UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

Sistema inalámbrico para operación a distancia de luminarias y tomacorrientes en un apartamento, por medio de un Smartphone con sistema operativo Android y comunicación vía Bluetooth.

TESIS

Para obtener el grado de:

Ingeniero mecánico electricista.

Presenta:

Luis Miguel Báez Martínez.

Director de tesis:

Dr. Alejandro Antonio Vega Ramírez.

FES Aragón, México 2015.

Índice.

Agradecimientos:

A mi madre por su apoyo incondicional en todo momento

A mi hermana Lorena Báez Martínez que me ha acompañado y apoyado siempre.

A mi asesor el Dr. Alejandro Antonio Vega Ramírez por haberme guiado en este trabajo.

A mi esposa Coral por estar a mi lado y ayudarme incondicionalmente.

A mis profesores por sus enseñanzas y su tiempo invertido para lograr formarme como ingeniero.

A mi cuñado José Carlos Goroztizaga Sánchez por haberme tendido la mano siempre que lo he necesitado.

Muchas gracias a todos por ayudarme y brindarme su apoyo.

Dedicatoria.

A mis hermanos Martina, Jorge, Lilia, Lorena y Victor.

A mis sobrinos y en especial a Armando Gómez Báez.

A mi compañero y amigo Juan Esteban Bonilla Navarrete.

A mis amigos y amigas.

A todas aquellas personas que han formado parte de mi vida a las que están presentes ahora y a las que forman parte de mis recuerdos, ocupó este espacio para dedicarles este trabajo de tesis por haber ayudado a mi formación.

Objetivo general:

Desarrollar un sistema electrónico capaz de controlar las luminarias y tomacorrientes de un apartamento, desde una aplicación en un teléfono inteligente con sistema operativo Android. Con el fin de disminuir el consumo de energía eléctrica.

Objetivos particulares:

- Realizar una comunicación entre un teléfono móvil y un circuito por medio de un puerto serie virtual Bluetooth.
- Crear una aplicación, asociada con un circuito, para poder realizar un control en luminarias y tomacorrientes.
- Tener comunicación entre el teléfono inteligente y el circuito en ambos sentidos, para controlar los interruptores y además saber si están encendidos o apagados, no importando si se manipulan directamente los contactos o por medio de la aplicación que desarrollaremos.

Introducción:

En este trabajo de tesis se desarrollara un sistema electrónico, él cuál nos permitirá controlar los interruptores de iluminación y los tomacorrientes, con el fin de ahorrar energía y pretende aumentar la comodidad en el sitio donde sea instalado, puede ser un apartamento, una habitación de hospital, una habitación de hotel, etcétera.

Sea instalado en un apartamento, nos permitirá tener a la mano el control de todas las luces y enchufes eléctricos, ayudando al ahorro de energía eléctrica, ya que algunas veces por descuido e incluso por pereza dejamos alumbrado o aparatos electrónicos funcionando siendo esto un desperdicio de electricidad. Con este sistema, al tenerlo sincronizado con el teléfono inteligente tenemos la oportunidad de monitorear los aparatos y luces, así solo tener encendidas las que realmente se estén utilizando.

Al implementarlo en un hospital puede ser acondicionado para que el paciente tenga el control y no sea dependiente de una enfermera para la manipulación de la luz o los aparatos electrónicos que necesite, por ejemplo la cama, el televisor, el radio, etcétera.

Al ser colocado el sistema en un hotel su finalidad es aumentar la comodidad del huésped, al interconectarlo con todos los servicios de que la habitación disponga como puede ser clima, TV, reproductor de música, luz, puertas, persianas, etcétera.

Para el desarrollo del sistema antes mencionado es necesario un dispositivo con sistema operativo Android que cuente con conexión Bluetooth. Se eligió el Smartphone porque en la actualidad puede considerarse una extensión de las personas, acompañándonos a todos lados y prácticamente en todo momento. Poco a poco se ha convertido en una herramienta indispensable para afrontar el acelerado ritmo de vida actual, puesto que lo utilizamos para un sin número de tareas, por ejemplo, para realizar llamadas, recibir mensajes, revisar nuestro correo electrónico, ver videos, escuchar música, consultar mapas, jugar, etcétera, teniendo una enorme lista de aplicaciones acrecentándose día con día. También se requiere de un circuito electrónico que al conectarse inalámbricamente con el teléfono inteligente sea capaz de interpretar las señales emitidas por este y de ese modo pueda reaccionar para lograr nuestro fin.

Por lo anterior, en el presente trabajo se abordan los siguientes temas:

En el primer capítulo daremos una reseña del Smartphone con sistema operativo Android y también de una herramienta para la creación de aplicaciones para dicho sistema, llamada MIT App Inventor.

El segundo capítulo se ocupará de la tecnología Bluetooth, primeramente diciendo de qué se trata, enseguida se menciona su historia y finalmente su funcionamiento.

En el tercer capítulo tocaremos el tema del microcontrolador PIC de microchip, su historia y sus principales características.

Por ultimo en el cuarto capítulo mencionaremos las herramientas que utilizamos y el proceso para el desarrollo del sistema inalámbrico de operación a distancia de luminarias y/o tomacorrientes, sus características y un presupuesto del proyecto.

Capítulo I. Smartphone con sistema operativo Android

En este capítulo nos enfocaremos en el Smartphone dando una definición de este dispositivo electrónico. También hablaremos del sistema operativo Android, sus orígenes y principales características, asimismo describiremos una herramienta para la creación de aplicaciones en el sistema operativo antes mencionado.

1.1 ¿Qué es un Smartphone?

Llamamos Smartphone (teléfono inteligente) a la familia de teléfonos celulares que disponen de un hardware y un sistema operativo, capaz de realizar tareas y funciones similares a las realizadas por las computadoras fijas o portátiles, añadiéndole funcionalidades extras a la realización y recepción de llamadas o mensajes. (Ilyas Mohammad, Ahson Syed A. 2006)

1.2 Principales características de los Smartphone

- Disponen de una serie de aplicaciones focalizadas a realizar funciones de organizador personal como calendarios, recordatorios, alertas, bloc de notas, etcétera, los cuales pueden comunicarse y sincronizarse con otros dispositivos, que pueden ser computadoras, tablets o celulares.
- Disponen de una conexión a internet, gracias a la red 3G y 4G, que permite navegar al igual que si se accediese desde una computadora de escritorio.
- Cuentan con conexión WI-FI.
- Poseen conexión Bluetooth.
- Disponen de una aplicación para la recepción y envió de correo electrónico.
- Pueden leer, editar y reproducir gran variedad de archivos como hojas de cálculo, editores de texto, archivos multimedia, entre otros.
- Permiten la descarga y ejecución de aplicaciones (App) desarrolladas por terceros, los cuales amplían nuevas funcionalidades, por ejemplo juegos, retoques fotográficos, lectores de libros electrónicos, navegador GPS, entre muchas otras.
- Tienen un teclado QWERTY físico o táctil, que nos permite y facilita la escritura de datos en el dispositivo.
- Por último y más importante, disponen de un sistema operativo apto para desarrollar todas las funciones antes descritas.

1.3 Sistema operativo Android

Android es un Sistema operativo basado en el kernel de Linux, diseñado principalmente para dispositivos móviles con pantalla táctil, como Smartphone o

tablets, inicialmente desarrollado por Android Inc. y respaldado económicamente por Google que finalmente lo compró en el año 2005. (http://source.android.com)

Android fue presentado en 2007 junto a la fundación del Open Handset Alliance: un consorcio de compañías de hardware, software y telecomunicaciones, en el que se encuentran Texas Instruments, Broadcom Corporation, Nvidia, Qualcomm, Samsung Electronics, Intel, LG, Sprint Nextel, Marvell Technology Group, Motorola y T-Mobile, entre otras, para avanzar en los estándares abiertos de los dispositivos móviles. El primer celular con Sistema Operativo Android fue el HTC Dream y se vendió en octubre de 2008.(http://www.smart-gsm.com) Figura 1.1.

Figura 1.1 HTC Dream.
Imagen tomada de Wikipedia.
<http://es.wikipedia.org/wiki/HTC_Dream>

1.3.1 Características del Sistema operativo Android

El sistema operativo Android desde su aparición y hasta la fecha está repuntando en ventas y al no ser un sistema operativo creado solo para Smartphone, ahora también se puede encontrar en relojes, televisores, autoestéreos, tablets, etcétera. Debido principalmente a sus ventajas y características las principales se mencionan a continuación:

- Plataforma realmente abierta. Es una plataforma de desarrollo libre basado en Linux y de código abierto. Una de sus grandes ventajas es que se puede usar y personalizar el sistema sin pagar regalías.
- Adaptable a cualquier tipo de hardware.
- Portabilidad asegurada. Las aplicaciones finales son desarrolladas en Java lo que nos asegura que podrán ser ejecutadas en todos aquellos dispositivos que soporten este lenguaje.
- Arquitectura basada en componentes inspirados en internet.
- Aceptable nivel de seguridad. Los programas se encuentran aislados unos de otros gracias al concepto de ejecución dentro de una caja que hereda de Linux.

Además, cada aplicación dispone de una serie de permisos que limitan su rango de actuación.

- Optimizado para baja potencia y poca memoria. Android utiliza la máquina Virtual Dalvik. Se trata de una implementación de Google de la máquina virtual de Java optimizada para dispositivos móviles.
- Alta calidad en gráficos y sonidos. Gráficos vectoriales suavizados, con animaciones inspiradas en Flash, gráficos en tres dimensiones basados en OpenLG. Incorpora codecs estándar de audio y video incluyendo H.264, MP3, ACC, etcétera.

1.4 MIT App Inventor

Es oficialmente una herramienta para desarrollo de aplicaciones de forma visual para celulares con sistema operativo Android y se divide en dos bloques principales, App Inventor Designer y App Inventor Blocks Editor. Permite también ver tu progreso en tiempo real con un emulador o inclusive conectando tu teléfono móvil a la computadora. (Moreno, 2013) Figura 1.2 Esquema MIT App Inventor.

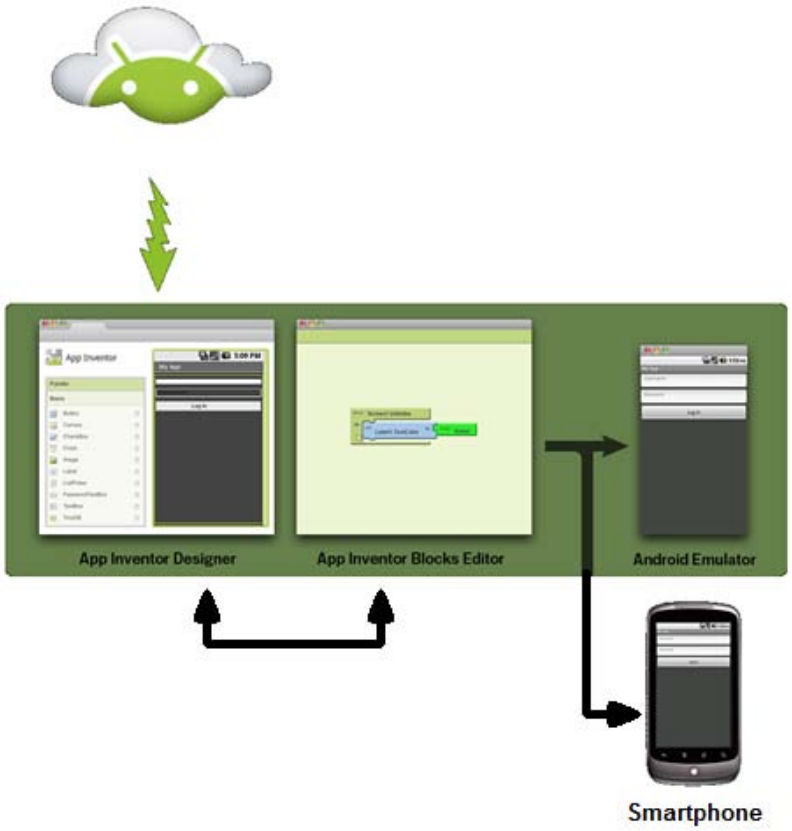

Figura 1.2 Esquema para la creación de aplicaciones en MIT App Inventor.
Ilustración tomada de recursostic.educacion.es
<http://recursostic.educacion.es/observatorio/web/es/software/programacion/1090-uso-de-appinventor-en-la-asignatura-de-tecnologias-de-la-comunicacion-y-la-informacion>

- App Inventor Designer. Aplicación Web

Es para el diseño de la interfaz de nuestra aplicación, aquí se seleccionan los componentes, su ubicación y las propiedades de dichos componentes. Figura 1.3.

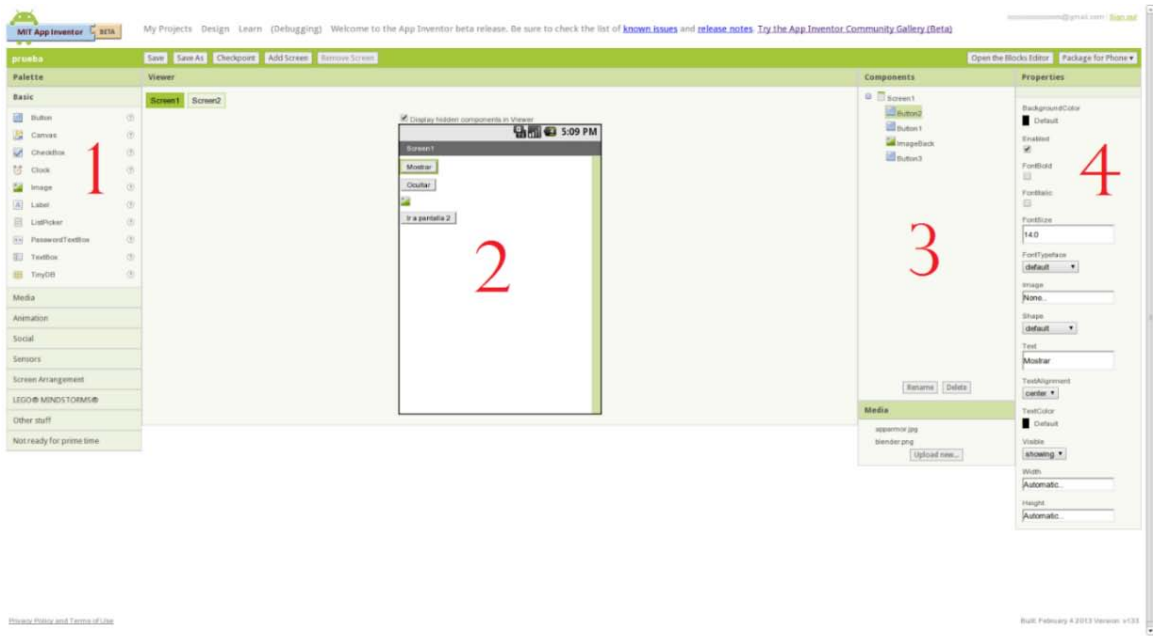

Figura 1.3 Interfaz App Inventor Designer.

Esta herramienta está compuesta por cuatro módulos:

1. Pallete (paleta). Sirve para seleccionar los componentes que tendrán nuestra aplicación, tales como botones, dibujos, imágenes, sonidos, etcétera.
2. Viewer (visor). Aquí se visualiza la interfaz de la aplicación, acomodando los componentes a nuestro gusto y dándole la apariencia que nosotros encontremos conveniente.
3. Components (componentes). En esta sección editamos los componentes, es decir, nos permite ponerle el nombre que nos convenga y podemos subir los componentes multimedia que vallamos a utilizar.
4. Properties (propiedades). Configuramos las propiedades de los componentes de nuestra aplicación, por ejemplo, el color de un botón o el tamaño de mismo.

Una vez teniendo la interfaz en el visor y los componentes modelados a nuestro parecer, podemos proseguir a la siguiente etapa, que es, abrir el editor de bloques. Esto se hace dando click en Open the blocks editor, menú ubicado en la parte superior derecha de nuestra interfaz en App Inventor Designer.

- App Inventor Blocks editor. Editor de bloques.

Es aquí donde especificamos como debe comportarse cada componente, esto se realiza uniendo bloques como en un rompecabezas. Figura 1.4.

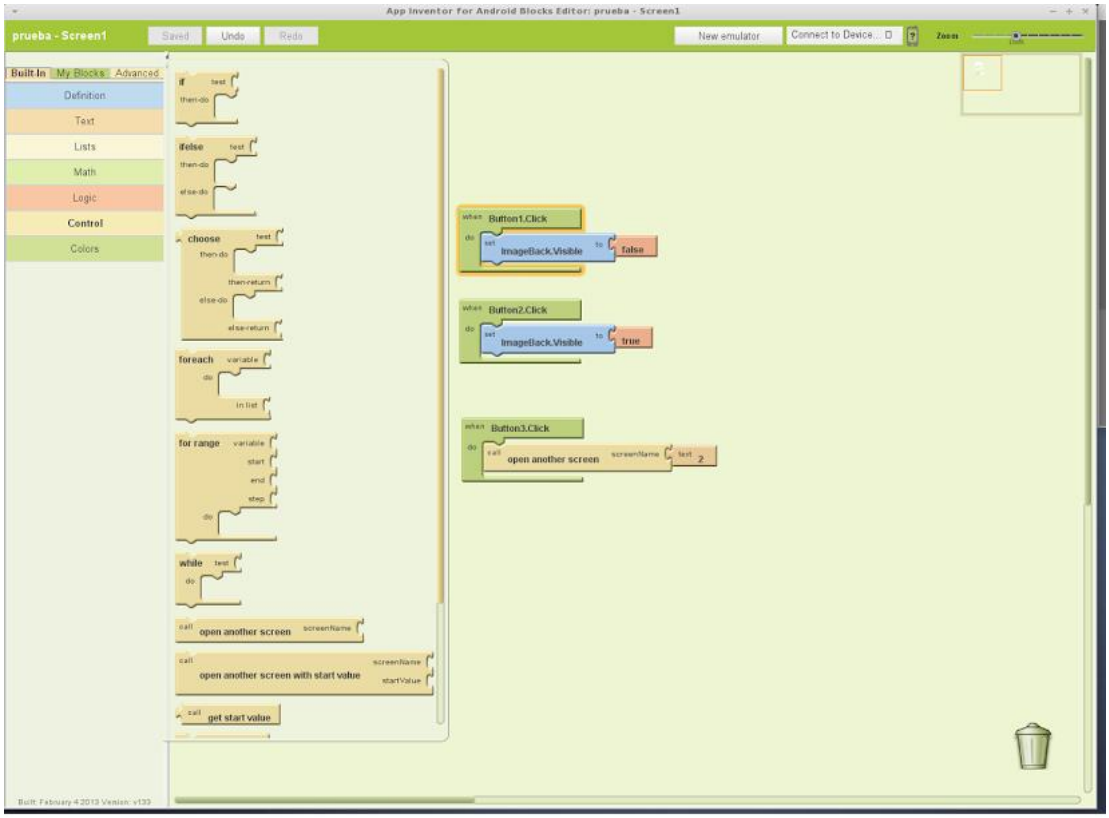

Figura 1.4 Interfaz de App Inventor Blocks Editor.

Esta Herramienta cuenta con tres menús principales Built-in, My Blocks y Advance. También con un área de trabajo donde uniremos los bloques, un menú para iniciar el emulador y otro para conectar nuestro dispositivo.

1. Built-in (incorporar). Al seleccionar este menú se desplegará un cuadro con las opciones que tenemos, por ejemplo, definir una variable, hacer una comparación, etcétera.
2. My Blocks (mis bloques). Este apartado muestra las componentes que previamente hemos seleccionado en App Inventor Designer.
3. Advance (avanzadas). Aquí se nos permite utilizar opciones avanzadas, como cambiar de tamaño a un botón, pedirle que realice una acción para todos los botones, etcétera.

Esto es lo que usaremos para la creación de nuestra aplicación, que es una pieza esencial para nuestro sistema de operación en luminarias y tomacorrientes.

Capitulo II. Comunicación vía Bluetooth

En este capítulo se define lo que es el Bluetooth, sus características fundamentales, asimismo contiene una reseña histórica del proceder del nombre. Conociéndolo ahora como una herramienta para la comunicación inalámbrica, en el cual nos apoyamos para el desarrollo del sistema de operación a distancia de luminarias y tomacorrientes en un apartamento.

2.1 ¿Qué es el Bluetooth?

El Bluetooth es un estándar abierto que posibilita la comunicación entre diversos dispositivos en un corto rango de conexión inalámbrica de radio frecuencia estándar. Básicamente era una tecnología para la sustitución de cables y hoy en día sus áreas de aplicación han encontrado ser incontables. (Prabhu & Reddi, 2006)

Las características fundamentales que se concentran por el estándar Bluetooth son:

- Baja complejidad en su arquitectura.
- Robustez con un precio bajo.
- Bajo consumo de energía.
- Soporte seguro para la voz y la transmisión de datos.

2.2 Historia del Bluetooth

El nombre Bluetooth es en honor a un rey Danés del siglo X de nombre Herald Blátand, quien fue rey de Dinamarca entre los años 940-981 D.C. De este rey se dice que tenía unas grandes habilidades comunicativas, que lo hicieron famoso y con las cuales comenzó el proceso de cristianización de la sociedad vikinga. (http://www.wayerless.com)

La historia del Bluetooth es relativamente corta. Las primeras investigaciones fueron realizadas por Ericsson Mobile Communications en 1994 y fue desarrollada por Jaap Haartsen y Mattisson Sven. Esta compañía promovió una iniciativa para estudiar la viabilidad de una interfaz por radio entre los teléfonos móviles y sus accesorios, con la característica y condición de que tuviese un bajo costo y pequeño consumo de energía. Hasta entonces, las tecnologías de comunicación basadas en el cable funcionaban con eficiencia, sin embargo, su instalación y configuración resultaban bastante dificultosas.

