UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA

UNAN – León

Facultad de Ciencias y Tecnología

Departamento de Computación

Desarrollo de un equipo que active el flujo eléctrico de un local de forma automática mientras existan personas presentes a través de la utilización de un sensor ultrasónico para prevenir el gasto innecesario de energía eléctrica.

Tesis para optar al título de

INGENIERO EN TELEMÁTICA

Presentado por:

Br. Hellen Junieth Solís Ballesteros.

Br. Roberto José Barreto Espinoza.

Br. José Abraham Laínez Juárez.

Tutor:

Msc. Denis Espinoza Hernández.

León, Enero de 2016

Contenido

Índice de Figuras:

RESUMEN

La domótica en estos días ha llamado mucho la atención y está siendo utilizada en varias áreas (industrias, edificios, hogares, etc.), ya que en corto tiempo ha presentado un gran avance en la tecnología, siendo capaz de controlar artefactos eléctricos y electrónicos.

Actualmente la mayoría de automatismos en los hogares, empresas, edificios, utilizan un control automático del flujo energético. El presente proyecto propone una alternativa en el control de los sistemas eléctricos, con la ayuda de un sensor de proximidad ultrasónico asociado a un temporizador que permite tener el control total al usuario asignándole un tiempo aproximado al equipo para activar y desactivar el flujo energético que alimenta los equipos eléctricos que se desee automatizar, de igual manera el hardware a utilizar permite la fácil interacción con el usuario ya que el control del equipo no es complicado. Este conjunto de herramientas permitirán el desarrollo de un sistema domótico que sea capaz de controlar el flujo de la energía eléctrica con un temporizador y a la vez es de fácil uso para el usuario.

INTRODUCCIÓN

1.1 Antecedentes

Para la presente investigación se ha hecho un cuidadoso estudio de proyectos antes realizados sobre este tema, obteniendo experiencias que se relacionan con nuestro proyecto. Así tenemos:

Estudios Similares:

- En La Universidad de Ingeniería UNI-Managua en el año 2013 se realizó un trabajo llamado Implementación de la Domótica para mayor confort, seguridad y eficiencia en residencias de Nicaragua. Éste proyecto realizaba un control de luminarias, electrodomésticos y climatización de una residencia y utilizaron autómatas programables, sensores, etc. para lograr el control de dicho proyecto que fue presentado como prototipo en forma de maqueta. Teniendo como autor del mismo a Cristóbal Zelaya.

- En la Universidad Nacional Autónoma UNAN-León los Lic. Arnoldo José Contreras y Lic. Jairo José Martínez Jirón presentaron el proyecto Modelación de un Sistema Automatizado para la administración del Invernadero del campus agropecuario de la UNAN-León (INVERNADERO INTELIGENTE), en el que daban una solución a la problemática de una administración inadecuada del invernadero ubicado en la Escuela de Agroecología de ésta Universidad.

- Otro proyecto similar fue realizado en La Universidad de Ingeniería UNI-Managua, Carlos Ruíz presentó un proyecto de sistema de iluminación para un edificio inteligente.

1.2 Planteamiento del Problema

Desde el comienzo de la civilización la tecnología ha estado al servicio de la humanidad. Desde la invención de la rueda, el fuego o la imprenta la tecnología ha estado ahí para facilitar la vida del hombre. Hoy en día la sociedad busca constantemente un sistema que le brinde comodidad en todos los aspectos de su vida cotidiana.

Uno de los puntos a tomar en cuenta es el incremento en el precio de la energía en Nicaragua. Para saber cuántos kilovatios (KW) se consumen al mes en un hogar, se debe conocer el consumo por hora de cada aparato y multiplicarlo por la cantidad de horas que se utiliza al día y la cantidad de días que se utiliza al mes.

Una bujía NO ahorradora consume por hora 0.10 KW, en un uso promedio de 5 horas al día por 30 días consumiría 15KW al mes. Mientras que una bujía ahorradora puesta en un cuarto consume 0.02 KWh al ser utilizada por un período de 5 horas al día por 30 días y el consumo final de ésta al mes sería 3KW.

En el caso de los abanicos, un abanico promedio consume 0.05KWh haciendo uso del mismo por un período de 8 horas al día por 30 días, el consumo final de éste electrodoméstico sería de 12KW al mes. El consumo promedio de una casa en la que se utilizan pocos electrodomésticos es de 133 KW. Actualmente a los nicaragüenses que consumen menos de 150KW al mes se les aplica subsidio energético. Ver Anexo 3: Consumo energético de una casa promedio en Nicaragua. Para mayor detalle.

Por todo lo anterior es necesario que siempre se cuide que los equipos no estén encendidos cuando no estén en uso y para ello en la mayoría de los hogares no se tiene más que el elemento visual.

1.3 Justificación

La energía eléctrica es vital para el desarrollo de Nicaragua. La existencia de este servicio garantiza la automatización de los sistemas productivos, lo que dinamiza la economía, tanto en la industria como en las actividades comerciales. En nuestros hogares, resulta indispensable su utilización para realizar la mayor parte de las actividades cotidianas, mejorando nuestra calidad de vida.

Por esta razón nosotros decidimos crear un sistema que permita administrar la energía de un determinado espacio cerrado (cuarto) en base a la utilización de un sensor de proximidad ultrasónico.

1.3.1 Originalidad

En Nicaragua son pocos los equipos ligados al área tecnológica que se han creado para solventar el problema del ahorro en el consumo energético, aunque existen algunas opciones a nivel eléctrico para ahorrar energía como luces LED y lámparas ahorrativas, la mejor solución que hemos encontrado es la automatización de las luces eléctricas ya que muchas veces por descuido olvidamos apagarlas lo que ocasiona un mayor gasto energético lo que conlleva a que tengamos una cuota más alta en el recibo de la energía eléctrica.

1.3.2 Alcance

Una vez instalado la placa Arduino a las conexiones eléctricas de la casa se podrán brindar los siguientes servicios:

- Circuito que conectado a un Sensor ultrasónico colocado en un área ofrezca cobertura, sea capaz de medir y procesar la Distancia o bien la presencia de un individuo para activar un módulo relé que encadenara corriente a diversos dispositivos electrónicos conectados a este.

- Automatizar las luces y electrodomésticos conectados a la red electica de Arduino.

1.3.3 Producto

Se entregará un equipo de uso simple para el usuario, que incorpore todas las funcionalidades que se requieren para brindar la automatización de las luces eléctricas del hogar, con lo que obtendremos el beneficio de un mejor control del gasto energético.

1.3.4 Impacto

Con la implementación de Arduino en nuestros hogares, negocios, etc. podremos tener un gran ahorro de energía y dinero, así como un mejor uso de nuestro sistema eléctrico ya que al tener este sistema no necesitaremos preocuparnos todo el tiempo por las luces encendidas y brindará a los usuarios preferencias de configuración para cada necesidad.

1.4 Objetivos

1.4.1 Objetivo General

Desarrollar un equipo que active el flujo eléctrico de un local de forma automática mientras existan personas presentes a través de la utilización de un sensor ultrasónico para prevenir el gasto innecesario de energía eléctrica.

1.4.2 Objetivos Específicos

- Crear el equipo para la administración del flujo energético utilizando la plataforma Arduino y un sensor de proximidad ultrasónico.
- Permitir la configuración del tiempo en que el flujo eléctrico se mantiene activado una vez que se detecta la presencia de alguien en el local.
- Evaluar el desempeño del equipo a través de su instalación en una habitación durante un período de pruebas.

MARCO TEÓRICO

2.1 ARDUINO

Arduino es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares.

