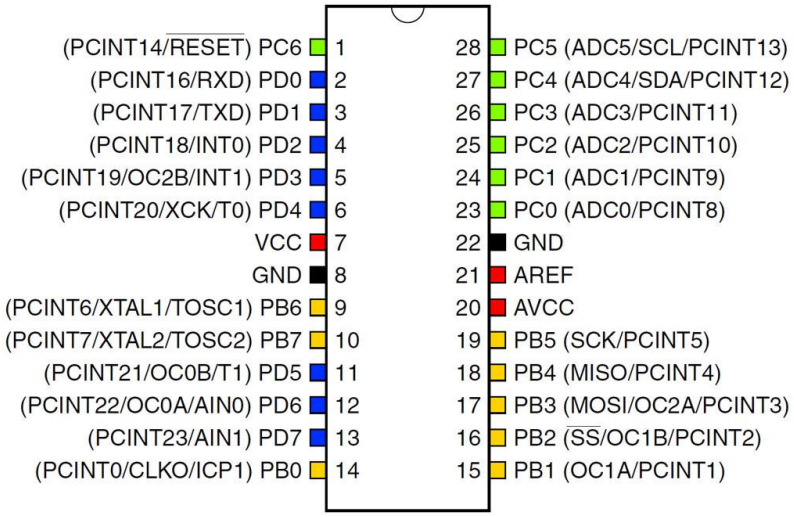


Figura II.5Patillaje del Microcontrolador ATmega328.

II.3.2. Memorias del microcontrolador.

Al igual que en el ATmega2560, el ATmega328 aloja diferentes tipos de memoria y capacidad así:

- 32 KB de memoria Flash.
- 2 KB de memoria SRAM.
- 1 KB de memoria EEPROM.

II.3.3. Registros del microcontrolador.

Su tamaño es muy reducido ya que tan solo tienen capacidad para almacenar unos pocos bits cada uno. Pero este factor es una de las características más importantes de cualquier microcontrolador, porque cuanto mayor sea el número de bits que “quepan” en sus registros, mayores serán sus prestaciones en cuanto a poder de cómputo y velocidad de ejecución. En chip ATmega328 es de 8 bits.

II.3.4. Protocolos de comunicación.

Los protocolos utilizados son similares a los del ATmega2560, es decir, el SPI e I2C.

Capítulo III: Unidades de entrada/salida.

CAPITULO III. UNIDADES DE ENTRADA/SALIDA.

III.1. Introducción.

Las unidades de entrada y salida son dispositivos a través de los cuales el microcontrolador se comunica con el exterior, es decir, la información es enviada y recibida a través de estas interfaces.

Existen diversos tipos de unidades de entrada (*Figura III.1)*, tales como joystick, ratón, teclado, etc. Estos dispositivos permiten introducir datos externos al microcontrolador para su posterior tratamiento. Dichos datos pueden provenir de distintas fuentes, siendo la principal el ser humano.

Figura III.1. Unidades de entrada.

La mayor parte de las unidades de salida (*Figura III.2)* se emplean para informar, comunicar, proyectar o para dar al usuario cierta información, de la misma forma que se encargan de convertir impulsos eléctricos en información legible para éste, es decir, son los que reciben información que es procesada previamente por el microcontrolador y la reproducen para que sea perceptible por el usuario.

Figura III.2. Unidades de salida.

Para aumentar la utilidad y prestaciones del sistema, se ha hecho uso de varios elementos de entrada y salida. La posibilidad de interactuar con el dispositivo mediante

pulsadores y observar resultados de forma telemática, ya sea mediante mensajería instantánea o Internet, hace que el dispositivo sea mucho más versátil. A continuación se detallan las características de las unidades de entrada/salida utilizadas para implementar el sistema.

III.2. Pulsador.

El pulsador (*Figura III.3)* es un dispositivo de entrada que permite el paso de corriente cuando es presionado [7]. El sistema detectará si el pulsador ha sido presionado o no por el usuario, realizando una tarea específica que ha sido anteriormente programada.

Figura III.3. Pulsador.

Para conectar el pulsador al microcontrolador tendremos que utilizar resistencias de protección. Existen dos configuraciones en función de si deseamos obtener un "0" o un "1" lógico al presionar el pulsador. Es por ello que se utilizan las llamadas resistencias de Pull-up y Pull-down, respectivamente. Ambas se conectan entre el PIN digital y una de las tensiones de referencia (5V o 0V), (*Figura III.4).*

- La resistencia de Pull-Up fuerza HIGH cuando el pulsador está abierto. Cuando está cerrado el PIN se pone a LOW, la intensidad que circula se ve limitada por esta resistencia.
- La resistencia de Pull-Down fuerza LOW cuando el pulsador está abierto. Cuando está cerrado el PIN se pone a HIGH, y la intensidad que circula se ve limitada por esta resistencia.

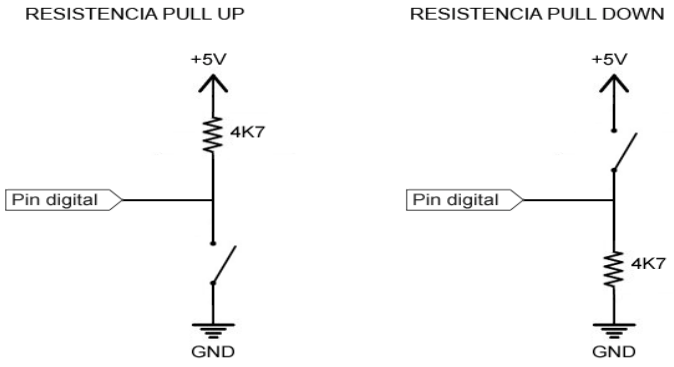


Figura III.4. Resistencia Pull up y Pulldown.

III.3. Pantalla LCD.

La pantalla LCD (*Figura III.5)* es un dispositivo de salida que permite mostrar información acerca del sistema [8] en forma de valores numéricos, texto, etc. Existen diversos tipos y tamaños, pero en este trabajo se ha optado por una retroiluminada de 20x4 (20 caracteres y 4 líneas).

Figura III.5. Pantalla LCD.

En el caso que nos ocupa, la pantalla mostrará los valores de los parámetros medidos. La comunicación entre el microcontrolador y la pantalla es realizada a través del protocolo I2C, lo que permite una mayor rapidez y simplificación del sistema.

III.4. Reloj en tiempo real.

El microcontrolador necesita conocer la fecha y la hora para poder realizar determinadas tareas, y su reloj interno posee ciertas limitaciones. Una manera sencilla y barata para conocer estos parámetros de forma fiable es mediante la utilización de un reloj en tiempo real (DS3231, *Figura III.6)*. Con este dispositivo es posible conocer en todo momento la hora además de día, mes y año de forma automática y válido hasta 2100 [9].

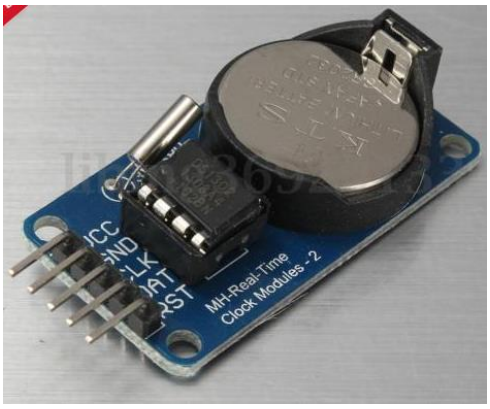


Figura III.6. Reloj en tiempo real.

El DS3231 se comunica con el microcontrolador mediante el protocolo I2C y tiene un consumo del orden de miliamperios.

III.5. Modem GSM.

Es un dispositivo que permite la conectividad inalámbrica del sistema mediante el uso de la red GSM controlado mediante comandos AT [10]. Es una posible solución para la comunicación entre el sistema y el usuario ya que, mediante el uso de SMS o llamadas, podremos obtener información acerca de los parámetros medidos. Para su utilización es necesario disponer de una tarjeta SIM y suficiente cobertura de red. La *Figura III.7* muestra el módulo GSM.

Figura III.7. Modem GSM.

El uso del modem GSM permite no limitarse al binomio pulsadores/display, sino que se amplía a las comunicaciones a distancia mediante las redes de telefonía móvil e Internet. Así, mediante la utilización de un teléfono móvil y el uso del modem GSM conectado al dispositivo, el usuario podrá recibir mensajes de alerta, solicitar determinadas información, etc. Su control se realiza mediante comandos AT a través de una UART del Arduino Mega 2560 (pines TX1 y RX1 de dicha placa).

En los casos en los que el dispositivo implementado sea utilizado como instrumento de medida en un laboratorio con objeto de obtener valores correspondientes a magnitudes como la intensidad de corriente, potencias activa y reactiva o el factor de potencia, entre otros, no se considera como necesaria la utilización de dicho modem.

III.6. Módulo Wifi.

Se ha hecho uso de un módulo ESP8266 (*Figura III.8)* que permite la conexión del dispositivo a Internet mediante el acceso a través de wifi [11]. Esto permite la consulta de datos mediante aplicación Android conectada internet.

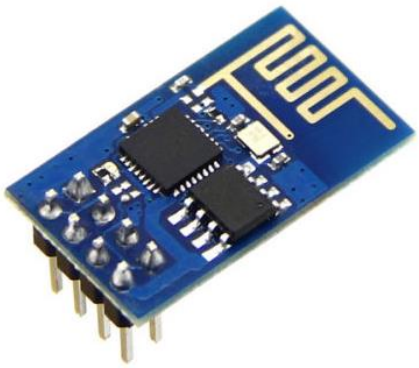


Figura III.8. Módulo Wifi.

La tensión de alimentación de este dispositivo se corresponde con 3,3 voltios, además utiliza comunicación UART y en modo de baja energía consume 10 microamperios. Dispone de un microcontrolador propio y 2 pines GPIO para poder funcionar de forma autónoma sin necesidad de microcontrolador externo.

Capítulo IV: Adquisición de datos.

CAPITULO IV. ADQUISICIÓN DE DATOS.

En este Capítulo se presentarán los circuitos acondicionadores de señal que han sido implementados a lo largo del proyecto. En un principio los circuitos fueron diseñados con una configuración determinada pero, a medida que se desarrollaba el proyecto, ha sido necesario un cambio parcial en cada uno de ellos. A continuación se presenta los circuitos y sus características más relevantes.

IV.1. Introducción.

Se puede definir la potencia eléctrica como la cantidad de energía absorbida por un elemento durante determinado tiempo [12] donde la energía consumida por un dispositivo eléctrico se mide en kilovatios-hora (kWh).

El objetivo principal del dispositivo realizado es tanto la medida de la potencia, como los diferentes parámetros asociados a la misma. Los tipos de potencia existentes son los siguientes:

- Potencia activa (P):

Es la potencia consumida por los circuitos y, por tanto, utilizada para determinar la demanda energética. La potencia activa se debe a los elementos resistivos y representa la capacidad de transformar energía eléctrica en trabajo. Su expresión analítica es mostrada en (1).

$$P = V * I * cos\gamma \qquad (1)$$

Siendo:

P= Potencia activa.

V= Voltaje.

I= Intensidad.

cosγ= Factor de potencia.

- Potencia reactiva (Q):

Esta potencia no se consume ni se genera en sentido estricto y solo aparece cuando existen bobinas y condensadores. Su expresión se muestra en (2).

$$Q = V * I * sen\gamma \qquad (2)$$

Siendo:

Q= Potencia reactiva.

V= Voltaje.

I= Intensidad.

senγ= Seno del ángulo de potencia.

- Potencia aparente (S):

También es conocida como potencia compleja, representa la suma vectorial de la potencia disipada en un circuito (potencia activa) y la utilizada para la formación de los campos eléctrico y magnético de sus componentes (potencia reactiva). En (3) se observa la definición analítica de la potencia aparente.

$$S = V * I \qquad (3)$$

Siendo:

S= Potencia aparente.

V= Voltaje.

I= Intensidad.

En la *Figura IV.1* podemos observar el llamado triángulo de potencias que relaciona las tres magnitudes descritas anteriormente con el denominado ángulo de potencia. Este triángulo es la mejor forma de ver y comprender de manera gráfica qué es el factor de potencia o coseno de "fi" (cosγ) y su estrecha relación con los restantes tipos de potencia presentes en un circuito eléctrico.

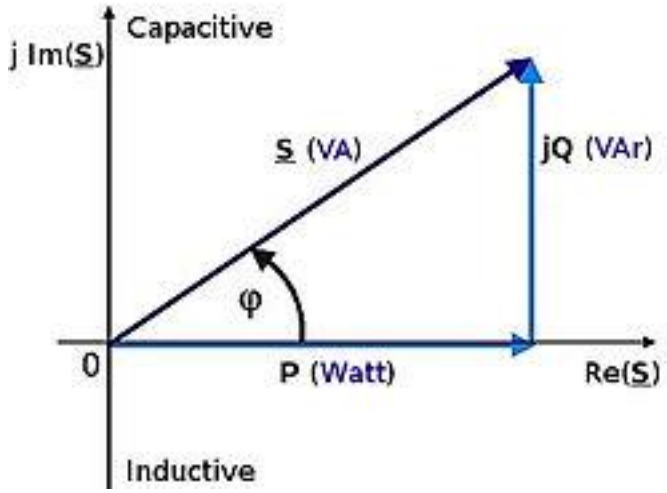


Figura IV.1. Triángulo de potencia.

Como se puede observar en el triángulo de potencia, el factor de potencia representa el valor del coseno del ángulo que se forma al representar gráficamente las potencias activa (P) y aparente (S), es decir, la relación existente entre la potencia real de trabajo y la total consumida por la carga o el consumidor conectado a un circuito eléctrico. Esta relación también se puede representar, de forma matemática, por medio de (4).

$$cos\gamma = \frac{P}{S} \qquad (4)$$

El objetivo del dispositivo es medir la potencia consumida en una vivienda, por lo que surge la necesidad de medir cada uno de los parámetros que influyen en el cálculo de las

potencias descritas anteriormente. Por tanto, ha sido indispensable determinar la tensión, la corriente y el factor de potencia para que el dispositivo funcione correctamente.

La idea inicial para tomar los valores de todos estos datos consistía en la utilización de circuitos detectores de cruce por cero [13]. Tal como se observa en la *Figura IV.2,* dicho circuito genera un "1" lógico (señal inferior) en su salida cada vez que se produce un cambio de polaridad en la señal de entrada (señal superior).

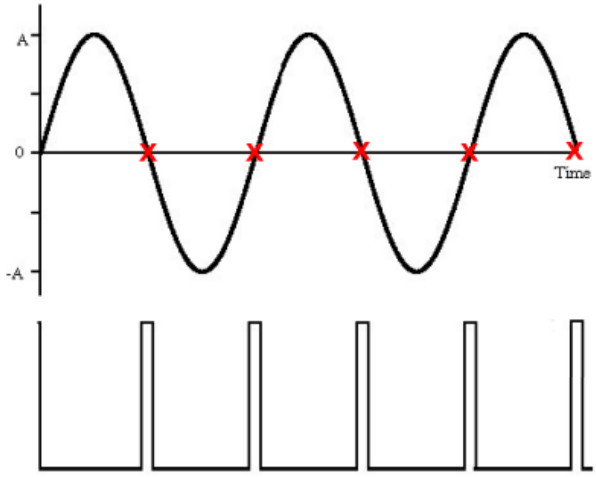


Figura IV.2.Señales del circuito detector de cruce por cero.

Si se utiliza un circuito detector de cruce por cero para las señales de tensión y de corriente se podría calcular el desfase entre ambas y, de esta manera, obtener el factor de potencia.

Como la señal de tensión no sufre distorsión y se conoce su periodo exacto (20 ms), se pueden medir sus flancos mediante un microcontrolador. Asimismo, serían medidos los flancos de la señal de corriente, y a través de la diferencia de tiempos obtenidos y simples operaciones matemáticas realizadas por el microcontrolador, se determinaría el valor del factor de potencia. Ya que la tensión tiene un valor fijo, y la corriente se mediría a través de un sensor, se hallarían todos los parámetros necesarios para el cálculo de potencias.

Sin embargo, este método no se ha podido llevar a cabo debido al elevado ruido presente al medir señales de corriente directamente en una vivienda. Como podemos observar en la *Figura IV.3,* cuando se mide directamente una señal de corriente de la red, en muchos casos no se suele obtener una señal sinusoidal perfecta, sino que, por el contrario, se obtiene una señal que contiene muchos armónicos, deformando por completo la sinusoide. Debido a este motivo, el uso de detectores de cruce por cero fue descartado.

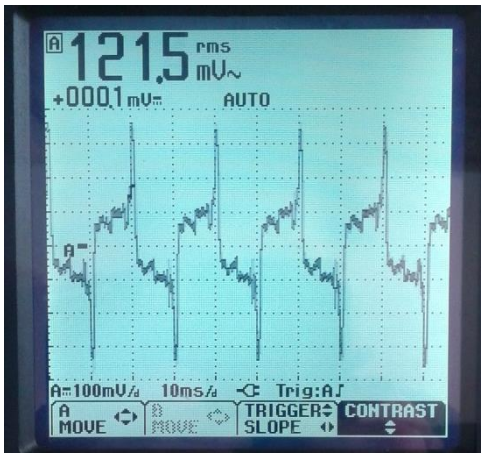


Figura IV.3.Señal tomada en vivienda.

La solución que finalmente ha sido desarrollada, será mostrada y explicada con detalle en el siguiente capítulo.

IV.2. Medida de tensión.

IV.2.1 Introducción.

Como ya se ha descrito anteriormente, uno de los parámetros más importantes a medir para poder calcular la potencia es la tensión. Sin embargo, el microcontrolador no es capaz de soportar los valores que se obtienen directamente de la red, es por ello, que se ha realizado un circuito acondicionador de tensión, de forma que se obtenga una señal réplica de la red con las características necesarias para que el microcontrolador sea capaz de muestrearla sin sufrir ningún tipo de daño. Este circuito dispone de una parte de potencia (donde la tensión de red entra al circuito) y otra de señal (donde los valores han sido reducidos y no hay peligro para el microcontrolador).

Básicamente se puede aislar un circuito mediante dos métodos. Por un lado se pueden utilizar transformadores y, por otro, haciendo uso de optoacopladores [14]. En este caso concreto, se ha optado por la segunda opción debido a que un transformador podría introducir desfases en las señales del circuito acondicionador, lo que produciría lecturas erróneas por parte del microcontrolador ya que habría un cambio significativo en el factor de potencia y, por tanto, en la potencia medida.

IV.2.2 Etapas y descripción del circuito.

Como se ha explicado anteriormente, el circuito de tensión posee una etapa de potencia y otra de señal.

➔ **Etapa de potencia.**

- Optoacoplador:

Ha sido necesario obtener la señal sinusoidal completa de la red para realizar la réplica de la tensión. Al utilizar un circuito optoacoplador, sólo se obtiene uno de los semiciclos de la señal, es por ello, que en la etapa de potencia se utilizó el integrado *MCT6* por sus características [15]. Dicho integrado, como podemos observar en la *Figura IV.4*, posee dos

optoacopladores que serán de gran utilidad para obtener cada uno de los semiciclos de la señal de red.

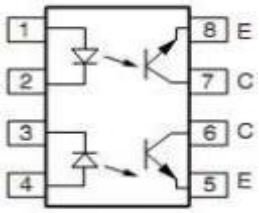


Figura IV.4. Integrado MCT6.

- Diodos 1N4007:

Se ha colocado un diodo*1N4007* en cada uno de los semiciclos con objeto de proteger los fotodiodos que se encuentran en el optoacoplador cuando trabajan en polarización inversa.

- Resistencias y potenciómetros:

Para limitar el paso de corriente se han colocado en serie, junto a cada uno de los diodos *1N4007,* resistencias de 200 kΩ. Además se han polarizado los fototransistores de la salida del optoacoplador, alimentándolos con 12V, y utilizando potenciómetros para poder realizar un ajuste en cada una de las señales.

En la *Figura IV.5* podemos observar la etapa de potencia del circuito de tensión. Asimismo, en la *Figura IV.6* se muestra la forma de la señal de salida del optoacoplador.

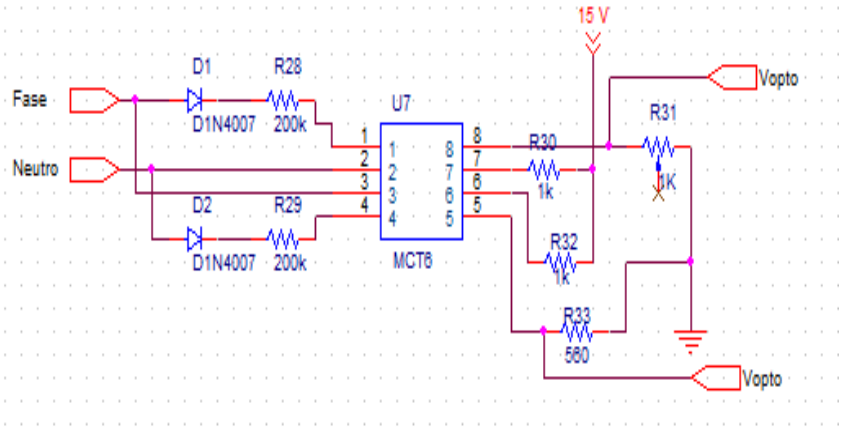


Figura IV.5. Etapa de potencia del circuito de tensión.

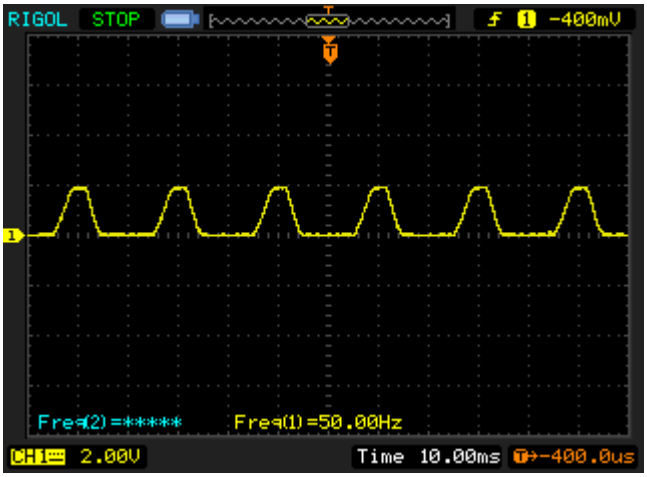


Figura IV.6. Señal de salida del optoacoplador.

➔ ***Etapa de señal.***

Toda la etapa de señal ha sido alimentada con ±12 y se ha hecho uso del *LM324 [16]*. Estos integrados permiten reducir de forma significativa las dimensiones del circuito, ya que cada uno de ellos posee cuatro *LM741* en su interior, lo que permite simplificar muchas conexiones, por ejemplo, de alimentación. El esquema del *LM*324 se muestra en la *Figura IV.7.*

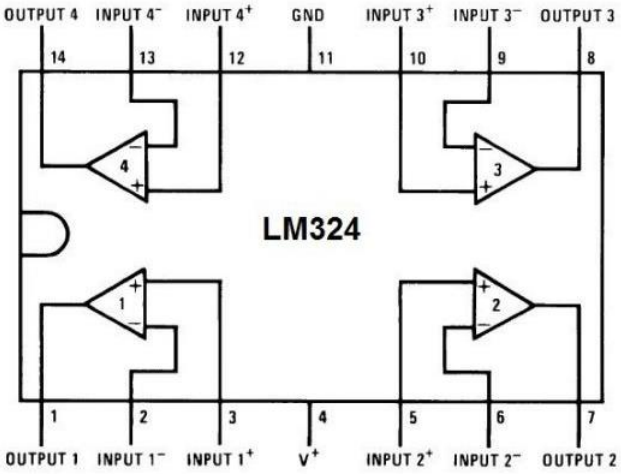


Figura IV.7. Integrado LM324.