Conforme avanzaba el proyecto del Bluetooth, este fue despertando el interés en otros fabricantes y enseguida se vio el gran abanico de posibilidades que ofrecía esta tecnología. A principios de 1998 se creó la SIG Special Interest Group (Grupo de Interés Especial), el cual estaba formado por Ericsson Mobile

Comunications, Nokia, IBM, Toshiba e Intel. Actualmente se han ido añadiendo otras empresas al grupo como 3Com Corporation, Lucent Technologies, Microsoft Corporation, Motorola Inc. y otras muchas más. Llegando hoy en día a estar formado por más de 2000 empresas del sector del cómputo y las telecomunicaciones. El propósito general de la SIG es establecer un estándar para la interfaz aérea junto con un software de control. Su fin es asegurar la interoperabilidad de los diversos equipos de diferentes fabricantes. (http://www.wayerless.com/)

2.3 Funcionamiento del Bluetooth.

El Bluetooth funciona en la banda médico-científica industrial (ISM: Industrial, Scientific and Medical), con rangos que van desde los 2.4 y hasta 2.48 GHz. Es una banda abierta en todo el mundo, es decir, no necesita licencia.

El estándar Bluetooth utiliza la técnica espectro ensanchado por saltos de frecuencia (FHSS: Frequency hopping Spread Spectrum), que consiste en dividir la banda de frecuencia de 2.402 – 2.480 GHz en 79 canales (denominados saltos) de 1 MHz de ancho cada uno, y después transmitir la señal utilizando una secuencia de canales que sea conocida tanto para la estación emisora como para la receptora. Por lo tanto, al cambiar de canales con una frecuencia de 1600 veces por segundo, el estándar puede evitar la interferencia con otras señales de radio. (Prabhu & Reddi, 2006)

La potencia de transmisión es de hasta 100 mW. La distancia nominal del enlace va desde 10 cm hasta los 10 m, pudiendo alcanzar los 100 m si se aumenta suficientemente la potencia.

Cuando un equipo Bluetooth está dentro de la cobertura de otro, estos pueden crear un enlace entre ellos. Hasta ocho unidades pueden comunicarse entre ellas y forman lo que se denomina una Piconet. La unión de varias Piconet se denomina Scartternet. En todas las Piconets solo puede haber una unidad maestra que normalmente es quien inicia la conexión, el resto de unidades Bluetooth se denominan esclavas.

Cada unidad de la red Bluetooth utiliza su identidad maestra y reloj nativo para seguir en el canal de salto. Cuando se establece la conexión, se añade un ajuste de reloj a la propia frecuencia nativa de la unidad esclava para poder sincronizarse la temporización nativa del maestro. Esté mantiene siempre constante su frecuencia, sin embargo, los ajustes producidos por las unidades esclavas para sincronizarse con el maestro solo son válidos mientras dura la conexión. Dentro de la misma área pueden coexistir diversas Piconets ya que cada una tiene una unidad maestra distinta, con su propia secuencia de saltos de canales y de fase. A medida que tenemos más redes Bluetooth en la misma área de cobertura, la probabilidad

de colisión aumenta, produciendo una degradación del espectro y reduciendo el rendimiento del sistema.

Una unidad Bluetooth puede participar secuencialmente en varias Piconets, gracias al sistema de multiplexaje por división de tiempo (TDM: Time-Division Multiplexing). Esto es posible siempre y cuando la unidad este activa solo en una a la vez. Para realizar este proceso, la unidad cuando se incorpora a la nueva red debe ajustar el offset de su reloj nativo y realizar los ajustes de configuración correspondientes a la nueva. Cuando una unidad abandona la red, la esclava informa al maestro actual que no estará disponible por un determinado periodo, que será en el que estará activa en otra. Durante su ausencia, el tráfico en la Piconet entre el maestro y otros esclavos continúan igualmente.

2.3.1 Sustento matemático para el procesamiento digital de señales

La radiocomunicación es la tecnología que posibilita la transmisión de señales mediante la modulación de su frecuencia o amplitud de ondas electromagnéticas. Estas ondas no requieren un medio físico de transporte, por lo que pueden propagarse a través del vacío.

Una onda de radio se origina cuando una partícula cargada por ejemplo, un electrón se excita a una frecuencia situada en la zona de radiofrecuencia (RF) del espectro electromagnético. Cuando la onda de radio actúa sobre un conductor eléctrico (la antena), induce en un movimiento de la carga eléctrica (corriente eléctrica) que puede ser transformado en señales portadoras de información.

Las redes bluetooth transmiten datos por medio de ondas de baja potencia. Se comunica en una frecuencia de entre 2.402 y 2.480 Ghz, la modulación que utiliza es GFSK Gaussian Frequency Shift Keying (modulación por desplazamiento de frecuencia gaussiana) es un tipo de modulación derivada de fsk Frequency Shift Keying (modulación por desplazamiento de frecuencia), agregando al sistema un filtro gaussiano después de la banda de pulsos, para cuando estos pasen, sean más suaves para limitar el ancho del espectro. Donde un 1 lógico es representado mediante una desviación positiva (incremento) de la frecuencia de la onda portadora, y un 0 mediante una desviación negativa (decremento) de la misma. Figura 2.1 ejemplo de señal GFSK.

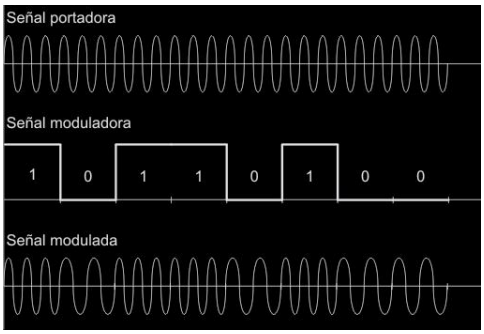

Figura 2.1 ejemplo de señal GFSK

Definiendo los pulsos $g_i\,(t), i = 0, \dots, M-1$ como

$$g_i(t) = sen\,(\omega_i t)\omega_T(t)$$

(2.1)

Siendo $\omega_{\mathrm{T}}(t)$ un pulso rectangular causal tenemos:

$$\omega_{\mathrm{T}}(t) = \begin{cases} 1 & 0 \le t \le \mathrm{T} \\ 0 & \text{resto} \end{cases}$$

(2.2)

La modulación FSK asigna a cada "símbolo" de entrada $A[n]\epsilon\{0,1, \dots, M-1\}$, el pulso $g_{A[n]}(t)$, de modo que si, por simplicidad, suponemos que M es una potencia de dos, y enviamos un nuevo pulso cada T segundos, entonces estaremos transmitiendo con régimen binario $\log_2(M)/\,(T)$ bits/seg. La señal trasmitida tiene la siguiente expresión.

$$x(t) = K \sum_n g_{A\,[n]}\,(t - nT)$$

(2.2)

Donde K es una constante relacionada con la energía medida por símbolo transmitido. Teniendo en cuenta las consideraciones del incremento de ancho de banda debido a la existencia de saltos bruscos de fase de la señal transmitida, nos limitaremos a una versión de la modulación FSK en la que se garantiza la continuidad de fase. Dicha modulación se denomina CPFSK continuous phase frecuency shift keying (modulación por desplazamiento de frecuencia de fase continua). La condición para que exista continuidad de fase es que las frecuencias ω_i empleadas en *(2.1)* sean tales que exista un número entero de ciclos dentro del periodo de símbolo T o, equivalentemente, para algún número entero N_i.

$$\omega_i = \frac{2_\pi\, N_i}{T}, i = 0, \dots, M-\ 1$$

(2.4)

Con esta condición es suficiente para garantizar la ortogonalidad de los pulsos $g_i(t)$. Podemos definir la base ortonormal

$$\phi_i(t) = \sqrt{\frac{2}{T}}\, sen(\omega_i t)\omega_T(t), i = 0, \dots\ , M-1$$

(2.5)

Y, a partir de ella, la señal transmitida (2.2) será

$$x(t) = \sqrt{E_s} \sum_n \phi\, A_{[n]}(t - nT)$$

(2.6)

Siendo E_S la energía por pulso transmitido. En la figura 2.2 se representa un fragmento de señal transmitida para una modulación CPFSK ortogonal binaria (*M = 2),* donde podemos observar claramente como se mantiene la continuidad de fase. Para esta misma modulacion binaria la figura 2.3 recoge en el espacio de señales la constelación empleada. De un modo genérico, las modulaciones que emplean este tipo de constelación reciben el nombre de modulaciones ortogonales.

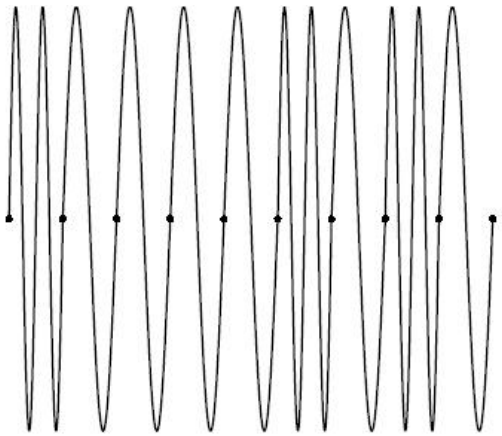

Figura 2.2 ejemplo de fragmento de señal ortogonal binaria.

Una forma de representar la modulación CPFSK consiste en considerar la transmisión de un pulso de frecuencia ω_i como una desviación con respecto a una frecuencia central, que denotaremos por ω_c y que se puede relacionar con una frecuencia nominal de "portadora". Por ejemplo, en el caso de la CPFSK ortogonal binaria, $\omega_c = (\omega_0 + \omega_1)/2$ y en la CPFSK ortogonal cuaternaria, $\omega_c = (\omega_0 + \omega_3)/2 = (\omega_1 + \omega_2)/2$. De forma alternativa en la CPFSK ortogonal binaria $\omega_0 = \omega_c - \frac{\pi}{T}; \omega_1 = \omega_c + \pi/T$ y en la CPFSK ortogonal cuaternaria $\omega_0 = \omega_c - \frac{3\pi}{T}; \omega_1 = \omega_c - \frac{\pi}{T}; \omega_2 = \omega_c + \frac{\pi}{T} y\, \omega_3 = \omega_c + 3_\pi/T$

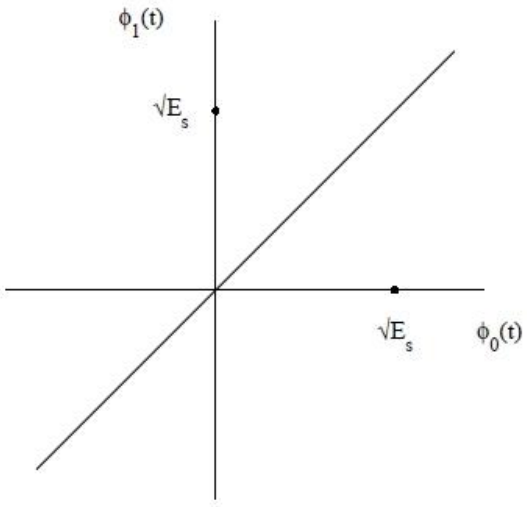

Figura 2.3 Representación en espacio de señales de la constelación empleada en la modulación CPFSK ortogonal binaria.

De este modo, la señal x_t puede reescribirse como

$$x(t) = \sqrt{\frac{2E_s}{T}} \sum_n sen\left(\omega_c t + I_{[n]} \frac{\pi t}{T}\right) \omega_T (t - nT)$$

(2.7)

Donde $I_{[n]} \epsilon \{\pm 1, \pm 3, \dots, \pm(M-1)\}$ sirve para seleccionar la frecuencia de transmisión durante la transmisión del pulso n-ésimo. Obsérvese que los $I_{[n]}$ pueden considerarse en cierto sentido"símbolos" de información, sin que ello implique la constelación unidimensional porque, como acabamos de ver, estamos trabajando con *M* dimensiones. En cualquier caso, la utilización de los $I_{[n]}$ permite ver la modulación como un auténtico desplazamiento de frecuencia con respecto a ω_c,lo que justifica el nombre FSK. El espectro de una señal multipulso, calculamos el espectro de potencia de un proceso de la forma

$$X(t) = K \sum_n g_{A[n]}(t - nT)$$

(2.8)

Donde $K = \sqrt{2E_s/T}$, los pulsos $g_i(t), i = 0, \dots, M-1$ tienen duración igual o menor que T segundos, y los símbolos $A_{[n]}$ son estadísticamente indispensables y uniformemente distribuidos en $\{0,1, \dots, M-1\}$. En primer lugar, es fácil comprobar que el proceso $X(t)$ es cicloestacionario de periodo T, por lo que la autocorrelación $R_x(t+\tau, t)$ es una función periódica de t y, por tanto, la función de autocorrelación promediada se puede calcular sobre un único periodo

$$\begin{aligned} R_x(t) &= \frac{1}{T} \int_0^T R_x(t+\tau, t) dt \\ &= \frac{K^2}{T} \sum_n \sum_t \int_0^T E\left\{g_{A\{n\}}(t+\tau-nT) g_{A[l]}(t-lT)\right\} dt \end{aligned}$$

(2.9)

Es conveniente introducir la media $\mu_g(t)$ de los pulsos trasmitidos, ya que, en general, esta no será nula

$$\mu_g(t) \doteq E\{g_{A[n]}(t)\} = \frac{1}{M} \sum_{i=0}^{M-1} g_i(t)$$

(2.10)

Donde la última igualdad es consecuencia del carácter equiproblable de los símbolos. También será conveniente definir, para todo $i, 0 \leq i \leq M-1, \tilde{g}_i(t) \doteq g_i(t) - \mu_g(t)$. Obsérvese que

$$E\{g_{A[n]}(t+\tau-nT)g_{A[l]}(t-lT\} = E\{g_{A[n]}(t-\tau-nT)\}\cdot E\{g_{A[l]}(t-lT)\}$$
$$= \mu_g(t+\tau-nT)\mu_g(t-lT), n \neq l$$

(2.11)

Para el caso $n = l$ tenemos

$$E\{g_{A[n]}(t+\tau-nT)g_{A[l]}(t-lT\} = E\{[\tilde{g}_{A[n]}(t+\tau-nT)+\mu_g(t+\tau-nT)]$$
$$\cdot\,[\tilde{g}_{A[n]}(t-nT)+\mu_g(t-nT)]\}$$
$$= E\{\tilde{g}_{A[n]}(t+\tau-nT)\tilde{g}_{A[n]}(t-nT)\}$$
$$+\,\mu_g(t+\tau-nT)\mu_g(t-nT)$$

(2.11)

Ahora estamos en condiciones de escribir (2.9) como

$$\bar{R}_x(\tau) = \frac{K^2}{T}\sum_n\sum_l\int_0^T \mu_g(t+\tau-nT)\mu_g(t-lT)dt$$
$$+\,\frac{k^2}{T}\sum_n\int_0^T E\{\tilde{g}_{A[n]}(t+\tau-nT)\tilde{g}_{A[n]}(t-nT)\}\,dt$$

(2.12)

Denotamos los dos sumandos anteriores por $\tilde{R}_{xd}(\tau)$ y $\tilde{R}_{xc}(\tau)$ respectivamente, mientras que sus transformadas respectivas son $S_{xd}(j\omega)$ y $S_{xc}(j\omega)$. A continuación se desarrollan las autocorrelaciones promediadas y sus espectros asociados.

$$\bar{R}_{xd}(\tau) = \frac{K^2}{T}\sum_n\sum_l\int_{-lT}^{T}\mu_g(t+\tau+lT-nT)\mu_g(t)\,dt$$
$$= \frac{K^2}{T}\sum_m\sum_l\int_{-lT}^{T-lT}\mu_g(t+\tau+mT)\,\mu_g(t)dt$$
$$= \frac{K^2}{T}\sum_m r_{\mu_g}(\tau+mT)$$
$$= \frac{K^2}{T}r_{\mu_g}(t) * \sum_m \delta(\tau+mT)$$

(2.12)

Donde $r_{\mu_g}(t) \doteq \mu_g(t) * \mu_g(-t)$ es la función de ambigüedad temporal de $\mu_g(t)$. Para calcular $S_{xd}(j\omega)$, solo tenemos que recordar que la transformada de un

tren de impulsos espaciados T segundos es otro tren de impulsos espaciados $2\pi/T\ rad/seg$. Por tanto,

$$S_{xd}(j\omega) = \frac{K^2}{T^2} S_{\mu_g}(j\omega) \sum_m \delta\left(j\omega - j\frac{2\pi m}{T}\right)$$

(2.12)

Donde $S_{\mu_g}(j\omega)$ es la trasformada de Fourier de $r_{\mu_g}(\tau) = \mu_g(\tau) * \mu_g(-\tau)$. A partir de (2.10) se puede notar que

$$S_{\mu_g}(j\omega) = \frac{1}{M^2}\left|\sum_{i=0}^{M-1} G_i(j\omega)\right|^2$$

(2.13)

Ecuación que al sustituirla en (2.12) obtenemos

$$S_{xd}(j\omega) = \frac{2E_s}{T}\frac{1}{(MT)^2}\left|\sum_{i=0}^{M-1} G_i(j\omega)\right|^2 \sum_k \delta\left(\omega - \frac{2\pi k}{T}\right)$$

(2.14)

Por lo que respecta a $\bar{R}_{xc}$, podemos escribir

$$\begin{aligned}\bar{R}_{xc}(\tau) &= \tfrac{K^2}{T}\textstyle\sum_n \int_{-nT}^{T-nT} E\left\{\tilde{g}_{A_{[n]}}(t+\tau)\tilde{g}_{A_{[n]}}(t)\right\} dt \\ &= \frac{K^2}{T}\int_{-\infty}^{\infty} E\left\{\tilde{g}_{A_{[n]}}(t+\tau)\tilde{g}_{A_{[n]}}(t)\right\} dt \\ &= \frac{K^2}{MT}\sum_{i=0}^{M-1}\int_{-\infty}^{\infty}\tilde{g}_i(t+\tau)\tilde{g}_i(t)dt \\ &= \frac{K^2}{MT}\left(\sum_{i=0}^{M-1} g_i(\tau) * g_i(-\tau) - M r_{\mu_g}(\tau)\right)\end{aligned}$$

(2.15)

Cuya transformada de Fourier es

$$S_{xc}(j\omega) = \frac{2E_s}{T}\frac{1}{MT}\left\{\sum_{i=0}^{M-1}|G_i(j\omega)|^2 - \frac{1}{M}\left|\sum_{i=0}^{M-1} G_i(j\omega)\right|^2\right\}$$

(2.16)

Donde $G_i(j\omega), i = 0, \dots, M-1$, es la transformada de Fourier del pulso $g_i(t)$.

Finalmente, el espectro de la señal transmitida será

$$S_x(j\omega) = S_{xc}(j\omega) + S_{xd}(j\omega) \tag{2.17}$$

2.3.2 Canal.

La banda ISM tiene muchas interferencias. Para solucionar este problema, el estándar Bluetooth utiliza un método de salto de frecuencia pseudo-aleatorio, en el que el canal queda dividido en intervalos de 625 µs, llamados slots, donde cada salto de frecuencia es ocupado por uno de estos.

Un paquete de datos ocupa un slot para la emisión y otro para la recepción y pueden ser usados alternativamente, dando lugar a un esquema de tipo duplexación por división de tiempo (TDD Time-Division Duplexing).

De esta manera se pueden conseguir transeptores de banda estrecha con gran inmunidad a las interferencias. (Prabhu & Reddi, 2006) Figura 2.4.

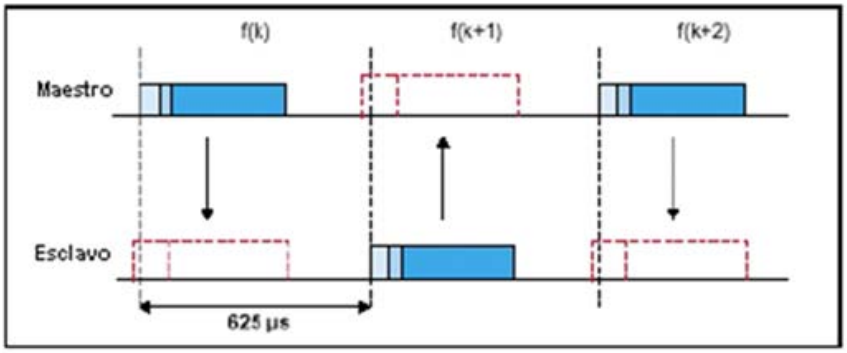

Figura 2.4 Se muestra la división de 625 µs y la asignación de los slots para evitar interferencias.
Esquema tomado de oscar.iitb.ac.in
<http://oscar.iitb.ac.in/onsiteDocumentsDirectory/Bluetooth/Bluetooth/Help/Technology%20Overview.htm>

2.3.3 Paquete.

Las unidades Bluetooth intercambian paquetes de datos entre ellas. Cada paquete está formado por un conjunto de slots. Todos los paquetes tienen un mismo formato. Constan de un código de acceso de 72 bits en el cual podemos encontrar la identidad de la unidad maestra. Después una cabecera de 54 bits en los cuales podemos encontrar información de control, tipo de paquete, bits de comprobación

de flujo, chequeo de errores, etcétera. Finalmente podemos encontrar el paquete con información, el cual puede variar su longitud entre 0 y 2754 bits. (Prabhu & Reddi, 2006) Figura 2.5.

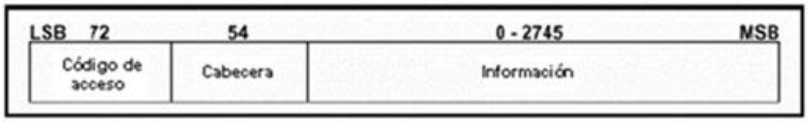

Figura 2.5 Estructura de un paquete.
Imagen tomada de oscar.iitb.ac.in
<http://oscar.iitb.ac.in/onsiteDocumentsDirectory/Bluetooth/Bluetooth/Help/Technology%20Overview.htm>

Los paquetes de datos están protegidos por un esquema de repetición automática de consulta (ARQ: Automatic Repeat-reQuest), en el cual los paquetes perdidos son automáticamente retrasmitidos. Con este sistema, si un paquete no llega a su destino, solo una pequeña parte de la información se pierde.

La voz no se retrasmite nunca, sin embargo, se utiliza un sistema de codificación muy robusto basado en una modulación variable de declive delta, que sigue la forma de la onda de audio y es muy resistente a los errores de bits.