El hardware consiste en una placa con un microcontrolador Atmel AVR y puertos de entrada/salida. Los microcontroladores más usados son el Atmega168, Atmega328, Atmega1280, ATmega8 por su sencillez y bajo coste que permiten el desarrollo de múltiples diseños. Por otro lado el software consiste en un entorno de desarrollo que implementa el lenguaje de programación Processing/Wiring y el cargador de arranque que es ejecutado en la placa.

Arduino se puede utilizar para desarrollar objetos interactivos autónomos o puede ser conectado a software tal como Adobe Flash, Processing, Max/MSP, Pure Data. Las placas se pueden montar a mano o adquirirse. El entorno de desarrollo integrado libre se puede descargar gratuitamente.

Arduino puede tomar información del entorno a través de sus entradas analógicas y digitales, puede controlar luces, motores y otros actuadores. El microcontrolador en la placa Arduino se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing). Los proyectos hechos con Arduino pueden ejecutarse sin necesidad de conectar a un computador.

El proyecto Arduino recibió una mención honorífica en la categoría de Comunidades Digital en el Prix Ars Electrónica de 2006.

2.2 Aplicaciones Con Arduino

El módulo Arduino ha sido usado como base en diversas aplicaciones electrónicas:

- Xoscillo: Osciloscopio de código abierto.
- Equipo científico para investigaciones.
- Arduinome: Un dispositivo controlador MIDI.
- OBDuino: Un económetro que usa una interfaz de diagnóstico a bordo que se halla en los automóviles modernos.
- Humane Reader: dispositivo electrónico de bajo coste con salida de señal de TV que puede manejar una biblioteca de 5000 títulos en una tarjeta microSD.

- The Humane PC: equipo que usa un módulo Arduino para emular un computador personal, con un monitor de televisión y un teclado para computadora.
- Ardupilot: software y hardware de aeronaves no tripuladas.
- ArduinoPhone: un teléfono móvil construido sobre un módulo Arduino.

2.3 Lenguaje de Programación Arduino

La plataforma Arduino se programa mediante el uso de un lenguaje propio basado en el lenguaje de programación de alto nivel Processing. Sin embargo, es posible utilizar otros lenguajes de programación y aplicaciones populares en Arduino, debido a que Arduino usa la transmisión serial de datos soportada por la mayoría de los lenguajes mencionados. Para los que no soportan el formato serie de forma nativa, es posible utilizar software intermediario que traduzca los mensajes enviados por ambas partes para permitir una comunicación fluida. Algunos ejemplos son:

- 3DVIA Virtools: aplicaciones interactivas y de tiempo real.
- Adobe Director
- BlitzMax (con acceso restringido)
- C
- C++ (mediante libSerial o en Windows)
- C#
- Cocoa/Objective-C (para Mac OS X)
- Isadora (Interactividad audiovisual en tiempo real)
- Instant Reality (X3D)
- Java
- Liberlab (software de medición y experimentación)
- Mathematica
- Matlab
- MaxMSP: Entorno gráfico de programación para aplicaciones musicales, de audio y multimedia
- Minibloq: Entorno gráfico de programación, corre también en las computadoras OLPC
- Perl
- Php
- Physical Etoys: Entorno gráfico de programación usado para proyectos de robótica educativa
- Processing
- Pure Data
- Python
- Ruby

- Scratch for Arduino (S4A): Entorno gráfico de programación, modificación del entorno para niños Scratch, del MIT)
- Squeak: Implementación libre de Smalltalk
- SuperCollider: Síntesis de audio en tiempo real
- VBScript
- Visual Basic .NET
- VVVV: Síntesis de vídeo en tiempo real

2.4 Microcontroladores

Recibe el nombre de controlador el dispositivo que se emplea para el control de uno o varios procesos.

Hace tres décadas, los controladores se construían exclusivamente con componentes de lógica discreta, posteriormente se emplearon los microprocesadores, que se rodeaban con chips de memoria y E/S sobre una tarjeta de circuito impreso. En la actualidad, todos los elementos del controlador se han podido incluir en un chip, el cual recibe el nombre de microcontrolador. Consiste principalmente en una computadora, contenida en un circuito integrado.

Un microcontrolador es un circuito integrado de alta escala de integración que incorpora la mayor parte de los elementos que configuran un controlador.

- Procesador o CPU
- Memoria RAM
- Memoria ROM
- Líneas de E/S para comunicarse con el exterior.

Las CPU´s tendiendo al tipo de instrucciones que utilizan pueden clasificarse en:

- CISC: (Complex Instruction Set Computer) Computadores de juego de instrucciones complejo, que disponen de un repertorio de instrucciones elevado (80, 100 o más), algunas de ellas muy sofisticadas y potentes, pero que como contrapartida requieren muchos ciclos de máquina para ejecutar las instrucciones complejas.
- RISC: (Reduced Instruction Set Computer) Computadores de juego de instrucciones reducido, en los que el repertorio de instrucciones es muy reducido, las instrucciones son muy simples y suelen ejecutarse en un ciclo máquina. Además los RISC deben tener una estructura pipe-line y ejecutar todas las instrucciones a la misma velocidad.
- SISC: (Specific Instriction Set Computer) Computadores de juego de instrucciones específico. En los microcontroladores destinados a aplicaciones muy concretas, el juego de instrucciones, además de

ser reducido, es "específico", es decir, las instrucciones se adaptan a las necesidades de la aplicación prevista.

El siguiente diagrama muestra un diagrama a bloques general de un microcontrolador:

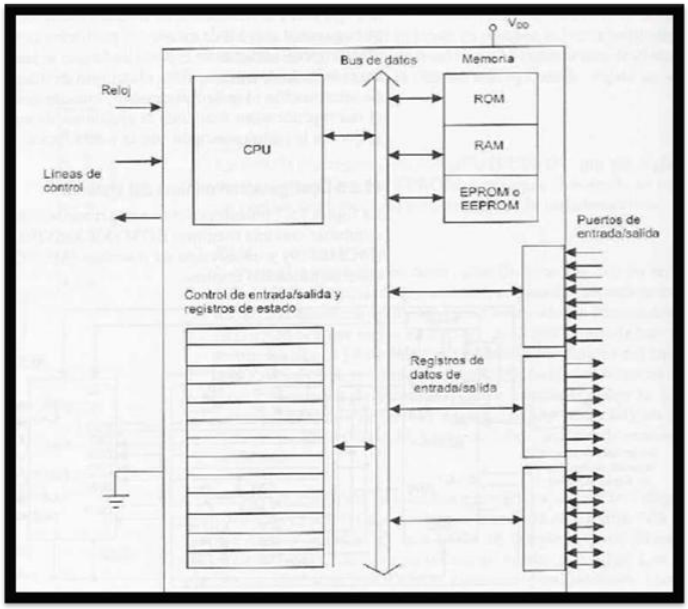


Figura 1: Diagrama de un Microcontrolador

Un microcontrolador tiene terminales para la conexión externa de entradas y salidas, alimentación eléctrica y señales de reloj y de control. Las conexiones de entrada y salida se agrupan en unidades denominadas puertos de entrada/salida. Por lo general estos puertos son de 8 bits. Los puertos pueden ser solo de entrada o de salida, pero también bidireccionales para que funcionen tanto como entradas o como saldas.

2.5 Sistema Domótico

El sistema domótico permite la automatización del hogar o de cualquier espacio de trabajo. Dicha automatización, sea cual sea su grado y las tecnologías empleadas, se conoce como domótica.