Las señales de los semiciclos que vienen del optoacoplador son llevadas a seguidores de tensión con objeto de acoplar esta etapa con la siguiente. Posteriormente se utilizan

amplificadores en distintas configuraciones con el fin de modificar y adecuar la señal para que pueda ser muestreada.

Para eliminar el ruido de alta frecuencia, una de las etapas está compuesta por un filtro antialiasing. Este filtro se encuentra configurado para suprimir las frecuencias superiores a la mitad de la frecuencia de muestreo, es decir, a 5 kHz. Está compuesto por una resistencia de 1kΩ y un condensador de 33nF.

El microcontrolador solo lee valores positivos hasta 5V, es por ello que se debe montar la señal sinusoidal sobre un nivel de continua de 2,5V. De esta manera se tomaría como valor 0V el que corresponde a la posición 512 (de los 1024 niveles) una vez que el conversor analógico-digital de 10 bits del microcontrolador transforme la señal. Además su valor será de 5Vpp, lo que la hace adecuada para su muestreo y posterior tratamiento. Las *Figuras IV.8* y *IV.9* muestran, respectivamente, la etapa de señal del circuito de tensión y la señal de red frente a la señal acondicionada.

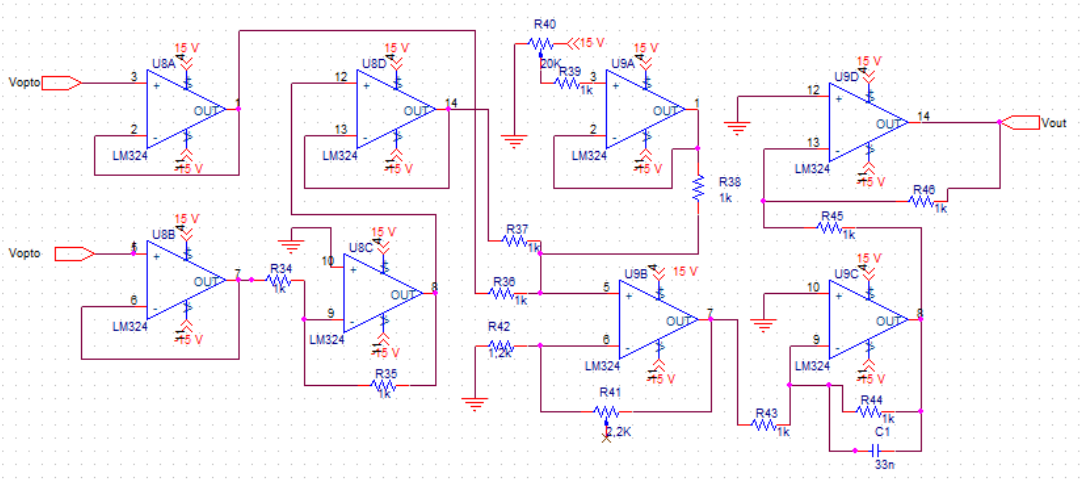


Figura IV.8. Etapa de señal.

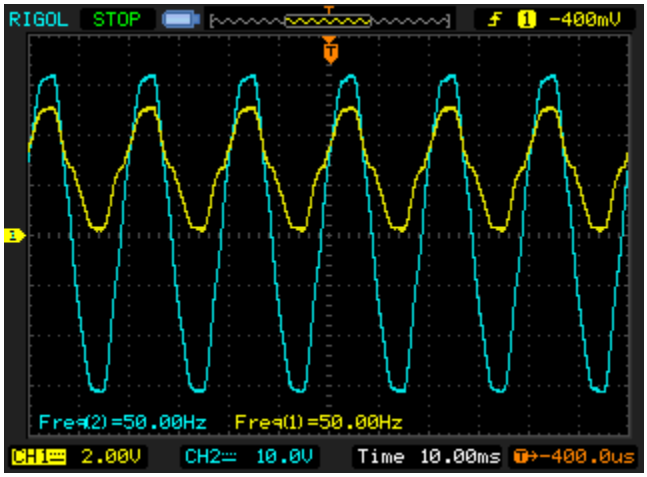


Figura IV.9. Señal de red y señal acondicionada.

IV.3. Medida de corriente.

IV.3.1 Introducción.

La corriente es el último parámetro necesario para calcular la potencia. Al igual que ocurría con el caso de la tensión, también es necesario acondicionar los valores de intensidad para que puedan ser muestreados correctamente por el microcontrolador.

Para poder realizar la lectura de la corriente, se han utilizado sensores que permiten obtener una señal de tensión proporcional a la intensidad circulante. Estas señales son acondicionadas, muestreadas y traducidas en un valor de corriente por el microcontrolador, lo que permitirá el cálculo de la potencia.

IV.3.2 Etapas y descripción del circuito.

El circuito de corriente posee como característica principal su capacidad para cambiar de escala con distintos umbrales de intensidad. Esta característica va ligada a los sensores que detectan el paso de corriente.

➔ **Componentes principales.**

- Sensor efecto Hall:

Se ha hecho uso del sensor de Efecto Hall *ACS712 [17]*, que permite obtener un valor de voltaje proporcional al de corriente con un límite de 40 A, un valor más que suficiente para el caso de una vivienda donde el valor máximo se corresponde, normalmente, con 25 A. Además, la señal de voltaje obtenida será de forma sinusoidal montada sobre 2,5 voltios de continua. El *ACS712* se muestra en la *Figura IV.10.*

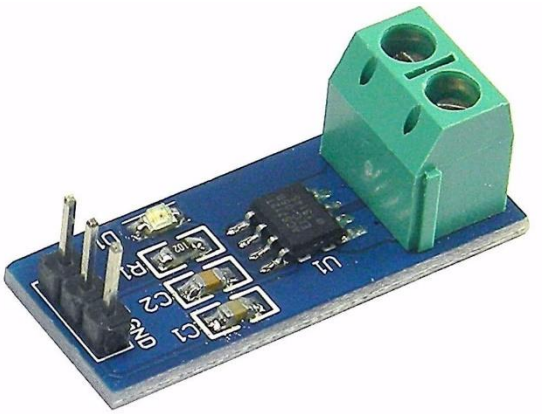


Figura IV.10. Sensor ACS712.

El principal problema que presenta este sensor es su sensibilidad. El *ACS712* no es capaz de detectar corrientes pequeñas, por lo que ha sido necesario utilizar otro que detecte las corrientes más bajas, permitiéndonos cubrir todo el rango de valores deseado y dotando de mayor precisión al dispositivo.

Al existir más de un sensor de corriente, es necesario la utilización de un sistema de escalas de forma que, en función del valor de la intensidad, leeremos los valores de un sensor o de otro.

- Sensor toroidal *HWCT004:*

Como se explicó anteriormente, el *ACS712* no es capaz de medir valores bajos de corriente, por lo que se ha hecho uso de otro más sensible, en este caso el *HWCT004 [18],* un toroide dotado de mayor sensibilidad y que se muestra en la *Figura IV.11.*

Figura IV.11. Sensor HWCT004.

Este sensor, como se verá a continuación, permite obtener mediante su correspondiente circuito acondicionador dos escalas de corriente, la primera para valores inferiores a 0,5 amperios y, la segunda para valores entre 0,5 y 3,3 amperios.

- Multiplexor *74HC4051:*

Para realizar el cambio de escala descrito anteriormente, ha sido necesario el uso del multiplexor analógico *74HC4051 [19].* Este integrado posee ocho entradas en tres de las cuales están conectadas las escalas existentes. El microcontrolador elegirá una entrada en función del valor de corriente, haciendo uso para ello de los terminales de selección S0 y S1 del *74HC4051*. En la *Figura IV.12* podemos observar el patillaje del multiplexor.

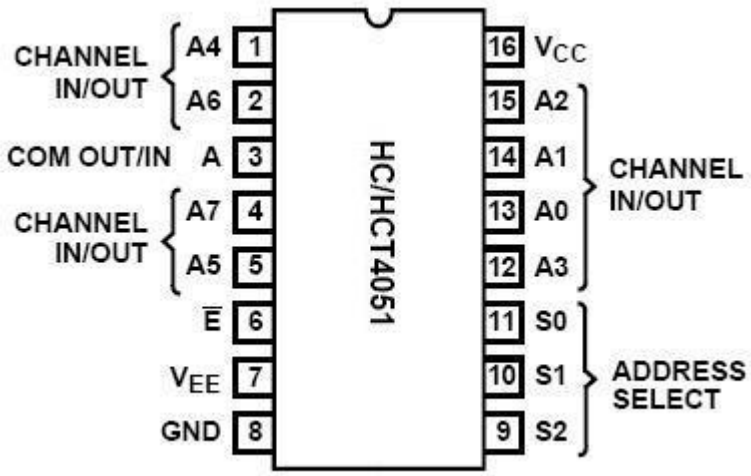


Figura IV.12. Multiplexor 74HC4051.

Cabe destacar que el circuito de corriente está alimentado a ±6,5V, de esta manera conseguimos que para valores altos de corriente, las escalas menores saturen a esta tensión, de forma que el microcontrolador queda protegido aunque los valores de intensidad aumenten.

- Integrados *LM324 y LM741:*

Al igual que en el caso del circuito de tensión, el de corriente ha sido implementado mediante el uso del *LM324*, lo que dota al circuito de mayor simpleza. Asimismo, en este caso se ha utilizado un único *LM741* necesario en la etapa final del circuito. Seguidamente, se detallará el tratamiento de la señal obtenida de cada sensor.

➔ **Descripción del circuito.**

Las señales de los sensores son llevadas hasta seguidores de tensión para posteriormente, mediante el uso de amplificadores en distintas configuraciones obtener valores adecuados para el muestreo de la señal.

Al igual que en el caso del circuito de tensión, en el de corriente se ha hecho uso de un filtro antialiasing configurado a 5kHz.

Las señales de cada una de las escalas se conectan a las entradas del multiplexor. El microcontrolador será el encargado de especificar los valores de control que decidirán que escala será la que se obtenga en la salida. Los rangos de corriente para cada escala son los siguientes:

- Escala 1: De 0 a 0,5 Amperios.
- Escala 2: De 0,5 a 3,3 Amperios.
- Escala 3: De 3,3 a 25 Amperios.

Estos valores han sido tomados en función al óptimo funcionamiento de cada sensor. El toroide es capaz de proporcionar dos escalas debido a que para valores de corriente bajos la señal será amplificada y para medianos atenuada, es decir, será el encargado de proporcionar los datos de la escala 1 y 2. El sensor de Efecto Hall obtendrá los datos para la escala 3.

Como resultado, en la salida del circuito de corriente se obtiene una señal montada sobre 2,5V de continua y 5Vpp de alterna, lo que la hace apta para ser muestreada. El microcontrolador sabe a qué escala pertenece la señal de salida y, por tanto, conocerá el factor para traducir estos valores de voltaje a corriente. En la *Figura IV.13* se puede observar una señal de tensión proporcional a otra de corriente. El circuito acondicionador de intensidad es mostrado en el apartado *Anexos* debido a las dimensiones que posee.

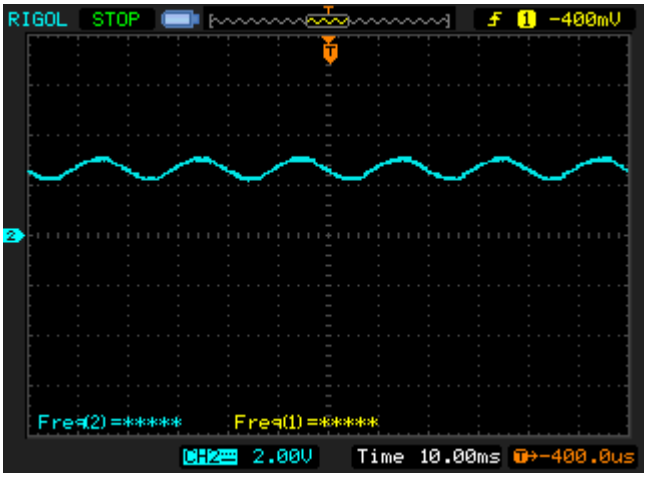


Figura IV.13. Señal de tensión proporcional a la corriente.

El tratamiento y procesado de señal, será realizado por los microcontroladores. En el siguiente Capítulo se explicará con más detalle cómo se ha llevado a cabo la obtención de la potencia y demás parámetros necesarios para el funcionamiento del sistema.

CAPÍTULO V: Procesado de datos y software de microcontroladores.

CAPITULO V. PROCESADO DE DATOS Y SOFTWARE DE MICROCONTROLADORES.

En el Capítulo anterior fueron presentados los circuitos empleados para obtener las señales acondicionadas proporcionales a la tensión e intensidad de corriente utilizadas para calcular los valores de potencia mediante el dispositivo. Los microcontroladores deben ejecutar un software específico para que dichas señales puedan ser interpretadas, de esta manera se obtendrán todos los parámetros que podrá consultar el usuario final. El tratamiento y procesado de los datos será explicado a continuación.

V.1. Introducción.

El funcionamiento del sistema está controlado por el microcontrolador ATmega2560 de Atmel presente en una tarjeta Arduino Mega 2560 que será encargado tanto de procesar la información que le llega desde dos microcontroladores ATmega328 presentes en tarjetas de Arduino Nano (N1 y N2), como de controlar las interfaces del sistema.

Además, el Arduino Mega será el encargado de seleccionar la escala correspondiente en el circuito de tensión mediante el multiplexor*74HC4051.* Los microcontroladores ATmega328 son los responsables del muestreo de las señales de tensión y corriente acondicionadas en los circuitos descritos en el Capítulo anterior.

Los Arduinos Nanos serán capaces, mediante el muestreo de señal, de aportar los valores instantáneos de la corriente y la tensión, permitiendo calcular los valores eficaces de intensidad (I), voltaje (V) y potencia (P). La resolución, por métodos numéricos, de sus correspondientes integrales permite calcular los valores medios y eficaces de I, V y P. En la *Figura V.1* se observa el diagrama general de bloques del sistema.

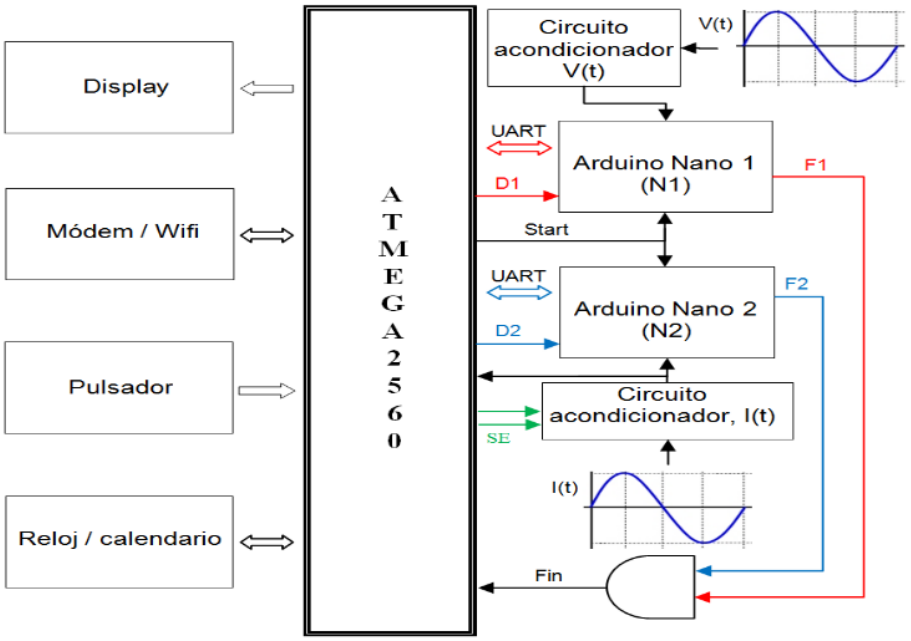


Figura V.1. Diagrama de bloques general.

Tal como se muestra en el diagrama de bloques, todo el sistema gira en torno al microcontrolador ATmega2560, al que denominaremos Maestro y será encargado de:

- Controlar el funcionamiento de los dos ATmega328 interpretando y procesando la información recibida desde ambos microcontroladores para obtener los datos requeridos por el usuario.
- Seleccionar la escala de corriente más adecuada.
- Mantener actualizada la información que se muestra en el display y atendiendo los requerimientos realizados por el usuario a través de los pulsadores.
- Gestionar el módem GSM y el acceso a Internet a través del módulo wifi.

En todo momento, las señales de tensión e intensidad estarán presentes en las entradas de los CAD de sus respectivos ATmega328 (N1 y N2). El Maestro emplea una salida serial (Start, *Figura V.1)* para ordenar el comienzo de la digitalización.

Cuando ésta ha finalizado, cada microcontrolador por separado lo indicará colocando en alta las salidas de fin de conversión F1 y F2, respectivamente (*Figura V.1*). Los datos digitales así obtenidos serán transferidos al Maestro cuando éste los solicite mediante las líneas D1 y D2. La *Figura V.2* muestra un cronograma simplificado de los eventos que tienen lugar para completar un ciclo de medida y donde se indica que cadena de eventos se inicia en cada uno de esos instantes.

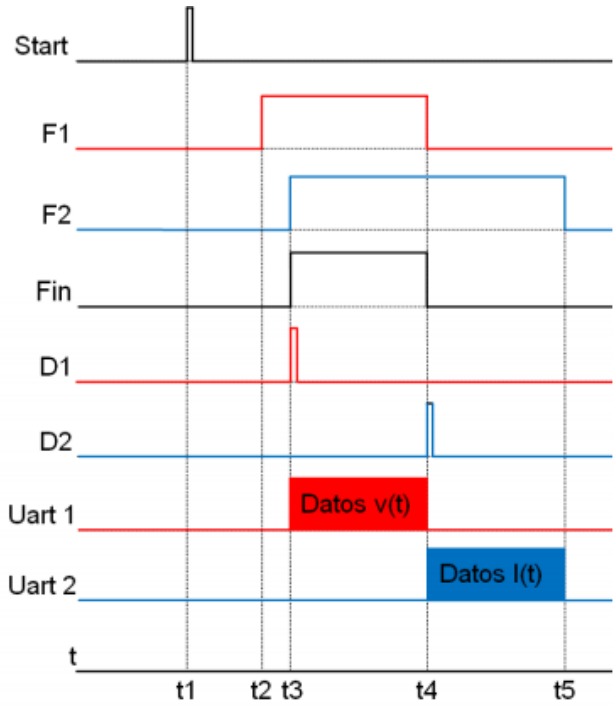


Figura V.2. Cronograma del muestreo.

Así, al principio de cada ciclo, las señales Start, F1, F2, Fin, D1 y D2, estarán a nivel bajo. En el instante t1 el Maestro genera un pulso que hará que los microcontroladores N1 y N2 comiencen a digitalizar las señales V(t) e I(t), respectivamente (Línea Start, *Figuras V.1 y V.2*).

Teniendo en cuenta que, en general, la forma de las señales anteriores será diferente y estarán desfasadas (salvo para cargas puramente resistivas) el tiempo para su digitalización podría variar. En el ejemplo de la *Figura V.2*, N1 finaliza dicha operación en el instante t2 y lo indica colocando en alta la línea F1. Algo más tarde (en t3) hace lo propio N2 y coloca su línea

de fin de conversión F2 en alta, haciendo que la salida de la puerta AND pase al mismo estado (línea Fin) para que el Maestro interprete que las señales V(t) e I(t) han sido digitalizadas.

Tras este primer paso, es necesario que los datos anteriores sean enviados al Maestro para su procesado, para lo cual éste envía un pulso a N1 a través de D1 para que se los remita vía UART. En el instante t4, cuando la transferencia ha finalizado, N1 pone a baja su línea F1 (pasando Fin al mismo estado) y repitiéndose el mismo proceso para N2 al recibir por su línea D2, procedente del Maestro, un nuevo pulso por el que le solicita el envío de sus datos.

Una vez finalizada la transferencia (t5, *Figura V.2*), N2 pasa F2 a baja y el sistema queda preparado para realizar el siguiente ciclo de medida. Con los datos muestreados de V(t) e I(t), el Maestro podrá realizar operaciones encaminadas a obtener los valores medios de las magnitudes anteriores y de la potencia, como media algebraica de sus valores instantáneos durante un periodo según se recoge en la ecuación (5) o sus valores eficaces, como la media cuadrática de los valores instantáneos durante un periodo completo según (6).

Para determinar la potencia tendremos en cuenta lo indicado en la expresión (7). Los resultados anteriores se podrán utilizar para obtener los factores de forma y amplitud.

$$Y_{med} = \frac{1}{T}\int_0^T y(t)dt \qquad (5)$$

$$Y_{ef} = \sqrt{\frac{1}{T}\int_0^T [y(t)]^2 dt} \qquad (6)$$

$$P(t) = y(t) = v(t)i(t) \qquad (7)$$

V.2. Software Arduino Mega.

En este Arduino podemos diferenciar claramente dos sub-procesos diferentes. El primero corresponde al *funcionamiento normal* y el segundo a la *solicitud los datos* a los Esclavos (*Figura V.3*).

En el primero de los procesos, el Maestro realiza las tareas de procesamiento de datos, interactúa con los sistemas hardware de salida instalados en el dispositivo, solicita a los Esclavos la toma de datos y, por último, realiza el cambio de escalas aumentando o disminuyendo la amplitud de la señales a digitalizar.

Por otro lado, el segundo proceso corresponde a la recepción de datos. Mediante una señal digital los Esclavos avisan que ya tomaron la información solicitada y, llegado ese momento, el Maestro sale de la ejecución del *programa normal* atendiendo única y exclusivamente a la recepción de los datos, primero los de corriente y luego los de tensión. Una vez terminado este proceso este volverá a la ejecución normal del programa.

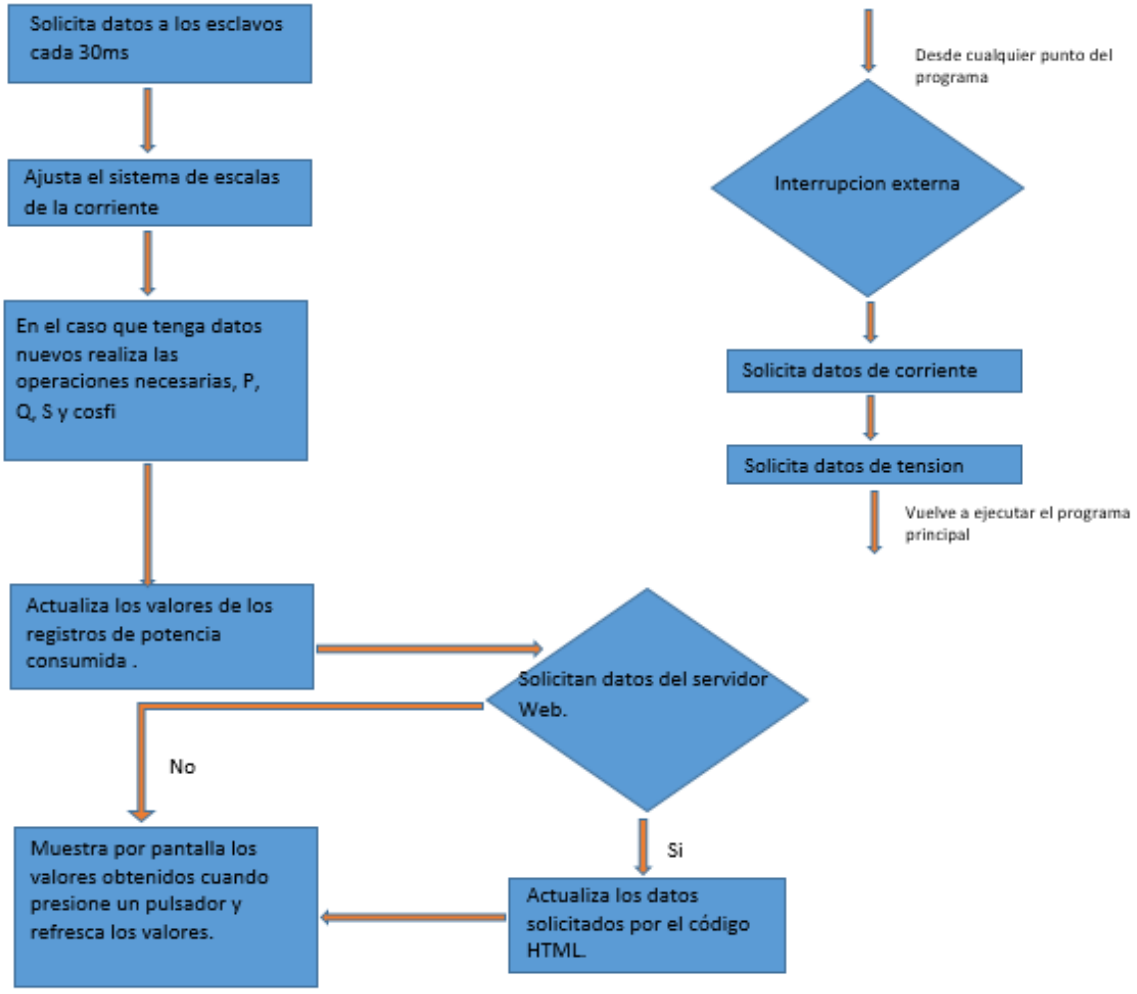


Figura V.3. Proceso Arduino Mega.