Los receptores de la Piconet comparan las señales que reciben con el código de acceso, si estas no coinciden, el paquete recibido no es considerado como válido en el canal y el resto de su contenido es ignorado.

2.3.4 Enlace físico.

En la especificación Bluetooth se definen dos tipos de enlace:

- Enlace de conexión orientada sincrona (SCO: Synchronous connection-oriented). Soportan conexiones asimétricas, punto a punto, usadas normalmente en conexiones de voz. Dichos enlaces están definidos en el canal y se reservan dos slots consecutivos (envío y retorno) en intervalos fijos.
- Enlace de baja conexión asincrono (ACL: Asynchronous Connection-Less). Soportan conmutaciones punto a punto, simétrico o asimétrico, típicamente usado en la transmisión de datos.

Un conjunto de paquetes se ha definido para cada tipo de enlace físico: para los SCO existen tres tipos de slot simple, cada uno con una portadora a una velocidad de 64 kbit/s. La transmisión de voz se realiza sin ningún mecanismo de protección, pero si el intervalo de las señales en el enlace SCO disminuye, se puede seleccionar una velocidad de corrección de envío de 1/3 ó 2/3. Para los enlaces ACL se ha definido el slot-1, slot-3 y slot-5. Cualesquiera de los datos pueden ser

enviados protegidos o sin proteger con una velocidad de corrección de 2/3. La máxima velocidad de envío es de 721 kbit/s en una dirección y 57.6 kbit/s en la otra. (Prabhu & Reddi, 2006)

2.3.5 Establecimiento de una conexión.

Para establecer la Piconet, la unidad maestra debe conocer la identidad del resto de unidades que están en modo standby en su radio de cobertura. El maestro que inicia la red Bluetooth trasmite el código de acceso continuamente en periodos de 10 ms, que es recibido por el resto de las unidades. El tren de 10 ms de acceso da diferentes saltos de portadora y se transmite repetidamente hasta que el receptor responde o bien se excede el tiempo de respuesta. Después que el receptor acepta éste paquete, ajustará su reloj para seleccionar el canal de salto correcto determinado por el emisor.

2.3.6 Arquitectura de protocolo.

La especificación Bluetooth define una arquitectura de protocolos semejante a la torre OSI (Open System Interconnection). Estos protocolos son utilizados dependiendo el tipo de sistema que se quiere implementar. Cada protocolo lo podemos tratar como una capa independiente con su correspondiente interfaz para que el sistema sea homogéneo. Figura 2.6 esta muestra la pila de protocolos del estándar Bluetooth. (http://bluehack.elhacker.net)

2.3.7 Perfiles.

Para asegurar la interoperabilidad entre dispositivos Bluetooth de diferentes fabricantes, se han delimitado una serie de perfiles. Estos definen los roles y capacidades para diferentes aplicaciones específicas y pueden abarcar distintas capas y protocolos para aumentar o disminuir el grado de seguridad. La figura 2.7 muestra la estructura de los perfiles. (http://bluehack.elhacker.net)

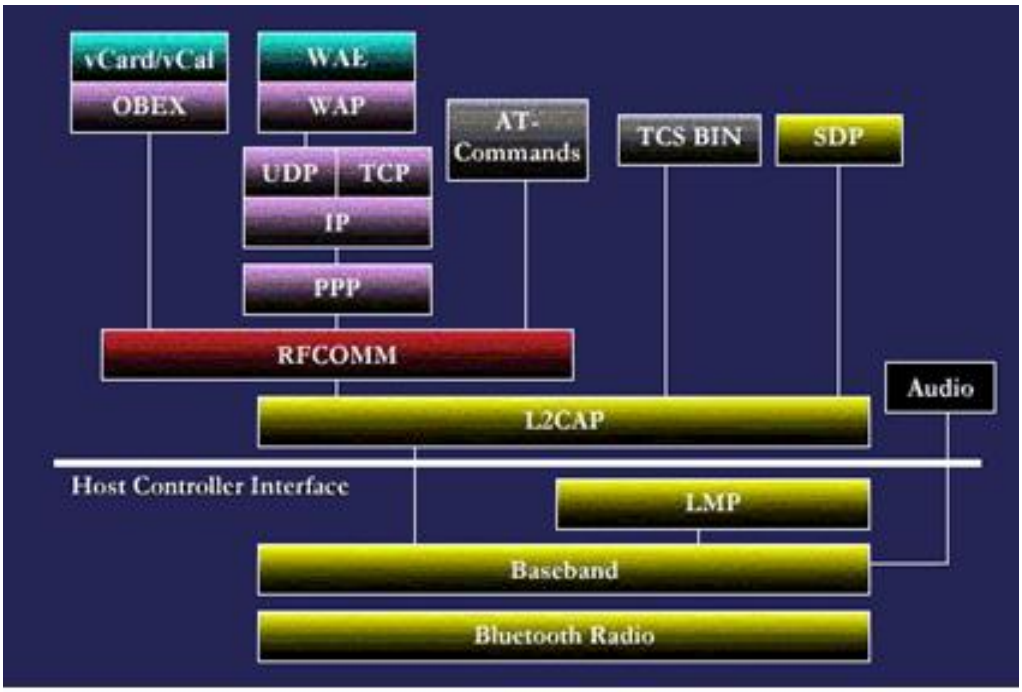

Figura 2.6 Muestra la pila de protocolos Bluetooth.
Imagen tomada de rooibo.wordpress.com
<http://rooibo.wordpress.com/category/protocolos/>

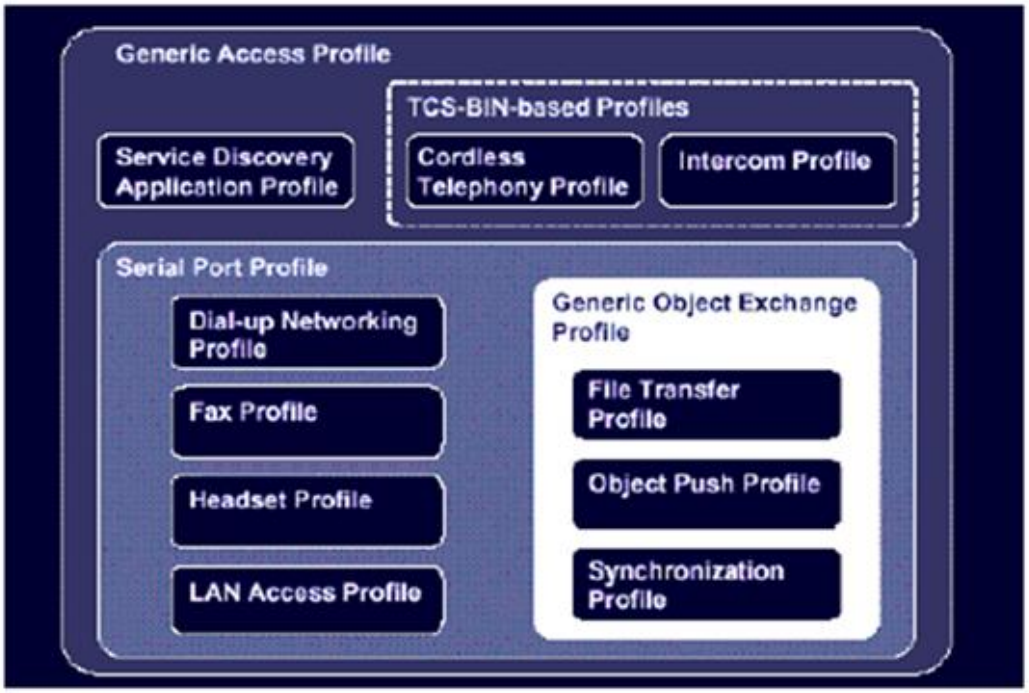

Figura 2.7 Se ve como están estructurados los perfiles.
Ilustración tomada de bluehack.elhacker.net
<http://bluehack.elhacker.net/proyectos/bluesec/bluesec.html>

2.3.8 Seguridad.

En el estándar Bluetooth se han definido unas medidas de seguridad para la protección de la información. Las medidas de seguridad se basan en tres puntos:

- Una rutina de pregunta-respuesta, para autenticación.
- Una corriente cifrada de datos, para encriptación.
- Generación de una clave de sesión.

Tres entidades son utilizadas en los algoritmos de seguridad: La dirección de la unidad Bluetooth, que es una entidad pública, una clave de usuario privada, una entidad secreta, y un número aleatorio, que es diferente por cada transacción. La dirección Bluetooth se puede obtener por medio de un procedimiento de consulta. La clave privada se deriva durante la inicialización y no es revelada posteriormente. El número aleatorio se genera en un proceso pseudo-aleatorio en cada unidad Bluetooth. (http://bluehack.elhacker.net)

2.4 Protocolo RF-COMM.

RF-COMM es la abreviatura del termino en ingles Radio frequency Communication (Comunicación por radio frecuencia). Es un conjunto simple de protocolos de transporte, construido por el protocolo L2CAP, Logical Link Control and Adaptation Protocol (Protocolo de control y adaptación del enlace lógico), el cual es utilizado dentro de la pila de protocolos de Bluetooth. Se utiliza para pasar paquetes con y sin orientación a la conexión a sus capas superiores, incluyendo tanto al Host Controller Interface (HCI: controlador principal de interfaz), como directamente al gestor del enlace. (http://www.bluetooth.com). Las funciones de L2CAP incluyen:

- Segmentación y reensamblado de paquetes. Acepta paquetes de hasta 64KB de sus capas superiores.
- Multiplexación de varias fuentes de paquetes, comprobando el protocolo de las capas superiores para así adaptarlo antes del reensamblaje.
- Proporcionar una buena gestión, para la transmisión unidireccional a otros dispositivos Bluetooth.
- Gestión de la calidad de servicio (QoS: Quality of Service), para los protocolos de las capas superiores. En esta fase negocia el tamaño máximo del campo de datos de las tramas. Con ello, evita que algún dispositivo envíe paquetes tan grandes que puedan desbordar al receptor.

L2CAP se utiliza para comunicarse sobre el acoplamiento ACL del anfitrión, y su conexión se establece después de que el enlace ACL haya sido configurado.

2.5 Comunicación serie.

La comunicación serial consiste en él envió de un bit de información de manera secuencial, esto es, un bit a la vez y a un ritmo acordado entre el emisor y el receptor.

La comunicación serial en las computadoras siguió los estándares definidos en 1969 por el estándar recomendado 232 (RS-232: Recommended Standar 232) que establece niveles de voltaje, velocidad de transmisión de los datos, etcétera. Por ejemplo, este protocolo establece niveles de voltaje de -12 V como un uno lógico y un nivel de +12 V como un cero lógico. Por su parte, los microcontroladores emplean por lo general 5 V como uno lógico y 0 V como cero lógico.

Existen en la actualidad diferentes ejemplos de puertos que comunican información de manera serial. El puerto serie que gradualmente ha sido reemplazado por el puerto USB (Universal Serial Bus) que permite mayor versatilidad en la conexión de múltiples dispositivos.

2.6 Modulo Bluetooth JY-MCU (HC-06)

Mencionamos este dispositivo porque es el que utilizaremos en nuestro proyecto. El modulo Bluetooth JY-MCU está constituido por un módulo HC-06 y una placa JY-MCU. Dispone de 4 pines Vcc, Gnd, TX y Rx, también cuenta con un led indicador de estado. Las principales características del módulo HC-06 son:

- Voltaje de alimentación de 3.3 V.
- Antena integrada.
- Velocidad por defecto de 9600 bps.
- Protocolo RF-COMM, conocido también como emulador de puerto serie.
- Dimensiones 27 x 13 x 2.2 mm.
- Peso de 10 gramos.

En la siguiente figura 2.8 se muestra un módulo HC-06.

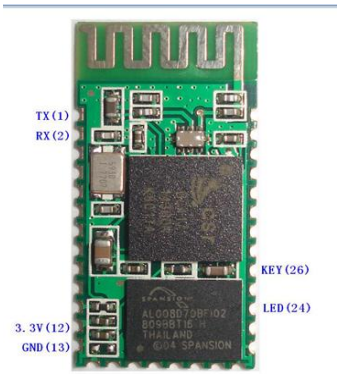

Figura 2.8 módulo HC-06.

Imagen tomada de la hoja de especificaciones del módulo HC-06.

La placa JY-MCU consta principalmente de un regulador de voltaje, para permitir una alimentación de 3.6 y hasta 6 V, pines marcados para facilitar la manipulación y un led indicador, que se encuentra parpadeando mientras está en espera de realizar un enlace Bluetooth y permanece encendido una vez realizada la vinculación, manteniéndose así hasta que está finalice. En la figura 2.9 podemos ver una placa JY-MCU.

Figura 2.9 Placa JY-MCU.
Foto tomada de www.ebay.com
<http://www.ebay.com/itm/JY-MCU-V1-02pro-Serial-Bluetooth-Interface-Board-with-Clear-Key-/291028600935?pt=LH_DefaultDomain_0&hash=item43c2a80867>

Al unir la placa JY-MCU y el módulo HC-06 se forma el módulo Bluetooth JY-MCU, es de clase 2, esto quiere decir que su potencia es de 2.5 mW y su alcance es de 10 m aproximadamente. Por su potencia encontramos una limitante en nuestro sistema, ya que para poder tener control sobre los interruptores, tenemos que estar en el radio de cobertura de nuestro dispositivo esto es ±10 m aproximadamente. En la Figura 2.10 podemos observar un módulo Bluetooth JY-MCU.

Figura 2.10 Módulo JY-MCU.

Capitulo III. Los PIC de Microchip

La evolución de la electrónica desde la aparición del circuito integrado ha sido constante. Actualmente podemos encontrar dispositivos cada vez más complejos ubicados en encapsulados más pequeños. Un ejemplo es un microcontrolador, este es un dispositivo electrónico encapsulado en un chip, capaz de ejecutar un programa. Este reúne en un solo integrado: microprocesador, memoria de programa, memoria de datos y puertos de entrada/salida. Además, también suelen disponer de otras características especiales como puertos serie, comparadores, convertidores analógico digital, etcétera. En el mercado existen gran variedad de marcas y características distintas que podremos utilizar dependiendo la aplicación a realizar. (Mandado, Menéndez, Fernandez & López, 2007)

Un microcontrolador ejecuta instrucciones, el conjunto de estas es lo que llamamos programa. Las instrucciones son leídas de la memoria de programa para ejecutarlas una detrás de otra.

Las instrucciones son operaciones simples como sumar, restar, escribir en un puerto, activar un bit de un dato, etcétera. Mediante estas órdenes básicas podemos realizar operaciones más complejas y así llegar al objetivo de la aplicación.

3.1 Historia de los PIC.

En 1965 la empresa GI (General Instruments) formó una división de microelectrónica destinada a generar las primeras arquitecturas viables de memoria EPROM, Erasable Programmable Read-Only Memory (Memoria de solo lectura programable borrable), y EEPROM, Electrically Erasable Programmable Read-Only Memory (Memoria de solo lectura programable y borrable eléctricamente). A principios de 1970 se creó el CP1600, que era un microprocesador bastante bueno pero no manejaba muy bien los puertos de Entrada/Salida, por esta razón, en 1975 se creó el PICmicro comúnmente llamado solo PIC, que es un chip que funcionaba en combinación con el CP1600 para mejorar el control de los puertos de entrada/salida. (Catsoulis, 2005)

Alrededor de 1980 GI reestructuró la empresa y creó la GI Microelectronics, que finalmente fue vendida a un grupo de inversores de capital de riesgo que crearon la actual Arizona Microchip Technology y centraron su negocio en la fabricación de PIC, memorias EPROM y EEPROM. Actualmente Microchip ha realizado un gran número de mejoras a la arquitectura original, adaptándola a las actuales tecnologías y al bajo costo de los semiconductores. (Catsoulis, 2005)

3.2 Características de los PIC.

Los PIC gracias a sus especificaciones son dispositivos muy confiables, potentes y a un precio competitivo. Sus principales características son:

- El procesador sigue el modelo Harvard. Este tipo de arquitectura se conecta de forma independiente y con dos buses distintos, uno a la memoria de instrucciones y otro a la de datos. La Figura 3.1 muestra un modelo Hardvard. La arquitectura Harvard permite al CPU, Central Processing Unit (unidad central de procesamiento), acceder simultáneamente a las dos memorias. Esto proporciona mayor velocidad, además de numerosas ventajas al funcionamiento del sistema.

Figura 3.1 Modelo de arquitectura Harvard.

- Se aplica la técnica de segmentación (pipe-line) en la ejecución de las instrucciones. La segmentación permite al procesador realizar al mismo tiempo la ejecución de una instrucción y la búsqueda del código de la siguiente, así puede ejecutar cada orden en un ciclo equivalente a cuatro ciclos de reloj. En cada etapa se realiza la búsqueda de una instrucción y la ejecución de la anterior. Las instrucciones de salto ocupan dos ciclos dado que la dirección de la siguiente instrucción no se conoce hasta que no se haya completado la bifurcación. Figura 3.2

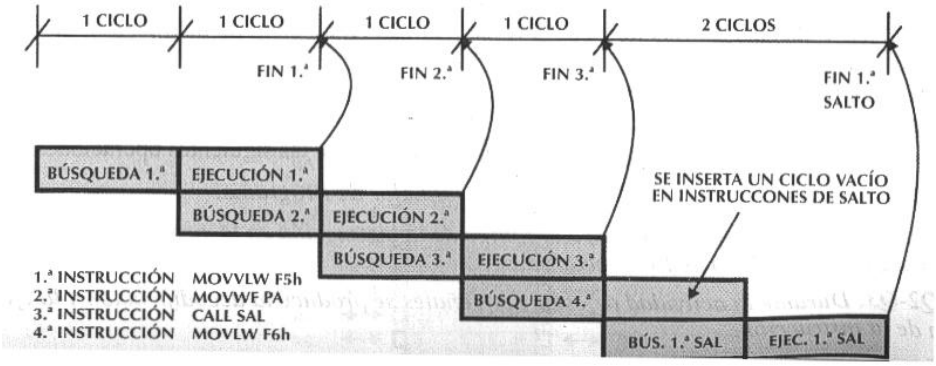

Figura 3.2 Técnica de segmentación.
Ilustración tomada de www.forosdeelectronica.com
<http://www.forosdeelectronica.com/f16/usando-pwm-mini-tutorial-271/>

- El formato de las instrucciones tiene la misma longitud. Esta característica es muy ventajosa en la optimización de la memoria de instrucciones y facilita enormemente la construcción de ensambladores y compiladores.
- Procesador RISC, Reduced Instruction Set Computer (computadora con conjunto de instrucciones reducida). Dependiendo de la gama del procesador (baja, media o alta) tienen mayor o menor número de instrucciones. Los modelos de la gama baja disponen de un repertorio de 33 instrucciones, 35 los de la gama media y más de 70 para los de alta.
- Todas las instrucciones son ortogonales. Cualquier instrucción puede manejar cualquier elemento de la arquitectura como fuente o como destino.
- Arquitectura basada en un banco de registros. Esto significa que todos los objetos del sistema (puertos de E/S, temporizadores, posiciones de memoria, etc.) están implementados físicamente como registros.
- Diversidad de modelos con prestaciones y recursos diferentes. La gran variedad de PIC permite que el usuario pueda seleccionar el más conveniente para su proyecto.
- Herramientas de soporte potentes y económicas. Las herramientas de software que nos facilita Microchip las podemos descargar fácilmente de su página Web www.microchip.com y son totalmente gratuitas. (Mandado *et al*, 2006)

3.3 La memoria.

Los PIC, al estar construidos con arquitectura Harvard, poseen dos bloques distintos de memoria, uno para la de programa y otro para la de datos. Estos dos son independientes entre ellos teniendo tamaño y longitudes de palabra distintas. Cada bloque posee su propio bus, de tal forma que el acceso a cada uno puede producirse durante el mismo ciclo del oscilador.

La memoria de datos puede dividirse en la RAM, Random Access Memory (Memoria de acceso aleatorio), de fines generales y en los Registros SFR, Specific Function Register (Registros de función específica). Esta contiene también los datos de la EEPROM, aunque no está introducida directamente en ella es registrada en forma indirecta. La memoria de datos se organiza en *bancos* y la cantidad de estos depende de la gama de PIC que estemos utilizando.

La memoria de programa también se le denomina de instrucciones. Aquí es donde nosotros escribimos nuestro programa. Los PIC con memoria tipo flash se pueden programar y borrar eléctricamente, la cantidad de memoria también depende de la gama de PIC que estemos utilizando. (Mandado et al, 2006)

3.3.1 Registros.

Los PIC utilizan una arquitectura basada en registros. Esto significa que todos los objetos del sistema, puertos de E/S, temporizadores, posiciones de memoria, etcétera, están implementados físicamente como registros. Todos los registros están ubicados en una posición específica de la memoria. (Mandado, et al, 2006). Existen diferentes tipos de registros:

- GPR: General Purpose Registers (registros de propósito general).
- SFR: Specific Function Register (registros de función específica).

3.3.2 Contador de programa.

El contador de programa denominado PC: Program Counter, es incrementado automáticamente en cada ciclo de instrucción, de tal manera que las instrucciones son leídas en secuencia desde la memoria. Ciertas instrucciones, tales como las bifurcaciones y las llamadas y retornos de subrutinas, interrumpen la secuencia al colocar un nuevo valor en el contador de programa.