La domótica aporta a la vivienda tradicional la posibilidad de controlar y gestionar de forma eficiente los sistemas existentes y equipos ya instalados (Sistemas de alarma, TV, teléfono, agua, cocina, refrigerador, eléctrico), mediante un sistema de gestión técnica inteligente, con el objetivo de permitir una mejor calidad de vida al usuario de dicha vivienda.

Las principales áreas socio-técnicas y sus funciones que incluyen la domótica son:

- **Automatización y control:** Abrir, cerrar, apagar, encender, regular dispositivos y actividades domésticas como iluminación, climatización, persianas, toldos, puertas, ventanas, cerraduras, riego, electrodomésticos, suministro de agua, gas, electricidad.
- **Gestión energética**: Conexión de dispositivos de calefacción y aire acondicionado según criterios de ahorro y confort, complemento de control de toldos y persianas para aprovechamiento de las energías naturales, control de alumbrados, racionalización de cargas eléctricas.
- **Seguridad:** Vigilancia automática de personas, bienes, e incidencias y averías, alarmas de intrusión y cámaras de vigilancia, alarmas personales, alarmas técnicas de incendio, humo, agua, gas, fallo de suministro eléctrico.

Para el presente proyecto se tomó en cuenta las características que brindan los sistemas domóticos que existen en el mercado, por lo que se optó por diseñar una electrónica que cumpla con las necesidades para la implantación del proyecto.

2.6 Señales Electrónicas

Es la representación de un fenómeno físico o estado material a través de una relación establecida; las entradas y salidas de un sistema electrónico serán señales variables.

En electrónica se trabaja con variables que toman la forma de Tensión o corriente estas se pueden denominar comúnmente señales. Las señales primordialmente pueden ser de dos tipos:

- **Variable analógica**: Son aquellas que pueden tomar un número infinito de valores comprendidos entre dos límites. La mayoría de los fenómenos de la vida real dan señales de este tipo (presión, temperatura, etc.).

- **Variable digital:** También llamadas variables discretas, entendiéndose por estas, las variables que pueden tomar un número finito de valores. Por ser de fácil realización los componentes físicos con dos estados diferenciados, es este el número de valores utilizado para dichas variables, que por lo tanto son binarias. Siendo estas variables más fáciles de tratar (en lógica serían los valores V y F) son los que generalmente se utilizan para relacionar varias variables entre sí y con sus estados anteriores.

2.7 Dispositivos Analógicos

Algunos ejemplos de estos son:

- Amplificador operacional: amplificación, regulación, conversión de señal, conmutación.
- condensador: almacenamiento de energía, filtrado, adaptación impedancias.
- Diodo: rectificación de señales, regulación, multiplicador de tensión.
- Diodo Zener: regulación de tensiones.
- Inductor: adaptación de impedancias.
- Potenciómetro: variación de la corriente eléctrica o la tensión.
- Relé: apertura o cierre de circuitos mediante señales de control.
- Resistor o Resistencia: división de intensidad o tensión, limitación de intensidad.
- Transistor: amplificación, conmutación.

2.8 Dispositivos Digitales

- Biestable: control de sistemas secuenciales.
- Memoria: almacenamiento digital de datos.
- Microcontrolador: control de sistemas digitales.
- Puerta lógica: control de sistemas combinacionales.

2.9 Sensor de Proximidad Ultrasónico

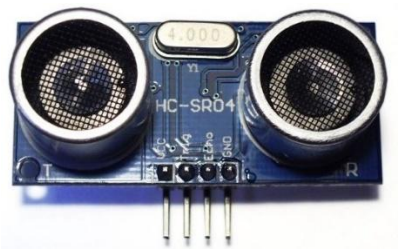


Figura 2: Sensor de Proximidad Ultrasónico

Un sensor de proximidad es un transductor que detecta objetos o señales que se encuentran cerca del elemento sensor. Existen varios tipos de sensores de proximidad según el principio físico que utilizan. Los más comunes son los interruptores de posición, los detectores capacitivos, los inductivos y los fotoeléctricos, como el de infrarrojos.

Los sensores de ultrasonidos son detectores de proximidad que trabajan libres de roces mecánicos y que detectan objetos a distancias de hasta 8m. El sensor emite impulsos ultrasónicos. Estos reflejan en un objeto, el sensor recibe el eco producido y lo convierte en señales eléctricas, las cuales son elaboradas en el aparato de valoración. Estos sensores trabajan solamente en el aire, y pueden detectar objetos con diferentes formas, superficies y de diferentes materiales. Los materiales pueden ser sólidos, líquidos o polvorientos, sin embargo, han de ser deflectores de sonido. Los sensores trabajan según el tiempo de transcurso del eco, es decir, se valora la distancia temporal entre el impulso de emisión y el impulso del eco.

Este sensor al no necesitar el contacto físico con el objeto ofrece la posibilidad de detectar objetos frágiles, como pintura fresca, además detecta cualquier material, independientemente del color, al mismo alcance, sin ajuste ni factor de corrección. Los sensores ultrasónicos tienen una función de aprendizaje para definir el campo de detección, con un alcance mínimo y máximo de precisión de 6mm. El problema que presentan estos dispositivos son las zonas ciegas y el problema de las falsas alarmas. La zona ciega es la zona comprendida entre el lado sensible del detector y el alcance mínimo en el que ningún objeto puede detectarse de forma fiable.

2.9.1 Descripción del Sensor

El HC-SR04 es un sensor de distancias por ultrasonidos capaz de detectar objetos y calcular la distancia a la que se encuentra en un rango de 2 a 450 cm. El sensor funciona por ultrasonidos y contiene toda la electrónica encargada de hacer la medición. Su uso es tan sencillo como enviar el pulso de arranque y medir la anchura del pulso de retorno. De muy pequeño tamaño, el HC-SR04 se destaca por su bajo consumo, gran precisión y bajo precio por lo que está reemplazando a los sensores polaroid en los robots más recientes. Es de fácil uso y programación con las placas de Arduino y microcontroladores.

2.9.2 Características

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

2.9.3 Pines de Conexión

- VCC
- Trig (Disparo del ultrasonido)
- Echo (Recepción del ultrasonido)
- GND

Distancia = {(Tiempo entre Trig y el Echo) * (V.Sonido 340 m/s)}/2

DISEÑO METODOLÓGICO

3.1 Materiales Utilizados

Para llevar a cabo la realización de nuestro proyecto hicimos uso de los siguientes materiales:

3.1.1 Materiales Hardware

Tabla 1 Materiales Hardware

Materiales	Descripción	Costo
Placa Arduino Mega 2560 Display LCD Breadboard o Protoboard Jumper Wires hembra y macho	Estos materiales los adquirimos de un Kit Arduino que contiene varios de los elementos que necesitábamos para nuestro proyecto.	$56
Módulo Relay	Control de la corriente eléctrica.	$9
Sensor de Proximidad Ultrasónico HC-SR04	Sensor utilizado para la creación de nuestro producto	$10
Lámparas de 25w	Empleadas para la presentación del Producto Final	C$35
Keypad	Teclado Matricial para establecer un temporizador para la manipulación del sensor.	$15
Laptop	Procesador Intel Core i5, 2.50GHz. RAM 4GB	$1000

Caja de cartón solido 8.4 x 2 pulgadas	Caja que contiene los elementos que conforman nuestro equipo. Forro del Equipo	
Silicona	Pegar el Equipo	C$12
Foamy	Forro del Equipo	C$20

3.1.2 Materiales Software

Tabla 2: Materiales Software

Software	**Descripción**
Sistema Operativo Windows 8.1	Sistema Operativo de 64 bits para instalar los programas necesarios para la realización de este proyecto.
Arduino	Compilador de software libre que utilizamos para programar nuestro proyecto.