V.3. Software Arduino Nano.

El programa instalado en ambos Arduinos Nano es idéntico, lo que significa que teóricamente invierten el mismo tiempo en su ejecución. Para que ello tenga lugar, han de recibir un pulso de Start *(Figura V.4*) proveniente del Maestro indicándoles el inicio de la toma de datos.

Una vez ocurrido lo anterior, ambos Esclavos comienzan dicho proceso con una frecuencia de 10k muestras/segundo durante un periodo de 20 milisegundos, tomando un total de 200 muestras. Transcurrido este tiempo y tomadas las muestras por ambos esclavos, se envía una señal digital al Maestro indicado que se ha finalizado.

Seguidamente, cuando el Maestro lo considere oportuno solicitará, y por ese orden, los datos de corriente y tensión a sus respectivos Esclavos, quedando disponibles para el siguiente proceso de captura.

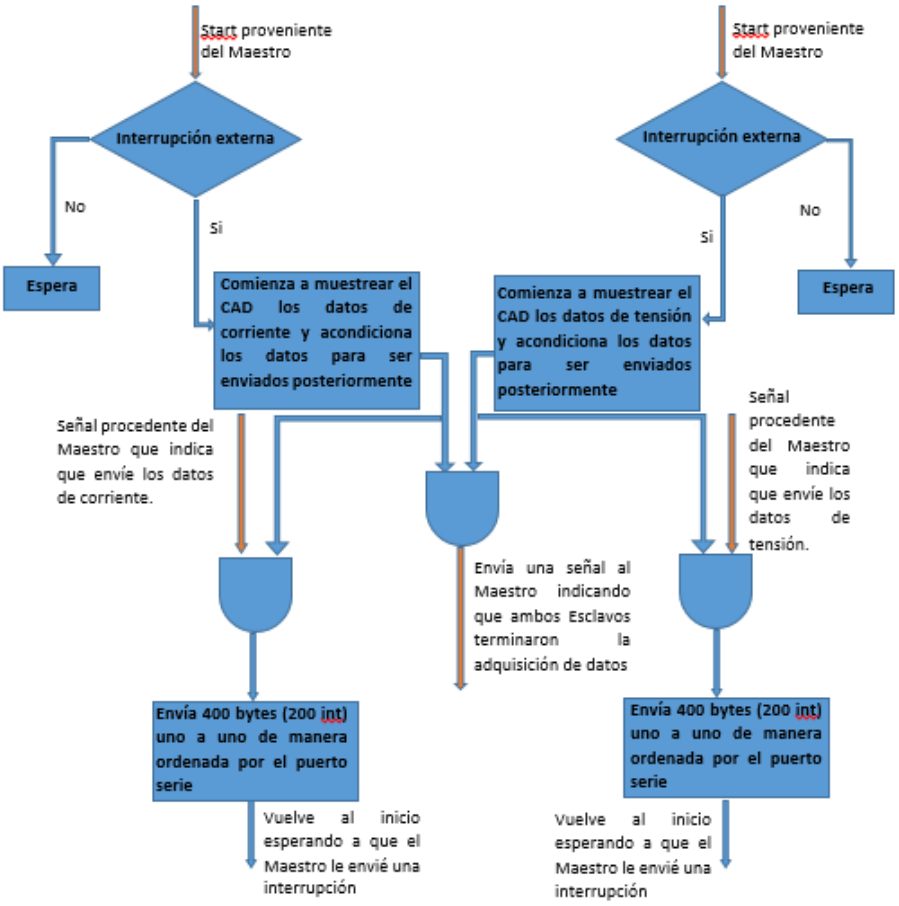


Figura V.4. Proceso Arduinos Nanos.

V.4. Entorno Arduino.

La programación de Arduino está caracterizada por el uso de un Entorno de Desarrollo Integrado (IDE) [20]. Se trata de un conjunto de herramientas software que permite a los programadores poder desarrollar sus propios programas con comodidad.

En el caso de Arduino el programa (también llamado “sketch”), permite comprobar la ausencia de errores y grabar la memoria del microcontrolador para que éste se convierta a partir de entonces en el ejecutor autónomo de dicho programa. La *Figura V.5* muestra el aspecto del entorno IDE perteneciente a Arduino.

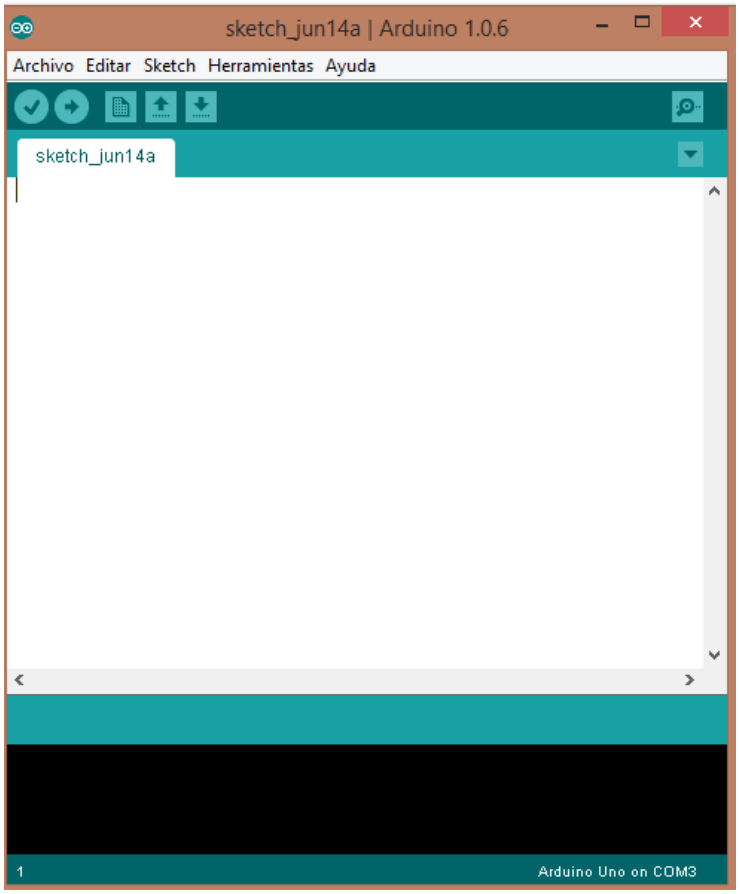


Figura V.5. IDE de Arduino.

Se puede observar que la ventada del IDE se divide en cinco grandes áreas. De arriba hacia abajo éstas son: la barra de menús y de botones, el editor de código propiamente dicho, la barra y consola de mensajes, y la barra de estado.

La zona del IDE donde se realizará la mayor parte del trabajo se corresponde con el editor de código, ya que es allí donde se escribirán los programas. Otra zona que se utiliza con frecuencia es la de botones, compuesta por los siguientes elementos:

Verify: este botón realiza dos funciones: comprueba que no haya ningún error en el código del programa y, si es correcto, se procede a compilar.

Upload: se pulsa este botón inmediatamente después de verify. Su función es ejecutar internamente el comando “avrdude” que permite cargar en la memoria de la placa Arduino el programa recientemente verificado y compilado.

New: Crea un nuevo sketch vacío.

Open: Desplega un menú con programas disponibles para abrir, tanto de realización propia como skecthes de ejemplo.

Save: Guarda el código del programa en un fichero.

Serial Monitor: Es una ventana IDE que permite enviar y recibir datos textuales a la placa Arduino mediante conexión serie.

El lenjuaje utilizado en la programación de skecth es el C++. Este lenguaje de alto nivel permite una mayor facilidad en el desarrollo del código ya que no es necesaria la programación en lenguaje ensamblador o de bajo nivel.

La mayor parte de los módulos compatibles con Arduino utilizan librerías específicas que contienen instrucciones muy útiles para simplificar el código. Es necesaria la importación de dichas librerías dentro del entorno IDE para proceder a su utilización. Asimismo, la barra de menú nos ofrece cinco entradas principales como se muestra en la *Figura V.6.*

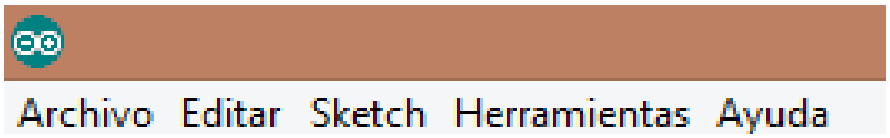


Figura V.6. IDE de Arduino.

La entrada de menú muestra acciones adicionales que complementan a las que se pueden realizar con los botones.

- Archivo: Además de ofrecer acciones estándar como crear un nuevo skecth, abrir uno existente, guardarlo, cerrarlo, cerrar el IDE en sí, etc; realiza también otras acciones interesantes tales como acceder a ejemplos de programas, imprimir el código por impresora, utilizar un programador externo, etc.
- Editar: Posee acciones triviales tales como deshacer y rehacer, cortar, copiar, pegar, comentar, descomentar, etc.
- Sketch: Se ofrece la acción de verificar/compilar descrita anteriormente, pero también permite la acción de importar librerías o añadir nuevas pestañas a un fichero.
- Herramientas: Entre otras acciones, se puede elegir el tipo de placa Arduino que se está utilizando, especificando además cuál es su puerto serie. Asimismo permite el uso de herramientas avanzadas para hacer el código más legible.
- Ayuda: Desde este menú se puede acceder a varias secciones de la página web de Arduino que contienen diferentes artículos, tutoriales y ejemplos de ayuda.

Capítulo VI: Descripción del sistema.

CAPÍTULO VI. DESCRIPCIÓN DEL SISTEMA.

En este Capítulo se hace referencia al funcionamiento del sistema desde el punto de vista de su uso, es decir, se expone y se explica cómo debe ser su utilización por parte del usuario.

VI.1 Introducción.

La instalación del sistema resulta bastante sencilla. Debe colocarse próximo a la caja de protección de forma que se conecte directamente en la derivación individual de la vivienda. Esto es importante para poder obtener los datos de potencia totales. Para proceder a su conexión, se deben conectar los cables de fase y neutro en los orificios correspondientes.

VI.2 Puesta a punto del sistema.

Una vez el dispositivo ha sido conectado, es necesaria la configuración inicial del mismo a través del puerto USB de la placas de Arduino Mega, lo que permitirá ajustar los parámetros correspondientes al Wifi, número telefónico para mandar mensajes, ajustes de alarma de corriente, potencia o factura de la luz, etc. A partir de ese momento el sistema comenzará a almacenar todos los datos de potencia correspondientes a la vivienda.

El dispositivo posee un display que se encenderá al pulsar un botón. La pantalla principal está compuesta por los parámetros de corriente, factura de la luz y potencia activa. Al pasar 10 segundos el menú cambiará y mostrará los valores referentes a las potencias reactiva y aparente así como el factor de potencia. En las *Figuras VI.2 y VI.3* se observa el aspecto de los dos menús disponibles en el sistema anteriormente señalados.

Figura VI.2. Pantalla inicial display.

Figura VI.3. Pantalla secundaria del display.

En caso de no detectar actividad, la pantalla se apagará pasado 1 minuto. Además, se avisará al usuario mediante mensajería instantánea si en algún momento se corta el suministro eléctrico.

La instalación de la aplicación Android se realiza de manera estándar. Se permitirá la consulta de todos los valores que también aparecen en el display así como los registros de potencia pertenecientes a los últimos meses o días. El aspecto de la aplicación se muestra en la *Figura VI.4.*

Figura VI.4. Aplicación Android del sistema.

Capítulo VII: Resultados experimentales.

CAPÍTULO VII. RESULTADOS EXPERIMENTALES.

En el presente Capítulo se exponen los resultados obtenidos durante la realización de diferentes pruebas encaminadas a testear el correcto funcionamiento del sistema. Asimismo, serán evaluados aquellos aspectos referentes a la exactitud del dispositivo a la hora de obtener los valores de los parámetros a medir.

VII.1 Introducción.

El objetivo de las pruebas realizadas se centró en la evaluación del sistema mediante la comparación de valores obtenidos mediante la utilización de un vatímetro comercial marca *Metrix* modelo *PX120* con los medidos haciendo uso del sistema implementado. Las *Figura VII.1* muestra el aspecto general del sistema mientras, las *Figuras VII.2 y VII.3* muestran las placas de circuito impreso que permiten la obtención de los datos y que se encuentran instaladas en el interior del dispositivo.

Figura VII.1. Dispositivo tras su montaje.

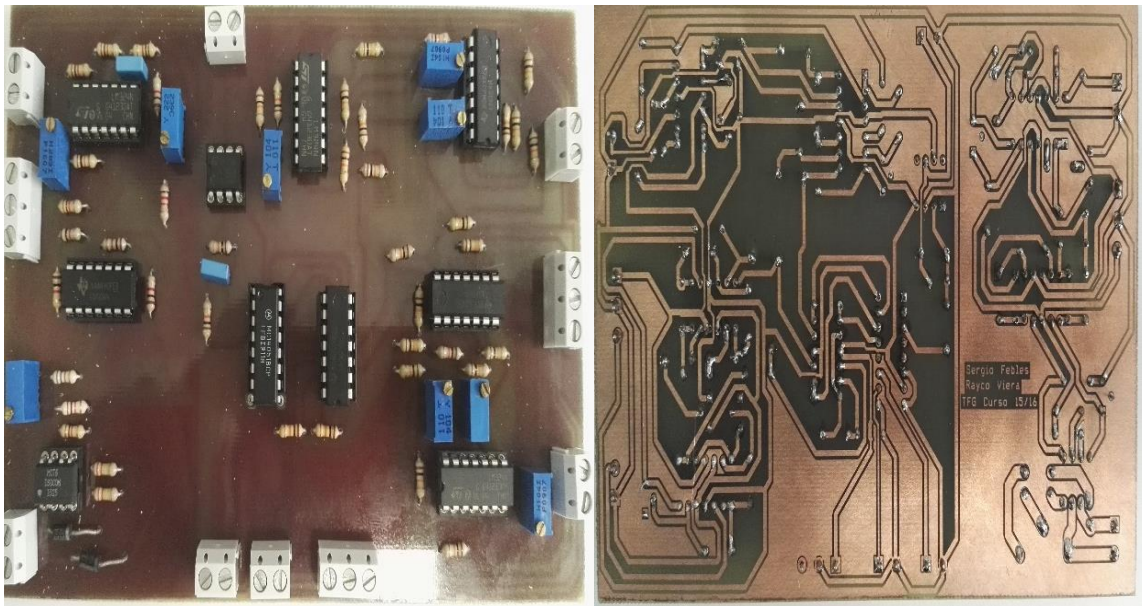


Figura VII.2. Placa con circuitos analógicos.

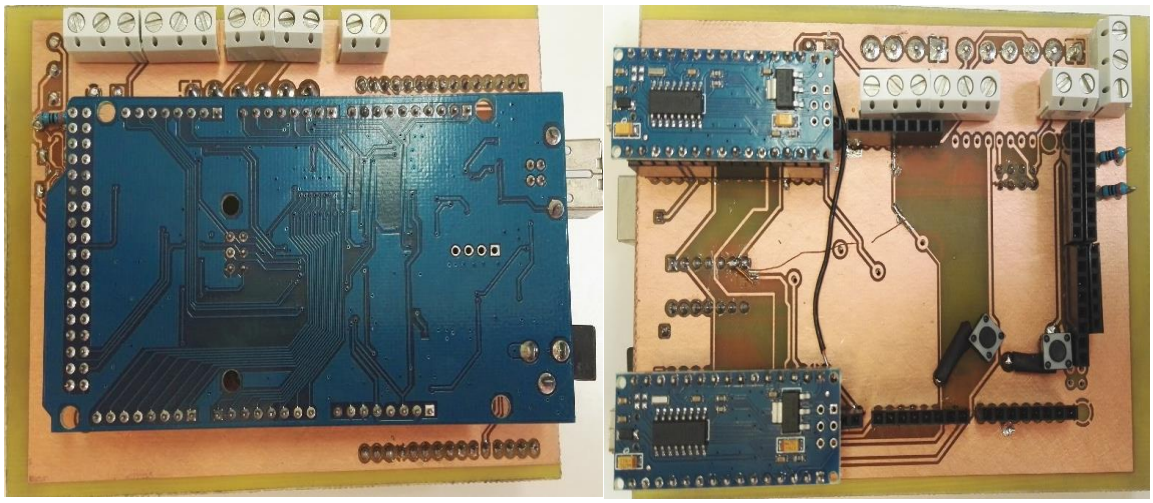


Figura VII.3. Placa con tarjetas Arduino.

VII.2. Vatímetro comercial.

Como se indicó anteriormente, para comprobar el correcto funcionamiento del sistema se realizaron medidas experimentales encaminadas a comparar los valores obtenidos mediante un vatímetro comercial con los correspondientes al sistema implementado. Para ello, fueron utilizadas cargas de diferente impedancia para obtener distintos consumos de corriente, de forma que se pudieran evaluar adecuadamente cada una de las escalas de intensidad utilizadas Las *Figuras VII.4 y VII.5* muestran, respectivamente, el aspecto real de señales proporcionales a la tensión y corriente correspondientes a una carga resistiva y a otra de tipo inductivo, que serán llevadas a los ATmega328 (Arduinos Nano, N1 y N2) para la obtención de magnitudes como intensidad de corriente, potencia, factor de potencia, etc.

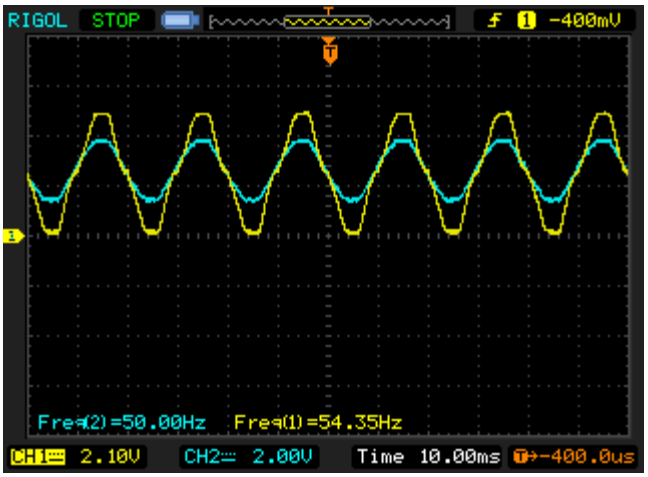


Figura VII.4. Señales de corriente (azul) y tensión (amarillo) para una carga resistiva.

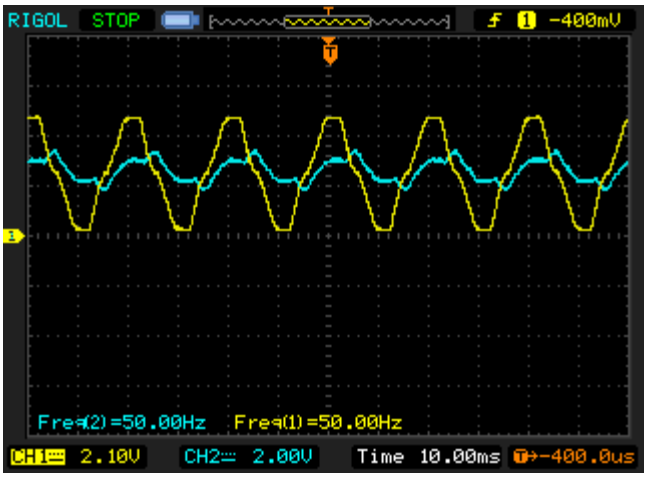


Figura VII.5. Señales de corriente (azul) y tensión (amarillo) para una carga inductiva.

Si bien la tensión permanece siempre constante (sería la de la red, señal amarilla en las *Figuras VII.4 y VII.5*), no sucede los mismo con la corriente que cambia tanto en amplitud, forma y desfase respecto a la tensión. De ahí, que ha sido necesario hacer uso de diferentes sensores y escalas.

- Escala 1:

Es la más sensible, emplea el sensor inductivo HWCT004, y se activa de forma automática cuando el Arduino Mega 2560 detecta consumos de corriente inferiores a 0,5 A. A modo de ejemplo, y para una carga de tipo inductiva, los resultados obtenidos por el vatímetro se observan en la *Figura VII.6,* mientras que los de la *VII.7* corresponden al sistema desarrollado.

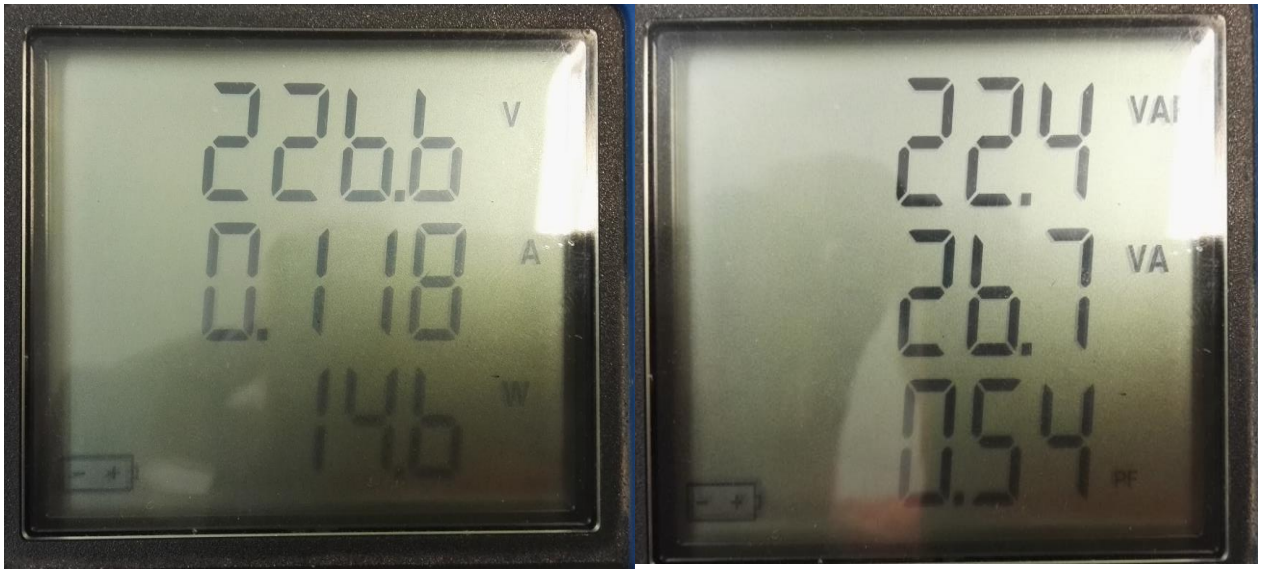


Figura VII.6. Resultados del vatímetro para escala 1.