Al resetearse el microprocesador, todos los bits del PC toman valor 1, de manera que la dirección de arranque del programa es siempre la última posición de memoria de programa. En esta posición se deberá poner una instrucción de salto al punto donde verdaderamente se inicia el programa. A diferencia de la mayoría de los microprocesadores convencionales, el PC es también accesible al programador como registro de memoria interna de datos; se encuentra en la posición 02. Es decir, cualquier instrucción común que opere sobre registros puede ser utilizada para alterar el PC y desviar la ejecución del programa. (Mandado, et al, 2006)

3.3.3 Stack o Pila.

El registro Stack (Pila) es del tipo LIFO Last In First Out (último en entrar primero en salir). El puntero que lo direcciona es invisible para el programador, solo se accede al ejecutarse una instrucción de salto o llamada a subrutina. En él se guarda el contenido del contador de programa incrementado en una unidad y dicho contenido se transfiere nuevamente al PC al ejecutarse la instrucción de retorno de subrutina. Las posiciones de la pila en el PIC son limitadas, lo que no permite hacer uso intensivo del anidamiento de subrutinas. (Mandado, et al, 2006)

3.3.4 Puertos de Entrada/Salida.

Un recurso imprescindible para los microcontroladores son los puertos de entrada y salida que sirven para comunicarse con los periféricos del mundo exterior. Dependiendo del PIC que estemos utilizando, estos tendrán más o menos puertos de E/S y también dependerá las funciones que puedan realizar, ya que pueden ser digitales, analógicos, etcétera. Los puertos utilizan la denominación de Puerto A, Puerto B, Puerto C, etcétera, dependiendo de la cantidad de puertos de que disponga el PIC. (Mandado, et al, 2006)

3.3.5 Temporizador/Contador.

Una exigencia en las aplicaciones de control es la regulación estricta de los tiempos que duran las diversas acciones que realiza el sistema. El dispositivo típico destinado a gobernar los tiempos recibe el nombre de temporizador o Timmer, *y* básicamente consiste en un contador ascendente o descendente, que determina un tiempo entre el valor que se le carga y el momento en que se produce su desbordamiento o paso por cero. (Mandado, et al, 2006). Figura 3.3.

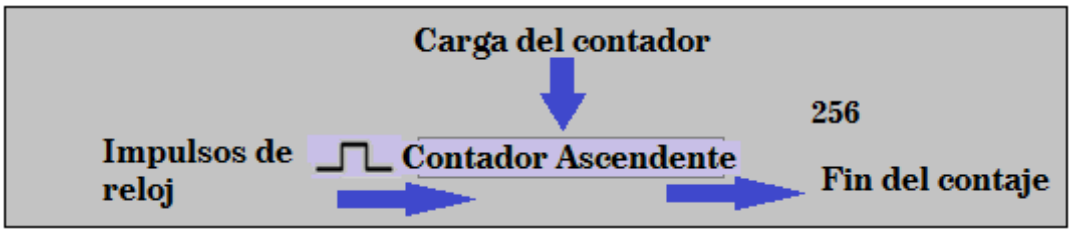

Figura 3.3 Esquema simplificado de un temporizador.

En este caso se trata de un contador ascendente, que se va cargado al ritmo de los impulsos de reloj hasta que llega a 256. La cantidad de temporizadores depende de la gama de PIC que se esté utilizando. La gama baja solo dispone de dos, uno actúa como principal y sobre él recae el control del tiempo de las operaciones del sistema. El otro recibe el nombre de WDT, Watchdog (perro guardián), y se encarga de que el programa no entre en un bucle y dejen de ejecutarse las instrucciones secuenciales. Para realizar esta labor de vigilancia, *da un paseo por el CPU* cada determinado tiempo y comprueba si el programa se ejecuta normalmente, en caso contrario, por ejemplo el control está detenido o a la espera de un acontecimiento que no se produce, actúa y provoca un reset, reinicializando todo el sistema.

Tanto el Temporizador principal, TMR0, como el WDT, a veces precisan controlar tiempos largos y aumentar la duración de los impulsos de reloj, que les incrementan o decrementan. Para cubrir esta necesidad, se dispone de un circuito

programable llamado divisor de frecuencia, que es utilizado por diversos rangos para poder realizar temporizaciones más largas.

3.3.6 Interrupciones.

Una interrupción consiste en una detención del programa en curso, para realizar una determinada rutina que atienda la causa que la ha provocado. Es una llamada a subrutina, que se origina por ejemplo con un desbordamiento en TMR1 o la activación del pin de interrupción RBO/INT. Tras la terminación de la rutina de interrupción, se retorna al programa principal en el punto en que se abandonó.

En las aplicaciones industriales, las interrupciones son una herramienta muy potente para atender los acontecimientos físicos en tiempo real. Existen diferentes tipos de interrupciones que dependen del modelo del PIC que estemos utilizando. En concreto, para saber los tipos que nuestro dispositivo admite se debe consultar la ficha técnica del mismo. (Mandado, et al, 2006)

3.3.7 Modos de direccionamiento.

Cuando programamos un PIC, a la hora de especificar los datos y operandos lo podemos hacer de 3 formas distintas, que llamamos modos de direccionamiento. (http://www.info-ab.uclm.es). Estas 3 formas son:

- Inmediato. El modo inmediato es cuando utilizamos el valor literal.
- Directo. Es cuando utilizamos un valor que apunta a una determinada posición de memoria.
- Indirecto. Lo utilizamos como operando, donde el registro INDF que accede a la posición que apunta el contenido del registro FSR ubicado en el área de datos.

3.4 Oscilador Externo.

En general, el microcontrolador requiere un circuito externo que le indique la velocidad a la que debe trabajar. A este circuito se le conoce con el nombre de oscilador o reloj, es muy simple pero de vital importancia para el funcionamiento del sistema.

Los PIC pueden utilizar cuatro tipos de osciladores diferentes:

- RC. Oscilador con resistencia y capacitor.
- XT. Cristal de cuarzo.
- HS. Cristal de alta velocidad.
- LP. Cristal para baja frecuencia y bajo consumo de potencia.

En el momento de programar el microcontrolador se debe especificar qué tipo de oscilador se usa. Esto se hace a través de unos fusibles llamados *fusibles de configuración*. Generalmente se suele utilizar un cristal de 4Mhz y por lo que internamente esta frecuencia queda dividida por cuatro, lo que hace que la frecuencia efectiva de trabajo sea de 1 MHz, provocando que cada instrucción se realiza en un microsegundo. El cristal debe ir acompañado de dos capacitores y se conecta como se muestra en la Figura 3.4.

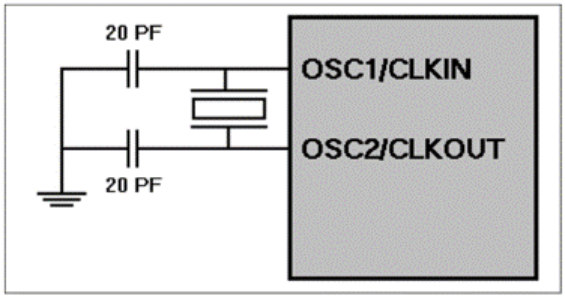

Figura 3.4 Diagrama de conexión de los capacitores y el Cristal.
Imagen tomada de la datasheet del PIC16F877A.

3.5 Herramientas de desarrollo

Microchip technology Inc. Ofrece una serie de herramientas de desarrollo totalmente gratuitas. Las podemos descargar directamente de la página www.michrochip.com. Con estas herramientas podemos programar los microcontroladores PIC y así realizar los proyectos con estos pequeños chips. El software que nos ofrece Microchip es el MPLAB X, un entorno de desarrollo integrado (Integrated development Environment, IDE) que corre en sistemas operativos Windows, mediante el cual se pueden desarrollar aplicaciones para los PIC. (http://www.microchip.com)

Este programa permite escribir, depurar y optimizar nuestros diseños con el PIC. Incluye un editor de texto, un simulador y un organizador de proyectos, además, soporta el emulador PICMASTER y otras herramientas de desarrollo de Microchip, como lo es el MPLAB PM3, EL PICSTART PLUS, entre otras.

3.6 Características del chip PIC16F877A

Para nuestro proyecto utilizamos un PIC16F877A, seleccionado por lo sencillo que es trabajar con él. Tiene un buen promedio en los parámetros de velocidad, consumo, tamaño, alimentación, etcétera, además de contar con abundante información para su uso y un buen precio. La figura 3.5 muestra la distribución de los pines en el PIC16F877A. Las principales características con las que cuenta son:

- Procesador de arquitectura RISC avanzada.
- Juego de 35 instrucciones con 14 bits de longitud. Todas ellas se ejecutan en un ciclo de instrucción, menos las de salto que tardan dos.
- Frecuencia de hasta 20 MHz.
- Hasta 8K palabras de 14 bits para la memoria del código, tipo flash.
- 368 bytes de memoria de datos RAM.
- 256 bytes de memoria de datos EEPROM.
- Hasta 14 fuentes de interrupción internas y externas.
- Pila con 8 niveles.
- Modo de direccionamiento directo, indirecto y relativo.
- Perro guardián (WDT).
- Código de protección programable.
- Modo sleep de bajo consumo.
- Voltaje de alimentación comprendido entre 2 y 5.5 volts.
- Bajo consumo, menos de 2mA a 5 V y 4 MHz.

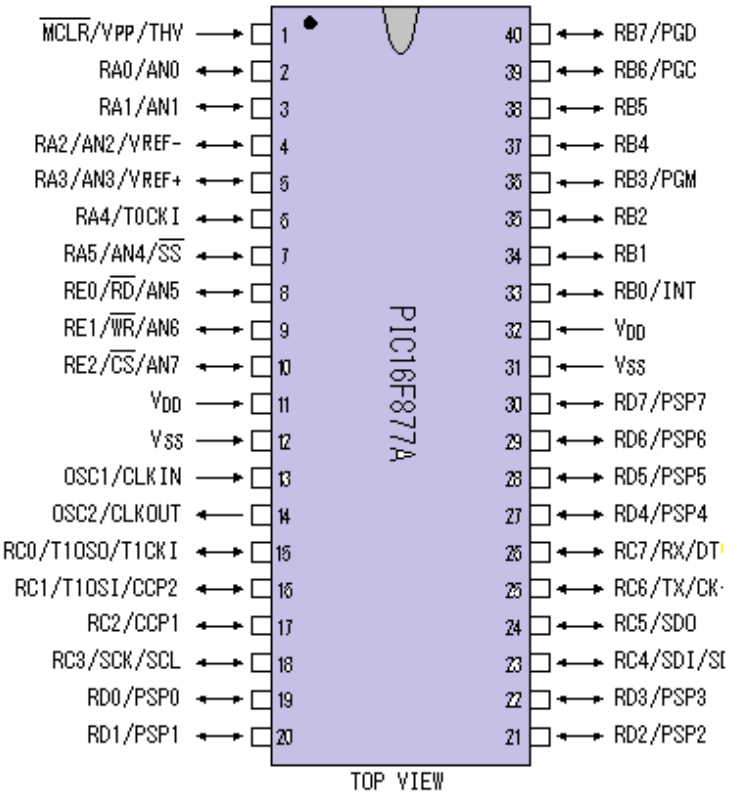

Figura 3.5 Distribución de los pines del PIC16F877A.

Hasta ahora hemos hecho una pequeña reseña de lo que son los PIC, y particularmente del PIC16F877A resaltando sus principales características

Capitulo IV. Diseño del sistema para la operación de luminarias y tomacorrientes.

En este capítulo se explica de forma general como se realizó el prototipo del sistema de operación de luminarias y tomacorrientes.

4.1 Esquema del sistema.

El sistema está constituido principalmente por tres elementos:

- Smartphone: Recibe y envía información para la manipulación de los interruptores. Este se conecta al módulo Bluetooth por medio de un puerto serie virtual.
- Circuito principal: Donde Encontramos el PIC y al módulo Bluetooth. Se encarga de procesar las señales enviadas por el Smartphone.
- Módulos AC: En los cuales están los triacs para el manejo de la corriente alterna. Controlan la *carga*, es decir, están conectados a las lámparas y contactos.

Los tres elementos reciben y envían información entre sí, como se muestra en la Figura 4.1

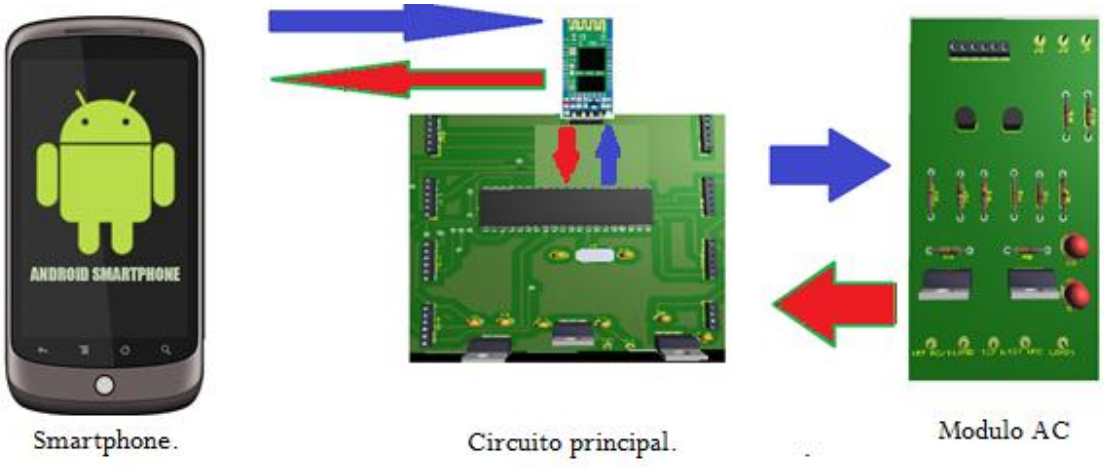

Figura 4.1 Esquema del sistema.

Para que se realice la comunicación entre el dispositivo Bluetooth y el PIC lo primero es hacer el enlace para la transmisión de datos, utilizando un programa para el microcontrolador. En este caso se empleó el lenguaje Assembler para realizar dicho programa.

4.2 Lenguaje Assembler.

Es un lenguaje de programación de bajo nivel para las computadoras, microprocesadores, microcontroladores y otros circuitos integrados programables. Implementa una representación simbólica de los códigos de máquina binarios y otras constantes necesarias para programar una arquitectura dada de CPU y constituye la representación más directa del código máquina específico para cada arquitectura legible por un programador. Esta representación es usualmente definida por el fabricante de hardware y está basada en los mnemónicos que simbolizan los pasos de procesamiento (las instrucciones), los registros del procesador, las posiciones de memoria y otras características. Un lenguaje ensamblador es específico de cierta arquitectura de computadora física (o virtual). (Mandado et al, 2006)

Un programa utilitario llamado compilador es usado para traducir sentencias del lenguaje ensamblador al código de máquina de la computadora. El compilador realiza una traducción más o menos isomorfa (un mapeo de uno a uno) desde las sentencias mnemónicas a las instrucciones y datos de máquina. Muchos sofisticados compiladores ofrecen mecanismos adicionales para facilitar el desarrollo del programa, controlar el proceso de ensamblaje, y la ayuda de depuración. Particularmente, la mayoría de los compiladores modernos incluyen una facilidad de macro, es decir, un conjunto de instrucciones previamente definidas para simplificar y reducir la cantidad de codificación repetitiva. Una vez definida la macro puede ser utilizada en lugar de un mnemónico. Cuando el ensamblador procesa tal sentencia, reemplaza la sentencia por las líneas del texto asociadas a ese macro, entonces las procesa como si hubieran existido en el archivo del código fuente original. (Mandado, et al, 2006)

Este lenguaje fue usado principalmente en los inicios del desarrollo de software, cuando aún no se contaba con potentes lenguajes de alto nivel y los recursos eran limitados. Actualmente se utiliza con frecuencia en ambientes académicos y de investigación, especialmente cuando se requiere la manipulación directa de hardware, alto rendimiento, o un uso de recursos controlado y reducido.

El código para la comunicación que diseñamos se encuentra en el Anexo 1, En la figura 4.2 se muestra un esquema del sistema de operación de luminarias y tomacorrientes.

Una vez teniendo el algoritmo para que nuestro PIC transfiera datos, tenemos que diseñar la aplicación para el teléfono móvil y así podamos enlazarlos e interactúen conjuntamente.

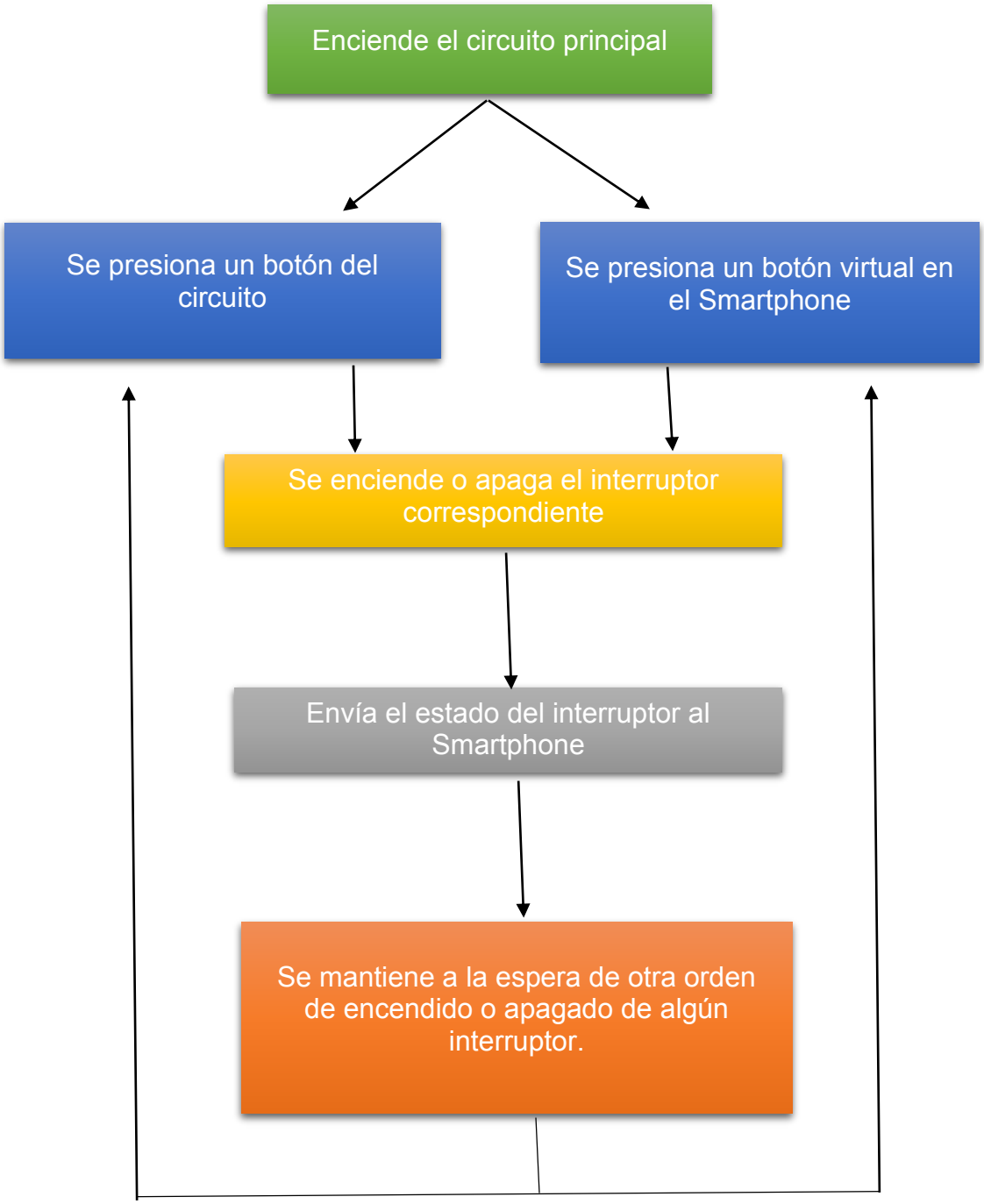

Figura 4.2 Esquema del funcionamiento del programa del PIC.

4.3 Diseño de la aplicación con MIT App Inventor.

MIT App Inventor nos permite efectuar la aplicación de una forma gráfica, por lo tanto, se muestran las imágenes con una breve explicación acerca de cómo se diseñó la app para el teléfono móvil. Para comenzar a trabajar primero ingresamos a nuestra cuenta, creamos nuestro proyecto e iniciamos el diseño de nuestra aplicación.

4.3.1 App Inventor Designer.

En la siguiente figura 4.3 se muestra la *primera pantalla* de nuestra aplicación y consiste en tres etiquetas, una imagen y dos botones. El primero *CONECTAR,* que nos permite iniciar la conexión del teléfono celular al módulo Bluetooth, y el segundo *SALIR,* que si es presionado se encarga de cerrar la aplicación.

Figura 4.3 Pantalla de inicio de nuestra aplicación.

En la siguiente figura 4.4 se muestran los componentes ocultos y los archivos multimedia utilizados para la creación de la aplicación. Estos son Bluetoothclient1, que nos permite la conexión entre el teléfono móvil y el circuito principal, Clock1 y Clock2, que ocuparemos para controlar los códigos de llegada enviados por el circuito principal.

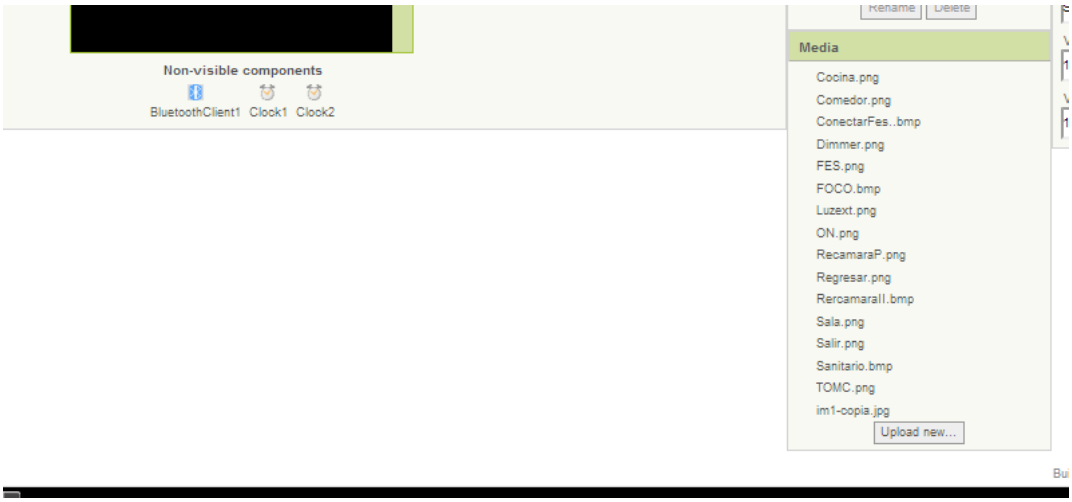

Figura 4.4 Componentes ocultos y los archivos multimedia.