3.2 Etapas del Trabajo

3.2.1 Etapa I: Recopilación de información

Para la realización de nuestro proyecto hemos dispuesto de las técnicas básicas de un proceso de investigación recolectando información teórica y práctica de diversas fuentes.

3.2.2 Etapa II: Diseño del equipo

Descripción:

En ésta etapa dimos inicio al diseño del equipo para la administración del flujo energético. Hicimos uso de un sensor en conjunto a la plataforma Arduino y definimos el esquema lógico de nuestro proyecto.

3.2.3 Etapa III: Creación y configuración del equipo

Descripción:

En ésta etapa realizamos la creación y configuración del tiempo en que el flujo eléctrico se mantiene activado una vez que se detecta la presencia de alguien en el local.

3.2.4 Etapa IV: Pruebas del equipo

Descripción:

En ésta etapa evaluamos el desempeño de éste a través de su instalación en una habitación durante un período de pruebas.

3.2.5 Etapa V: Informe Final

Descripción:

Redactamos y estructuramos el documento final en el que explicamos paso a paso el proceso de nuestro proyecto y sus resultados.

RESULTADOS

En esta sección se aborda el desarrollo de las diversas etapas del proyecto.

4.1 Diseño del Equipo

Para dar inicio a nuestro proyecto lo principal fue diseñar un esquema lógico en el que plasmamos la idea del equipo que deseábamos crear.

4.1.1 Funcionamiento del equipo

En este apartado explicaremos la estructura interna y el funcionamiento del equipo final.

4.1.2 Definición del esquema de funcionamiento lógico

El funcionamiento básico del equipo se basa en la detección de movimiento; un sensor ultrasónico ubicado en una zona de cobertura en la que los usuarios suelen estar expuestos. Al momento en el que el sensor encuentre movimiento activará inmediatamente el flujo eléctrico del local durante un tiempo determinado. Gracias a las funcionalidades definidas en el teclado este tiempo puede variar de acuerdo a las preferencias del usuario. El código que se ejecuta de forma recursiva nos permite reutilizar el tiempo delimitado por el usuario hasta que este disponga detenerlo.

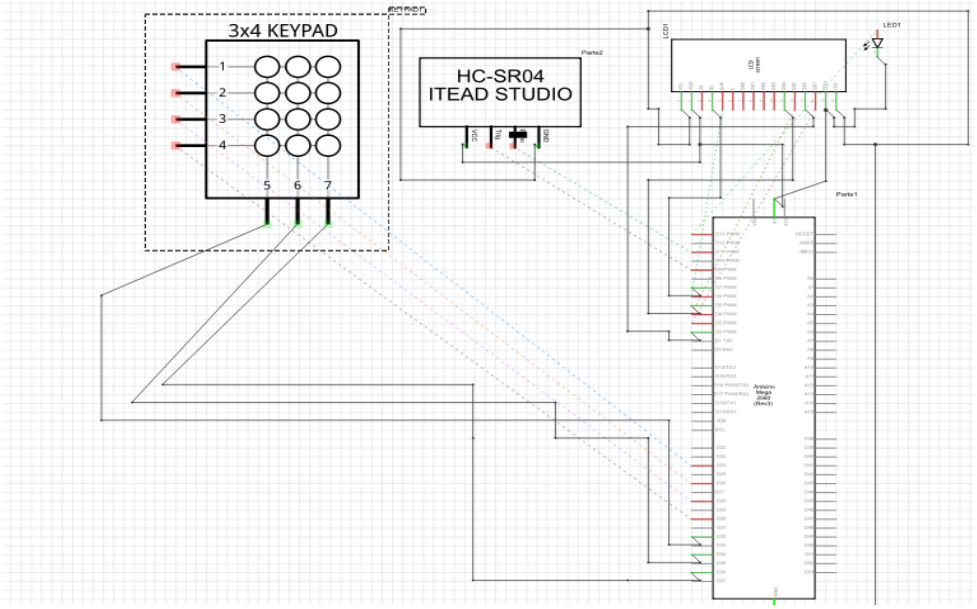
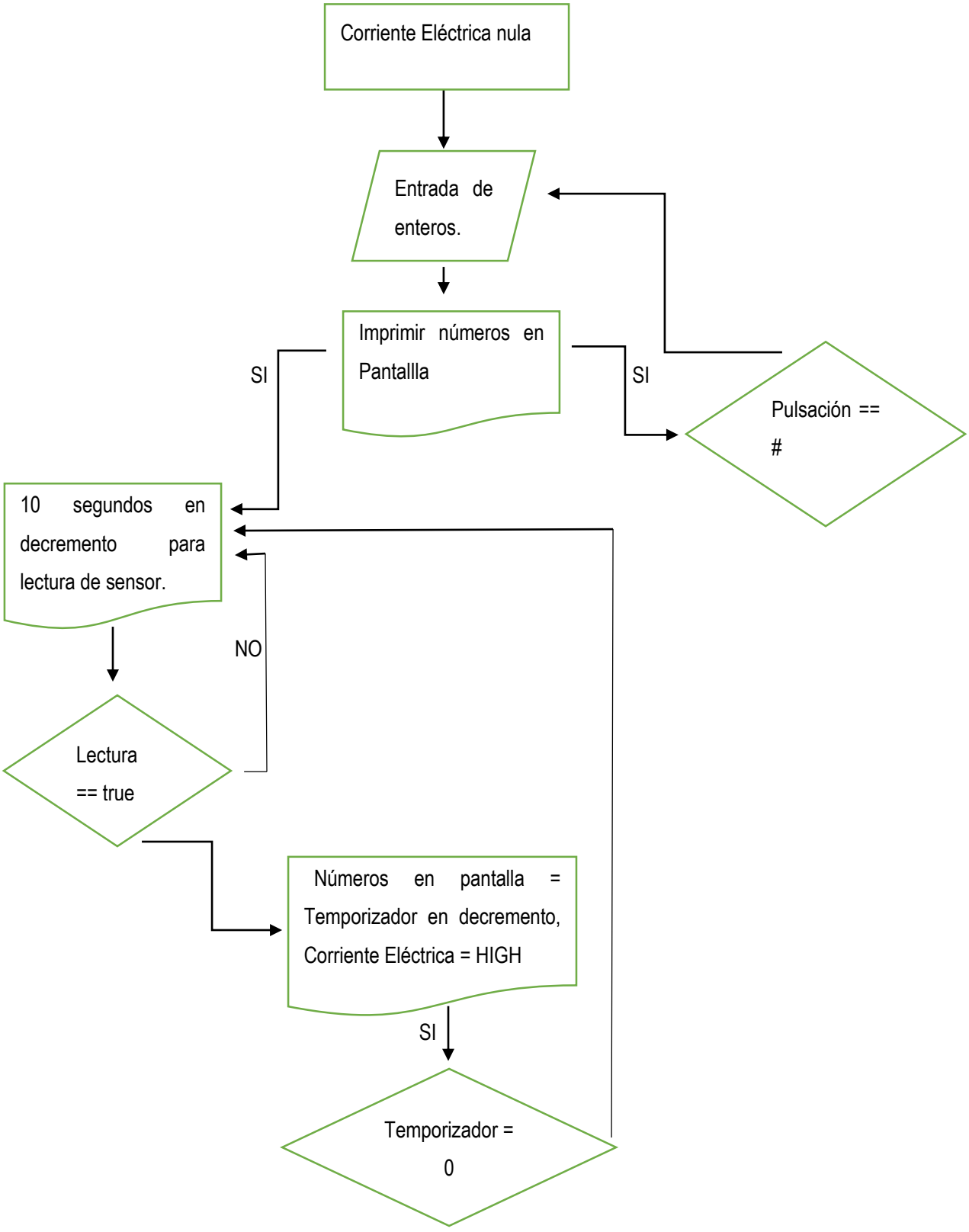


Figura 3: Esquema Funcionamiento Lógico

Una vez que el temporizador termina definimos un tiempo estático, este tiempo se utiliza para otorgar un tiempo de realizar una nueva lectura antes de cortar el flujo energético a fin de que si existe alguna persona en el local el temporizador se reinicie. Si en dicho tiempo no se detecta una presencia en el medio el tiempo concluirá y se procederá al apagado de los equipos.