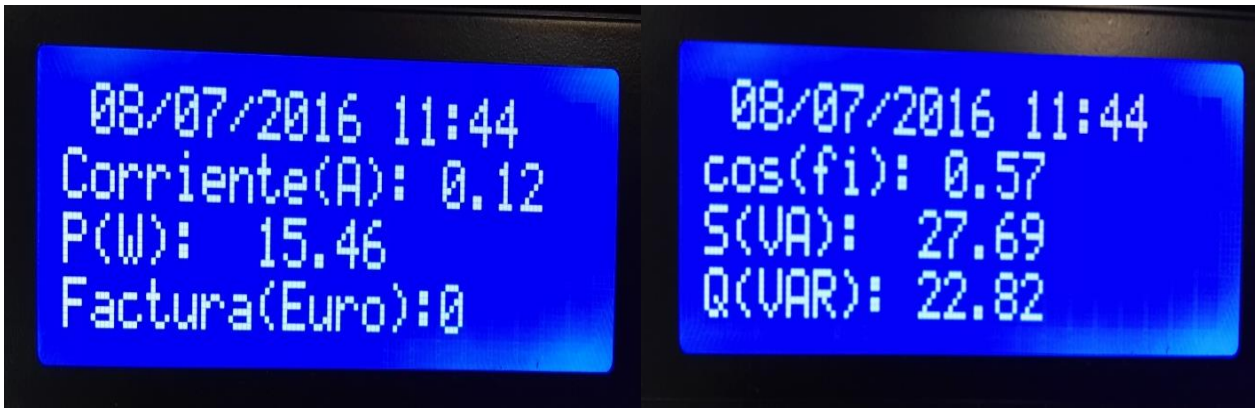


Figura VII.7. Resultados del sistema para escala 1.

- Escala 2:

Es una escala intermedia, que utilizando nuevamente el sensor inductivo HWCT004, se emplea para cargas que generen consumos de corriente comprendidos entre 0,5 y 3,3 A. En las *Figuras VII.8* y *VII.9* se observan, respectivamente, los resultados obtenidos por el vatímetro comercial y por el sistema implementado para una carga de tipo resistivo.

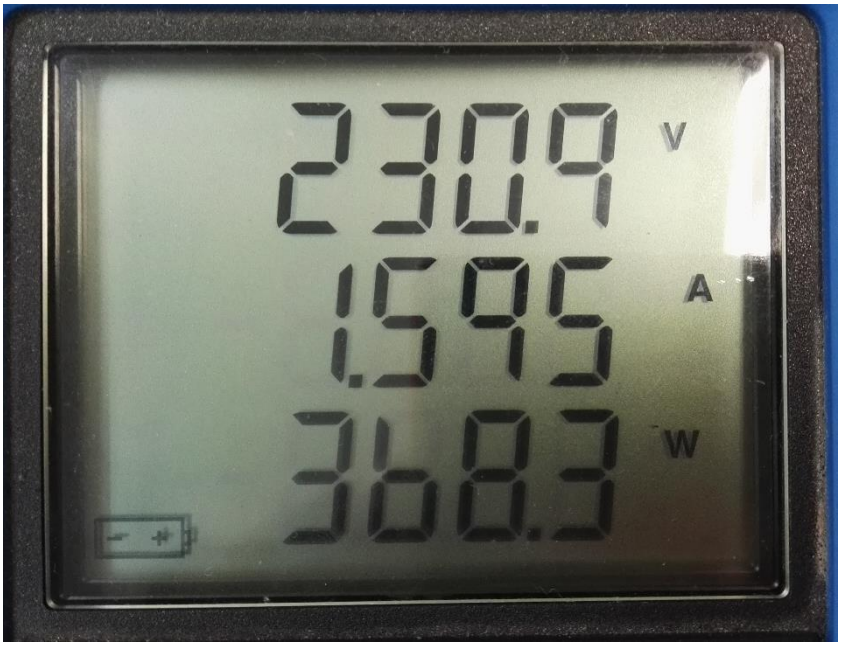


Figura VII.8. Resultados del vatímetro para escala 2.

Figura VII.9. Resultados del sistema para escala 2.

- Escala 3:

Por último, utilizando el sensor de Efecto Hall ACS712, se ha procedido a conectar cargas que generen valores intensidad de corriente superiores a 3,3 A. Las *Figuras VII.10* y *VII.11* muestran un ejemplo de los resultados obtenidos en la presenta escala.

De igual manera a lo indicado en la descripción de la Escala 1, la utilización de las Escalas 2 y 3 es decidida por el Arduino Mega 2560 en virtud de la corriente que circula en cada momento por la carga.

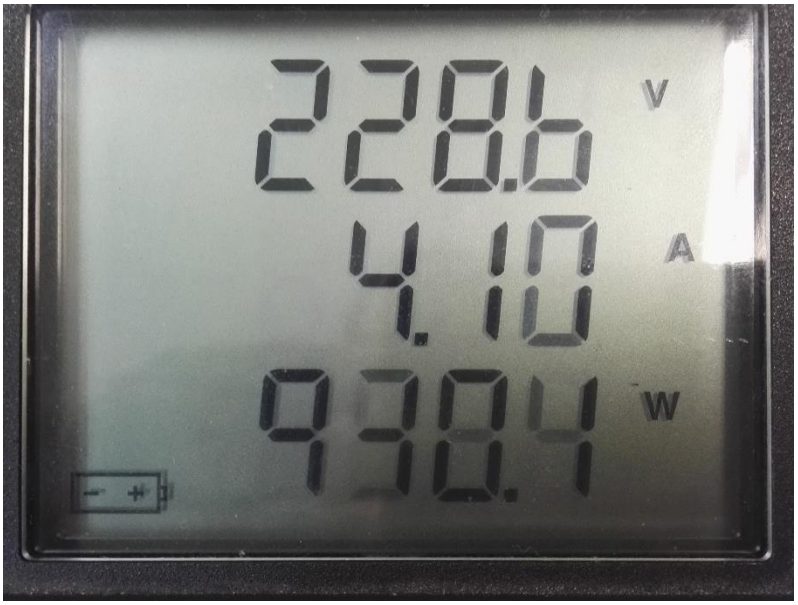


Figura VII.10. Resultados del vatímetro para escala 3.

Figura VII.11. Resultados del sistema para escala 3.

VII.3 Resultados.

Tras múltiples pruebas realizadas, se han podido obtener valores de las diferencias máximas que se han podido determinar entre los valores medidos con el diseño realizado en este trabajo y un vatímetro comercial, lo que permite una evaluación global sobre la fiabilidad del dispositivo. Las *Tablas VII.1 y VII.2* muestran la diferencia (en %) para cada una de las escalas y magnitudes medidas.

Diferencia	% P (KW)	% S (VA)	% I(Amperios)
Escala 1	1.79	5.01	0
Escala 2	1.76	6.29	1.61
Escala 3	2.17	2.67	1.88

Tabla VII.1. Diferencia para cada escala y magnitud con carga resistiva.

Diferencia	% P (KW)	% Q (KVAR)	% S (VA)	% I(Amperios)
Escala 1	6	10	7.78	8.6
Escala 2	1	4.2	2.3	5.3

Tabla VII.2. Diferencia para cada escala y magnitud con carga inductiva.

Con los datos mostrados en la tabla anterior, se puede concluir en el correcto funcionamiento del sistema ya que muestra una diferencia media de 4%.

Capítulo VIII: Presupuesto.

Capítulo VIII. Presupuesto.

En este Capítulo se detallará el presupuesto del proyecto. Para ello se ha dividido en gasto de materiales, mano de obra y coste de ejecución. Se debe tener en cuenta que los precios de los componentes son orientativos debido a que éstos están en continuo cambio.

VIII.1 Coste de materiales.

DESCRIPCIÓN	CANTIDAD	COSTE UNITARIO (€/U)	COSTE TOTAL (€)
ARDUINO NANO	2	4,5	9
ARDUINO MEGA	1	12	12
MODEM GSM	1	14	14
MÓDULO WIFI	1	4,5	4,5
PULSADOR	2	0,05	0,1
PANTALLA LCD	1	10	10
RELOJ TIEMP REAL	1	2,5	2,5
CONECTORES	3	0,55	1,65
RESISTENCIAS	50	0,036	1,8
CONDENSADORES	5	0,25	1,25
INTERRUPTOR	1	2,05	2,05
PLACA PCB	1	7,2	7,2
DIODO 1N4007	2	0,1	0,2
TIRA DE CONECTORES	5	0,55	2,75
RELÉ	1	1,1	1,1
SENSOR EFECTO HALL	1	3	3
TOROIDE	1	5	5
MULTIPLEXOR	1	2,25	2,25
LM741	1	0,25	0,25
LM324	7	0,5	3,5
MCT6	1	3,7	3,7
BATERÍA	1	15	15
TOTAL COSTE MATERIALES			102,8

VIII.2 Coste de mano de obra.

CONCEPTO	CANTIDAD (H)	COSTE UNITARIO(€/U)	COSTE TOTAL (€)
TIEMPO DE ANÁLISIS	120	60,5	7260
TIEMPO DE CODIFICACIÓN	300	82,5	24750
TIEMPO DE IMPLEMENTACIÓN	80	25	2000
TIEMPO DE DOCUMENTACIÓN	40	30	1200
TOTAL COSTES MANO DE OBRA			35210

VIII.3 Coste total del proyecto.

COSTES TOTAL (€)	
TOTAL COSTES MATERIALES (MT)	102,8
TOTAL COSTES MANO DE OBRA (MO)	35210
GASTOS GENERALES: 6% (MT+MO)	2118,77
BENEFICIO INDUSTRIAL: 13% (MT+MO)	4590,66
COSTE TOTAL DEL PROYECTO	42022,23

Aportaciones y conclusiones.

CONCLUSIONES.

- Se ha diseñado e implementado un vatímetro que hace uso de circuitos acondicionadores de señal para la obtención de la tensión V(t) e intensidad I(t) proporcionales a la potencia a medir.
- Para la medida de la corriente se recurrió a sensores de tipo inductivo y de Efecto Hall, siendo innecesarios para el caso de la tensión.
- El vatímetro es gestionado mediante un microcontrolador ATmega2560 que, entre otros cometidos, supervisa el muestreo de las señales V(t) e I(t) mediante sendos microcontroladores ATmega328 para obtener sus valores instantáneos. Ello ha posibilitado:
- La resolución, por métodos numéricos, de las integrales necesarias para el cálculo de los valores medios y eficaces de I (A) y P(KW).
- Determinar el factor de potencia (cosγ), Q (KVAR) y S(VA).
- En todo momento, el usuario dispone de un display de 4x20 caracteres como interfaz gráfico donde visualizar y seleccionar opciones de medida.
- Las prestaciones del dispositivo fueron incrementadas con objeto de ser utilizado como un sistema capaz de gestionar íntegramente toda la información relativa a la potencia consumida en una vivienda. Así, el usuario podrá:
- Conocer, entre otros aspectos, la energía consumida durante un día, mes o año con solo acceder a la opción del menú que le permite visualizarlo en el display.
- Disponer, desde cualquier parte del mundo, de toda la información que el sistema es capaz de suministrar. Para ello, éste fue dotado de el módulo de WIFI ESP8266 que lo conecta a Internet y desde donde se podrá acceder mediante su correspondiente URL.
- Ser informado, mediante mensajería SMS, cuando se produce un corte del suministro de energía eléctrica en la vivienda. Para tal fin, se recurrió a la utilización de un modem GSM modelo SIM800L.

POSIBLES MEJORAS.

Respecto a la aplicación final obtenida, ésta podría ser mejorada en aspectos tales como:

- Mejorar el dispositivo haciendo uso de tecnología SMD. Ello daría lugar a una disminución de las dimensiones y peso del dispositivo final, facilitando con ello una mejor instalación.
- Optimización del software empleado para la interpretación de las señales. Esto daría lugar a una mayor fiabilidad del sistema.
- Mejorar la aplicación Android introduciendo más opciones (avisos, configuraciones a distancia, etc).
- Sustitución de la pantalla LCD, por pantalla táctil que permita la representación de las señales en tiempo real.
- Ampliación de menús de la pantalla.

Bibliografía.

BIBLIOGRAFÍA.

[1] http://sobrelabombilla.blogspot.com.es/2010/09/creacion-y-evolucion.html.

[2] https://es.wikipedia.org/wiki/Guerra_de_las_corrientes

[3] Libro “Arduino. Curso práctico de formación “ de Óscar Torrente Artero.

[4] http://www.atmel.com/

[5] http://www.atmel.com/devices/atmega2560.aspx

[6] http://www.atmel.com/devices/atmega328.aspx

[7] http://www.luisllamas.es/2014/09/leer-un-pulsador-con-arduino/

[8] https://es.wikipedia.org/wiki/Pantalla_de_cristal_l%C3%ADquido

[9] http://www.prometec.net/relojes-rtc/

[10] https://www.arduino.cc/en/Main/ArduinoGSMShield

[11] http://www.prometec.net/esp8266/

[12] https://es.wikipedia.org/wiki/Potencia_el%C3%A9ctrica

[13] https://es.scribd.com/doc/232735097/Circuito-Detector-de-Cruce-Por-Cero

[14] https://es.wikipedia.org/wiki/Optoacoplador

[15] https://www.fairchildsemi.com/products/optoelectronics/phototransistor-optocouplers/phototransistor-output-dc-sensing-input/MCT6.html

[16] http://www.ti.com/product/LM324

[17] http://saber.patagoniatec.com/sensor-de-corriente-acs712-5-20-30a-ptec-arduino-pic/

[18] https://wholesaler.alibaba.com/product-detail/HWCT004-AC-Micro-Current-Detection-Module_1991577798.html?spm=a2700.7724857.0.0.GpWdrX

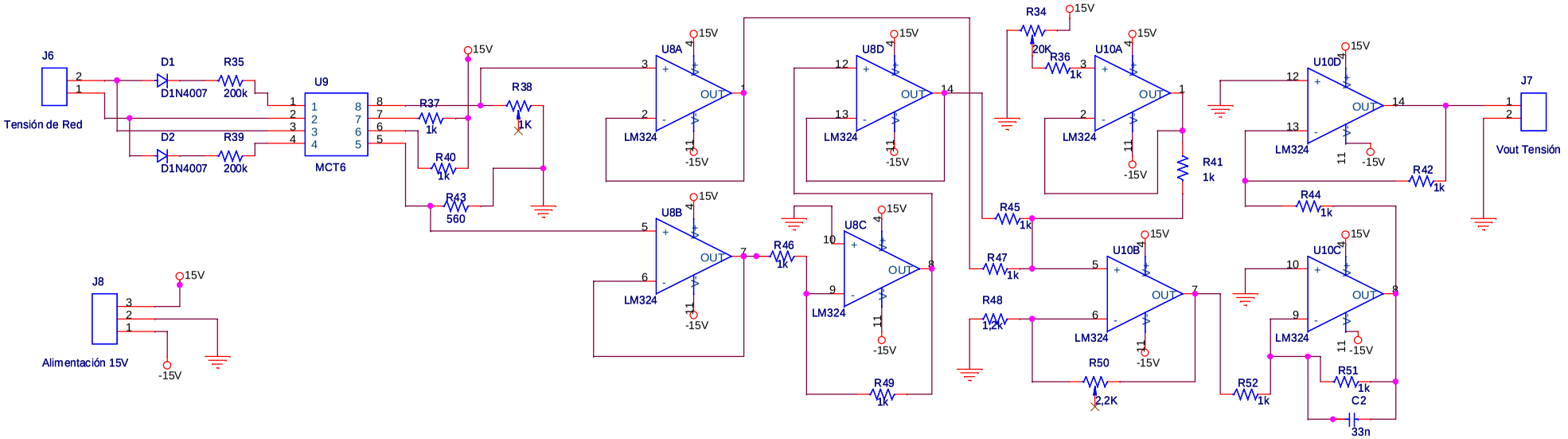
[19] http://www.datasheetcatalog.com/datasheets_pdf/7/4/H/C/74HC4051.shtml

[20] https://openwebinars.net/tutorial-arduino-ide-arduin

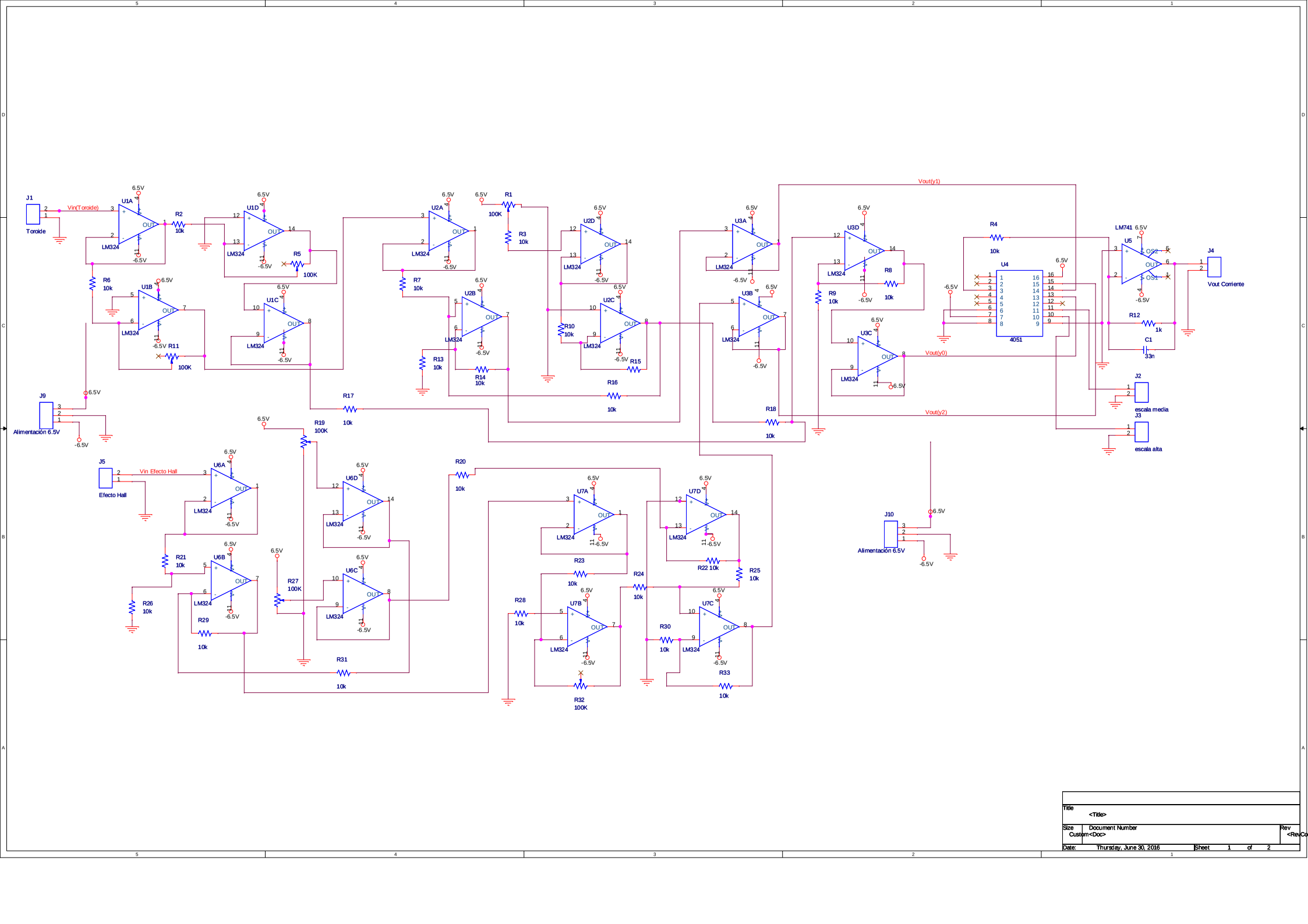
Anexo 1. Esquemáticos.

Anexo 1. Esquemáticos.

En este anexo, se exponen los esquemas electrónicos de los diferentes circuitos diseñados durante la realización del proyecto.



Title	<Title>	
Size: Custom	Document Number: <Doc>	Rev: <RevCode>
Date: Wednesday, June 29, 2016	Sheet 1 of 2	

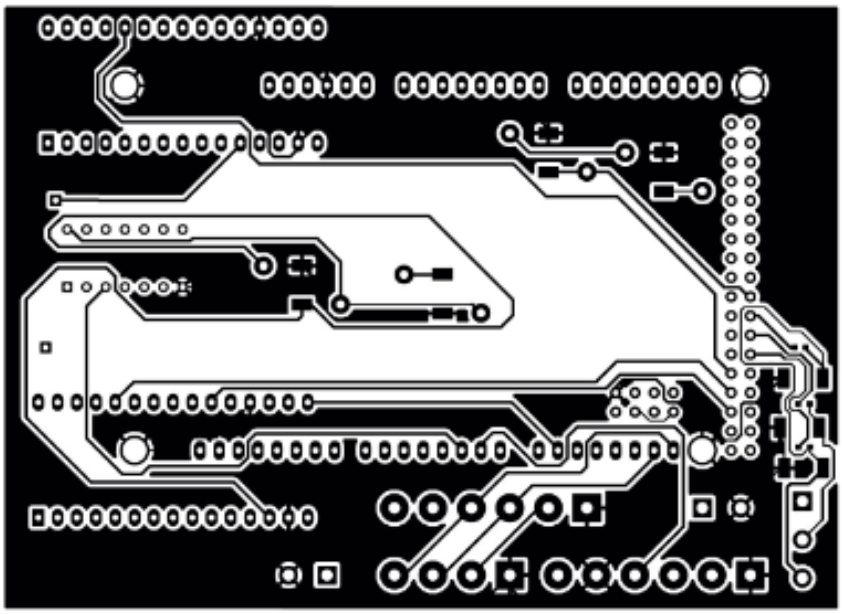

J1
Toroide
Vin(Toroide)
U1A
U1B
U1C
U1D
LM324
R2
10k
R5
100K
R6
10k
R11
100K
6.5V
-6.5V
U2A
U2B
U2C
U2D
R1
100K
R3
10k
R7
10k
R10
10k
R13
10k
R14
10k
R15
R16
10k
U3A
U3B
U3C
U3D
R8
10k
R9
10k
R17
10k
R18
10k
Vout(y1)
Vout(y0)
Vout(y2)
R4
10k
U4
4051
U5
LM741
R12
1k
C1
33n
J4
Vout Corriente
J2
escala media
J3
escala alta
J9
Alimentación 6.5V
J5
Efecto Hall
Vin Efecto Hall
U6A
U6B
U6C
U6D
R19
100K
R20
10k
R21
10k
R26
10k
R27
100K
R29
10k
R31
10k
U7A
U7B
U7C
U7D
R22 10k
R23
10k
R24
10k
R25
10k
R28
10k
R30
10k
R32
100K
R33
10k
J10
Alimentación 6.5V
Title
<Title>
Size
Custom<Doc>
Document Number
<Doc>
Rev
<RevCo
Date:
Thursday, June 30, 2016
Sheet
1
of
2

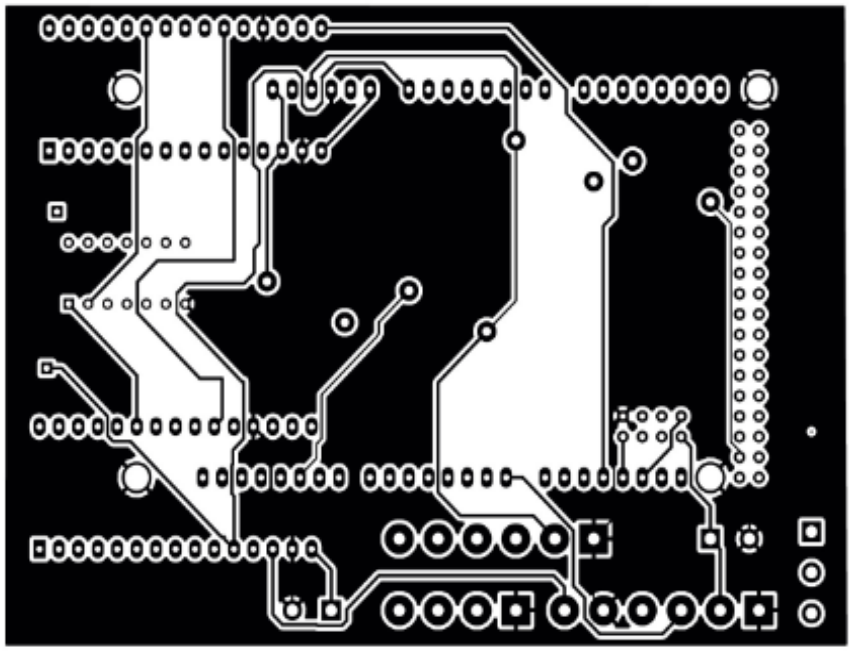
Anexo 2. Fotolitos

Anexo 2. Esquemáticos.