A continuación se muestra *la segunda pantalla* de nuestra aplicación, en la figura 4.5, y está conformada por el menú para seleccionar la habitación donde se quiere aplicar el control de luminaria y/o tomacorriente. Consta de ocho botones que muestran imágenes representativas de la habitación, además, de un botón para regresar a la pantalla anterior y uno más para cerrar la aplicación.

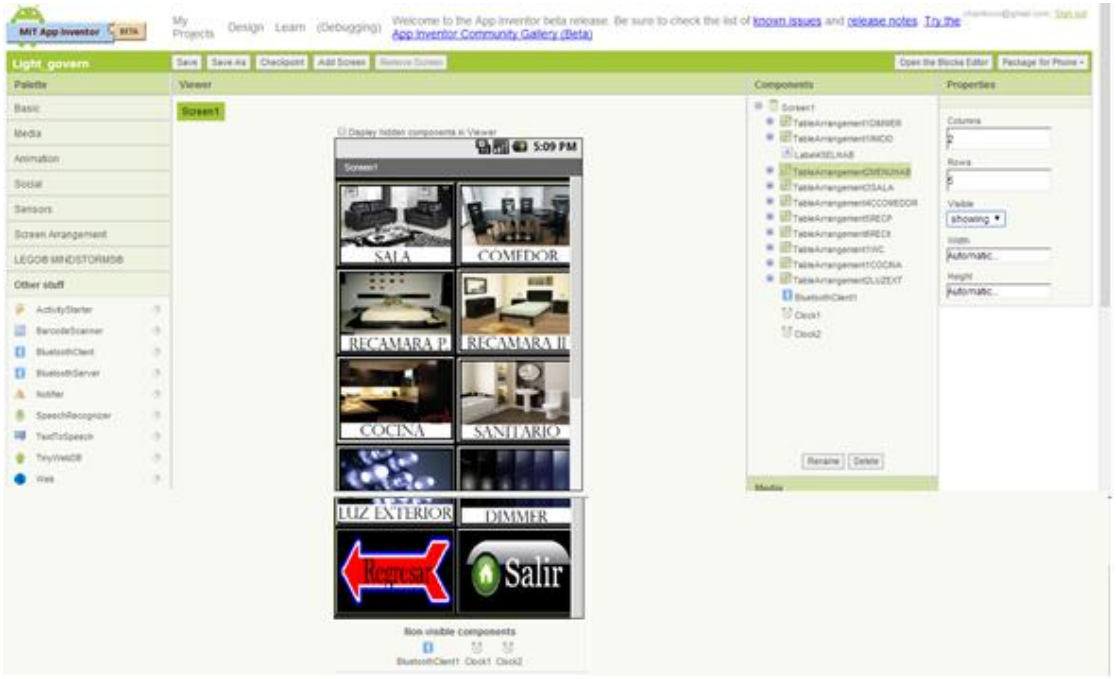

Figura 4.5 Pantalla menú habitaciones.

La siguiente figura 4.6 muestra *la pantalla sala.* Este diseño se repite siete veces, solo cambia el nombre de las habitaciones y en ellas se envían los códigos para el control de los interruptores. Está formada por tres etiquetas, dos imágenes y cuatro botones. El botón *REGRESAR* se encarga de llevarnos al *menú*

habitaciones, el botón *SALIR* cierra la aplicación y los botones *ON* se ocupan para encender o apagar la luminaria y/o el tomacorriente. También son utilizados para informarnos del estado de los interruptores, ya que la palabra *ON* alterna con la palabra *OFF,* dependiendo si esta encendido o apagado respectivamente.

Figura 4.6 Pantalla sala para el control de dos interruptores.

La figura 4.7 muestra la pantalla de control *DIMMER,* consta de seis etiquetas y siete botones. Los botones *OFF* son para seleccionar el nivel de intensidad en la salida del atenuador, el botón *REGRESAR* nos dirige al menú anterior y el botón *SALIR* se ocupa de cerrar la aplicación.

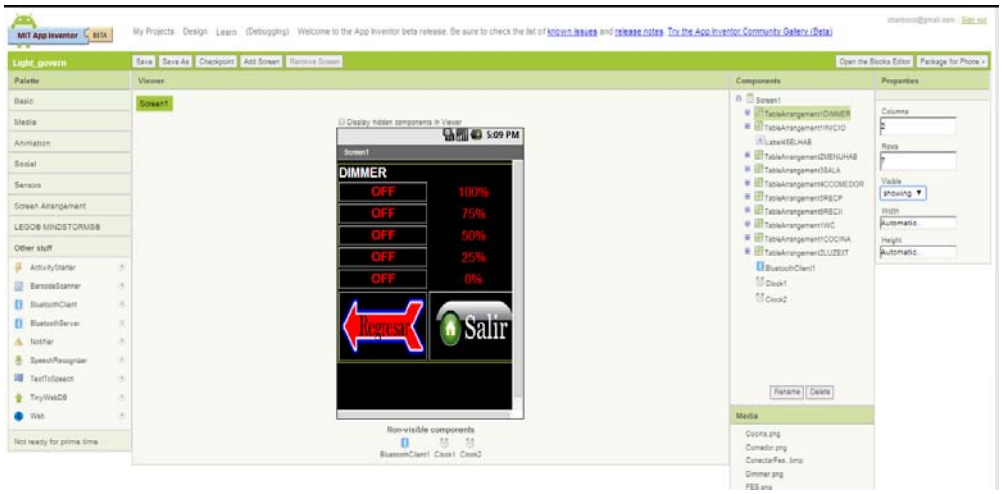

Figura 4.7 Pantalla *DIMMER.*

Básicamente es el diseño de la aplicación. Ahora nos vamos a la siguiente herramienta, que es donde le diremos a cada componente como debe comportarse para que funcione en conjunto con nuestro programa del PIC.

4.3.2 App inventor Blocks Editor.

Esta herramienta nos permite decirle a las componentes como deben comportarse, de una manera gráfica. Por tal motivo pondré imágenes explicando el funcionamiento de las conexiones realizadas en App Inventor Blocks Editor. En la figura 4.8 se muestran todos los elementos de la aplicación en el área de trabajo.

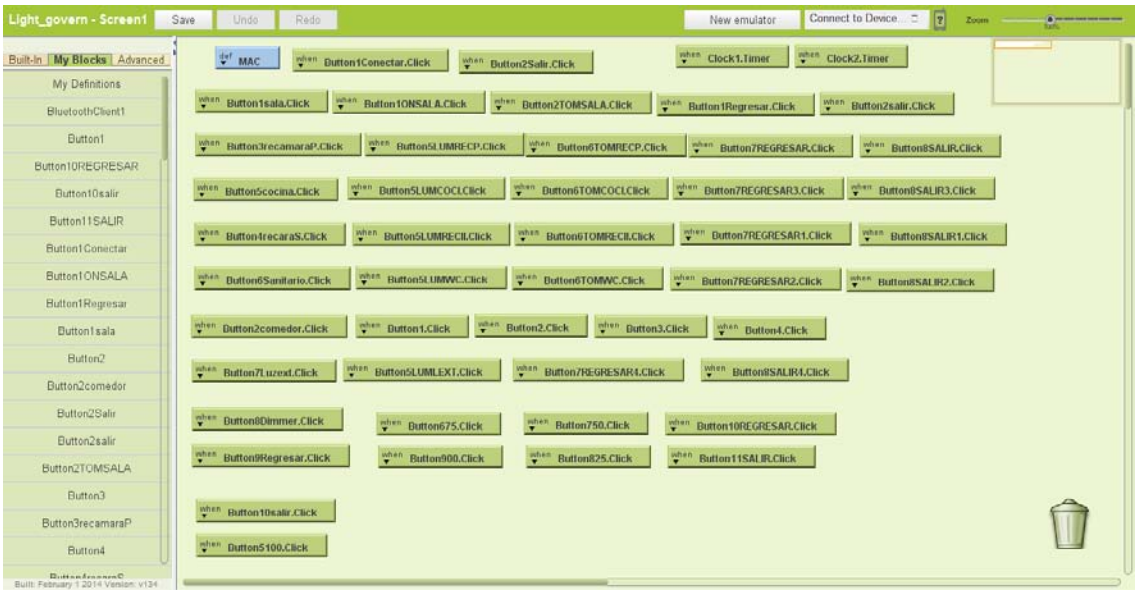

Figura 4.8 Componentes de la aplicación en el área de trabajo.

En la figura 4.9 se muestran los elementos que conforman la *pantalla inicio* y a continuación se describe la acción que realiza cada bloque.

Primero definimos la dirección MAC, Media Access Control (control de acceso al medio) o dirección física de nuestro modulo Bluetooth, por medio del bloque MAC, ya antes declarado.

Seleccionamos *Button1CONECTAR.Click,* le damos la instrucción que, al presionar dicho botón, de la orden de que se conecte el teléfono celular con el módulo Bluetooth JY-MCU, enseguida oculte la *pantalla inicio* y muestre la *pantalla menú habitaciones*.

Finalmente tomamos el *Button2Salir.Click* y le indicamos que al pulsarlo cierre la aplicación. La figura 4.10 Muestra la *pantalla Inicio* en nuestro teléfono móvil.

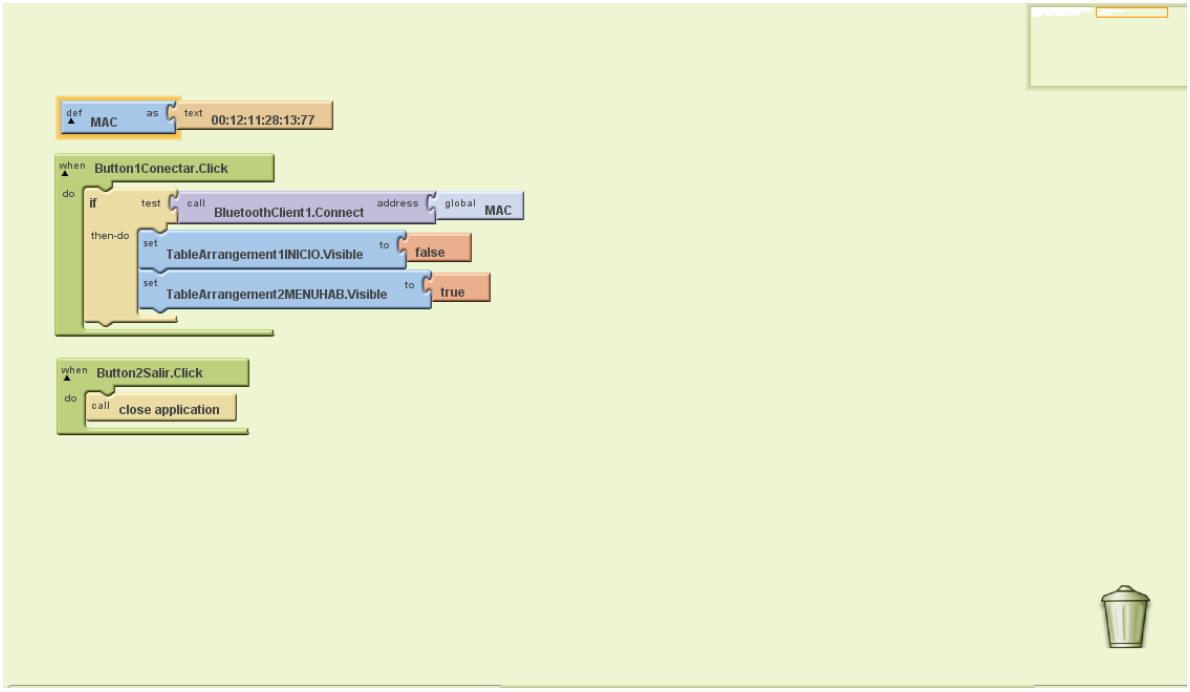

Figura 4.9 Componentes de la pantalla inicio.

Figura 4.10 Pantalla inicio.

En la próxima figura 4.11 se exponen los componentes de la pantalla *menú habitaciones,* y enseguida se explica la función de cada bloque.

El bloque *Button1Sala.Click* manda una “a” al módulo JY-MCU, oculta la *pantalla menú habitaciones* y muestra la *pantalla Sala.* Así es el comportamiento de la mayoría de los bloques, solo cambia la letra que envían al dispositivo Bluetooth y la pantalla que muestra. Por ejemplo, si presiono el botón *button3RecamaraP.Click* envía una “b” y nos dirige a la pantalla *RECAMARA P.* Este comportamiento cambia

únicamente en los bloques *Button8Dimmer.click, Button1Regresar.Click* y *Button2salir.* Lo que es distinto en estos es que no envían ninguna letra al dispositivo Bluetooth, El primero nos direcciona a la *pantalla DIMMER,* el segundo nos dirige a la *pantalla Inicio* y el tercero cierra la aplicación.

En la figura 4.12 se muestra la pantalla *menú habitaciones* en nuestro teléfono celular.

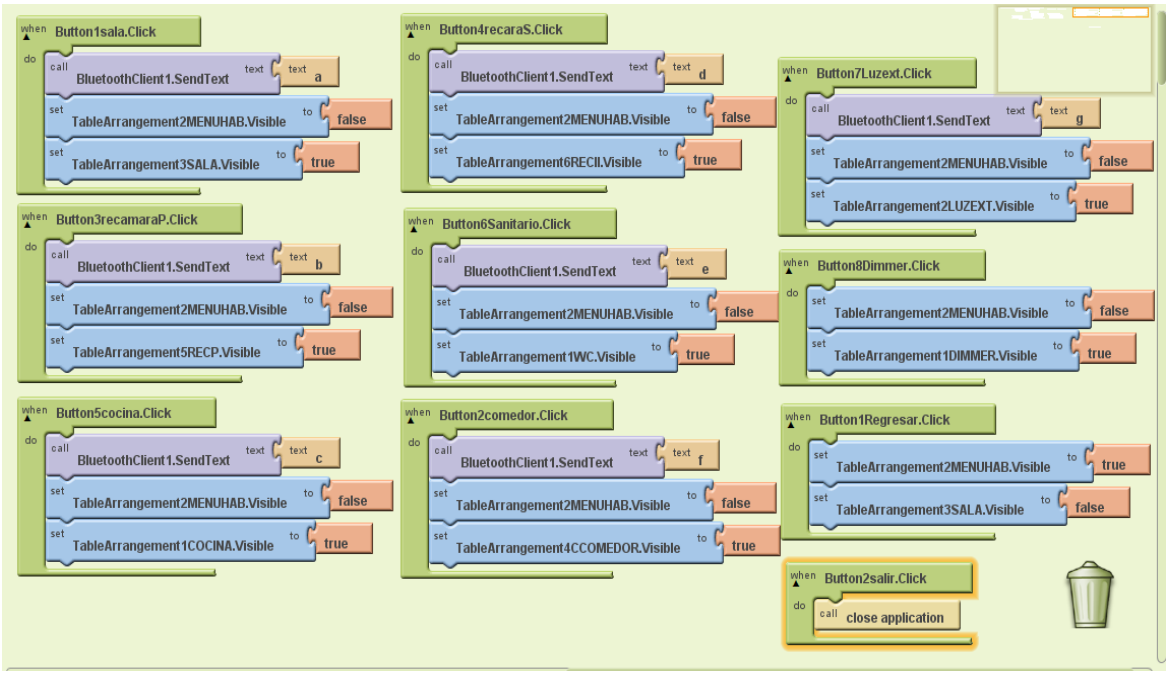

Figura 4.11 Elementos de la pantalla menú habitaciones.

Figura 4.12 Pantalla menú habitaciones.

En la siguiente figura 4.13 se muestran las componentes de la *pantalla Sala* y a continuación se explica la acción de cada bloque. Como ya se mencionó, esta pantalla se repite siete veces, solo cambia el nombre de la pantalla y la tetra que envían al dispositivo JY-MCU. Esta pantalla consta de cuatro bloques, el primero *Button1ONSALA.Click,* envía una letra “A”, el segundo *Button2TOMSALA.Click*, transmite una letra “B”, el tercero *Button1Regresar.Click,* esconde la *pantalla Sala* y muestra la *pantalla menú habitaciones,* y por último el cuarto bloque *Button2salir.Click* cierra la aplicación.

En la figura 4.14 podemos apreciar la pantalla sala en nuestro teléfono móvil.

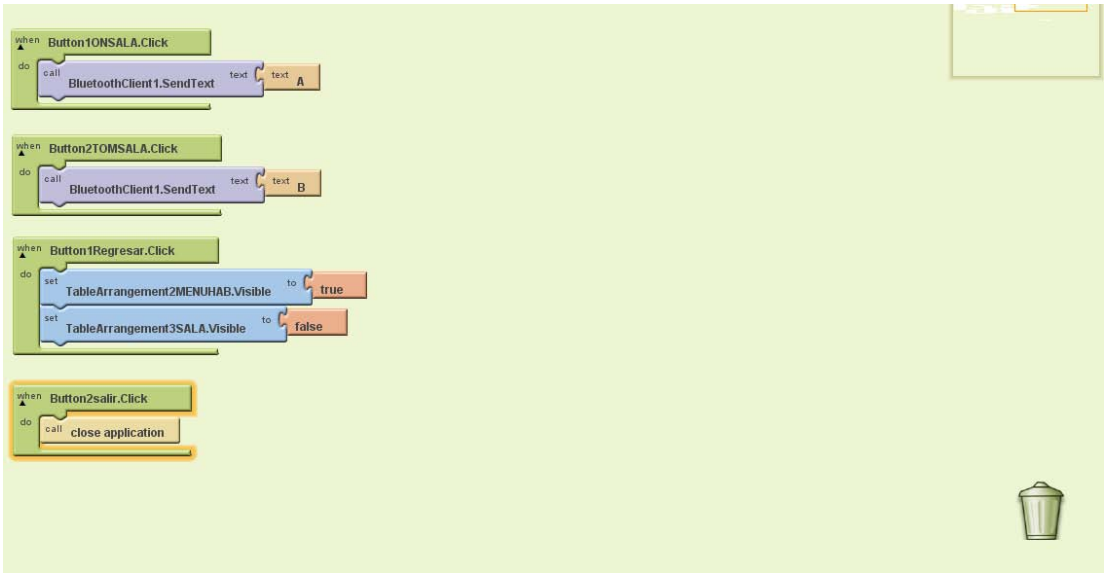

Figura 4.13 Bloques de la pantalla Sala.

Figura 4.14 Pantalla sala en el teléfono móvil.

En la figura 4.15 podemos apreciar los elementos de la pantalla *DIMMER,* y enseguida una descripción de los bloques que la componen.

El bloque *Button5100.Click,* es similar a los bloques *Button675.Click, Button750.Click, Button825.Click,* y estos se encargan de enviar una letra al dispositivo Bluetooth “V”, “W”, “X” y ”Y”, respectivamente. También se encargan de cambiar el texto *OFF* en rojo por *ON* en verde al ser elegidos y el color de la etiqueta de rojo a amarillo. Por ejemplo, si selecciono el botón *OFF 25%,* este cambiará a *ON 25%,* como se muestra en la figura 4.16, donde los botones están en OFF y el color de sus etiquetas es rojo, excepto el botón 25% que se encuentra en ON verde y su etiqueta es color amarillo. El bloque *Button900.Click* envía una “Z” y regresa a rojo todos los botones y etiquetas, indicando que está apagado el atenuador. Igual que en pantallas anteriores, el bloque *Button10REGRESAR* se ocupa de dirigirnos al *menú habitaciones,* y el bloque *Button11SALIR.Click cierra la aplicación.*

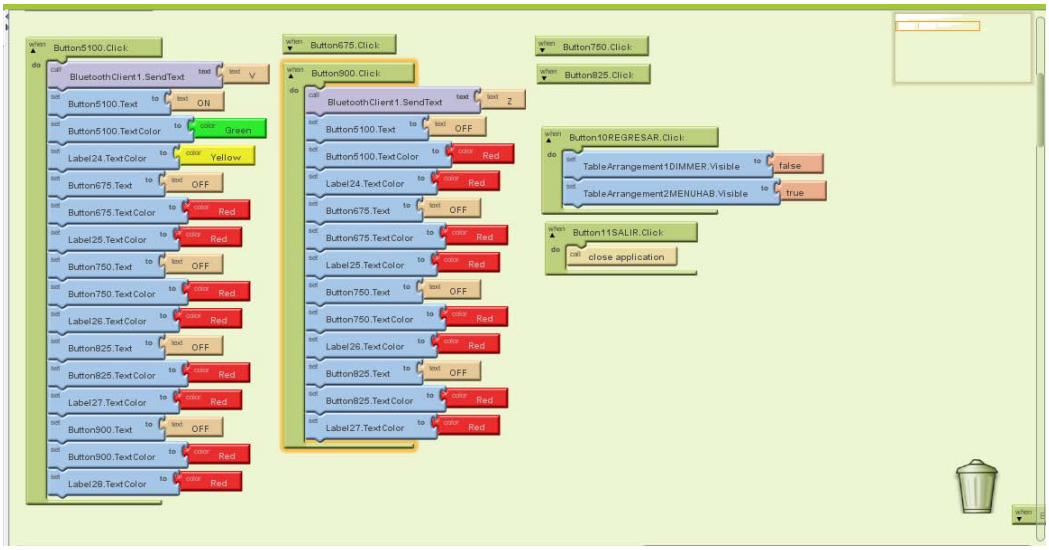

Figura 4.15 Bloques que componen la pantalla DIMMER.

Figura 4.16 Muestra la pantalla DIMMER en nuestro teléfono celular.

En la Figura 4.17 se encuentran los códigos de llegada. Al ser una gran columna, no se pueden apreciar de forma clara, por tal motivo en la figura 4.18 se

muestra solo una parte, para dar una breve explicación de su funcionamiento en nuestra aplicación.