4.2 Creación y configuración del equipo

En la siguiente imagen se muestra el diseño físico de nuestro equipo y las conexiones necesarias para su funcionamiento. En él se ven reflejadas las conexiones correspondientes al LCD, Keypad y el Sensor de Proximidad Ultrasónico.

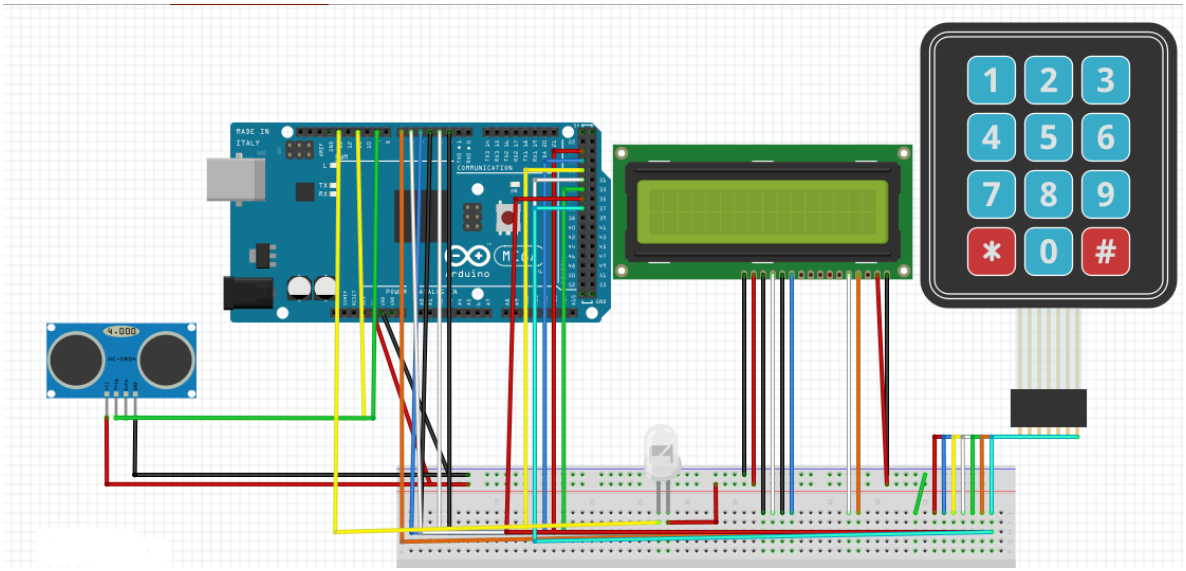


Figura 4: Esquema de Funcionamiento Físico

Un elemento importante de nuestro proyecto es el sensor de proximidad ultrasónico. A continuación se muestra el fragmento de código con el que el sensor realiza las funciones que queremos que realice.

```
KeyPress = keysPins.getKey();
if (int(KeyPress) != 0  ) {   // siempre realizamos todo con la condición que las pulsaciones sean diferentes de 0.
  switch(KeyPress) {   // entramos a un switch capturando lo que tenga nuestra variable KeyPress
  case '#':     // si es # detenemos todo e inicializamos la cadena a 0 nuevamente, detenemos el flujo energético.
    digitalWrite(PinRele, HIGH);    // Se pasa un valor alto al pin correspondiente a la variable PinRele
    currentTimeValue[0] = '0';   // Igualamos a 0 todos los valores indexados de la cadena
    currentTimeValue[1] = '0';
    currentTimeValue[2] = '0';
    currentTimeValue[3] = '0';
    showTime();   // invocamos un método que simplemente muestra los valores del array
    lpcount = 0;
```

```
    timerSeconds = 0;
    aux = 11;
    break;

  case '*':  // si es * realizamos las siguientes acciones
    tempVal[0] = currentTimeValue[0];
    tempVal[1] = currentTimeValue[1];
    tempVal[2] = 0;
    timerSeconds = atol(tempVal) * 60;

    tempVal[0] = currentTimeValue[2];
    tempVal[1] = currentTimeValue[3];
    tempVal[2] = 0;
    timerSeconds = timerSeconds + atol(tempVal);
    break;

  default:
    currentTimeValue[0] = currentTimeValue[1];
    currentTimeValue[1] = currentTimeValue[2];
    currentTimeValue[2] = currentTimeValue[3];
    currentTimeValue[3] = KeyPress;
    showTime();
    break;
  }
}
if(int(KeyPress) != 0) {
  if(KeyPress == '#') {
    digitalWrite(PinRele, HIGH);
    displayCodeEntryScreen();
    currentTimeValue[0] = '0';
    currentTimeValue[1] = '0';
    currentTimeValue[2] = '0';
    currentTimeValue[3] = '0';
    showTime();
```

```
      lpcount = 0;
      timerSeconds = 0;
    }
  }
  else {
    if(lpcount > 9 && timerSeconds > 0) {
      lpcount = 0;
      --aux;
      decrement(); // invocamos un evento que simplemente nos muestra la variable aux en decremento
      digitalWrite(trigPin, LOW);    //  este cambio entre LOW Y HIGH calibra la precisión del sensor entre disparos de
ráfaga
      delayMicroseconds(2);

      digitalWrite(trigPin, HIGH);
      delayMicroseconds(10);

      digitalWrite(trigPin, LOW);

      duration = pulseIn(echoPin, HIGH);
      distance = microsecondsToCentimeters(duration);    // la formula la obtenemos de un ejemplo propio de arduino
      if (distance >= 20 || distance <= 0){
      }
      else {
        digitalWrite(PinRele, LOW);
        while(timerSeconds > 0) {
          CountDown();
          lcd.clear();
          lcd.setCursor(0,0);
          lcd.print("Distancia:");
          lcd.setCursor(10,0);
          lcd.print(distance);
          lcd.setCursor(1,1);
          lcd.print("Restante:");
          lcd.setCursor(10,1);
```

```
        char count[6];
        sprintf(count,"%d:%.2d", (timerSeconds/60), (timerSeconds - ((timerSeconds/60)*60)));
        lcd.print(count);

        KeyPress = keysPins.getKey();
        if(int(KeyPress) != 0) {
          if(KeyPress = '#') {
            break;
          }
        }
      }
      tempVal[0] = currentTimeValue[0];
      tempVal[1] = currentTimeValue[1];
      tempVal[2] = 0;

      timerSeconds = atol(tempVal) * 60;

      tempVal[0] = currentTimeValue[2];
      tempVal[1] = currentTimeValue[3];
      tempVal[2] = 0;

      timerSeconds = timerSeconds + atol(tempVal);
      aux = 11;
    }
    if(aux == 0) {
      digitalWrite(PinRele, HIGH);
      aux = 11;
    }
  }
  ++ lpcount;
  delay(100);
 }
}
```