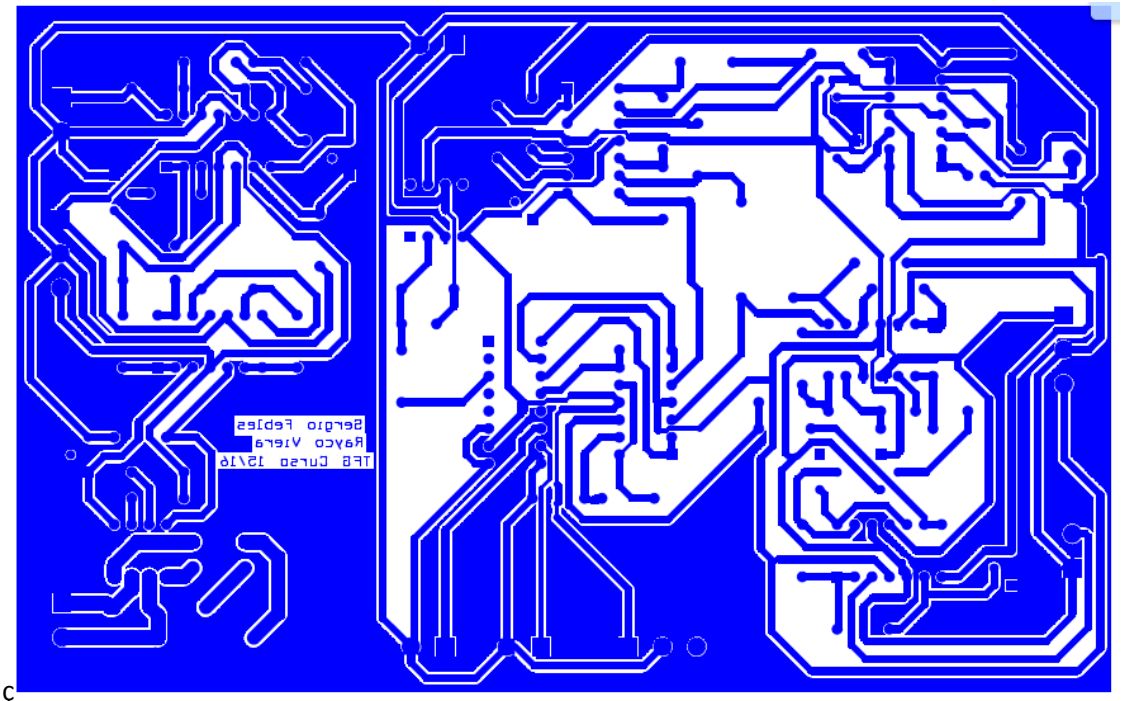
A continuación, se adjuntan los fotolitos desarrollados para obtener las placas de circuito impreso que se han realizado en el proyecto. La primera muestra la Capa Top y Bottom de la placa impresa que contiene los microcontroladores y sus unidades de entrada/ salida asociadas mientras, la segunda, se corresponde con la Capa Bottom de los circuitos acondicionares.

Capas Bottom y Top de microcontroladores.





Capa Bottom circuitos acondicionadores.



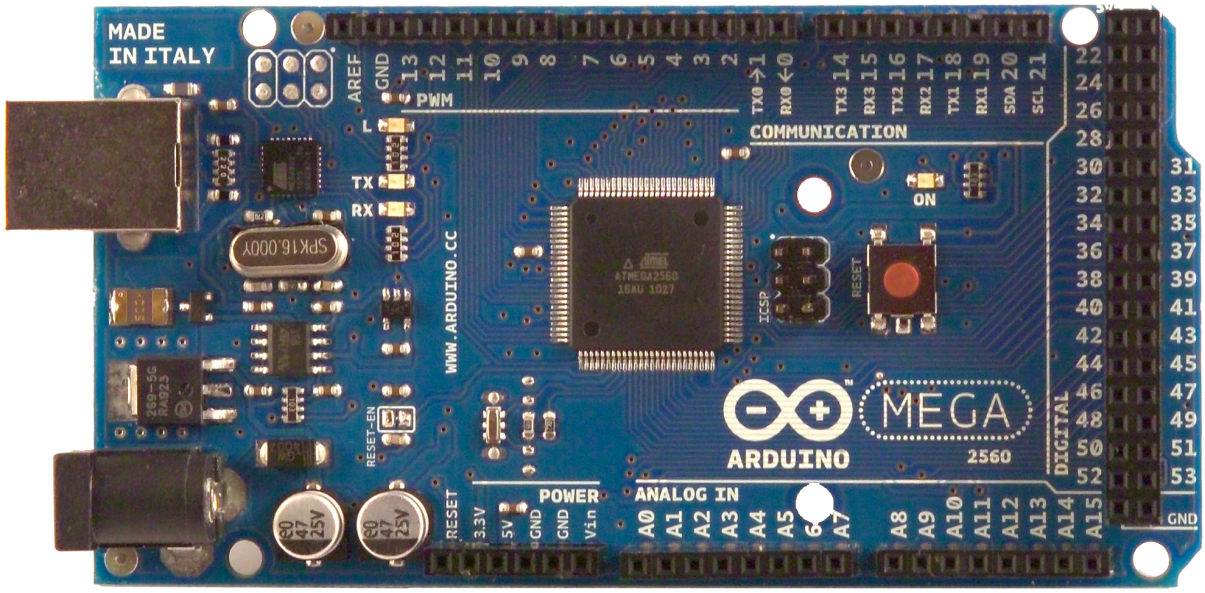
Ç

Anexo 3. Datasheet.

Anexo 3. Datasheet.

En este documento, se exponen los datasheet de aquellos componentes más relevantes utilizados en el desarrollo del sistema.

Arduino MEGA 2560



Product Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Index

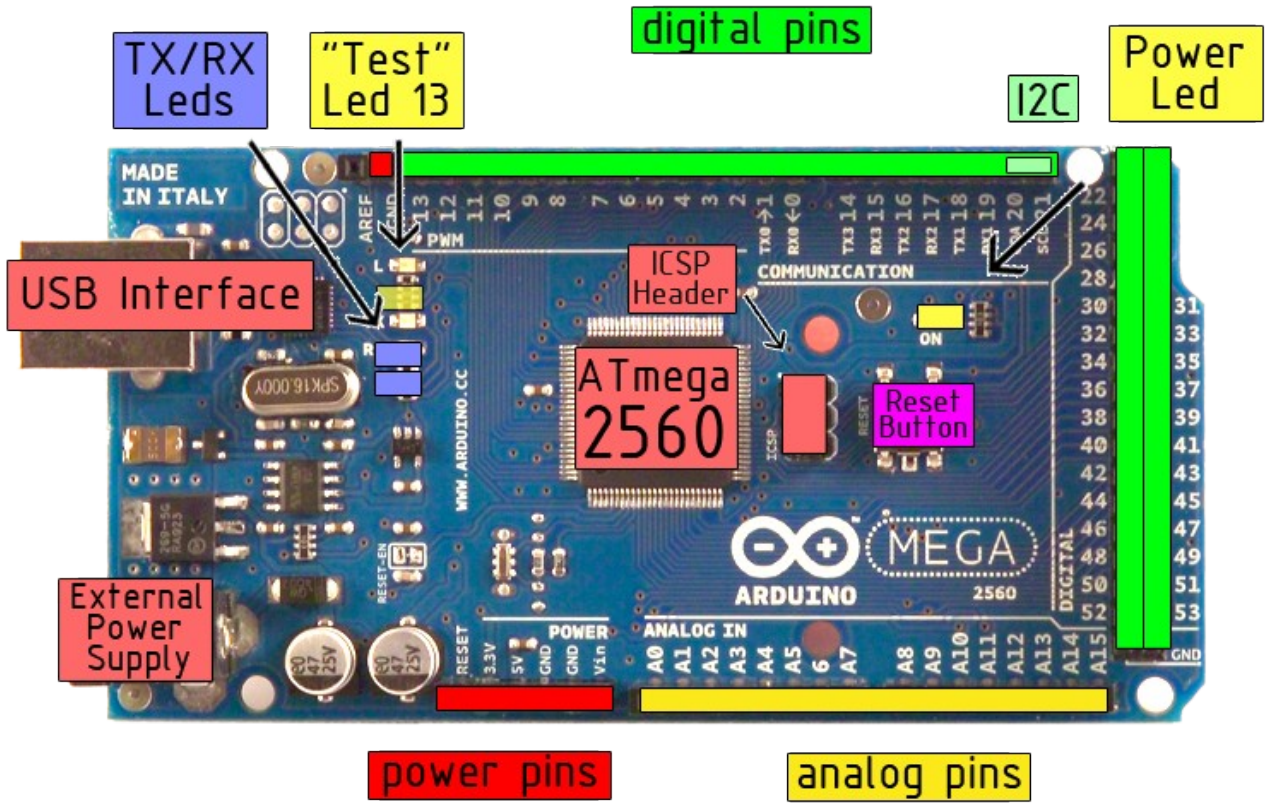
Technical Specification

EAGLE files: arduino-mega2560-reference-design.zip Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

the board



Power

The Arduino Mega2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **I^2C: 20 (SDA) and 21 (SCL).** Support I^2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I^2C pins on the Duemilanove.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega's digital pins.

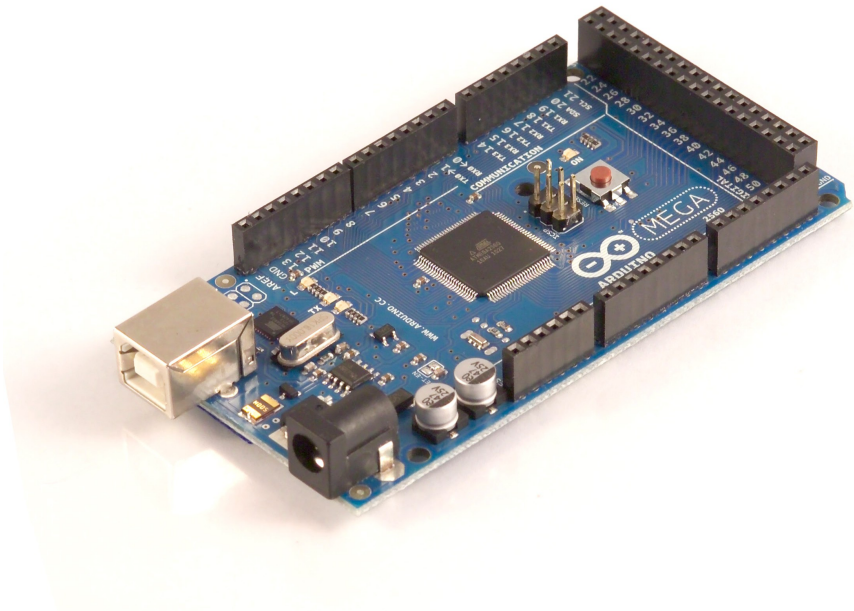
The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the ATmega2560 datasheet.

Programming

The Arduino Mega2560 can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Atmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.



Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

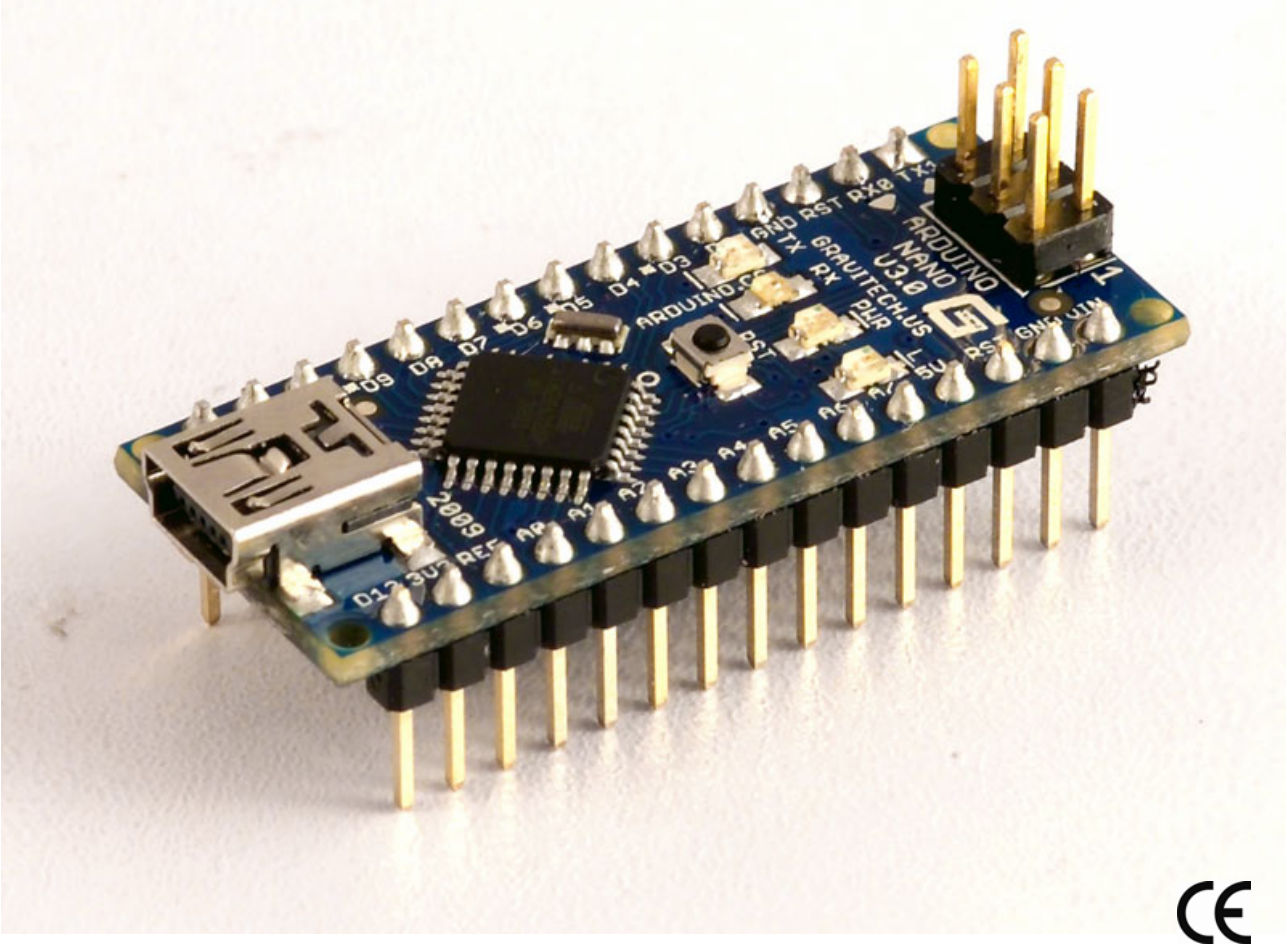
The Arduino Mega has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega is designed to be compatible with most shields designed for the Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega and Duemilanove / Diecimila. **Please note that I^2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).**

Arduino Nano



Product Overview

The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328 (Arduino Nano 3.0) or ATmega168 (Arduino Nano 2.x). It has more or less the same functionality of the Arduino Duemilanove, but in a different package. It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one. The Nano was designed and is being produced by Gravitech.

Index

Technical Specification

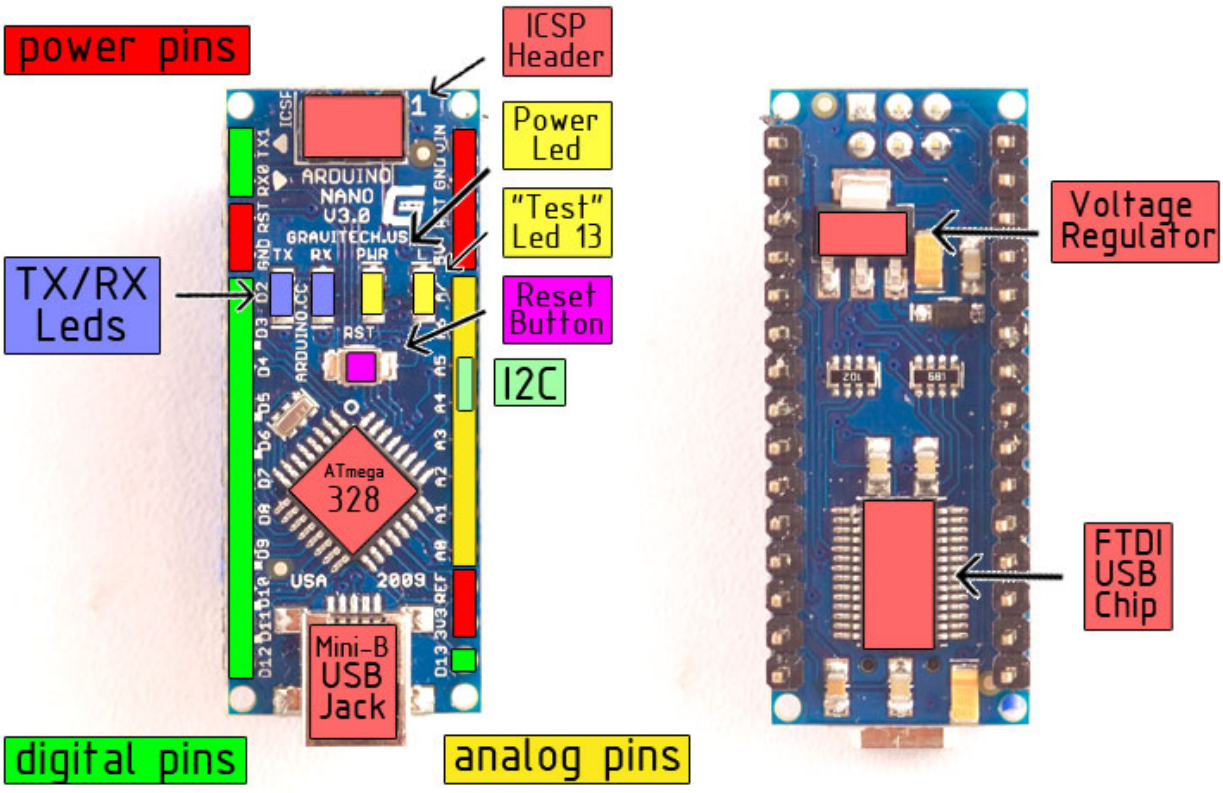
Arduino Nano 3.0 (ATmega328): schematic, Eagle files.

Arduino Nano 2.3 (ATmega168): manual (pdf), Eagle files. *Note:* since the free version of Eagle does not handle more than 2 layers, and this version of the Nano is 4 layers, it is published here unrouted, so users can open and use it in the free version of Eagle.

Summary

Microcontroller	Atmel ATmega168 or ATmega328
Operating Voltage (logic level)	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (ATmega168) or 32 KB (ATmega328) of which 2 KB used by bootloader
SRAM	1 KB (ATmega168) or 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) or 1 KB (ATmega328)
Clock Speed	16 MHz
Dimensions	0.73" x 1.70"

the board



Power

The Arduino Nano can be powered via the Mini-B USB connection, 6-20V unregulated external power supply (pin 30), or 5V regulated external power supply (pin 27). The power source is automatically selected to the highest voltage source.

The FTDI FT232RL chip on the Nano is only powered if the board is being powered over USB. As a result, when running on external (non-USB) power, the 3.3V output (which is supplied by the FTDI chip) is not available and the RX and TX LEDs will flicker if digital pins 0 or 1 are high.

Memory

The ATmega168 has 16 KB of flash memory for storing code (of which 2 KB is used for the bootloader); the ATmega328 has 32 KB, (also with 2 KB used for the bootloader). The ATmega168 has 1 KB of SRAM and 512 bytes of EEPROM (which can be read and written with the EEPROM library); the ATmega328 has 2 KB of SRAM and 1 KB of EEPROM.

Input and Output

Each of the 14 digital pins on the Nano can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip.
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Nano has 8 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the analogReference() function. Additionally, some pins have specialized functionality:

- **I^2C: 4 (SDA) and 5 (SCL).** Support I^2C (TWI) communication using the Wire library (documentation on the Wiring website).

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and ATmega168 ports.

Communication

The Arduino Nano has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega168 and ATmega328 provide UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An FTDI FT232RL on the board channels this serial communication over USB and the FTDI drivers (included with the Arduino software) provide a virtual com port to software on the computer. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the FTDI chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Nano's digital pins.

The ATmega168 and ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmega168 or ATmega328 datasheet.

Programming

The Arduino Nano can be programmed with the Arduino software (download). Select "Arduino Diecimila, Duemilanove, or Nano w/ ATmega168" or "Arduino Duemilanove or Nano w/ ATmega328" from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega168 or ATmega328 on the Arduino Nano comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Nano is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the FT232RL is connected to the reset line of the ATmega168 or ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Nano is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Nano. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

CD74HCT4052, CD54/74HC4053, CD54/74HC54053
HIGH-SPEED CMOS LOGIC ANALOG MULTIPLEXERS/DEMULTIPLEXERS

Check for Samples: CD/74HC4051, CD54/74HCT4051, CD54/74HC4052,

FEATURES

- **Wide Analog Input Voltage Range. . ±5 V Max**
- **Low ON Resistance**
  - **70 Ω Typical (V_{CC} – V_{EE} = 4.5 V)**
  - **40 Ω Typical (V_{CC} – V_{EE} = 9 V)**
- **Low Crosstalk Between Switches**
- **Fast Switching and Propagation Speeds**
- **Break-Before-Make Switching**
- **Wide Operating Temperature Range –55°C to 125°C**
- **CD54HC/CD74HC Types**
  - **Operation Control Voltage . . . . . 2 V to 6 V**
  - **Switch Voltage . . . . . 0 V to 10 V**

  «
- **CD54HCT/CD74HCT Types**
  - **Operation Control Voltage . . . 4.5 V to 5.5 V**
  - **Switch Voltage . . . . . 0 V to 10 V**
  - **Direct LSTTL Input Logic Compatibility V_{IL} = 0.8 V Max, V_{IH} = 2 V Min**
  - **CMOS Input Compatibility I_I ≤ 1 µA at V_{OL}, V_{OH}**

DESCRIPTION

These devices are digitally controlled analog switches which utilize silicon gate CMOS technology to achieve operating speeds similar to LSTTL with the low power consumption of standard CMOS integrated circuits.

These analog multiplexers/demultiplexers control analog voltages that may vary across the voltage supply range (i.e., V_{CC} to V_{EE}). They are bidirectional switches thus allowing any analog input to be used as an output and vice-versa. The switches have low ON resistance and low OFF leakages. In addition, all three devices have an enable control which, when high, disables all switches to their OFF state.