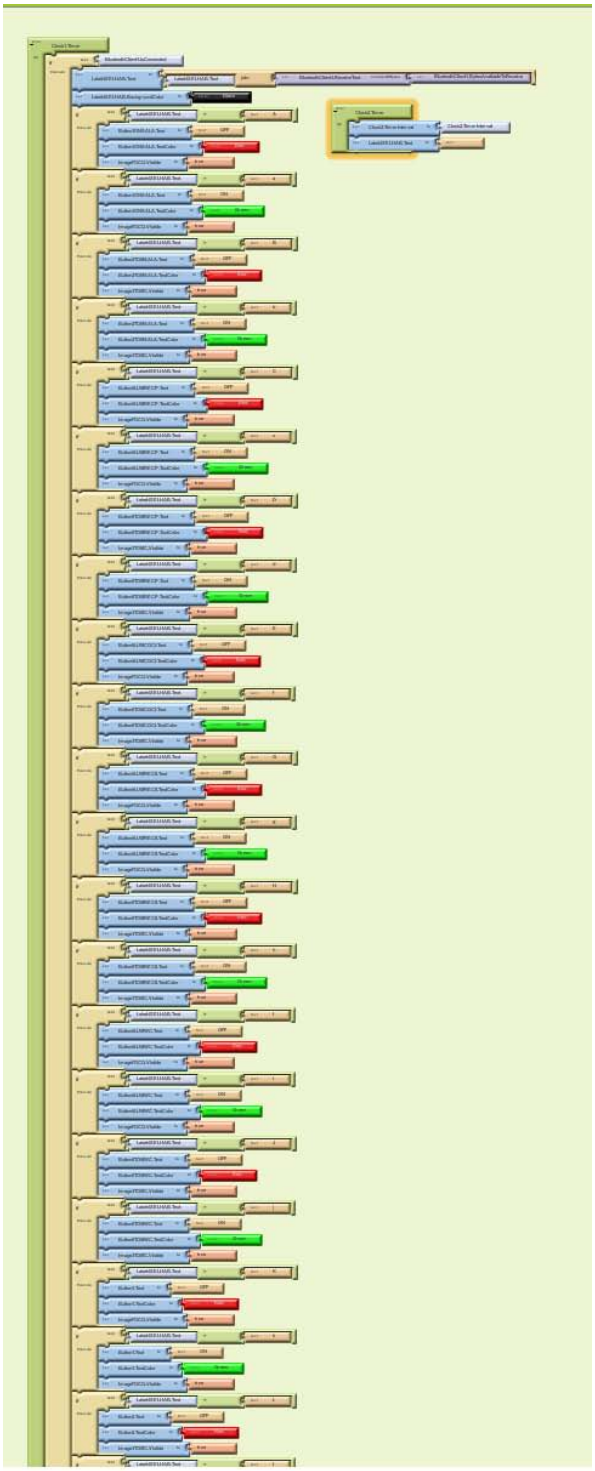

Figura 4.17 Bloques pertenecientes a los códigos de llegada.

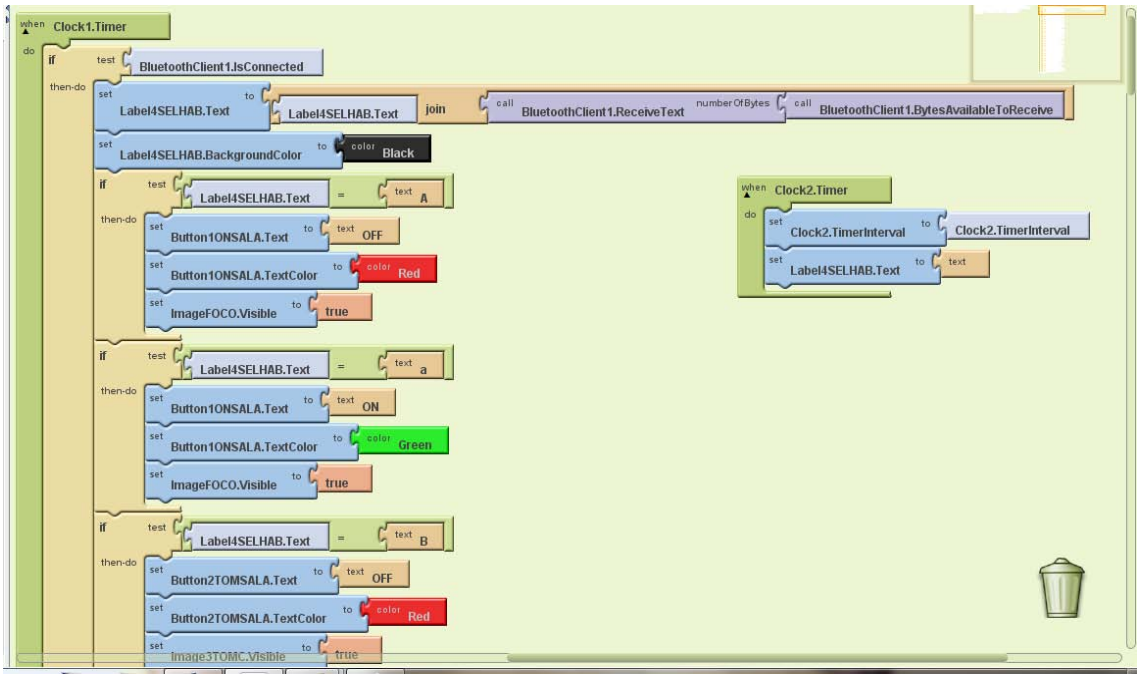

Figura 4.18 Algunos bloques de los códigos de llegada.

La componente Clock1 se encarga de procesar estos códigos. El bloque *Clock1.Timmer,* recibe una *"letra"* la coloca en una etiqueta y luego la compara para hacer la acción correspondiente.

Por ejemplo si recibe una letra "A", cambia el texto del botón *ON* a *OFF* y el color del mismo de verde a rojo, esto lo hace en el botón correspondiente, en este caso el botón de la luminaria Sala. Si recibe una letra "a" hace lo opuesto, es decir, el texto del botón *OFF* lo cambia por *ON* y el color del mismo de rojo a verde. De esta manera sabemos sí está encendido o apagado el interruptor que sea de nuestro interés.

El funcionamiento para el resto de los botones, es igual, cambiando el texto y el color dependiendo del dato recibido.

El Bloque *Clock2.Timmer*, se ocupa de borrar la etiqueta donde se alberga la "letra" para ser comparada, con el fin de que no haya confusión y no se acumulen las "letras" en dicha etiqueta.

Con esto terminamos el diseño de la aplicación, ahora, para poder llevarla a nuestro teléfono celular, MIT App Inventor nos da las siguientes opciones en el menú Package for phone (paquete para el teléfono), este menú lo podemos encontrar en la parte superior de nuestra pantalla en App Inventor Designer, que se muestran en la figura 4.19

Figura 4.19 Menu package for phone App Inventor Designer.

En el menú mencionado elegimos la opción Download to Connected Phone (descargar al teléfono conectado), para esto, previamente hay que conectar nuestro teléfono móvil a la computadora, por medio de un cable USB, una vez conectado nuestro dispositivo, seleccionamos en el menú de App Inventor Blocks Editor Connect to Device (Conectar dispositivo) en este caso 80A354046043372304, como se muestra en la figura 4.20

Figura 4.20 Menu Connect to Device en App Inventor Blocks Editor.

Esperamos a que se conecte, esto es, cuando podamos ver la aplicación en nuestro dispositivo. Una vez que es visible la *pantalla inicio* en el teléfono móvil, procedemos a dar click en el menú Download to Connected Phone. Se inicia la descarga e instalación de nuestra aplicación, al finalizar, tendremos el mensaje

Application successfully downloaded to phone (aplicación descargada correctamente en el teléfono), como se muestra en la figura 4.21

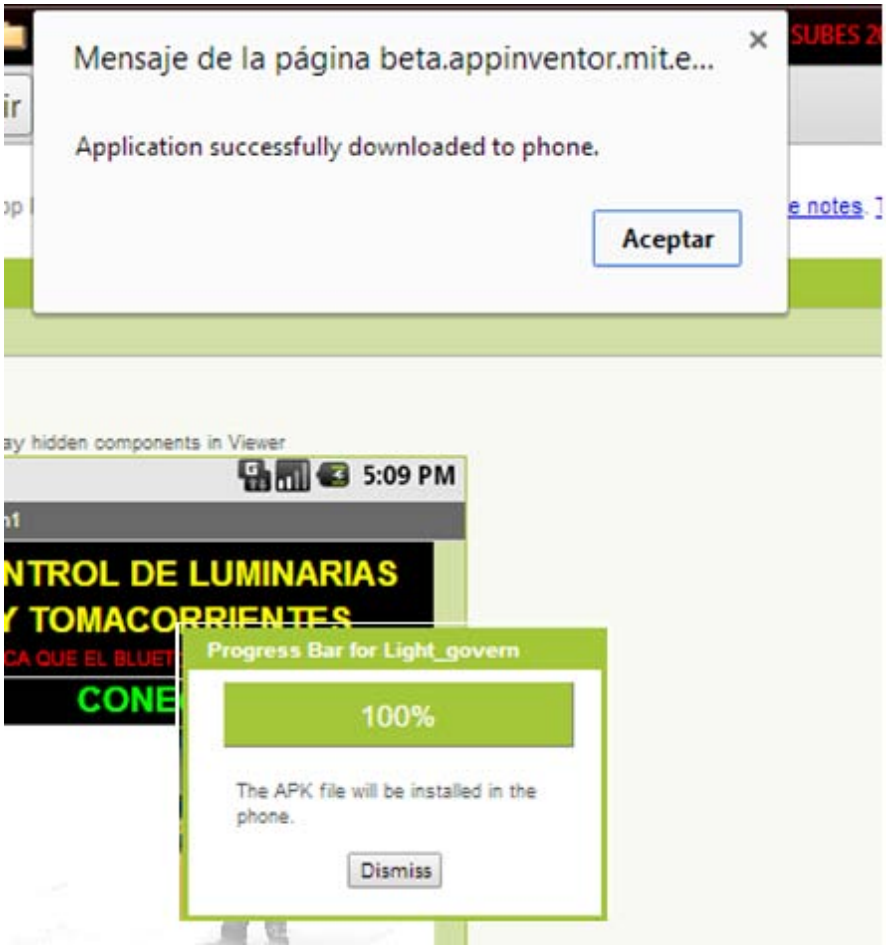

Figura 4.21 Mensaje que la aplicación ha sido instalada correctamente en nuestro dispositivo.

Ahora la aplicación ya está en el teléfono móvil y la podremos utilizar en conjunto con nuestro circuito para el control de luminarias y tomacorrientes.

4.4 Circuito para el control de luminarias y tomacorrientes en Proteus desing suite 8.0.

Para la elaboración del circuito, nos apoyamos en un software, tanto para hacer el diagrama del circuito como para realizar el diseño de la placa de circuito impreso.

4.4.1 Proteus desing suite 8.0.

Proteus es un software para diseñar y simular circuitos electrónicos, permite la captura esquemática y diseño de la placa de circuito impreso. Es desarrollado por Labcenter Electronics. Los componentes del software son:

- ISIS Schematic Capture. Herramienta para la captura del esquema.
- ProSpice Simulation SPICE. Simulación de circuitos.
- ARES PCB printed circuit board (placa de circuito impreso) Layout. Diseño de PCB.
- VSM Virtual System Modelling (sistema virtual de modelado). Simulación de varias familias de microcontroladores en tiempo real. La figura 4.22 muestra la pantalla de inicio de Proteus desing suite 8.0.

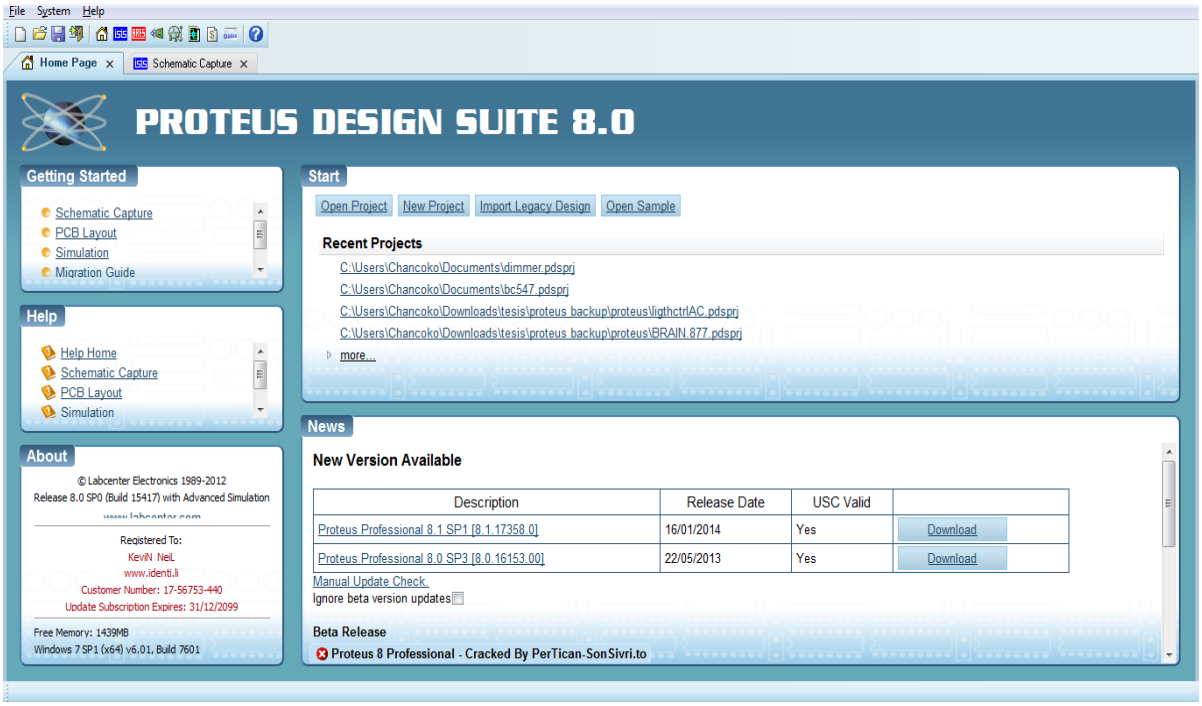

Figura 4.22 Pantalla de inicio Proteus.

4.4.2 Circuito principal.

Para la elaboración de nuestro esquema del circuito principal, primero seleccionamos las componentes a utilizar en ISIS, a continuación se enlistan con la letra y número que ocupan en el diagrama, estas son:

- U1. PIC16F877A.
- X1. Cristal de 4 MHz.
- U2, U3 y U4. MC7805CT.

- C7 y C8. Capacitor de 22 pF.
- C1, C3 y C5. Capacitor de 0.33 µF.
- C2, C4 y C6. Capacitor de 0.1 µF.
- BLUETOOTH y JM. Conector SIL-4.
- JA-B, JC-D, JE-F, JG-H, JI-J y JK-L. Conector SIL-6.
- Vcc y Gnd. Pin.

La figura 4.23 Muestra la selección de los componentes en ISIS de Proteus.

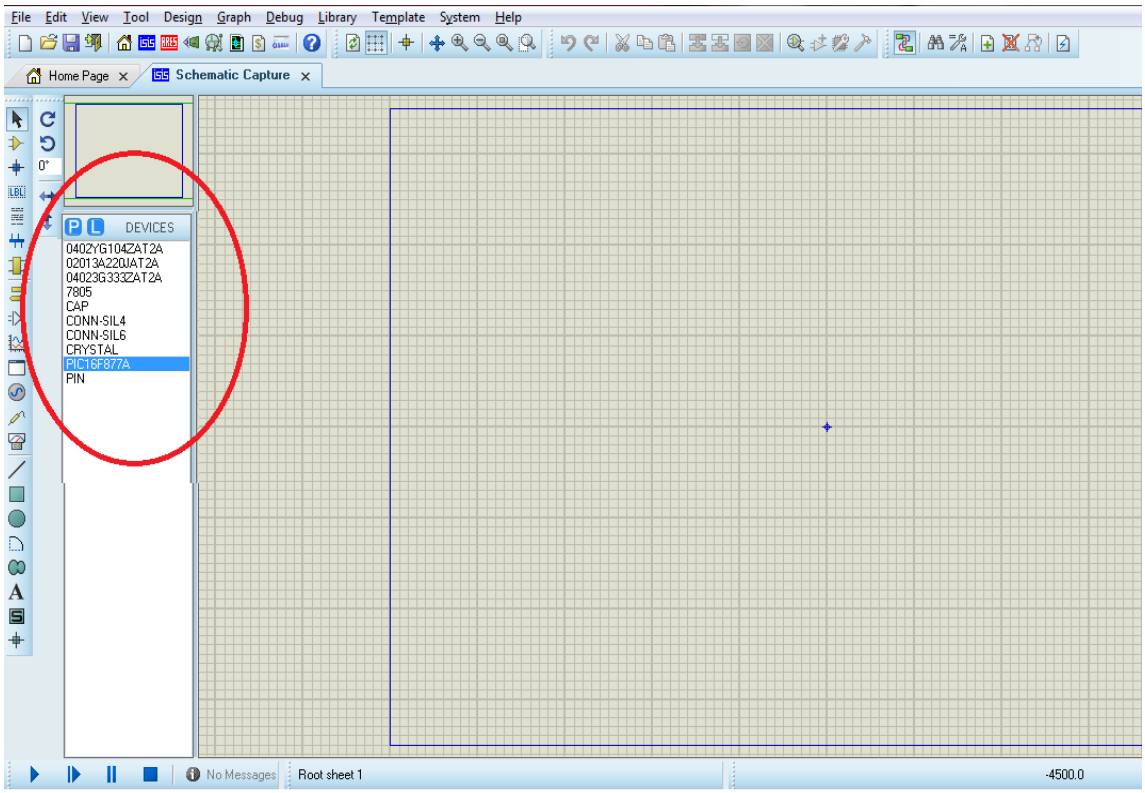

Figura 4.23 Componentes del circuito principal.

Una vez teniendo estas componentes en nuestro menú *DEVICES* podemos arrastrarlas a nuestra área de trabajo y unirlas para que funcionen como nosotros esperamos.

La figura 4.23 muestra el circuito principal en el área de trabajo.

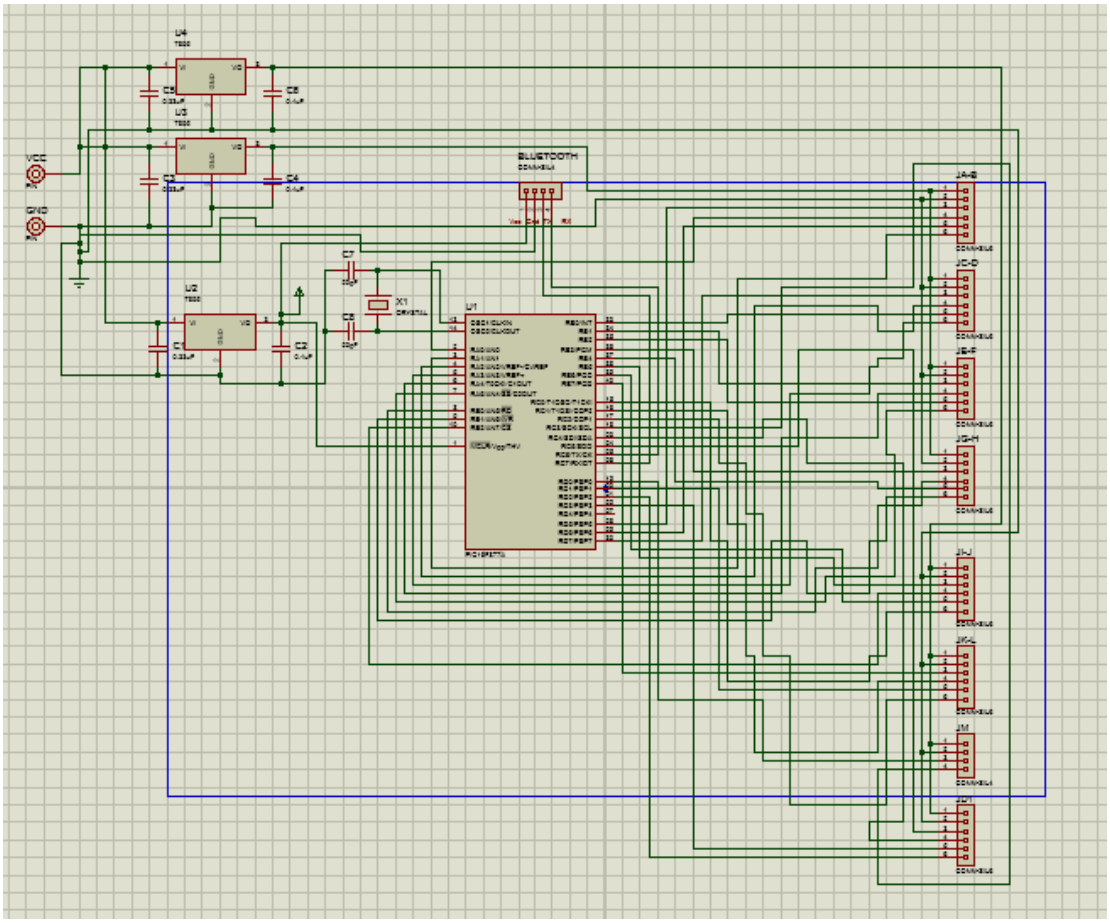

Figura 4.24 Esquema del circuito principal en ISIS.

En nuestro esquema, el PIC trabaja con un cristal de 4 MHz conectado en OSC1 y OSC2 (pines 13 y 14), con sus respectivos capacitores de 22 pF. Para el funcionamiento de esté el pin $\overline{MCLR}$ (master clear pin 1) debe ir conectado a VCC.