Otro de los elementos importantes es el teclado. El fragmento de código en cual obtenemos el valor numérico que se introduce en el teclado y lo pasamos al temporizador es el siguiente:

```
    case '*':  // si es * realizamos las siguientes acciones
  tempVal[0] = currentTimeValue[0];
  tempVal[1] = currentTimeValue[1];
  tempVal[2] = 0;
  timerSeconds = atol(tempVal) * 60;   // aquí convertimos la cadena a su valor numérico

  tempVal[0] = currentTimeValue[2];
  tempVal[1] = currentTimeValue[3];
  tempVal[2] = 0;
  timerSeconds = timerSeconds + atol(tempVal); // aquí también para los segundos
  break;

default:
  currentTimeValue[0] = currentTimeValue[1];
  currentTimeValue[1] = currentTimeValue[2];
  currentTimeValue[2] = currentTimeValue[3];
  currentTimeValue[3] = KeyPress;
  showTime();
  break;
```

En el código anterior como mencionamos antes pasamos el valor de una cadena o bien un array de caracteres de longitud 3, gracias a la instancia que creamos del tipo KeyPad podemos obtener las pulsaciones del teclado en tiempo real y pasarlas 1 a 1 a otra cadena de caracteres llamado currentTimeValue la cual se posiciona de manera indexada de izquierda a derecha. Por ultimo invocamos un método llamado showTime el cual muestra los valores del estado actual.

4.2.1 Manipulación del equipo

Establecimos una herramienta (Teclado) para manipular el temporizador con el que el usuario podrá controlar el tiempo de actividad de la corriente eléctrica de la placa Arduino.

4.2.1.1 Funcionamiento del Temporizador asociado a nuestro equipo

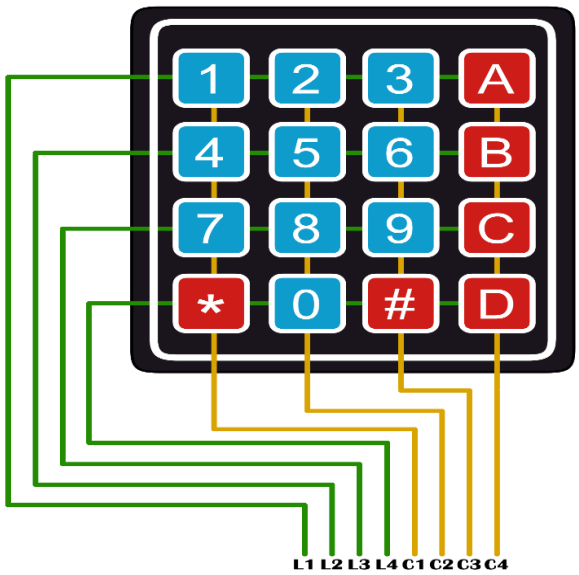


Figura 5: Keypad

Teclado matricial 4x4 en el que para el funcionamiento de nuestro equipo utilizamos 4x3 (correspondientes a las filas y las columnas), dicho tablero corresponde a una entrada de tiempo en formato mm/ss (minutos y segundos) con el cual se define el tiempo en el que permanecerá en actividad la corriente eléctrica y posteriormente dos botones para inicio (*) y parada (#) de dicho temporizador.

4.2.1.2 Conexión del módulo Relay

Para realizar la transición de energía de la placa hacia el exterior hicimos uso del módulo Relay el cual nos permite conmutar entre alto y bajo voltaje.

Su funcionamiento se basa en el fenómeno electromagnético. Cuando la corriente atraviesa la bobina, produce un campo magnético que magnetiza un núcleo de hierro dulce (ferrita). Este atrae al inducido que fuerza a los contactos a tocarse. Cuando la corriente se desconecta vuelven a separarse.

4.2.1.3 Sistema de enfriamiento del equipo

Hicimos uso de un cooler para la ventilación del equipo de esta manera evitamos que se recaliente y que eso afecte su correcto funcionamiento.

4.2.1.4 Producto Terminado

Figura 6: Producto Terminado

4.3 Pruebas del equipo

Para dar inicio a éste proyecto realizamos pruebas básicas de los complementos por separado (sensor, pantalla LCD, etc.) estudiando, analizando y aprendiendo el funcionamiento de lo que estará expuesto al entorno (sensor de proximidad ultrasónico).

Una vez habiendo obtenido los conocimientos necesarios procedimos a hacer las pruebas pertinentes para el funcionamiento final del equipo colocándolo en una habitación para comprobar su alcance máximo, su radio y precisión al momento de detectar movimiento.

Colocamos el equipo en un cuarto, lo conectamos a la corriente eléctrica, conectamos al equipo dos luces eléctricas y un abanico a como se muestra en la siguiente imagen:

Figura 7: Pruebas del Equipo

La prueba tuvo una duración de 10 minutos en los que el equipo detectaba la proximidad de los cuerpos en movimiento lo que mantenía encendidas las luces y el abanico, luego de ese período al dejar de detectar movimiento ambos aparatos se apagaron.

Figura 8: Pruebas del Equipo

Al realizar las pruebas pertinentes de nuestro dispositivo final observamos el comportamiento del sensor integrado en nuestro sistema; en condiciones óptimas evaluamos su funcionamiento en conjunto con los demás componentes hardware. Observamos que el sensor se vuelve más sensible al determinar el máximo alcance que este ofrece, en otra instancia; el sensor reacciona a pulsos en el entorno, tales como vibraciones fuertes, también a cambios repentinos de luz.

ASPECTOS FINALES

5.1 Conclusiones

Realizar éste proyecto nos ha dado la oportunidad de aprender mucho sobre el mundo de la domótica y un poco de electrónica en general.

Se logró crear un equipo que brinda a los usuarios un sistema automatizado que ayuda a tener un mejor control del flujo energético utilizando el sensor de proximidad ultrasónico cumpliendo con los objetivos planteados al iniciar el proyecto. Este equipo fue desarrollado utilizando lenguaje de programación Arduino y una serie de herramientas hardware necesarias para el funcionamiento del mismo. Al realizar las pruebas comprobamos el funcionamiento eficaz del producto.

La domótica facilita la gestión integrada de los diferentes dispositivos del hogar Además de esto, actúa de forma inteligente ya que permite programar diferentes escenarios que se ajusten a nuestras necesidades.

5.2 Recomendaciones

RECOMENDACIONES PARA EL USO DEL EQUIPO

- Realizar un test o bien verificar el correcto funcionamiento eléctrico en el hogar antes de la instalación del equipo.
- El sensor utilizado en este proyecto funciona perfectamente analizando las condiciones y dimensiones de un entorno definido; no obstante, si se necesita una mayor cobertura se recomienda el uso del sensor PIR ya que la solución actual funciona adecuadamente cuando el alcance de detección es de 4.5 mts.

RECOMENDACIONES AL DEPARTAMENTO

- Fomentar el desarrollo e implementación de proyectos relacionados a la domótica, ya que esta ayuda a desarrollar aún más la creatividad del estudiante.
- Profundizar la implementación de Arduino a un mayor nivel como por ejemplo bajo administración con servidores y protección de acceso en la red.

BIBLIOGRAFIA

Artero, Ó. T. (18 de Enero de 2013). *Arduino. Curso práctico de formación.* México: RC Libros. Obtenido de history - VIDEOPLACE.