ORDERING INFORMATION(1)

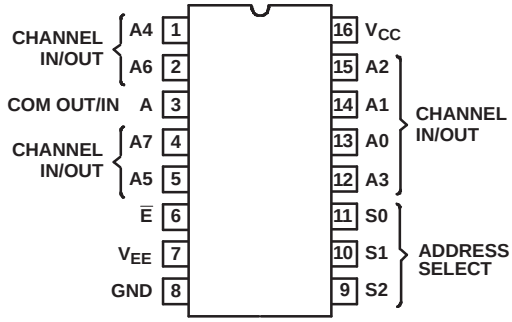
PART NUMBER	TEMP. RANGE (°C)	PACKAGE
CD54HC4051F3A	–55 to 125	16 Ld CERDIP
CD54HC4052F3A	–55 to 125	16 Ld CERDIP
CD54HC4053F3A	–55 to 125	16 Ld CERDIP
CD54HCT4051F3A	–55 to 125	16 Ld CERDIP
CD74HC4051E	–55 to 125	16 Ld PDIP
CD74HC4051M	–55 to 125	16 Ld SOIC
CD74HC4051MT	–55 to 125	16 Ld SOIC
CD74HC4051M96G3	–55 to 125	16 Ld SOIC
CD74HC4051NSR	–55 to 125	16 Ld SOP
CD74HC4051PWR	–55 to 125	16 Ld TSSOP
CD74HC4051PWT	–55 to 125	16 Ld TSSOP
CD74HC4052E	–55 to 125	16 Ld PDIP
CD74HC4052M	–55 to 125	16 Ld SOIC
CD74HC4052MT	–55 to 125	16 Ld SOIC
CD74HC4052M96G3	–55 to 125	16 Ld SOIC
CD74HC4052NSR	–55 to 125	16 Ld SOP
CD74HC4052PW	–55 to 125	16 Ld TSSOP
CD74HC4052PWR	–55 to 125	16 Ld TSSOP

(1) When ordering, use the entire part number. The suffixes 96 and R denote tape and reel. The suffix T denotes a small-quantity reel of 250.

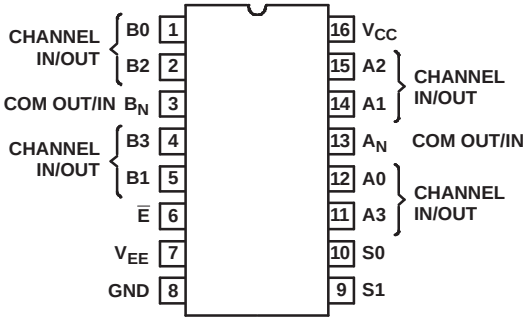
ORDERING INFORMATION(1) (continued)

PART NUMBER	TEMP. RANGE (°C)	PACKAGE
CD74HC4052PWT	–55 to 125	16 Ld TSSOP
CD74HC4053E	–55 to 125	16 Ld PDIP
CD74HC4053M	–55 to 125	16 Ld SOIC
CD74HC4053MT	–55 to 125	16 Ld SOIC
CD74HC4053M96G3	–55 to 125	16 Ld SOIC
CD74HC4053NSR	–55 to 125	16 Ld SOP
CD74HC4053PW	–55 to 125	16 Ld TSSOP
CD74HC4053PWRG3	–55 to 125	16 Ld TSSOP
CD74HC4053PWT	–55 to 125	16 Ld TSSOP
CD74HCT4051E	–55 to 125	16 Ld PDIP
CD74HCT4051M	–55 to 125	16 Ld SOIC
CD74HCT4051MT	–55 to 125	16 Ld SOIC
CD74HCT4051M96	–55 to 125	16 Ld SOIC
CD74HCT4052E	–55 to 125	16 Ld PDIP
CD74HCT4052M	–55 to 125	16 Ld SOIC
CD74HCT4052MT	–55 to 125	16 Ld SOIC
CD74HCT4052M96	–55 to 125	16 Ld SOIC
CDHCT4053E	–55 to 125	16 Ld PDIP
CDHCT4053M	–55 to 125	16 Ld SOIC
CDHCT4053MT	–55 to 125	16 Ld SOIC
CDHCT4053M96	–55 to 125	16 Ld SOIC
CDHCT4053PWR	–55 to 125	16 Ld TSSOP
CDHCT4053PWT	–55 to 125	16 Ld TSSOP

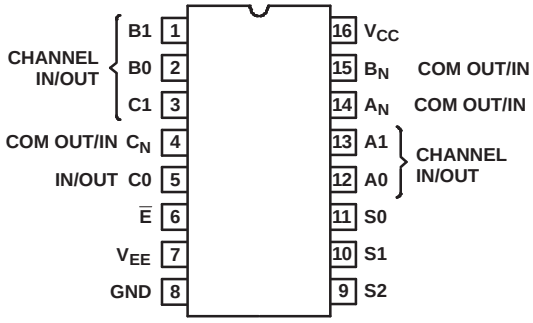
CD54HC4051, CD54HCT4051
(CERDIP)
CD74HC4051
(PDIP, SOIC, SOP, TSSOP)
CD74HCT4051
(PDIP, SOIC)
TOP VIEW



CD54HC4052
(CERDIP)
CD74HC4052
(PDIP, SOIC, SOP, TSSOP)
CD74HCT4052
(PDIP, SOIC)
TOP VIEW



CD54HC4053
(CERDIP)
CD74HC4053
(PDIP, SOIC, SOP, TSSOP)
CD74HCT4053
(PDIP, SOIC, TSSOP)
TOP VIEW



FUNCTIONAL DIAGRAM OF HC/HCT4051

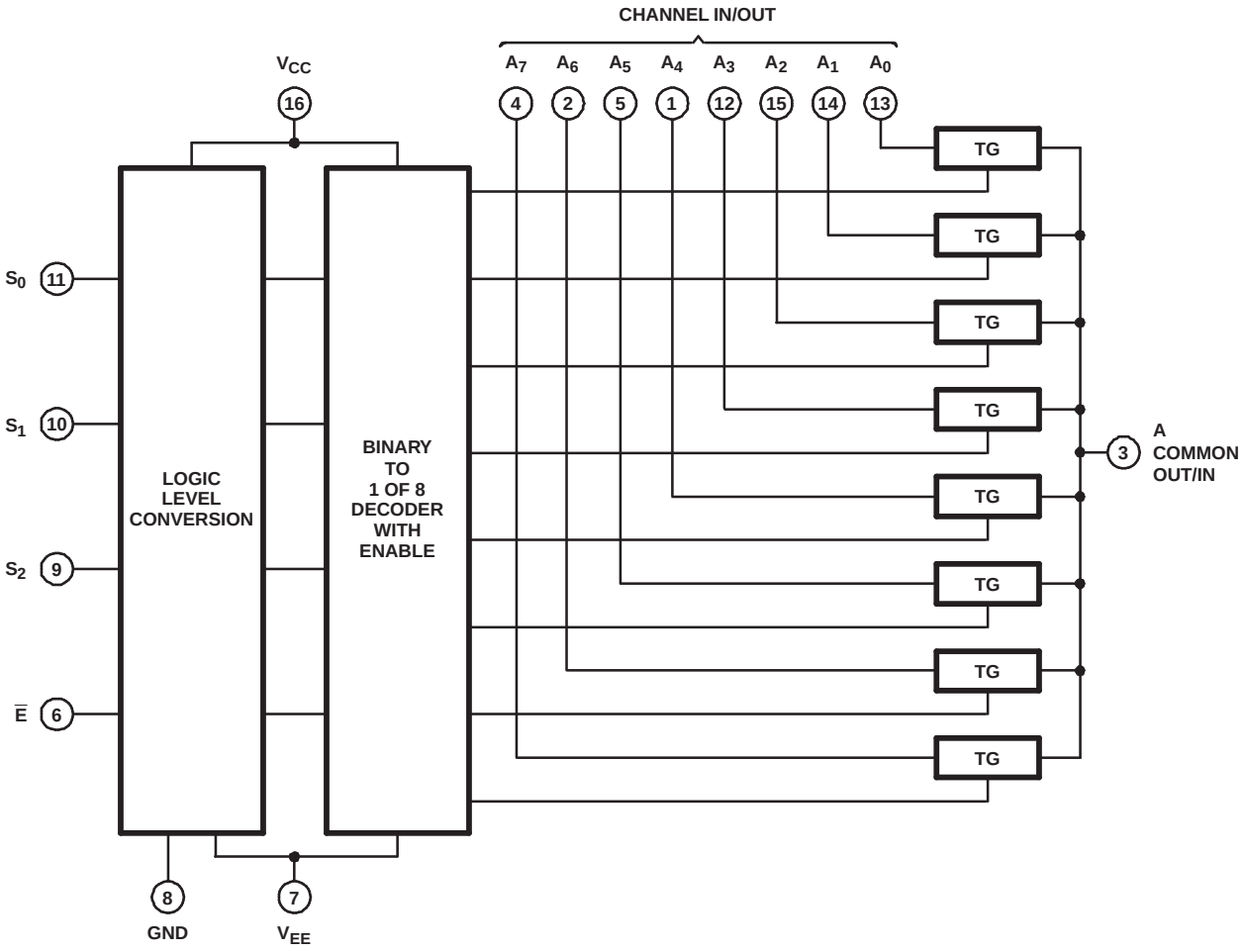


Table 1. TRUTH TABLE 'HC/CD74HCT4051 (1)

INPUT STATES				ON CHANNELS
ENABLE	S_2	S_1	S_0	
L	L	L	L	A0
L	L	L	H	A1
L	L	H	L	A2
L	L	H	H	A3
L	H	L	L	A4
L	H	L	H	A5
L	H	H	L	A6
L	H	H	H	A7
H	X	X	X	None

(1) X = Don't care

FUNCTIONAL DIAGRAM OF HC4052, CD74HCT4052

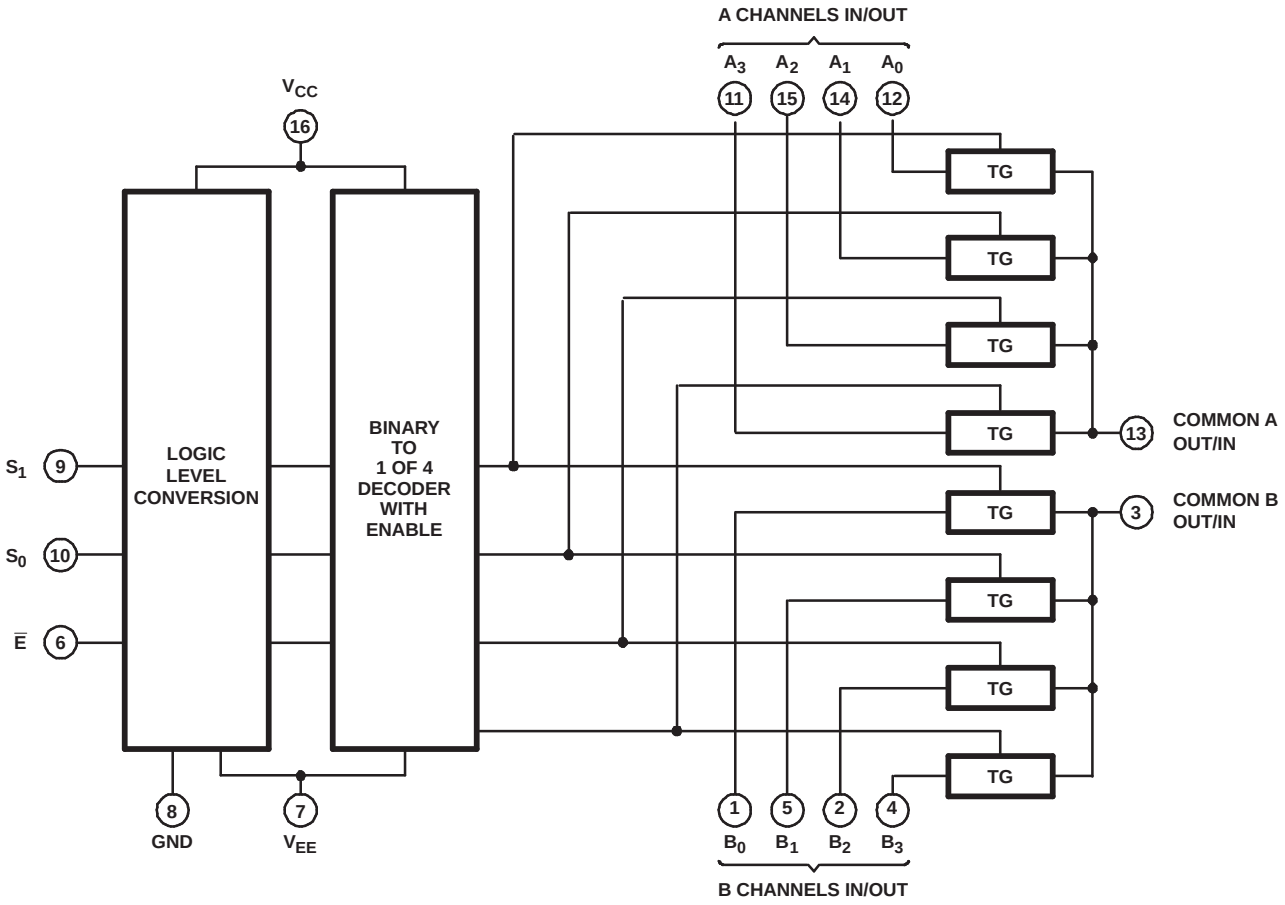


Table 2. FUNCTION TABLE 'HC4052, CD74HCT4052(1)

INPUT STATES			ON CHANNELS
ENABLE	S_1	S_0	
L	L	L	A0, B0
L	L	H	A1, B1
L	H	L	A2, B2
L	H	H	A3, B3
H	X	X	None

(1) X = Don't care

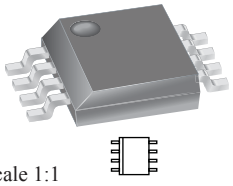
Fully Integrated, Hall Effect-Based Linear Current Sensor IC with 2.1 kVRMS Isolation and a Low-Resistance Current Conductor

Features and Benefits

- Low-noise analog signal path
- Device bandwidth is set via the new FILTER pin
- 5 µs output rise time in response to step input current
- 80 kHz bandwidth
- Total output error 1.5% at T_A = 25°C
- Small footprint, low-profile SOIC8 package
- 1.2 mΩ internal conductor resistance
- 2.1 kVRMS minimum isolation voltage from pins 1-4 to pins 5-8
- 5.0 V, single supply operation
- 66 to 185 mV/A output sensitivity
- Output voltage proportional to AC or DC currents
- Factory-trimmed for accuracy
- Extremely stable output offset voltage
- Nearly zero magnetic hysteresis
- Ratiometric output from supply voltage

TÜV America
Certificate Number:
U8V 06 05 54214 010

Package: 8 Lead SOIC (suffix LC)



Approximate Scale 1:1

Description

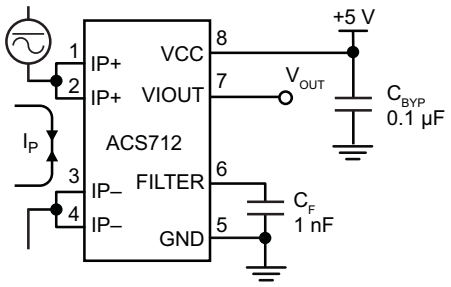
The Allegro™ ACS712 provides economical and precise solutions for AC or DC current sensing in industrial, commercial, and communications systems. The device package allows for easy implementation by the customer. Typical applications include motor control, load detection and management, switch-mode power supplies, and overcurrent fault protection. The device is not intended for automotive applications.

The device consists of a precise, low-offset, linear Hall circuit with a copper conduction path located near the surface of the die. Applied current flowing through this copper conduction path generates a magnetic field which the Hall IC converts into a proportional voltage. Device accuracy is optimized through the close proximity of the magnetic signal to the Hall transducer. A precise, proportional voltage is provided by the low-offset, chopper-stabilized BiCMOS Hall IC, which is programmed for accuracy after packaging.

The output of the device has a positive slope (>$V_{IOUT(Q)}$) when an increasing current flows through the primary copper conduction path (from pins 1 and 2, to pins 3 and 4), which is the path used for current sampling. The internal resistance of this conductive path is 1.2 mΩ typical, providing low power loss. The thickness of the copper conductor allows survival of

Continued on the next page…

Typical Application



Application 1. The ACS712 outputs an analog signal, V_{OUT}. that varies linearly with the uni- or bi-directional AC or DC primary sampled current, I_P, within the range specified. C_F is recommended for noise management, with values that depend on the application.

Description (continued)

the device at up to 5× overcurrent conditions. The terminals of the conductive path are electrically isolated from the signal leads (pins 5 through 8). This allows the ACS712 to be used in applications requiring electrical isolation without the use of opto-isolators or other costly isolation techniques.

The ACS712 is provided in a small, surface mount SOIC8 package. The leadframe is plated with 100% matte tin, which is compatible with standard lead (Pb) free printed circuit board assembly processes. Internally, the device is Pb-free, except for flip-chip high-temperature Pb-based solder balls, currently exempt from RoHS. The device is fully calibrated prior to shipment from the factory.

Selection Guide

Part Number	Packing*	T_A (°C)	Optimized Range, I_P (A)	Sensitivity, Sens (Typ) (mV/A)
ACS712ELCTR-05B-T	Tape and reel, 3000 pieces/reel	–40 to 85	±5	185
ACS712ELCTR-20A-T	Tape and reel, 3000 pieces/reel	–40 to 85	±20	100
ACS712ELCTR-30A-T	Tape and reel, 3000 pieces/reel	–40 to 85	±30	66

*Contact Allegro for additional packing options.

Absolute Maximum Ratings

Characteristic	Symbol	Notes	Rating	Units
Supply Voltage	V_{CC}		8	V
Reverse Supply Voltage	V_{RCC}		–0.1	V
Output Voltage	V_{IOUT}		8	V
Reverse Output Voltage	V_{RIOUT}		–0.1	V
Output Current Source	$I_{IOUT(Source)}$		3	mA
Output Current Sink	$I_{IOUT(Sink)}$		10	mA
Overcurrent Transient Tolerance	I_P	1 pulse, 100 ms	100	A
Nominal Operating Ambient Temperature	T_A	Range E	–40 to 85	ºC
Maximum Junction Temperature	T_J(max)		165	ºC
Storage Temperature	T_{stg}		–65 to 170	ºC

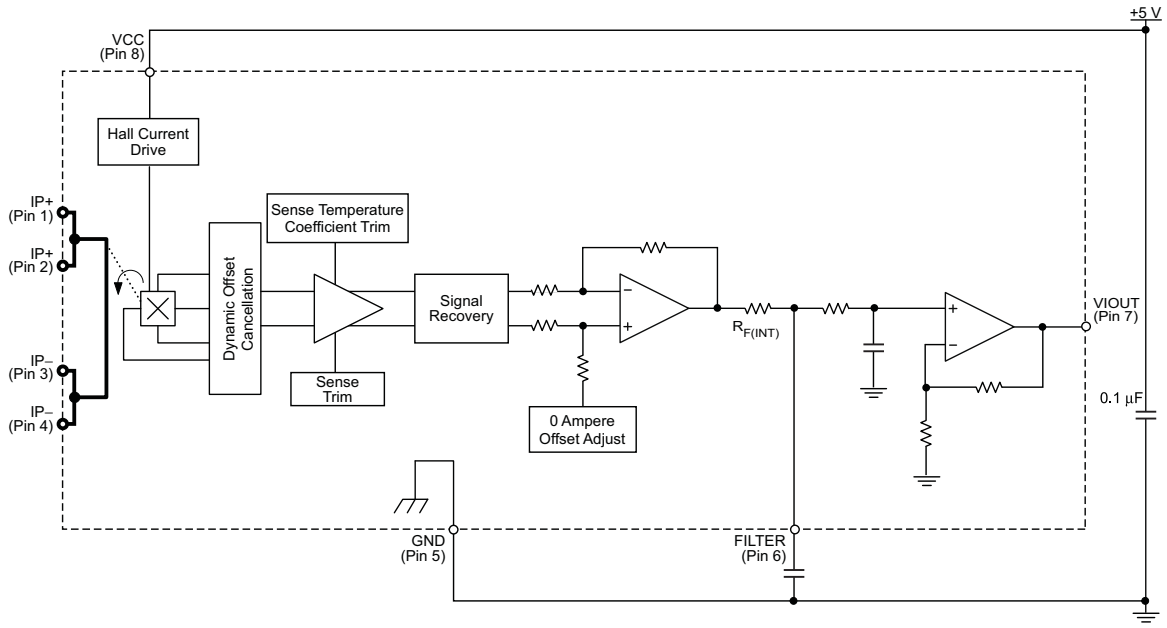
Isolation Characteristics

Characteristic	Symbol	Notes	Rating	Unit
Dielectric Strength Test Voltage*	V_{ISO}	Agency type-tested for 60 seconds per UL standard 60950-1, 1st Edition	2100	VAC
Working Voltage for Basic Isolation	V_{WFSI}	For basic (single) isolation per UL standard 60950-1, 1st Edition	354	VDC or V_{pk}
Working Voltage for Reinforced Isolation	V_{WFRI}	For reinforced (double) isolation per UL standard 60950-1, 1st Edition	184	VDC or V_{pk}

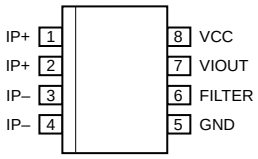
* Allegro does not conduct 60-second testing. It is done only during the UL certification process.

Parameter	Specification
Fire and Electric Shock	CAN/CSA-C22.2 No. 60950-1-03 UL 60950-1:2003 EN 60950-1:2001

Functional Block Diagram



Pin-out Diagram



Terminal List Table

Number	Name	Description
1 and 2	IP+	Terminals for current being sampled; fused internally
3 and 4	IP–	Terminals for current being sampled; fused internally
5	GND	Signal ground terminal
6	FILTER	Terminal for external capacitor that sets bandwidth
7	VIOUT	Analog output signal
8	VCC	Device power supply terminal

COMMON OPERATING CHARACTERISTICS[1] over full range of T_A, C_F = 1 nF, and V_{CC} = 5 V, unless otherwise specified

Characteristic	Symbol	Test Conditions	Min.	Typ.	Max.	Units
ELECTRICAL CHARACTERISTICS						
Supply Voltage	V_{CC}		4.5	5.0	5.5	V
Supply Current	I_{CC}	V_{CC} = 5.0 V, output open	–	10	13	mA
Output Capacitance Load	C_{LOAD}	VIOUT to GND	–	–	10	nF
Output Resistive Load	R_{LOAD}	VIOUT to GND	4.7	–	–	kΩ
Primary Conductor Resistance	$R_{PRIMARY}$	T_A = 25°C	–	1.2	–	mΩ
Rise Time	t_r	$I_P = I_P$(max), T_A = 25°C, C_{OUT} = open	–	3.5	–	µs
Frequency Bandwidth	f	–3 dB, T_A = 25°C; I_P is 10 A peak-to-peak	–	80	–	kHz
Nonlinearity	E_{LIN}	Over full range of I_P	–	1.5	–	%
Symmetry	E_{SYM}	Over full range of I_P	98	100	102	%
Zero Current Output Voltage	$V_{IOUT(Q)}$	Bidirectional; I_P = 0 A, T_A = 25°C	–	V_{CC} × 0.5	–	V
Power-On Time	t_{PO}	Output reaches 90% of steady-state level, T_J = 25°C, 20 A present on leadframe	–	35	–	µs
Magnetic Coupling[2]			–	12	–	G/A
Internal Filter Resistance[3]	$R_{F(INT)}$			1.7		kΩ

[1]Device may be operated at higher primary current levels, I_P, and ambient, T_A, and internal leadframe temperatures, T_A, provided that the Maximum Junction Temperature, T_J(max), is not exceeded.

[2]1G = 0.1 mT.

[3]$R_{F(INT)}$ forms an RC circuit via the FILTER pin.

COMMON THERMAL CHARACTERISTICS[1]

			Min.	Typ.	Max.	Units
Operating Internal Leadframe Temperature	T_A	E range	–40	–	85	°C
					Value	Units
Junction-to-Lead Thermal Resistance[2]	$R_{\theta JL}$	Mounted on the Allegro ASEK 712 evaluation board			5	°C/W
Junction-to-Ambient Thermal Resistance	$R_{\theta JA}$	Mounted on the Allegro 85-0322 evaluation board, includes the power consumed by the board			23	°C/W

[1]Additional thermal information is available on the Allegro website.