Los conectores JA-B, JC-D, JE,F, JG-H, JI-J y JK-L cuentan con seis líneas que son Vcc, Gnd, Tx1, Rx1, Tx2, Rx2 respectivamente, donde Tx1 se conecta al puerto RD5 del PIC y este nos controla un interruptor, Rx1 se conecta al puerto RA0 para recibir una señal de un botón ubicado en el módulo AC JA-B y así poder indicarle al PIC que queremos hacer el control del puerto RD5 por medio de este y no por el teléfono móvil. De esta manera están conectados el resto de los módulos a los diferentes puertos del PIC, excepto el módulo JM y el módulo JD1 el módulo JM, es un conector SIL-4 y solo cuenta con Vcc, Gnd, Tx1 y Rx1 por lo tanto este módulo solo tiene el control de un interruptor, no como los anteriores que tienen el control para dos. El módulo JD1 solo puede controlarse por medio del teléfono celular así que no cuenta con botones independientes para el control, de igual forma

es un conector SIL-6 y sus líneas son Vcc, Gnd, Tx1, Tx2, Tx3 y Tx4. Los conectores se relacionan como se muestra en la tabla 4.1

Tabla 4.1 Relación entre los conectores y los puertos del PIC.

Conector “Letra”	Puerto Tx	Puerto Rx
A	RD5	RA0
B	RD6	RA1
C	RD7	RA2
D	RB0	RA3
E	RB1	RA4
F	RB2	RA5
G	RB3	RE0
H	RB4	RE1
I	RB5	RE2
J	RB6	RC0
K	RB7	RC1
L	RD1	RC2
M	RD0	RC3
D1-V	RC5	
D2-W	RC4	
D3-X	RD3	
D4-Y	RD2	

Existe otro conector SIL-4 para la conexión del módulo Bluetooth, tiene cuatro líneas Vcc, Gnd, Tx y Rx. Se conecta Vcc (pin 1) a Vcc, Gnd (pin 2) a Gnd, Tx del dispositivo Bluetooth (pin 3) al puerto Rx del PIC (pin 26) y Rx del módulo Bluetooth (pin 4) al puerto Tx del PIC (pin 25).

Para la alimentación del circuito hay 3 MC7805CT, esto es, uno para alimentar al PIC y al módulo Bluetooth, otro para alimentar 4 módulos AC (JA-B, JC-D, JE-F, JG-H) y el último para alimentar a los 4 módulos restantes (JI-J, JK-L, JM Y JD1). Se realizó de esta manera para distribuir el consumo de la corriente y no exigir demasiado a cada regulador, el circuito se alimenta con una fuente de 9 V y 500 mA. Ahora que ya tenemos nuestro circuito conectado procedemos al diseño de la tarjeta de circuito impreso en ARES.

Para el diseño de la PCB guardamos nuestro trabajo en alguna carpeta, después seleccionamos el icono de ARES y proseguimos a colocar las componentes en el área de trabajo, una vez realizado esto, configuramos el tamaño de las pistas, el número de caras y procedemos al ruteo automático.

Y con eso será suficiente para tener ya el diseño de nuestra PCB, La siguiente figura 4.25 muestra el diseño terminado, es de dos caras y un tamaño de 10x10 cm.

Una vez terminado el diseño, se puede proceder a grabar la placa, utilice el método del papel transfer y lo único que se tiene que hacer es imprimir las caras en este papel y colocarlo en la placa, pasarle la plancha y queda transmitido nuestro diseño, para proceder al grabado, en mi caso empleé cloruro férrico para esto, finalmente teniendo la PCB grabada podemos proceder a hacerle las perforaciones necesarias.

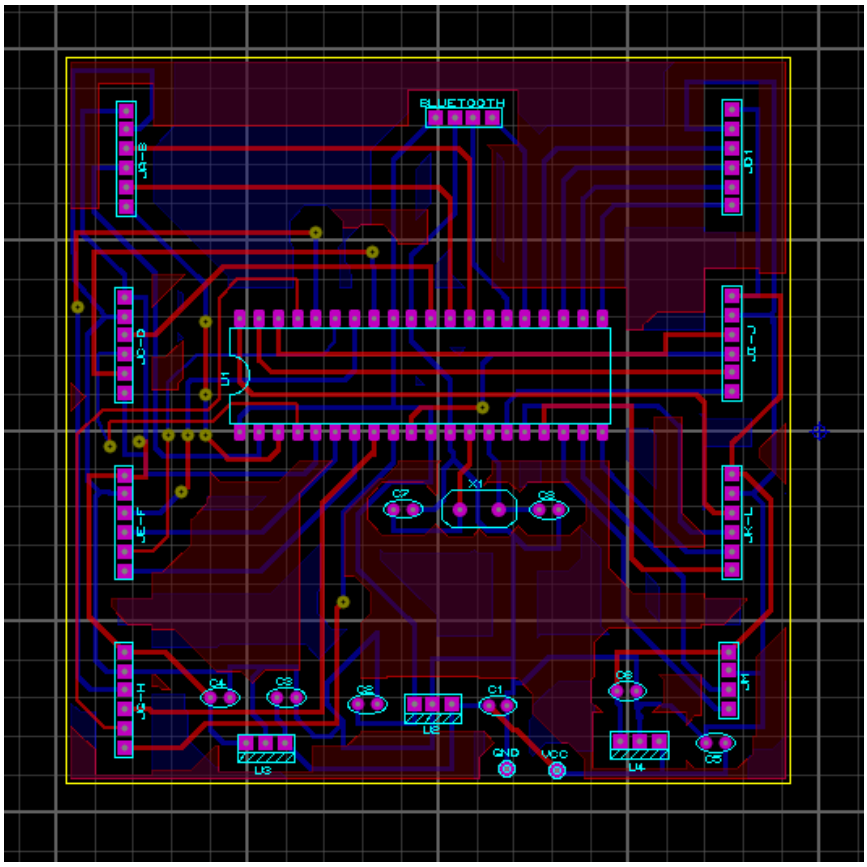

Figura 4.25 Diseño PCB en ARES terminado.

Con las perforaciones terminadas, colocamos y soldamos las componentes y estará listo para conectarle los módulos AC. Debe quedar como se muestra en la Figura 4.26.

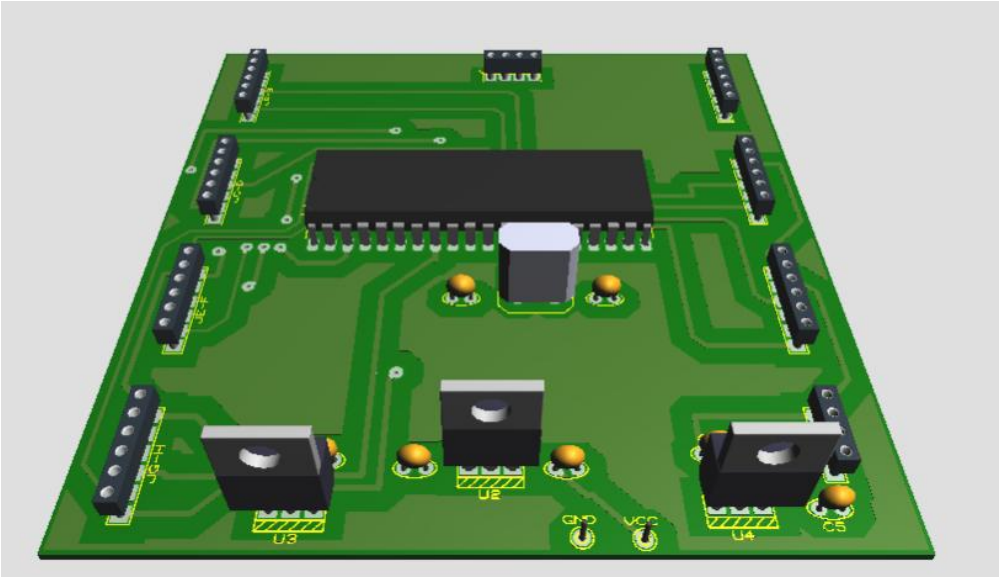

Figura 4.26 Esquema 3D del circuito principal.

En la figura 4.27 podemos ver el Circuito principal terminado.

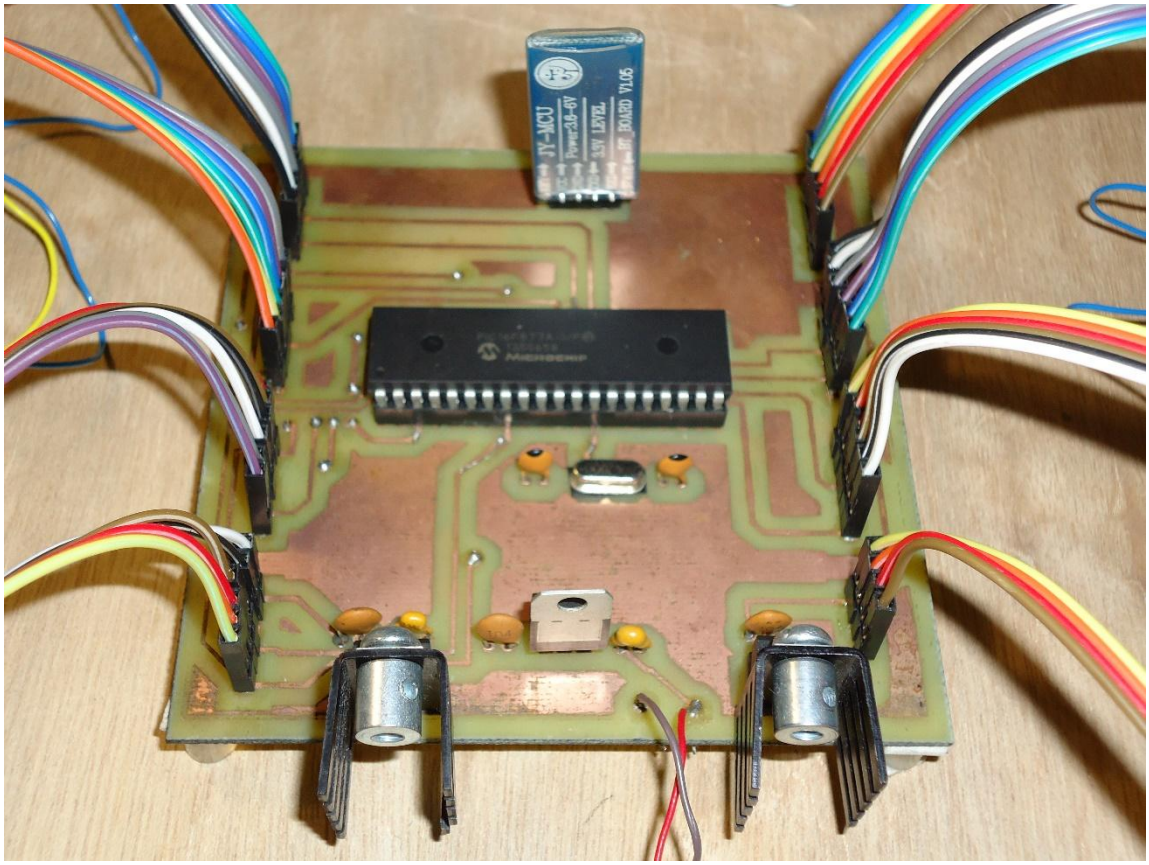

Figura 4.27 Circuito principal terminado.

4.4.3 Módulos AC.

Para el diseño del módulo AC se siguió el mismo camino, primero se eligieron los componentes para el diseño del circuito, en este caso:

- Q1 y Q2. BC547.
- J1. Conector SIL-6.
- U1 y U2. 2N6071A.
- D1 y D2. LED azul y rojo de 3MM.
- R1 y R5. Resistencias de 1.5 kΩ.
- R2 y R6. Resistencias de 100 Ω.
- R3 y R7. Resistencias de 330 Ω.
- R4, R8. Resistencias de 560 Ω.
- R9 y R10. Resistencias de 1 KΩ.
- LOAD, 127VAC, 127 N, LOAD1 y 127/AC1. Pin.
- Fusible.

La figura 4.28 Muestra las componentes en ISIS de Proteus.

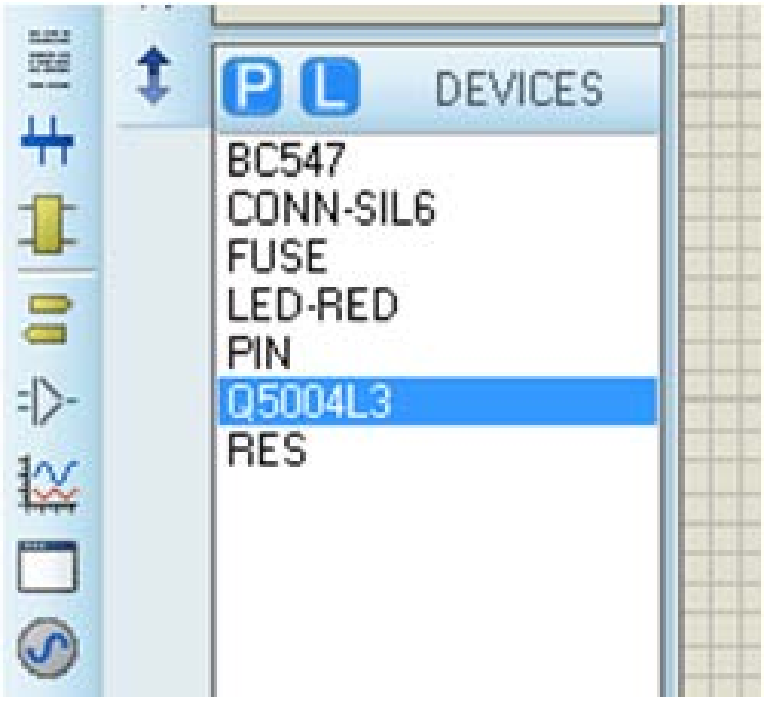

Figura 4.28 Componentes del módulo AC.

En la figura 4.28 se muestra Q5004L3, ya que en la librería no existía el componente 2N6071A y solo se puso led-red, ya que sus dimensiones son iguales a las del led blue y se utilizó este para ambos casos.

Ahora se puede proceder a realizar las conexiones necesarias. Obteniendo de resultado el diagrama que se expone en la figura 4.29

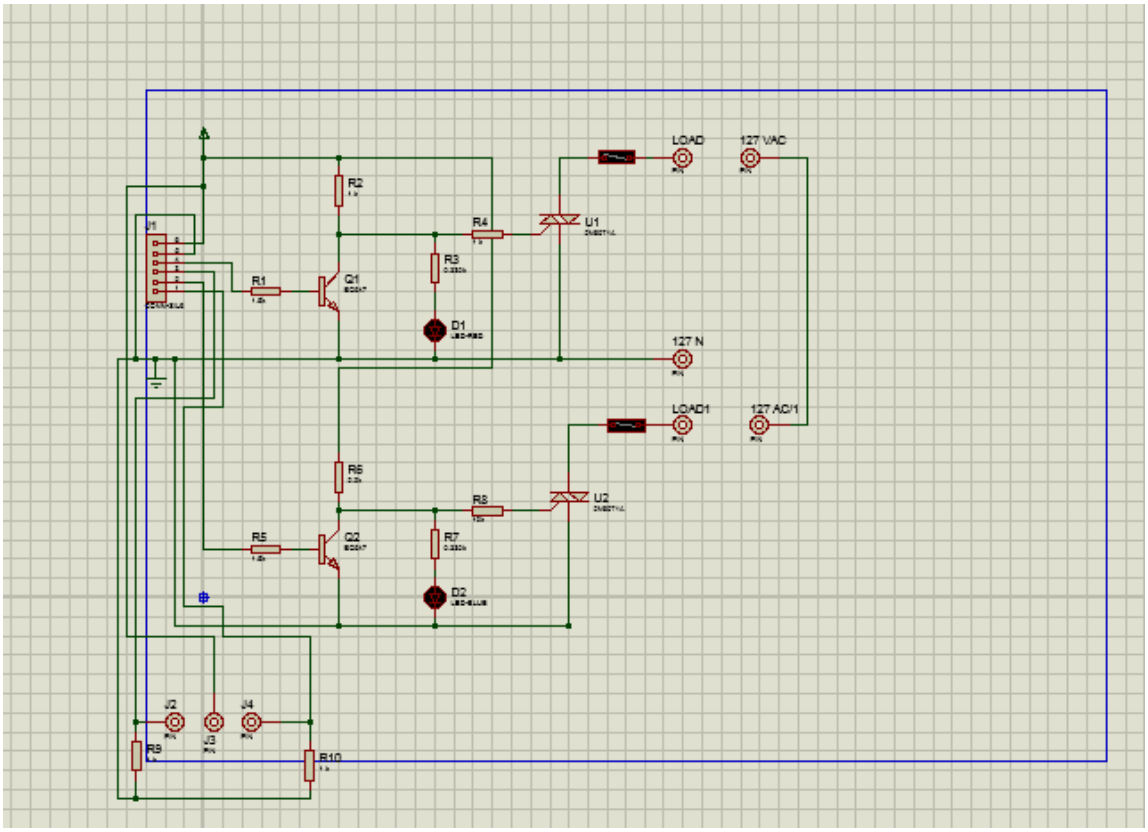

Figura 4.29 Diagrama del circuito módulo AC.

El conector SIL-6 cuenta con seis líneas Vcc, Gnd, Tx1, Rx1, Tx2 y Rx2, Donde Tx1 está conectado a la resistencia R1 (de 1.5 KΩ) y a la salida, esta se conectada a la base del transistor Q1 (BC547 esto con el fin de utilizarlo en corte y saturación para disparar el Triac), el transistor está conectado a Vcc en el colector por medio de R2 (de 470 Ω) y de esta conexión se desprende R3 (330 Ω) y R4 (1kΩ). R3 se dirige al led D1 y este a Gnd, mientras R4 se conecta a Gate de U1. MT2 se une a Gnd y al neutro de la línea y por ultimo MT1, donde se coloca el fusible y este debe ir conectado a la carga que se pretende controlar, el otro extremo de la carga se conecta a la fase de la línea 127 V AC.

Rx1 está conectado a R9 (de 1 KΩ) y se encuentra enlazado a Gnd (Esto con el fin de tener una referencia para la señal transmitida al PIC), y también está conectada a J2 donde se pretende ubicar el botón para hacer el control desde este, J3 solo se conecta a Vcc aquí es donde se conecta el otro extremo de nuestro botón (Con el propósito de mandarle una señal al PIC y de esta forma cambiar el estado del interruptor). De igual manera se conecta Tx2 pero a Q2 para disparar a U2, mientras hace lo propio para mandar señales al PIC Rx2.

Solo se hizo este diseño para el sistema de control de luminarias y toma corrientes, debido a que lo que se pretende es solo mostrar un prototipo, sin instalarlo en algún lugar específico. De lo contrario si se desea instalar primero se deben hacer los cálculos de las *cargas* que se pretenden controlar y con base en esto se diseña el circuito correspondiente.

De igual manera que en el circuito principal se tiene que hacer el diseño de la PCB de este circuito usando el mismo método obtuve el diseño que se muestra en la figura 4.30

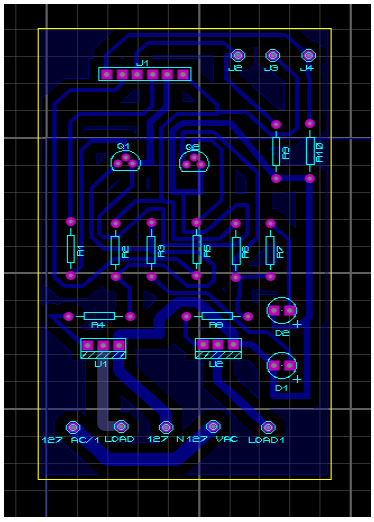

Figura 4.30 Diseño de la PCB módulo AC y figura en 3D del mismo.

Pero esta vez al ser un circuito más simple solo se diseñó sobre una cara y un tamaño de 5x10 cm. Dando un resultado final como se muestra en la siguiente figura 4.31.

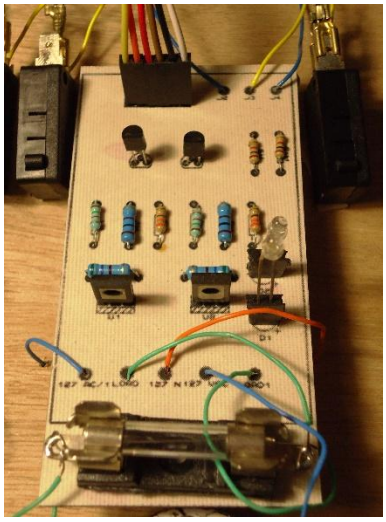

Figura 4.31 Circuito módulo AC.

En el Anexo II. Se tiene la memoria de cálculo para el disparo del Triac con el transistor.

4.4.4 Circuito DIMMER.

Para realizar este circuito se utilizó un NE555 como oscilador astable, controlando la atenuación por medio de PWM: Pulse Wide Modulation (modulación por ancho de pulso). Las componentes a utilizar son:

- U1. NE555.
- Q1, Q2, Q3 Y Q4, BC547.
- C1, Capacitor electrolítico de 0.1 µF.
- C2, Capacitor cerámico de 0.1 µF.
- Q5, IRFZ46N.
- R1, 220 KΩ.
- R2, 47 KΩ.
- R3, 10 KΩ.
- R4, 100 Ω
- R6, R7, R8 Y R9 1.5 KΩ
- J1, Conector sil-6.
- J2 y J3, pines.