Bräunl, T. (2012). *"Sparkfun Inventor's Kit" para Educadores* (Second Edition ed.). Obtenido de www.wikipedia.com,tienda robotica.com

electronilab. (s.f.). Obtenido de http://electronilab.co/

Evans, B. W. (2010). *Arduino: Manual de Programación.* Obtenido de http://www.arduino.cc/en/Booklet/HomePage

freelibro. (2012). Obtenido de http://www.freelibros.org

Gasto energético Nicaragua. (21 de Febrero de 2013). Obtenido de http://www.elnuevodiario.com.ni/nacionales/313519-batalla-150-kilovatios/

Gutiérrez, J. M. (2011). *Prácticas con Arduino Nivel I.* Mexico . Obtenido de https://es.wikipedia.org/wiki/Microcontrolador

Wikipedia. (s.f.). *Microcontroladores*. Obtenido de wikipedia.org: https://es.wikipedia.org/wiki/Microcontrolador

ANEXOS

Anexo 1: Iniciando con Arduino

Como primer paso debemos descargar el software de Arduino e instalarlo en nuestro entorno de trabajo favorito (Windows-Linux).

Al terminal la instalación lanzamos la aplicación y empezamos a realizar algunas pruebas, podemos empezar cargando y corriendo los ejemplos que ya trae la aplicación.

Damos clic a la siguiente secuencia: **File > Examples > 1.Basics > Blink.**

Figura 9: Iniciando Con Arduino

Seleccionamos nuestra placa: Tools > Board

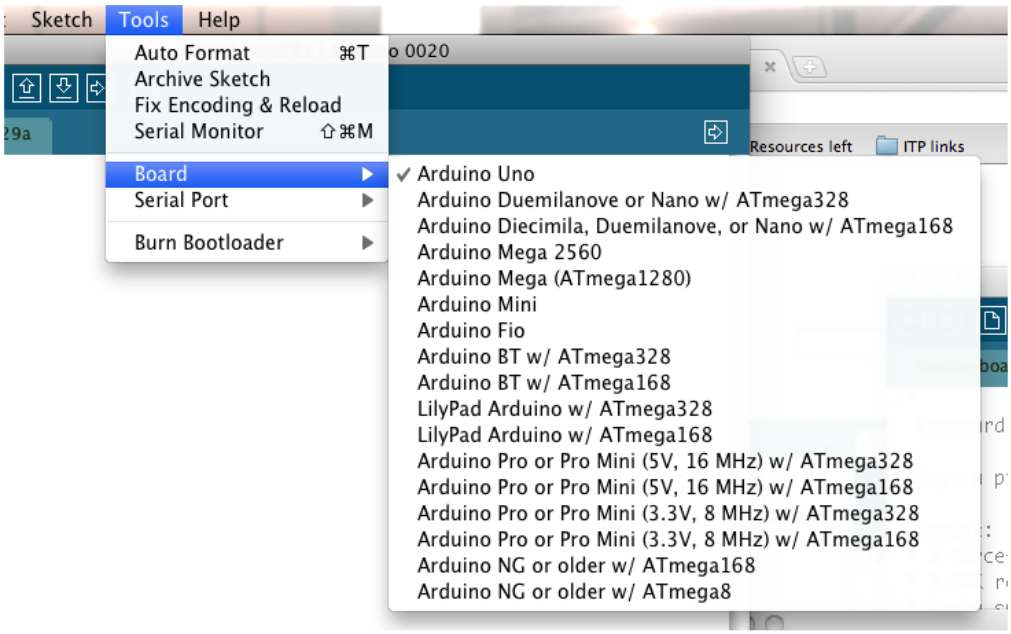


Figura 10: Iniciando Con Arduino

Seleccionamos nuestro puerto serial.

Seleccione el dispositivo de serie de la placa Arduino desde el menú Tools | Serial Port. Esto es probable que sea COM3 o mayor (COM1 y COM2 son generalmente reservados para los puertos serie de hardware). Para averiguarlo, podemos desconectar nuestra placa Arduino y volver a abrir el menú; la entrada que desaparece debe ser la placa Arduino. Volvemos a conectar el tablero y seleccionamos ese puerto serie.

Por ultimo cargamos el proyecto dando clic a la opción Upload.

Figura 11: Iniciando Con Arduino

Anexo 2: Características de la placa Arduino ATMega 2560

- Microcontrolador: ATMega 2560
- Voltaje de operación: 5V
- Voltaje de alimentación: 7-12V
- Memoria Flash: 256 Kbytes 8 Kbytes para bootloader
- Memoria SRAM: 8 KBytes
- Memoria EEPROM: 4 KBytes
- Velocidad de reloj: 16 MHz
- Digital I/O : 54
- Canales PWM: 15
- Entradas analógicas : 16 10bits
- Corriente máxima por I/O Pin: 20 mA
- USB: USB para programación
- USB Host: ATMega 2560 ADK
- Botón: Reset

Anexo 3: Consumo energético de una casa promedio en Nicaragua.

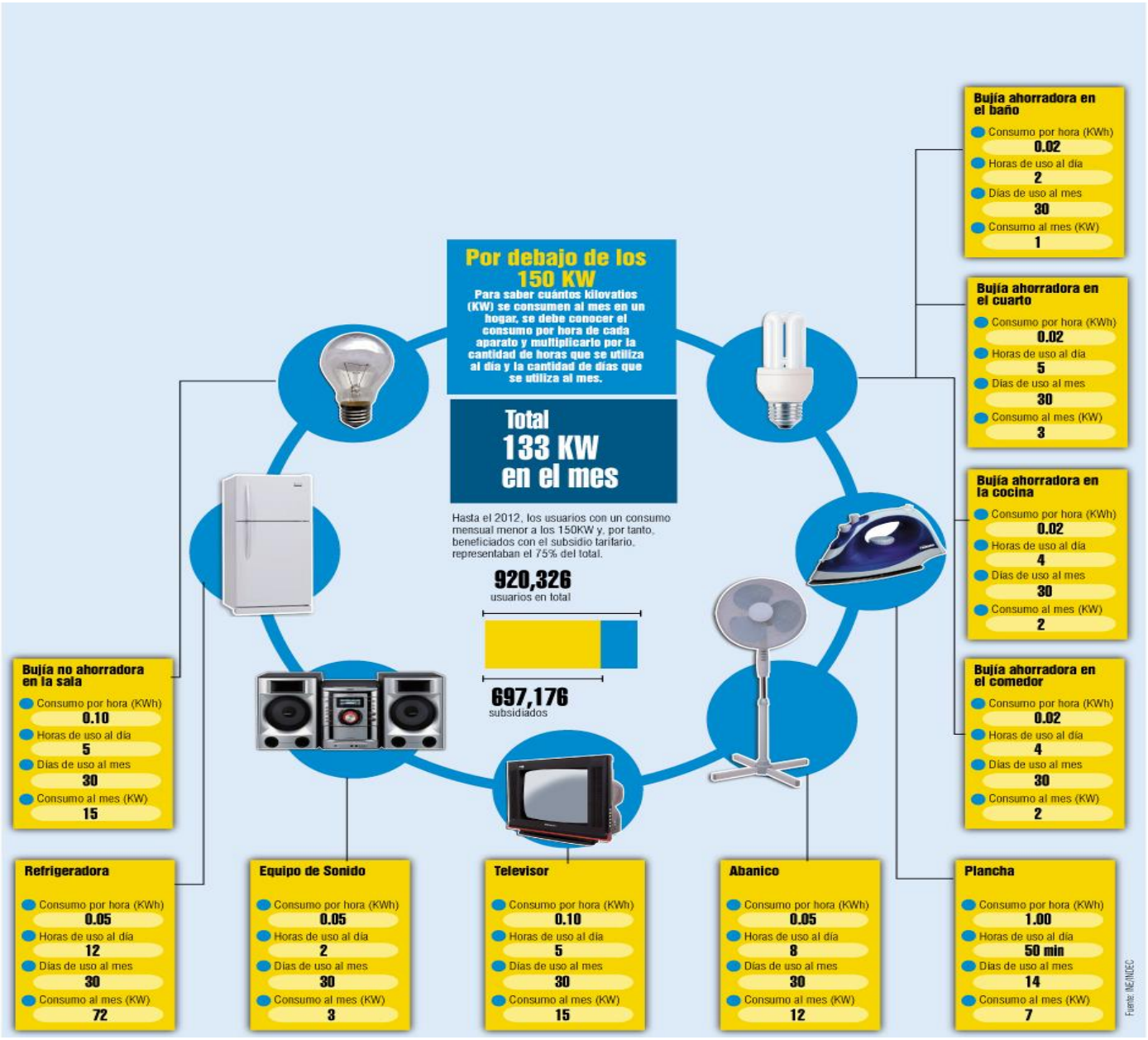


Figura 12: Consumo Energético en una Vivienda Nicaragüense. Fuente: El Nuevo Diario, 7 de marzo de 2014.