[2]The Allegro evaluation board has 1500 mm^2 of 2 oz. copper on each side, connected to pins 1 and 2, and to pins 3 and 4, with thermal vias connecting the layers. Performance values include the power consumed by the PCB. Further details on the board are available from the Frequently Asked Questions document on our website. Further information about board design and thermal performance also can be found in the Applications Information section of this datasheet.

x05B PERFORMANCE CHARACTERISTICS[1] T_A = –40°C to 85°C, C_F = 1 nF, and V_{CC} = 5 V, unless otherwise specified

Characteristic	Symbol	Test Conditions	Min.	Typ.	Max.	Units
Optimized Accuracy Range	I_P		–5	–	5	A
Sensitivity	Sens	Over full range of I_P, T_A = 25°C	180	185	190	mV/A
Noise	$V_{NOISE(PP)}$	Peak-to-peak, T_A = 25°C, 185 mV/A programmed Sensitivity, C_F = 47 nF, C_{OUT} = open, 2 kHz bandwidth	–	21	–	mV
Zero Current Output Slope	$\Delta V_{OUT(Q)}$	T_A = –40°C to 25°C	–	–0.26	–	mV/°C
		T_A = 25°C to 150°C	–	–0.08	–	mV/°C
Sensitivity Slope	ΔSens	T_A = –40°C to 25°C	–	0.054	–	mV/A/°C
		T_A = 25°C to 150°C	–	–0.008	–	mV/A/°C
Total Output Error[2]	E_{TOT}	I_P =±5 A, T_A = 25°C	–	±1.5	–	%

[1]Device may be operated at higher primary current levels, I_P, and ambient temperatures, T_A, provided that the Maximum Junction Temperature, $T_{J(max)}$, is not exceeded.

[2]Percentage of I_P, with I_P = 5 A. Output filtered.

x20A PERFORMANCE CHARACTERISTICS[1] T_A = –40°C to 85°C, C_F = 1 nF, and V_{CC} = 5 V, unless otherwise specified

Characteristic	Symbol	Test Conditions	Min.	Typ.	Max.	Units
Optimized Accuracy Range	I_P		–20	–	20	A
Sensitivity	Sens	Over full range of I_P, T_A = 25°C	96	100	104	mV/A
Noise	$V_{NOISE(PP)}$	Peak-to-peak, T_A = 25°C, 100 mV/A programmed Sensitivity, C_F = 47 nF, C_{OUT} = open, 2 kHz bandwidth	–	11	–	mV
Zero Current Output Slope	$\Delta V_{OUT(Q)}$	T_A = –40°C to 25°C	–	–0.34	–	mV/°C
		T_A = 25°C to 150°C	–	–0.07	–	mV/°C
Sensitivity Slope	ΔSens	T_A = –40°C to 25°C	–	0.017	–	mV/A/°C
		T_A = 25°C to 150°C	–	–0.004	–	mV/A/°C
Total Output Error[2]	E_{TOT}	I_P =±20 A, T_A = 25°C	–	±1.5	–	%

[1]Device may be operated at higher primary current levels, I_P, and ambient temperatures, T_A, provided that the Maximum Junction Temperature, T_J(max), is not exceeded.

[2]Percentage of I_P, with I_P = 20 A. Output filtered.

x30A PERFORMANCE CHARACTERISTICS[1] T_A = –40°C to 85°C, C_F = 1 nF, and V_{CC} = 5 V, unless otherwise specified

Characteristic	Symbol	Test Conditions	Min.	Typ.	Max.	Units
Optimized Accuracy Range	I_P		–30	–	30	A
Sensitivity	Sens	Over full range of I_P, T_A = 25°C	63	66	69	mV/A
Noise	$V_{NOISE(PP)}$	Peak-to-peak, T_A = 25°C, 66 mV/A programmed Sensitivity, C_F = 47 nF, C_{OUT} = open, 2 kHz bandwidth	–	7	–	mV
Zero Current Output Slope	$\Delta V_{OUT(Q)}$	T_A = –40°C to 25°C	–	–0.35	–	mV/°C
		T_A = 25°C to 150°C	–	–0.08	–	mV/°C
Sensitivity Slope	ΔSens	T_A = –40°C to 25°C	–	0.007	–	mV/A/°C
		T_A = 25°C to 150°C	–	–0.002	–	mV/A/°C
Total Output Error[2]	E_{TOT}	I_P = ±30 A, T_A = 25°C	–	±1.5	–	%

[1]Device may be operated at higher primary current levels, I_P, and ambient temperatures, T_A, provided that the Maximum Junction Temperature, T_J(max), is not exceeded.

[2]Percentage of I_P, with I_P = 30 A. Output filtered.

LM324, LM324A, LM324E, LM224, LM2902, LM2902E, LM2902V, NCV2902

Single Supply Quad Operational Amplifiers

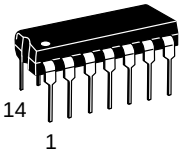
The LM324 series are low−cost, quad operational amplifiers with true differential inputs. They have several distinct advantages over standard operational amplifier types in single supply applications. The quad amplifier can operate at supply voltages as low as 3.0 V or as high as 32 V with quiescent currents about one−fifth of those associated with the MC1741 (on a per amplifier basis). The common mode input range includes the negative supply, thereby eliminating the necessity for external biasing components in many applications. The output voltage range also includes the negative power supply voltage.

Features

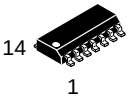
- Short Circuited Protected Outputs
- True Differential Input Stage
- Single Supply Operation: 3.0 V to 32 V
- Low Input Bias Currents: 100 nA Maximum (LM324A)
- Four Amplifiers Per Package
- Internally Compensated
- Common Mode Range Extends to Negative Supply
- Industry Standard Pinouts
- ESD Clamps on the Inputs Increase Ruggedness without Affecting Device Operation
- NCV Prefix for Automotive and Other Applications Requiring Unique Site and Control Change Requirements; AEC−Q100 Qualified and PPAP Capable
- These Devices are Pb−Free, Halogen Free/BFR Free and are RoHS Compliant

ON Semiconductor®

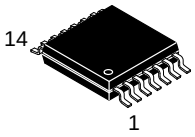
www.onsemi.com



PDIP−14
N SUFFIX
CASE 646

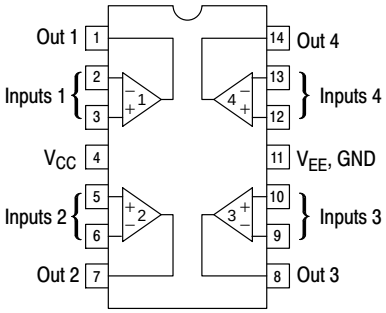


SOIC−14
D SUFFIX
CASE 751A



TSSOP−14
DTB SUFFIX
CASE 948G

PIN CONNECTIONS



ORDERING INFORMATION

See detailed ordering and shipping information in the package dimensions section on page 10 of this data sheet.

DEVICE MARKING INFORMATION

See general marking information in the device marking section on page 11 of this data sheet.

MAXIMUM RATINGS (T_A = +25°C, unless otherwise noted.)

Rating	Symbol	Value	Unit
Power Supply Voltages			Vdc
Single Supply	V_{CC}	32	
Split Supplies	V_{CC}, V_{EE}	±16	
Input Differential Voltage Range (Note 1)	V_{IDR}	±32	Vdc
Input Common Mode Voltage Range	V_{ICR}	−0.3 to 32	Vdc
Output Short Circuit Duration	t_{SC}	Continuous	
Junction Temperature	T_J	150	°C
Thermal Resistance, Junction−to−Air (Note 2) Case 646	$R_{\theta JA}$	118	°C/W
Case 751A		156	
Case 948G		190	
Storage Temperature Range	T_{stg}	−65 to +150	°C
Operating Ambient Temperature Range	T_A		°C
LM224		−25 to +85	
LM324, LM324A, LM324E		0 to +70	
LM2902, LM2902E		−40 to +105	
LM2902V, NCV2902 (Note 3)		−40 to +125	

Stresses exceeding those listed in the Maximum Ratings table may damage the device. If any of these limits are exceeded, device functionality should not be assumed, damage may occur and reliability may be affected.

1. Split Power Supplies.
2. All $R_{\theta JA}$ measurements made on evaluation board with 1 oz. copper traces of minimum pad size. All device outputs were active.
3. *NCV2902 is qualified for automitive use.*

ESD RATINGS

Rating	HBM	MM	Unit
ESD Protection at any Pin (Human Body Model − HBM, Machine Model − MM)			
NCV2902 (Note 3)	2000	200	V
LM324E, LM2902E	2000	200	V
LM324DR2G, LM2902DR2G	200	100	V
All Other Devices	2000	200	V

ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 V, V_{EE} = GND, T_A = 25°C, unless otherwise noted.)

Characteristics	Symbol	LM224			LM324A			LM324, LM324E			LM2902, LM2902E			LM2902V/NCV2902			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Input Offset Voltage V_{CC} = 5.0 V to 30 V V_{ICR} = 0 V to V_{CC} −1.7 V, V_O = 1.4 V, R_S = 0 Ω	V_{IO}																mV
T_A = 25°C		–	2.0	5.0	–	2.0	3.0	–	2.0	7.0	–	2.0	7.0	–	2.0	7.0	
T_A = T_{high} (Note 4)		–	–	7.0	–	–	5.0	–	–	9.0	–	–	10	–	–	13	
T_A = T_{low} (Note 4)		–	–	7.0	–	–	5.0	–	–	9.0	–	–	10	–	–	10	
Average Temperature Coefficient of Input Offset Voltage T_A = T_{high} to T_{low} (Notes 4 and 6)	$\Delta V_{IO}/\Delta T$	–	7.0	–	–	7.0	30	–	7.0	–	–	7.0	–	–	7.0	–	μV/°C
Input Offset Current	I_{IO}	–	3.0	30	–	5.0	30	–	5.0	50	–	5.0	50	–	5.0	50	nA
T_A = T_{high} to T_{low} (Note 4)		–	–	100	–	–	75	–	–	150	–	–	200	–	–	200	
Average Temperature Coefficient of Input Offset Current T_A = T_{high} to T_{low} (Notes 4 and 6)	$\Delta I_{IO}/\Delta T$	–	10	–	–	10	300	–	10	–	–	10	–	–	10	–	pA/°C
Input Bias Current	I_{IB}	–	−90	−150	–	−45	−100	–	−90	−250	–	−90	−250	–	−90	−250	nA
T_A = T_{high} to T_{low} (Note 4)		–	–	−300	–	–	−200	–	–	−500	–	–	−500	–	–	−500	
Input Common Mode Voltage Range (Note 5) V_{CC} = 30 V	V_{ICR}																V
T_A = +25°C		0	–	28.3	0	–	28.3	0	–	28.3	0	–	28.3	0	–	28.3	
T_A = T_{high} to T_{low} (Note 4)		0	–	28	0	–	28	0	–	28	0	–	28	0	–	28	
Differential Input Voltage Range	V_{IDR}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	–	–	V_{CC}	V
Large Signal Open Loop Voltage Gain R_L = 2.0 kΩ, V_{CC} = 15 V, for Large V_O Swing	A_{VOL}	50	100	–	25	100	–	25	100	–	25	100	–	25	100	–	V/mV
T_A = T_{high} to T_{low} (Note 4)		25	–	–	15	–	–	15	–	–	15	–	–	15	–	–	
Channel Separation 10 kHz ≤ f ≤ 20 kHz, Input Referenced	CS	–	−120	–	–	−120	–	–	−120	–	–	−120	–	–	−120	–	dB
Common Mode Rejection, R_S ≤ 10 kΩ	CMR	70	85	–	65	70	–	65	70	–	50	70	–	50	70	–	dB
Power Supply Rejection	PSR	65	100	–	65	100	–	65	100	–	50	100	–	50	100	–	dB

4. LM224: T_{low} = −25°C, T_{high} = +85°C
   LM324/LM324A/LM324E: T_{low} = 0°C, T_{high} = +70°C
   LM2902/LM2902E: T_{low} = −40°C, T_{high} = +105°C
   LM2902V & NCV2902: T_{low} = −40°C, T_{high} = +125°C
   NCV2902 is qualified for automotive use.
5. The input common mode voltage or either input signal voltage should not be allowed to go negative by more than 0.3 V. The upper end of the common mode voltage range is V_{CC} −1.7 V, but either or both inputs can go to +32 V without damage, independent of the magnitude of V_{CC}.
6. Guaranteed by design.

FAIRCHILD SEMICONDUCTOR®

February 2010

MCT6, MCT61, MCT62
Dual Phototransistor Optocouplers

Features

- Two isolated channels per package
- Two packages fit into a 16 lead DIP socket
- Choice of three current transfer ratios
- Underwriters Laboratory (U.L.) recognized File E90700
- VDE approved for IEC60747-5-2

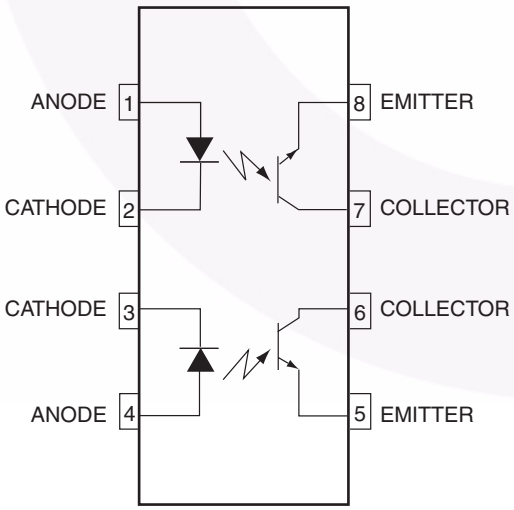
Applications

- AC line/digital logic – isolate high voltage transients
- Digital logic/digital logic – eliminate spurious grounds
- Digital logic/AC triac control – isolate high voltage transients
- Twisted pair line receiver – eliminate ground loop feedthrough
- Telephone/telegraph line receiver – isolate high voltage transients
- High frequency power supply feedback control – maintain floating grounds and transients
- Relay contact monitor – isolate floating grounds and transients
- Power supply monitor – isolate transients

Description

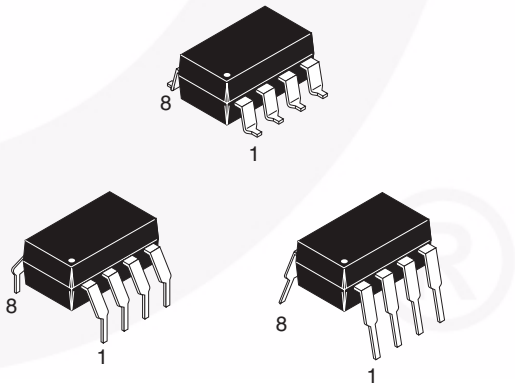
The MCT6X Optocouplers have two channels for density applications. For four channel applications, two-packages fit into a standard 16-pin DIP socket. Each channel is an NPN silicon planar phototransistor optically coupled to a gallium arsenide infrared emitting diode.

Schematic



Equivalent Circuit

Package Outlines



Absolute Maximum Ratings

Stresses exceeding the absolute maximum ratings may damage the device. The device may not function or be operable above the recommended operating conditions and stressing the parts to these levels is not recommended. In addition, extended exposure to stresses above the recommended operating conditions may affect device reliability. The absolute maximum ratings are stress ratings only.

Symbol	Rating	Value	Unit
TOTAL DEVICE			
T_{STG}	Storage Temperature	-55 to +150	°C
T_{OPR}	Operating Temperature	-55 to +100	°C
T_{SOL}	Lead Solder Temperature (Refer to Reflow Temperature Profile)	260 for 10 sec	°C
P_D	Total Device Power Dissipation @ T_A = 25°C	400	mW
	Derate above 25°C	5.33	mW/°C
EMITTER (Each channel)			
I_F	Forward Current – Continuous	60	mA
I_F(pk)	Forward Current – Peak (PW = 1µs, 300pps)	3	A
V_R	Reverse Voltage	3.0	V
P_D	LED Power Dissipation @ T_A = 25°C	100	mW
	Derate above 25°C (Total Input)	1.3	mW/°C
DETECTOR (Each channel)			
I_C	Collector Current – Continuous	30	mA
P_D	Detector Power Dissipation @ T_A = 25°C	150	mW
	Derate above 25°C	2.0	mW/°C

Electrical Characteristics (T_A = 25°C unless otherwise specified)

Individual Component Characteristics

Symbol	Parameter	Test Conditions	Min.	Typ.*	Max.	Units
EMITTER						
V_F	Input Forward Voltage	I_F = 20mA		1.2	1.5	V
V_R	Reverse Voltage	I_R = 10µA	3.0	25		V
I_R	Reverse Current	V_R = 5V		0.001	10	µA
C_J	Junction Capacitance	V_F = 0V, f = 1MHz		50		pF
DETECTOR						
BV_{CEO}	Collector-Emitter Breakdown Voltage	I_C = 1.0mA, I_F = 0	30	85		V
BV_{ECO}	Emitter-Collector Breakdown Voltage	I_E = 100µA, I_F = 0	6	13		V
I_{CEO}	Collector-Emitter Dark Current	V_{CE} = 10V, I_F = 0		5	100	nA
C_{CE}	Capacitance	V_{CE} = 0V, f = 1MHz		8		pF

Transfer Characteristics

Symbol	Characteristic	Test Conditions	Min.	Typ.*	Max.	Units
SWITCHING CHARACTERISTICS (AC)						
t_{on}	Non-Saturated Turn-on Time	R_L = 100Ω, I_C = 2mA, V_{CC} = 10V		2.4		µs
t_{off}	Non-Saturated Turn-off Time			2.4		µs
CURRENT TRANSFER RATIO, COLLECTOR-EMITTER (DC)						
CTR	MCT6	I_F = 10mA, V_{CE} = 10V	20			%
	MCT61	I_F = 5mA, V_{CE} = 5V	50			
	MCT62		100			
$V_{CE(sat)}$	Saturation Voltage	I_F = 16mA, I_C = 2mA		0.15	0.40	V

Isolation Characteristics

Symbol	Characteristic	Test Conditions	Min.	Typ.*	Max.	Units
V_{ISO}	Input-Output Isolation Voltage	$I_{I-O} \leq$ 10µA, t = 1min.	5000			Vac(rms)
R_{ISO}	Isolation Resistance	V_{I-O} = 500VDC	10^{11}			Ω
C_{ISO}	Isolation Capacitance	f = 1MHz		0.5		pF

*All typicals at T_A = 25°C

Anexo 4. Código del programa del microcontrolador

Anexo 4. Código del programa del microcontrolador.