Anexo 4: Código del proyecto.

```
#include <LiquidCrystal.h>
#include <Keypad.h>

LiquidCrystal lcd (2,3,4,5,6,7);   // definimos los pines para la lcd

const byte Filas = 4;          //KeyPad de 4 filas
const byte Cols = 3;           //y 4 columnas
#define trigPin 10
#define echoPin 12
#define PinRele 13
char currentTimeValue[4];   // definiendo un array de 4 elementos para los estados del temporizador
char KeyPress;
int aux = 11;
long distance, duration;
char tempVal[3];
int timerSeconds,lpcount,keep = 0;  // variables para el cálculo del temporizador

char Keys [ Filas ][ Cols ] =
{
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'*','0','#'}
};

byte rowPins[Filas] = { 24,26,28,30 };
byte colsPins[Cols] = { 32,34,36};

Keypad keysPins = Keypad(makeKeymap(Keys),rowPins,colsPins,Filas,Cols);  // creamos un objeto de la clase Keypad y le pasamos como parametro los objetos de la matriz

void setup () {     // espacio de nombre, aqui se declaran las variables que necesitemos, de tipo pin, ETC. correspondientes a la protoboard

Serial.begin (9600);  // abrimos el canal para que se pueda realizar la transmisión por el.
lcd.begin(16,2); // definimos la cantidad de caracteres y las filas del lcd, conociendo sus especificaciones tecnicas
displayCodeEntryScreen();

pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(PinRele,OUTPUT);    // pin led modo salida
digitalWrite(PinRele, HIGH);
currentTimeValue[0] = '0';
currentTimeValue[1] = '0';
currentTimeValue[2] = '0';
currentTimeValue[3] = '0';
showTime();
}
```

```
void loop() {   // viene a ser el constructor por defecto, se ejecuta n veces
KeyPress = keysPins.getKey();
  if (int(KeyPress) != 0  ) {   // siempre realizamos todo con la condición que las pulsaciones sean diferentes de
0.
    switch(KeyPress) {   // entramos a un switch capturando lo que tenga nuestra variable KeyPress
    case '#':    // si es # detenemos todo e inicializamos la cadena a 0 nuevamente, detenemos el flujo energético.
      digitalWrite(PinRele, HIGH);
      currentTimeValue[0] = '0';
      currentTimeValue[1] = '0';
      currentTimeValue[2] = '0';
      currentTimeValue[3] = '0';
      showTime();
      lpcount = 0;
      timerSeconds = 0;
      aux = 11;
      break;

    case '*':  // si es * realizamos las siguientes acciones
      tempVal[0] = currentTimeValue[0];
      tempVal[1] = currentTimeValue[1];
      tempVal[2] = 0;
      timerSeconds = atol(tempVal) * 60;
      tempVal[0] = currentTimeValue[2];
      tempVal[1] = currentTimeValue[3];
      tempVal[2] = 0;
      timerSeconds = timerSeconds + atol(tempVal);
      break;

    default:
      currentTimeValue[0] = currentTimeValue[1];
      currentTimeValue[1] = currentTimeValue[2];
      currentTimeValue[2] = currentTimeValue[3];
      currentTimeValue[3] = KeyPress;
      showTime();
```

```
      break;
    }
  }
  if(int(KeyPress) != 0) {
    if(KeyPress == '#') {
      digitalWrite(PinRele, HIGH);
      displayCodeEntryScreen();
      currentTimeValue[0] = '0';
      currentTimeValue[1] = '0';
      currentTimeValue[2] = '0';
      currentTimeValue[3] = '0';
      showTime();
      lpcount = 0;
      timerSeconds = 0;
    }
  }
  else {
    if(lpcount > 9 && timerSeconds > 0) {
      lpcount = 0;
      --aux;
      decrement(); // invocamos un evento que simplemente nos muestra la variable aux en decremento
      digitalWrite(trigPin, LOW);   //  este cambio entre LOW Y HIGH calibra la precisión del sensor entre disparos de ráfaga
      delayMicroseconds(2);
      digitalWrite (trigPin, HIGH);
      delayMicroseconds(10);
      digitalWrite(trigPin, LOW);

      duration = pulseIn(echoPin, HIGH);
      distance = microsecondsToCentimeters(duration);     // la formula la obtenemos de un ejemplo propio de arduino
      if (distance >= 20 || distance <= 0){
      }
      else {
```

```
  digitalWrite(PinRele, LOW);
  while(timerSeconds > 0) {
    CountDown();
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Distancia:");
    lcd.setCursor(10,0);
    lcd.print(distance);
    lcd.setCursor(1,1);
    lcd.print("Restante:");
    lcd.setCursor(10,1);
    char count[6];
   sprintf(count,"%d:%.2d", (timerSeconds/60), (timerSeconds - ((timerSeconds/60)*60)));
    lcd.print(count);
    KeyPress = keysPins.getKey();
    if(int(KeyPress) != 0) {
      if(KeyPress = '#') {
        break;
      }
    }
  }
  tempVal[0] = currentTimeValue[0];
  tempVal[1] = currentTimeValue[1];
  tempVal[2] = 0;
  timerSeconds = atol(tempVal) * 60;
  tempVal[0] = currentTimeValue[2];
  tempVal[1] = currentTimeValue[3];
  tempVal[2] = 0;
  timerSeconds = timerSeconds + atol(tempVal);
  aux = 11;
 }
 if(aux == 0) {
  digitalWrite(PinRele, HIGH);
  aux = 11;
```

```
    }
   }
    ++ lpcount;
    delay(100);
  }
}

void CountDown()
{
  --timerSeconds;
  delay(1000);
}
void decrement()
{
  lcd.clear();    // limpiamos la pantalla
  lcd.setCursor(0,0);       // posicionamos el indicador de la pantalla en posicion 1 de la fila 0
  lcd.print("En espera..");
  char temp[6];
  sprintf(temp,"%d:%.2d", (aux/60), (aux - ((aux/60)*60)));
  lcd.setCursor(11,0);
  lcd.print(temp);
}
void showTime()
{
  lcd.setCursor(10,1);
  lcd.print(currentTimeValue[0]);
  lcd.print(currentTimeValue[1]);
  lcd.print(":");
  lcd.print(currentTimeValue[2]);
  lcd.print(currentTimeValue[3]);
}
void displayCodeEntryScreen()
{
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Tiempo en mm:ss:");
}
long microsecondsToCentimeters(long microseconds)
{
  // la velocidad del sonido es 340 m/s o 29 microsegundos por cm
  // The ping travels out and back, so to find the distance of the
  // object we take half of the distance travelled.
  return microseconds / 29 / 2;
}
```