A continuación, se muestra el código en lenguaje C++ del programa cargado en los microcontroladores.

```
#include <Wire.h>

//#include <RTClib.h>

/*----------------------------------------------------------------MENÚ DE LA
PANTALLA DEL VATÍMETRO EN TIEMPO
REAL-----------------------------------------------------*/
//#include <Wire.h>
#include <LCD.h>                // Cargamos las librerias correspondientes a las
pantallaS y el reloj
#include<LiquidCrystal_I2C.h>
#include "DS3231.h"
/*-------------------------------------------------------------------DEFINICION
DEL
RELOJ-------------------------------------------------------------------------
--*/
#define INTERVALO_MEDICION 1000 // Medir temperatura cada 1 segundos (se renueva
internamente en el DS3231 cada 64 segundos)
#define ESPERA_ERROR 1000 // Tiempo de espera antes de volver a medir si se ha
producido un error
#define ELEMENTOS_MATRIZ_FECHA 7 //guardaremos los dás de la semana
/*-----------------------------------------------------------------
DECLARACION DE LAS VARIABLES UTILIZADAS PARA
PANTALLA-----------------------------------------------------------*/
int pulsador = 0; // variable que muestrea cuando el pulsador de menu ha sido
accionado
long tiempoPantalla = 30000; // Nos indica cuanto tiempo va a esperar la pantalla
para que se apague automaticamente
long TiempoInicial1 = 0; // Cada vez que apretemos algun pulsador comenzaremos el
tiempo desde el principio
long TiempoInicial2 = 0; // Cada vez que apretemos algun pulsador comenzaremos el
tiempo desde el principio
int cronometro = 0;
int temperatura = 0;
char *puntero_fecha;
int sumador = 0;
int pantalla = 0;
charbuffer_fecha[ELEMENTOS_MATRIZ_FECHA];
DS3231 reloj;
long tiempovariable = 5000;

long distancia;
long tiempo;
double x = 123;
String W = " ";
char w ;
#define DEBUG true
byte arrayBajoC [200];
byte arrayAltoC [200];
```

```
byte arrayBajoT [200];
byte arrayAltoT [200];
float Integral [200];
float INTV [200];
float INTI [200];
unsigned int SPROD [200];
unsigned int SPRD1 = 0;
unsigned int SV = 0;
unsigned int SI = 0;
unsigned int ArrayCorriente[200];
unsigned int ArrayTension[200];
double Vrms = 0;
double Irms = 0;

int longArray = 200;
byte listoParaEnviar = 8;
int disminuirEscala = 0;
int aumentaEscala = 0;
int conta = 0;
int conta1 = 0 ;
int escalaAlta = 0;
int escalaIntermedia = 1;
int escalaBaja = 0;
double K = 3.300000;
byte LEDbajo = 35;
byte LEDintermedio = 33;
byte LEDalto = 31;
int contador = 0;
int ATTDatos = 0 ;
int cont = 0;
byte ComenzarLectura = 24;
byte Automatico = 22;
byte Manual = 37;
byte solicitarDatCorriente = 26;
byte solicitarDatTension = 28;
int permiso = 0;
double PotenciaInstantanea = 0;
long previousMillis = 0;        // will store last time LED was updated
byte automatico = 0;
long intervalOn = 10;           // medio segundo  ON
long intervalOff = 20;         // cinco segundos OFF
double REGminutos = 0 ;
double REGhora = 0 ;
double REGdias = 0 ;
double REGmes = 0 ;
double REGano = 0 ;
double registroAcumuladoP = 0;
double registroAcumuladoVRMS = 0;
double registroAcumuladoIRMS = 0;
```

```
int contadorIntegrales = 0;
int minActual = 0;
byte minutos = 0;
byte permiso1 = 0;
double S = 0;
double fi = 0;
double Q = 0;
double FMP = 1.05;
double FMI = 0.83;
double FMV = 1.04;
double Potencia = 0;
double Intensidad = 0;
double Voltaje = 0;
byte NoResistiva = 0;
LiquidCrystal_I2C  lcd(0x3F, 2, 1, 0, 4, 5, 6, 7);
void setup() {
 Serial3.begin(115200);
 Serial2.begin(115200);
  Serial.begin (115200);
 Serial1.begin(115200);

  sendData("AT+RST\r\n", 2000, DEBUG); // rst
  sendData("AT+CWMODE=3\r\n", 1000, DEBUG); //  access point
  sendData("AT+CIFSR\r\n", 1000, DEBUG); // get ip address
  sendData("AT+CIPMUX=1\r\n", 1000, DEBUG); // configure for multiple connections
  sendData("AT+CIPSERVER=1,80\r\n", 1000, DEBUG); // turn on server on port 80
  pinMode ( listoParaEnviar , OUTPUT);
  pinMode ( LEDintermedio , OUTPUT);
  pinMode ( LEDbajo , OUTPUT);
  pinMode ( LEDalto , OUTPUT);
  pinMode ( ComenzarLectura, OUTPUT);
  pinMode ( Automatico, INPUT);
  pinMode (Manual, INPUT);
  Wire.begin();

  lcd.begin (20, 4); // Conectamos una pantalla de tipo 20x4.
  lcd.setBacklightPin(3, POSITIVE); // Configuramos que inicialmente se encuentre
apagada
 lcd.setBacklight(LOW);
  pinMode(10, INPUT);          // Configuramos el pin digital 10 como una
entrada nos indicara cuando  el pulsador de seleccion es accionado
  cronometro = 0;             // para que el reloj empiece inmediatamente
  pinMode(9, OUTPUT); /*activación del pin 9 como salida: para el pulso
ultrasónico*/
  pinMode(8, INPUT); /*activación del pin 8 como entrada: tiempo del rebote del
ultrasonido*/
  pinMode (solicitarDatCorriente , OUTPUT);
  pinMode (solicitarDatTension , OUTPUT);
  pinMode ( 2, INPUT);
```

```
  digitalWrite ( LEDintermedio , HIGH);
}



void loop() {
  attachInterrupt( 0, interrupcion , RISING);//interrupcion externa
  if (permiso1 == 0) {
    minActual = minutos + 1;
    permiso1 = 1;
  }
  if (ATTDatos == 1) {
    recibeCorriente();
    recibeTension();
  } else if (ATTDatos == 0) {
    envia();
    escalas();
    cosfi();
    integral();
    registro();
    web();
    Lcd ();
  }
}
void interrupcion() {
  ATTDatos = 1;
}
void recibeCorriente() {
  if (ATTDatos == 1 && permiso == 0) {
    digitalWrite ( solicitarDatCorriente, HIGH);
    if (contador == 0) {
      while (Serial3.available()) {
        memset(arrayBajoC, 0, sizeof(arrayBajoC)); //memset borra el contenido
del array  "muestreo" desde la posición 0 hasta el final sizeof
        memset(arrayAltoC, 0, sizeof(arrayAltoC));
        Serial3.readBytes(arrayBajoC, 65);
        Serial3.readBytes(&arrayBajoC[64], 65);
        Serial3.readBytes(&arrayBajoC[129], 70);
        Serial3.readBytes(arrayAltoC, 65);
        Serial3.readBytes(&arrayAltoC[64], 65);
        Serial3.readBytes(&arrayAltoC[129], 70);;
         contador = 1;
      }
    }
    else if (contador == 1 && ATTDatos == 1) {
      for (int i = 0; i < longArray; i++) {
        ArrayCorriente[i] = ((arrayAltoC[i] << 8) | (arrayBajoC[i])) ;
      }
      /*for (int i = 0; i < longArray; i++) {
```

```
      Serial.println(arrayBajoC[i],BIN);
      Serial.println(arrayAltoC[i],BIN);
       }*/
     /*for (int u = 0; u < longArray ; u++) {
      Serial.println(ArrayCorriente[u]); //sumar offset
       }*/
     contador = 0;
    digitalWrite (solicitarDatCorriente , LOW);
     permiso = 1;
    }
  }
}
void recibeTension() {
  if (ATTDatos == 1 && permiso == 1) {
   digitalWrite ( solicitarDatTension, HIGH);
    if (cont == 0) {
     while (Serial2.available()) {
      memset(arrayBajoT, 0, sizeof(arrayBajoT)); //memset borra el contenido
del array  "muestreo" desde la posición 0 hasta el final sizeof
      memset(arrayAltoT, 0, sizeof(arrayAltoT));
      Serial2.readBytes(arrayBajoT, 65);
      Serial2.readBytes(&arrayBajoT[64], 65);
      Serial2.readBytes(&arrayBajoT[129], 70);
      Serial2.readBytes(arrayAltoT, 65);
      Serial2.readBytes(&arrayAltoT[64], 65);
      Serial2.readBytes(&arrayAltoT[129], 70);
       cont = 1;
     }
    }
    else if (cont == 1 && ATTDatos == 1) {
     for (int i = 0; i < longArray; i++) {
      ArrayTension[i] = ((arrayAltoT[i] << 8) | (arrayBajoT[i]));
     }
     /*for (int i = 0; i < longArray; i++) {
      Serial.println(arrayBajoT[i],BIN);
      Serial.println(arrayAltoT[i],BIN);
       }*/
     /*for (int u = 0; u < longArray ; u++) {
      Serial.println(ArrayTension[u]);
       }*/
     cont = 0;
    digitalWrite (solicitarDatTension , LOW);
     ATTDatos = 0;
     permiso = 2;
    }
  }
}
void integral() {
```

```
  if (permiso == 2) {
   //-----------------------------------------Calculamos Potencia
Instantanea-----------------------------------------------------------
    PotenciaInstantanea = 0;
   memset(Integral, 0, sizeof(Integral)); //memset borra el contenido del array
"muestreo" desde la posición 0 hasta el final sizeof

    for (int i = 0; i < longArray; i++) {
     Integral[i] = ((0.636 * ArrayTension[i]) - (325.266)) * (((0.00276 * K *
ArrayCorriente[i]) - K * 1.4142) * (0.0001)); //sumar offset corriente
    }
    /*for (int u = 0; u < longArray ; u++) {
     Serial.println(Integral[u]);
      }*/
    PotenciaInstantanea = 0;
    for (int i = 0; i < longArray; i++) {
     PotenciaInstantanea = PotenciaInstantanea + Integral[i] / 0.02;
    }
  //-----------------------------------------------Calculamos
Vrms-------------------------------------------------------------------
    double Vrms1 = 0;
    Vrms = 0;
    for (int i = 0 ; i < longArray; i++) {
     INTV[i] = pow((0.636 * ArrayTension[i] - 325.266), 2);
    }
    for (int i = 0 ; i < longArray; i++) {
     Vrms1 = INTV[i] * 0.0001 + Vrms1;
    }
    Vrms = sqrt(Vrms1 / 0.02);
  //-----------------------------------------------Calculamos
Irms-------------------------------------------------------------------
    double Irms1 = 0;
    Irms = 0;
    for (int i = 0 ; i < longArray; i++) {
     INTI[i] = pow(((0.00276 * K * ArrayCorriente[i]) - K * 1.4142), 2);
    }
    for (int i = 0 ; i < longArray; i++) {
     Irms1 = INTI[i] * 0.0001 + Irms1;
    }
    Irms = sqrt(Irms1 / 0.02);

   registroAcumuladoIRMS = Irms * FMI + registroAcumuladoIRMS;
   registroAcumuladoVRMS = Vrms * FMV + registroAcumuladoVRMS;
   registroAcumuladoP = PotenciaInstantanea + registroAcumuladoP;
   contadorIntegrales ++;
    permiso = 0;
  }

}
```

```
void cosfi() {

  if (permiso == 3) {
   Serial.println("aqui1");

    double Vrms1 = 0;
    //Vrms = 0;
    for (int i = 0 ; i < longArray; i++) {
      INTV[i] = ((((2 * 1.1442 * 230) / 1023) * ArrayTension[i] - (230 * 1.4142))
* ((((2 * 1.1442 * 230) / 1023) * ArrayTension[i]) - (230 * 1.4142)) * 0.001) ;
    }
    for (int i = 0 ; i < longArray; i++) {
      Vrms1 = INTV[i] + Vrms1;
    }
    Vrms = sqrt(Vrms1 / 0.02);
   Serial.println("aqui1");

   Serial.println(Vrms);
    permiso = 0;
  }



}
void registro() {

  if (contadorIntegrales == 10) {

   S = ((registroAcumuladoVRMS / 10) * (registroAcumuladoIRMS / 10));
   REGminutos = (registroAcumuladoP / 10) * FMP;
   fi = REGminutos / S;
   Q = sqrt(pow(S , 2) - pow ( REGminutos , 2));

   REGhora = (REGminutos / 1000) * 60;
   REGdias = REGhora * 5;
   REGmes = REGdias * 30;
   REGano = REGmes * 12;
   if ( fi < 0, 9 ) {
     NoResistiva  = 1;

    } else {
     NoResistiva = 0;
    }
   Serial.print("Potencia activa: ");
   Serial.println(REGminutos);
   Serial.print("Potencia Reactiva: ");
   Serial.println(Q);
   Serial.print("Potencia Aparente: ");
```

```
   Serial.println(S);
   Serial.print("Voltaje Eficaz: ");
   Serial.println(registroAcumuladoVRMS / 10);
   Serial.print("Intensida Eficaz: ");
   Serial.println(registroAcumuladoIRMS / 10);

   Serial.print("fi: ");
   Serial.println(fi);

   Serial.println(K);
   Serial.println(FMP);
   Serial.println(NoResistiva);
    Potencia = REGminutos;
   Voltaje = registroAcumuladoVRMS / 10;
   Intensidad = registroAcumuladoIRMS / 10;
   //Serial.println(REGhora);
   //Serial.println(REGdias);
   //Serial.println(REGmes);
   //Serial.println(REGano);
    registroAcumuladoP = 0;
    registroAcumuladoVRMS = 0;
    registroAcumuladoIRMS = 0;
    contadorIntegrales = 0;
    minActual++;
  }
  if (minutos == minActual ) {
    //registroAcumulado = 0;
    contadorIntegrales = 0;
    minActual = minutos + 1;
  }

}

// --------------------------------Almacenar datos recibidos
corriente-------------------------------------



void envia() {
  if  ((digitalRead(Manual) == HIGH) && (ATTDatos == 0)) {
    digitalWrite(ComenzarLectura , HIGH);
     automatico = 0;
  }
  if ( (digitalRead(Manual) == LOW ) && (ATTDatos == 0)) {
    digitalWrite(ComenzarLectura , LOW);
  }

  if  ((digitalRead(Automatico) == HIGH) && (ATTDatos == 0)) {
    automatico = !automatico ;
```

```
  }

  if ((automatico == 1) && (ATTDatos == 0)) {
    unsigned long currentMillis = millis();

    if (currentMillis - previousMillis > intervalOff) {
      previousMillis = currentMillis;
      digitalWrite(ComenzarLectura , HIGH);
    }
    else {
      if (currentMillis - previousMillis > intervalOn) {
        previousMillis = currentMillis;
        digitalWrite(ComenzarLectura , LOW);
        automatico = 0;
       }
    }
  }

}




void escalas() {

 //---------------------------------------DISMINUYE
ESCALA-------------------------------------------------
  double lectura = analogRead(A0);
  if ( (lectura < 462) || (lectura > 562)) {
    conta = 0;
  }
  if ((462 < lectura) && (lectura < 562)) {     // cuando los valores de la
lectura esten siempre entre 412 y 612 aumentará uno a la cuenta
    conta++;                                    // cuando esta cuenta llegue a
vente quiere decr que la señal se encuentra entre esos dos parámetros
  }                                             // y necesitamos que aumente la
escala para tener un muestreo mas preciso
  if (conta >= 15) {                            // en cambio si nunca llega a
tener los 20 valores entre un rango previamente dicho no hará falta tener
    disminuirEscala = 1;                         // amplificar la señal.
  } else if ( conta <= 15) {
    disminuirEscala = 0;
  }

 //---------------------------------------AUMENTA
ESCALA-------------------------------------------------
  if ( lectura > 1015) {
    conta1 ++;
```

```
  }
  if ( conta1 <= 15) {
    aumentaEscala = 0;
  }
  if ( conta1 >= 15) {
    aumentaEscala = 1;
  }
  // --------------------------------SECUENCIA DE MOVIMIENTO ENTRE
ESCALAS-------------------------------------

  if ((escalaIntermedia == 1) && (aumentaEscala == 1)) {   // si estoy en la
escala intermedia y me solucita que aumente de escala
    escalaAlta = 1;                                          // paso a la escala
alta
    digitalWrite ( LEDalto , HIGH);
    digitalWrite ( LEDintermedio , LOW);
    digitalWrite ( LEDbajo, LOW);
    K = 25.000000;// multiplicando los valores del CAD por un factor de escala de
30 amperios

         if (NoResistiva == 0) {
    FMP = 1.2;
    FMI = 0.88;
    FMV = 1.056;
    }
    else if ( NoResistiva == 1) {
   FMP = 1.2;
    FMI = 0.88;
    FMV = 1.056;
    }
    conta1 = 0;                                              // y me pone a cero
el contador correspondiente de la condicion.
    escalaIntermedia = 0;
  }
  if ( (escalaAlta == 1) && disminuirEscala == 1) {       // si estoy en la
escala alta y me solicitan que baje de escala
    escalaIntermedia = 1;                                    // va a la escala
intermedia
    digitalWrite ( LEDalto , LOW);
    digitalWrite ( LEDintermedio , HIGH);
    digitalWrite ( LEDbajo, LOW);
    K = 3.3000000; // y multiplico lods valores del CAD por un factor de escala
de 10 amperios

     if (NoResistiva == 0) {
         FMP = 1.05;
    FMI = 0.83;
    FMV = 1.04;
    }
```

```
    else if ( NoResistiva == 1) {
        FMP = 1.05;
    FMI = 0.83;
    FMV = 1.04;
    }
    escalaAlta = 0;
  }
  if ((escalaIntermedia == 1) && ( disminuirEscala == 1)) { // si estoy en la
escala intermedia y me solicitan que disminuya de escala
    escalaBaja = 1;                                          // paso a la escala
mas pequeña
    digitalWrite ( LEDalto , LOW);
    digitalWrite ( LEDintermedio , LOW);
    digitalWrite ( LEDbajo, HIGH);
    K = 0.5000000;                                                  // y
multiplico los valores del CAD por un factor de escala de un amperio
    if (NoResistiva == 0) {
      FMP = 1.8;
      FMI = 1.19;
      FMV = 1.1;
    }
    else if ( NoResistiva == 1) {
      FMP = 1.6;
      FMI = 1.2;
      FMV = 1.1;
    }
    escalaIntermedia = 0;
  }
  if ((escalaBaja == 1) && (aumentaEscala == 1)) {       // si estoy en la escala
baja y me solicitan que aumente de escala
    escalaIntermedia = 1;                                   // paso a la escala
intermedia
    digitalWrite ( LEDalto , LOW);
    digitalWrite ( LEDintermedio , HIGH);
    digitalWrite ( LEDbajo, LOW);
    K = 3.3000000;                                                 // multiplico
los valores del CAD por un factor de escala de 10 amperios
    if (NoResistiva == 0) {
      FMP = 1.05;
      FMI = 0.83;
      FMV = 1.04;
    }
    else if ( NoResistiva == 1) {
      FMP = 1.05;
      FMI = 0.83;
      FMV = 1.04;
    }
    conta1 = 0;                                             // y pongo a cero el
contador correspondiente de la condición
```

```
    escalaBaja = 0;
  }
  delay (70);
}
void web()
{
  if ( w == '\n')                     // Sin han pulsado intro
  { if ( W.indexOf("P13") > 0 )    // Comprobamos si P13 esta incluido en el
string
    { //digitalWrite( 13, !digitalRead(13)) ;
      Serial.println("Invirtiendo pin 13");
    }
    W = "" ;  w = ' ' ;                    // Limpiamos las variables
  }
  if (Serial1.available()) // check if the esp is sending a message
  {
    w = Serial1.read() ;
    W = W + w ;

   if (Serial1.find("+IPD,"))
    {

     int connectionId = Serial1.read() - 48; // subtract 48 because the read()
function returns
     // the ASCII decimal value and 0 (the first decimal number) starts at 48
      // '0' - 48 = 0
      // '1' - 48 = 1
      //  C:\Users\Rayco\Desktop
     String webpage = "<head><meta http-equiv=""refresh=10"" content=""10"">";
      webpage = "<BODY BGCOLOR=""#92BCF1"" >";



     webpage += "<h1>Potencia consumida en una vivienda</h1><h2>";

     webpage += "<ul><li>Potencia instantanea: ";
      webpage += REGminutos;
      webpage +=  "wh</li>";
     webpage += "<li>Potencia consumida en las ultimas horas: ";
      webpage += REGhora;
      webpage +=  "Kwh</li>";
     webpage += "<li>Potencia consumida en el ultimo dia: ";
      webpage += REGdias;
      webpage +=  "Kwh</li>";
     webpage += "<li>Potencia consumida en el ultimo mes: ";
      webpage += REGmes;
      webpage +=  "Kwh</li>";
     webpage += "<li>potencia consumida en el ultimo a&ntilde;o: ";
      webpage += REGano;
```

```
     webpage +=  "Kwh</li></ul>";

     webpage += "</body>";
     webpage += "</h2>";
     String cipSend = "AT+CIPSEND=";
     cipSend += connectionId;
     cipSend += ",";
     cipSend += webpage.length();
     cipSend += "\r\n";

     sendData(cipSend, 1000, DEBUG);
     sendData(webpage, 1000, DEBUG);


     String closeCommand = "AT+CIPCLOSE=";
     closeCommand += connectionId; // append connection id
     closeCommand += "\r\n";

     sendData(closeCommand, 3000, DEBUG);
    }
  }
}

String sendData(String command, const int timeout, boolean debug)
{
  String response = "";

  Serial1.print(command); // send the read character to the Serial1

  long int time = millis();

  while ( (time + timeout) > millis())
  {
   while (Serial1.available())
    {

     // The esp has data so display its output to the serial window
     char c = Serial1.read(); // read the next character.
     response += c;
    }
  }

  if (debug)
  {
   Serial.print(response);
  }

  return response;
}
```

```
void Lcd () {
  pulsador = digitalRead(10); // regitramos pulsador de menu
  if (millis() > cronometro)
  {
    temperatura = reloj.leer_temperatura();
    if (temperatura > reloj.temperatura_maxima() || temperatura < reloj.
temperatura_minima())
    {
      cronometro = millis() + ESPERA_ERROR;
    }
    else
    {
      cronometro = millis() + INTERVALO_MEDICION;
      reloj.cargar_fecha_hora();
      puntero_fecha = reloj.valor_fecha_hora();
      for (sumador = 0; sumador < ELEMENTOS_MATRIZ_FECHA; sumador++)
       {
        buffer_fecha[sumador] = *(puntero_fecha + sumador);

       }
    }
  }
  if (pulsador == HIGH) {

    lcd.setBacklight(HIGH);
    TiempoInicial1 = millis();
    TiempoInicial2 = millis();
  }
  if (pantalla == 0) {

    lcd.setCursor (0, 0);
    //    lcd.print(pantalla);
    lcd.setCursor (1, 0);
   lcd.print(reloj.fecha_humana());
    lcd.setCursor (11, 0);
    lcd.print(" ");
    lcd.setCursor (12, 0);
   lcd.print(reloj.hora_humana());
    lcd.setCursor (17, 0);
    lcd.print("   ");
    lcd.setCursor (0, 1);
   lcd.print("cos(fi):");
    lcd.setCursor (9, 1);
    lcd.print(fi);                        // Mensaje a mostrar en pantalla para
este case
    lcd.setCursor (0, 2);
   lcd.print("S(VA):");
    lcd.setCursor (8, 2);
    lcd.print(Q);
```

```
    lcd.setCursor (0, 3);
    lcd.print("Q(VAR):");
    lcd.setCursor (8, 3);
    lcd.print(S);

  }

  if (pantalla == 1) { // detectamos la pulsacion de selec,nuestra primera
pulsacion debe ser sobre el pulsador de seleccion, en caso contrario la pantalla
no encendera
    lcd.setCursor (0, 0);
    // lcd.print(pantalla);
    lcd.setCursor (1, 0);
    lcd.print(reloj.fecha_humana());
    lcd.setCursor (11, 0);
    lcd.print(" ");
    lcd.setCursor (12, 0);
    lcd.print(reloj.hora_humana());
    lcd.setCursor (17, 0);
    lcd.print("   ");
    lcd.setCursor (0, 1);
    lcd.print("Corriente(A):");
    lcd.setCursor (14, 1);
    lcd.print(Intensidad);                      // Mensaje a mostrar en pantalla
para este case
    lcd.setCursor (0, 2);
    lcd.print("P(W):");
    lcd.setCursor (7, 2);
    lcd.print(Potencia);
    lcd.setCursor (0, 3);
    lcd.print("Factura(Euro):");
    lcd.setCursor (14, 3);
    lcd.print(distancia);

  }




/*-----------------------------------------------------------------TEMPORIZAD
OR-----------------------------------------------------------------------------
----------------------*/
  if (millis() >= (TiempoInicial1 + tiempoPantalla)) {
    TiempoInicial1 = millis(); // Comienza la cuenta del temporizador de nuevo
    lcd.clear();              // borramos el anterior mensaje que se pudiera
encontrar en la pantalla.
```

```
    // Cuando el tiempo que hemos configurado ha pasado, la pantalla se apaga
automaticamente y las variables toman el mismo valor que al incio
    lcd.setBackLight(LOW);
  }

  if (millis() >= (TiempoInicial2 + tiempovariable)) {
    TiempoInicial2 = millis(); // Comienza la cuenta del temporizador de nuevo
    pantalla = ! pantalla;
    lcd.clear();
    Serial.println(pantalla);
  }



}
```

```
int longArray = 200; //  cambiamos 130 por 200
unsigned int muestreo [200]; //cambiamos 130 por 200
byte arrayBajo [200];
byte arrayAlto [200];
int contador = 0;
byte listo = 7;
byte enviaDatos = 8;
void setup()
   {   Serial.begin(115200);
    pinMode ( listo , OUTPUT);
    pinMode ( enviaDatos , INPUT);
   attachInterrupt( 1, interupcion , RISING);//interrupcion externa

   }
void loop()
   {
    enviar();

    }

void interupcion() {
     memset(muestreo, 0,sizeof(muestreo));//memset borra el contenido del array
"muestreo" desde la posición 0 hasta el final sizeof
      for(int i=0; i<longArray; i++) {
     muestreo[i]=analogRead(A7);
    }
    for (int i = 0; i < longArray; i++) {
     arrayBajo[i]=muestreo[i] & 255;
       }
    for (int i = 0; i < longArray; i++) {
     arrayAlto[i]=(muestreo[i]>> 8) & 255;
       }
     digitalWrite(listo, HIGH);
      contador = 1;
}


void enviar(){
 if (digitalRead (enviaDatos) == HIGH && contador == 1){
  /*for (int i = 0; i < longArray; i++) {
    Serial.println (arrayBajo [i], BIN);
    Serial.println (arrayAlto [i] , BIN);
     Serial.println (analogRead (A1));
}*/


   Serial.write(arrayBajo,65);
   Serial.write(&arrayBajo[64],65);
   Serial.write(&arrayBajo[129],70);
```

```
 Serial.write(arrayAlto,65);
 Serial.write(&arrayAlto[64],65);
 Serial.write(&arrayAlto[129],70);
  digitalWrite(listo, LOW);
   contador = 0;
   }
}
```

```
int longArray = 200; //  cambiamos 130 por 200
unsigned int muestreo [200]; //cambiamos 130 por 200
byte arrayBajo [200];
byte arrayAlto [200];
int contador = 0;
byte listo = 7;
byte enviaDatos = 8;
void setup()
   {   Serial.begin(115200);
    pinMode ( listo , OUTPUT);
    pinMode ( enviaDatos , INPUT);
   attachInterrupt( 1, interupcion , RISING);//interrupcion externa

   }
void loop()
   {
    enviar();

    }

void interupcion() {
     memset(muestreo, 0,sizeof(muestreo));//memset borra el contenido del array
"muestreo" desde la posición 0 hasta el final sizeof
      for(int i=0; i<longArray; i++) {
     muestreo[i]=analogRead(A7);
    }
    for (int i = 0; i < longArray; i++) {
     arrayBajo[i]=muestreo[i] & 255;
       }
    for (int i = 0; i < longArray; i++) {
     arrayAlto[i]=(muestreo[i]>> 8) & 255;
       }
     digitalWrite(listo, HIGH);
      contador = 1;
}


void enviar(){
 if (digitalRead (enviaDatos) == HIGH && contador == 1){
  /*for (int i = 0; i < longArray; i++) {
    Serial.println (arrayBajo [i], BIN);
    Serial.println (arrayAlto [i] , BIN);
     Serial.println (analogRead (A1));
}*/


   Serial.write(arrayBajo,65);
   Serial.write(&arrayBajo[64],65);
   Serial.write(&arrayBajo[129],70);
```

```
 Serial.write(arrayAlto,65);
 Serial.write(&arrayAlto[64],65);
 Serial.write(&arrayAlto[129],70);
  digitalWrite(listo, LOW);
   contador = 0;
   }
}
```

GLOSARIO.

KB	Kilobyte.
SRAM	Static Random Access Memory.
EEPROM	Electrically Erasable Programmable Read-Only Memory.
CPU	Central Processing Unit.
I2C	Inter-Integrated Circuit.
SDA	Data Line.
SCL	Clock Line.
SPI	Serial Peripherals Interfaces.
PIC	Programmable Integrate Circuit.
LCD	Liquid Crital Display.
GSM	Global System for Mobile.
SMS	Short Message Service.
SIM	Subscriber Identity Module.
TCP/IP	Transmission Control Protocol/Internet Protocol.
IDE	Integrated Development Environment.