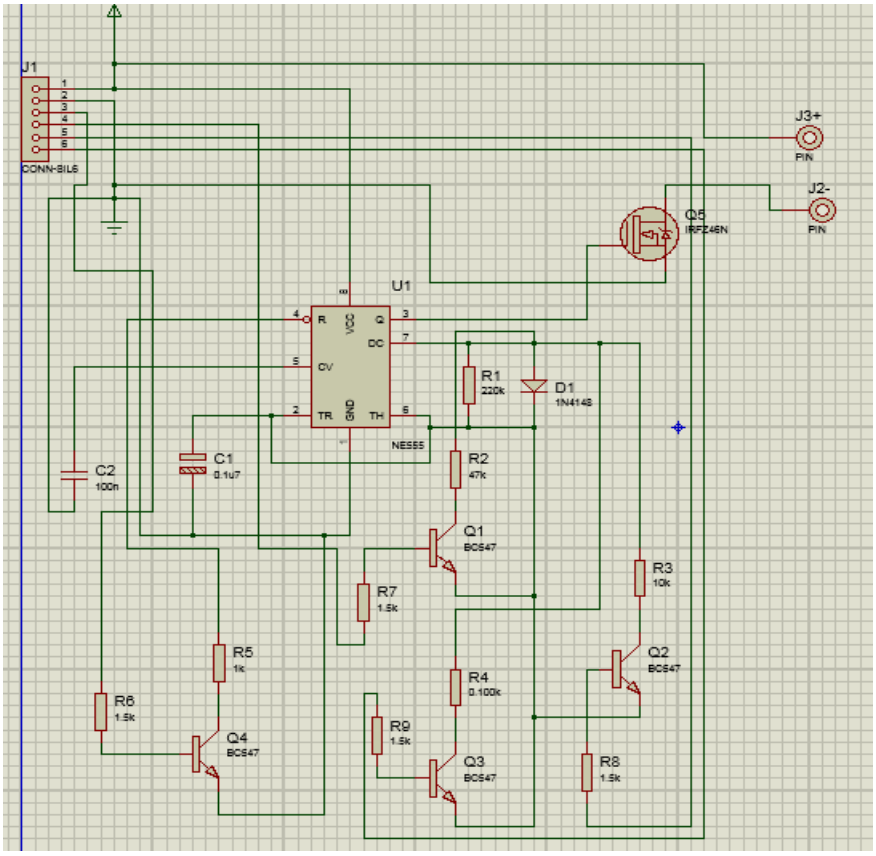

Figura 4.32 Diagrama del circuito DIMMER

Teniendo el diagrama en ISIS, podemos proceder al diseño de la PCB como en los circuitos anteriores. Obteniendo el diseño que se muestra en la figura 4.33

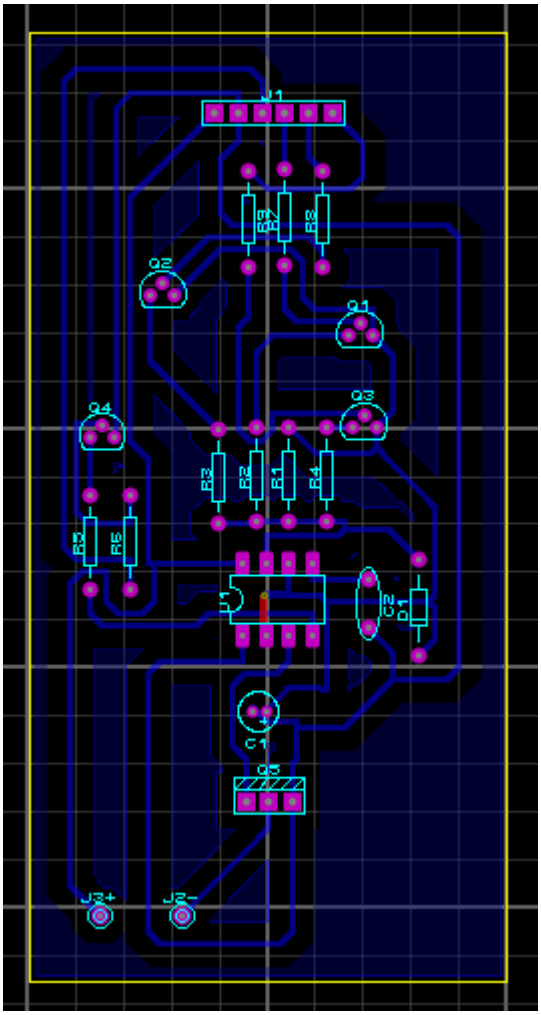

Figura 4.33 Diseño de la PCB circuito DIMMER.

Ahora, para interconectar el Circuito principal con los Módulos AC y el circuito DIMMER, utilice cables dupont hembra-hembra, para simular las luminarias y tomacorrientes, empleé focos de 7.5 W con socket E12 y la lámpara que será conectada al DIMMER es una lámpara de 13 leds.

4.4.5 Presupuesto del prototipo.

Para concluir este capítulo daré el presupuesto del prototipo, para dar una referencia en tal caso que el sistema se quiera instalar, al ser solo un prototipo aún no se suman los costos del acabado para la instalación, ni tampoco los costos de esta, en la siguiente tabla 4.2 se aprecia los componentes y sus costos.

Tabla 4.2 Costos de los componentes del prototipo.

Componente	Costo por unidad $	Unidades requeridas	Costo por componente $
PIC16F877A	70	1	70
JY-MCU	200	1	200
MOC7805CT	6	3	18
CRISTAL 4Mhz	5	1	5
Capacitor 22 pF	1	2	2
Capacitor 0.33uF	1	3	3
Capacitor 0.1 uF	1	3	3
Tira 36 pines	2	3	6
BC547	1.5	13	19.5
Resistencia 1/4	0.25	60	15
2N6071A	4	14	56
Porta fusible	5.5	14	77
fusible	2	14	28
Tira pin hembra	15	1	15
Cable dupont	2.25	46	103.5
Jack dc	5	1	5
P.fenólica 10x10	15	1	15
P.fenolica 5x10	5	8	40
Led 3MM	1	17	17
Total	--	--	628

Los beneficios que se espera obtener con la instalación de este sistema son: El ahorro de energía y aumentar la comodidad en el hogar al tener los switch a la mano para poder operarlos desde cualquier parte del apartamento.

Conclusiones:

El Bluetooth cada vez está más presente en nuestras vidas, junto con otras tecnologías inalámbricas, esto da pie para pensar que en un corto o mediano plazo ya nos olvidaremos de los cables, lo que facilita las conexiones entre todos nuestros dispositivos y haciendo más cómoda nuestra vida.

Este trabajo de tesis nos dio la oportunidad de poner en práctica todo lo aprendido en la universidad y, aunque solo se realizó un prototipo del sistema, nos dimos cuenta del trabajo que conlleva realizar un proyecto de esta envergadura, pues para diseñar el sistema requerimos de toda nuestra curiosidad y conocimientos adquiridos.

Principalmente se requirió de un planeamiento meticuloso, ya que se tuvo que idear conjuntamente todo el sistema, para que trabajara correctamente. Primero se realizó el algoritmo para el funcionamiento del PIC, una vez terminado, Proteus, gracias a su módulo VSM, me permitió simularlo y depurarlo, para realizar las primeras pruebas y enlaces con el modulo Bluetooth, realizando la primera conexión con un puerto serie Bluetooth virtual de mi computadora. Continuamos con MIT App Inventor y el diseño de la aplicación. Este, al hacer que su manejo sea gráfico, tiene algunas limitantes en cuanto a las opciones que ofrece, sin embargo, al final cumplió mis expectativas, ya que la aplicación funciona muy bien y posee una interfaz agradable.

El realizar este proyecto nos llenó de una gran satisfacción, debido a que nos demandó consultar material nuevo para el manejo de nuevas herramientas para nosotros, refiriéndome al PIC y a MIT App Inventor, lo que nos permitió adquirir nuevas capacidades. Además, nos llevó a conocer nuevas cosas, ver que hay muchas más que aún tenemos que aprender, que la carrera nos dio los cimientos, para entender y permitir adentrarnos, tanto como queramos en este mundo de conocimientos, logrando manipular los diferentes tipos de dispositivos existentes. También nos dio un manejo aceptable del software en general teniendo la capacidad de operar programas nuevos, dando relevancia a esto, ya que cada día hay nuevo software o mejoras en los antiguos y debemos adaptarnos, porque los encontramos en todos los ámbitos de nuestra vida, como por ejemplo, en la industria. Gracias a esta capacidad que desarrollamos, no se nos dificulta y al contrario nos encontramos a la expectativa de los nuevos programas, tratando de aprender más sobre esta materia y sobre todo lo que nos es de interés.

Al final se logró diseñar un circuito, en conjunto con una aplicación capaz de controlar luminarias y tomacorrientes simulados por lámparas. En general los objetivos se cumplieron, ahora solo falta acoplarlo a un espacio de la vida cotidiana y reafirmar su utilidad así como su correcto funcionamiento.

Anexo I. Programa PIC.

```
;*****************************************************************************************************
; PROGRAMA: Sistema de control de luminarias y toma corrientes para apartamento igual o menor a 100 m2
; Función : Encender o apagar luces y energetizar o desenergetizar toma corrientes para apartamento
;
;       Serial port config: 9600 baudios,8 bits,paridad = none
;
;  Revision : 1.2       07/12/2013          Programa para : PIC16F877a
; CPU Clock : 4 MHz                    Reloj instruccion : 4 MHz = 0.250 µs
;      WDT : Deshabilitado               Tipo de reloj : XT.
;


; Code Prot : OFF                      velocidad usart: 9600 baudios (Kbps)
;*****************************************************************************************************

;*****************************************************************************************************
**********************
        LIST   P=16F877a        ;Se indica el modo de procesador
        INCLUDE   <P16f877a.inc>  ;se incluye la definicion de los registros internos
        __CONFIG      _CP_OFF & _PWRTE_ON & _WDT_OFF & _XT_OSC & _BODEN_OFF &_LVP_OFF &_CPD_OFF &_WRT_OFF
        ;Configuracion de los registros de operacion

        CBLOCK 0X20 ;Definicion de los registros para el monitoreo de las salidas
        DATO
        REGA
        REGB
        REGC
        REGD
        REGE
        REGF
        REGG
        REGH
        REGI
        REGJ
        REGK
        REGL
        REGM
        REGD0
        REGD1
        REGD2
        REGD3
        ENDC

        ORG 0X00 ;declaro el origen del programa en la dirección de reset
        GOTO INICIO ;voy al inicio real del programa
        ORG 0X04 ;declaro el vector de interrupción
;*****************************CODIGO                                                    DE INTERRUPCION*****************************************************************************************
        MOVF RCREG,W
        MOVWF DATO
        BCF PIR1,RCIF
```

```
        RETFIE

;************Configuracion de puertos ********************************************************************
INICIO
        BANKSEL TRISA
        MOVLW B'00000000'
        MOVWF TRISA
        MOVLW B'00000110'
        MOVWF ADCON1
        MOVLW B'00000000'
        MOVWF TRISE  ;Puertos de entrada
        MOVLW B'00000000'
        MOVWF TRISB
        MOVLW B'00000000'
        MOVWF TRISD  ;Puertos de salida


;**********CONFIGURAR PUERTOS PARA LA ENTRADA DE DATOS DE LA USART Y E/S**********
        BANKSEL TRISC
        BSF TRISC,7 ;Configuro RC7/Rx como entrada
        BCF TRISC,6 ;Configuro RC6/Tx como salida
        BSF TRISC,0
        BSF TRISC,1
        BSF TRISC,2
        BSF TRISC,3
        BCF TRISC,4
        BCF TRISC,5

        BCF STATUS,RP0 ; Cambio a banco 0
        CALL USARTINI ;Llamo a subrutina para configurar la usart

;**************Programa principal*************************************************************************

      BANKSEL PORTD
      MOVLW B'11111111'
      MOVWF PORTD
      MOVWF PORTB
      BCF PORTC,4
      BSF PORTC,5
      BCF PORTD,2
      BCF PORTD,3


      MOVLW B'11111111'
      MOVWF REGA
      MOVWF REGB
      MOVWF REGC
      MOVWF REGD
      MOVWF REGE
      MOVWF REGF
      MOVWF REGG
      MOVWF REGH
      MOVWF REGI
      MOVWF REGJ
      MOVWF REGK
      MOVWF REGL
```

```
	MOVWF REGM
START
	BANKSEL PORTA



	MOVF DATO,W
	SUBLW "a"
	BTFSS STATUS,Z
	GOTO Sb
	GOTO AB
Sb
	MOVF DATO,W
	SUBLW "b"
	BTFSS STATUS,Z
	GOTO Sc
	GOTO CD
Sc
	MOVF DATO,W
	SUBLW "c"
	BTFSS STATUS,Z
	GOTO Sd
	GOTO EF
Sd
	MOVF DATO,W
	SUBLW "d"
	BTFSS STATUS,Z
	GOTO Se
	GOTO GH
Se
	MOVF DATO,W
	SUBLW "e"
	BTFSS STATUS,Z
	GOTO Sf
	GOTO IJ
Sf
	MOVF DATO,W
	SUBLW "f"
	BTFSS STATUS,Z
	GOTO Sg
	GOTO KL
Sg
	MOVF DATO,W
	SUBLW "g"
	BTFSS STATUS,Z
	GOTO BA
	GOTO MN

BA
	BTFSC PORTA,0
	GOTO ONAE

	MOVF DATO,W     ;Coloco el dato recibido el en registro
	SUBLW "A"       ;REG Dato y lo comparo con una
	BTFSS STATUS,Z  ;Literal
	GOTO BE
```

```
      GOTO ONAS
BE
      BTFSC PORTA,1
      GOTO ONBE

      MOVF DATO,W
      SUBLW "B"
      BTFSS STATUS,Z
      GOTO CE
      GOTO ONBS
CE
      BTFSC PORTA,2
      GOTO ONCE

      MOVF DATO,W
      SUBLW "C"
      BTFSS STATUS,Z
      GOTO DEE
      GOTO ONCS
DEE
      BTFSC PORTA,3
      GOTO ONDE

      MOVF DATO,W
      SUBLW "D"
      BTFSS STATUS,Z
      GOTO EE
      GOTO ONDS
EE
      BTFSC PORTA,4
      GOTO ONEE

      MOVF DATO,W
      SUBLW "E"
      BTFSS STATUS,Z
      GOTO FE
      GOTO ONES
FE
      BTFSC PORTA,5
      GOTO ONFE

      MOVF DATO,W
      SUBLW "F"
      BTFSS STATUS,Z
      GOTO GE
      GOTO ONFS
GE
      BTFSC PORTE,0
      GOTO ONGE

      MOVF DATO,W
      SUBLW "G"
      BTFSS STATUS,Z
      GOTO HE
      GOTO ONGS
HE
```

```
      BTFSC PORTE,1
      GOTO ONHE

      MOVF DATO,W
      SUBLW "H"
      BTFSS STATUS,Z
      GOTO IE
      GOTO ONHS
IE
      BTFSC PORTE,2
      GOTO ONIE

      MOVF DATO,W
      SUBLW "I"
      BTFSS STATUS,Z
      GOTO JE
      GOTO ONIS
JE
      BTFSC PORTC,0
      GOTO ONJE

      MOVF DATO,W
      SUBLW "J"
      BTFSS STATUS,Z
      GOTO KE
      GOTO ONJS
KE
      BTFSC PORTC,1
      GOTO ONKE

      MOVF DATO,W
      SUBLW "K"
      BTFSS STATUS,Z
      GOTO LE
      GOTO ONKS
LE
      BTFSC PORTC,2
      GOTO ONLE

      MOVF DATO,W
      SUBLW "L"
      BTFSS STATUS,Z
      GOTO ME
      GOTO ONLS
ME
      BTFSC PORTC,3
      GOTO ONME

      MOVF DATO,W
      SUBLW "M"
      BTFSS STATUS,Z
      GOTO DV
      GOTO ONMS
;*******¡NSTRUCCIONES PARA EL CONTROL DEL DIMMER*********************************
DV
```

```
        MOVF DATO,W
        SUBLW "V"
        BTFSS STATUS,Z
        GOTO DWW
        MOVLW "V"
        MOVWF TXREG
        BCF PORTC,5
        BCF PORTC,4
        BCF PORTD,3
        BCF PORTD,2
        CLRF DATO
        GOTO START
DWW
        MOVF DATO,W
        SUBLW "W"
        BTFSS STATUS,Z
        GOTO DX
        BSF PORTC,4
        BCF PORTC,5
        BCF PORTD,3
        BCF PORTD,2
        CLRF DATO
        GOTO START
DX
        MOVF DATO,W
        SUBLW "X"
        BTFSS STATUS,Z
        GOTO DY
        BSF PORTD,3
        BCF PORTC,4
        BCF PORTC,5
        BCF PORTD,2
        CLRF DATO
        GOTO START
DY
        MOVF DATO,W
        SUBLW "Y"
        BTFSS STATUS,Z
        GOTO DZ
        BSF PORTD,2
        BCF PORTC,4
        BCF PORTD,3
        BCF PORTC,5
        CLRF DATO
        GOTO START
DZ
        MOVF DATO,W
        SUBLW "Z"
        BTFSS STATUS,Z
        GOTO START
        BSF PORTC,5
        BCF PORTC,4
        BCF PORTD,3
        BCF PORTD,2
        CLRF DATO
        GOTO START
```

```
;******CHEQUEO DE LOS INTERRUPTORES****************************************

AB
      BTFSC REGA,0
      GOTO BB
      MOVLW "a"
      MOVWF TXREG
      CLRF DATO
      CALL RETARDO1
BB
      BTFSC REGB,0
      GOTO  AB1
      MOVLW "b"
      MOVWF TXREG
      CLRF DATO
      CALL RETARDO1
AB1
      BTFSS REGA,0
      GOTO BB1
      MOVLW "A"
      MOVWF TXREG
      CLRF DATO
      CALL RETARDO1
BB1
      BTFSS REGB,0
      GOTO START
      MOVLW "B"
      MOVWF TXREG
      CLRF DATO
      GOTO START
CD
      BTFSC REGC,0
      GOTO DD
      MOVLW "c"
      MOVWF TXREG
      CLRF DATO
      CALL RETARDO1
DD
      BTFSC REGD,0
      GOTO  CD1
      MOVLW "d"
      MOVWF TXREG
      CLRF DATO
      CALL RETARDO1
CD1
      BTFSS REGC,0
      GOTO DD1
      MOVLW "C"
      MOVWF TXREG
      CLRF DATO
      CALL RETARDO1
DD1
      BTFSS REGD,0
      GOTO START
      MOVLW "D"
      MOVWF TXREG
```

```
        CLRF DATO
        GOTO START
EF
        BTFSC REGE,0
        GOTO FF
        MOVLW "e"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
FF
        BTFSC REGF,0
        GOTO  EF1
        MOVLW "f"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
EF1
        BTFSS REGE,0
        GOTO FF1
        MOVLW "E"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
FF1
        BTFSS REGF,0
        GOTO START
        MOVLW "F"
        MOVWF TXREG
        CLRF DATO
        GOTO START
GH
        BTFSC REGG,0
        GOTO HH
        MOVLW "g"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
HH
        BTFSC REGH,0
        GOTO  GH1
        MOVLW "h"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
GH1
        BTFSS REGG,0
        GOTO HH1
        MOVLW "G"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
HH1
        BTFSS REGH,0
        GOTO START
        MOVLW "H"
        MOVWF TXREG
```

```
        CLRF DATO
        GOTO START
IJ
        BTFSC REGI,0
        GOTO JJ
        MOVLW "i"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
JJ
        BTFSC REGJ,0
        GOTO  IJ1
        MOVLW "j"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
IJ1
        BTFSS REGI,0
        GOTO JJ1
        MOVLW "I"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
JJ1
        BTFSS REGJ,0
        GOTO START
        MOVLW "J"
        MOVWF TXREG
        CLRF DATO
        GOTO START
KL
        BTFSC REGK,0
        GOTO LL
        MOVLW "k"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
LL
        BTFSC REGL,0
        GOTO  KL1
        MOVLW "l"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
KL1
        BTFSS REGK,0
        GOTO LL1
        MOVLW "K"
        MOVWF TXREG
        CLRF DATO
        CALL RETARDO1
LL1
        BTFSS REGL,0
        GOTO START
        MOVLW "L"
        MOVWF TXREG
```

```
        CLRF DATO

        GOTO START
MN
        BTFSC REGM,0
        GOTO MM
        MOVLW "m"
        MOVWF TXREG
        CLRF DATO

        CALL RETARDO1
MM
        BTFSS REGB,0
        GOTO  START
        MOVLW "M"
        MOVWF TXREG
        CLRF DATO

        GOTO START
;*****************************************************************************************************************
ONAE
        BTFSC REGA,0
        GOTO OFFAE
        BSF PORTD,5
        MOVLW B'11111111'
        MOVWF REGA

        CALL RETARDO
        MOVLW 'A'
        MOVWF TXREG
        GOTO START
OFFAE
        BCF PORTD,5
        MOVLW B'00000000'
        MOVWF REGA
        CALL RETARDO
        MOVLW "a"
        MOVWF TXREG
        GOTO START

ONBE
        BTFSC REGB,0
        GOTO OFFBE
        BSF PORTD,6
        MOVLW B'11111111'
        MOVWF REGB
        CALL RETARDO
        MOVLW 'B'
        MOVWF TXREG
        GOTO START
OFFBE
        BCF PORTD,6
        MOVLW B'00000000'
        MOVWF REGB
        CALL RETARDO
        MOVLW "b"
```

```
        MOVWF TXREG
        GOTO START

ONCE
        BTFSC REGC,0
        GOTO OFFCE
        BSF PORTD,7
        MOVLW B'11111111'
        MOVWF REGC
        CALL RETARDO
        MOVLW 'C'
        MOVWF TXREG
        GOTO START
OFFCE
        BCF PORTD,7
        MOVLW B'00000000'
        MOVWF REGC
        CALL RETARDO
        MOVLW "c"
        MOVWF TXREG
        GOTO START

ONDE
        BTFSC REGD,0
        GOTO OFFDE
        BSF PORTB,0
        MOVLW B'11111111'
        MOVWF REGD
        CALL RETARDO
        MOVLW 'D'
        MOVWF TXREG
        GOTO START
OFFDE
        BCF PORTB,0
        MOVLW B'00000000'
        MOVWF REGD
        CALL RETARDO
        MOVLW "d"
        MOVWF TXREG
        GOTO START

ONEE
        BTFSC REGE,0
        GOTO OFFEE
        BSF PORTB,1
        MOVLW B'11111111'
        MOVWF REGE
        CALL RETARDO
        MOVLW 'E'
        MOVWF TXREG
        GOTO START
OFFEE
        BCF PORTB,1
        MOVLW B'00000000'
        MOVWF REGE
        CALL RETARDO
```

```
        MOVLW "e"
        MOVWF TXREG
        GOTO START

ONFE
        BTFSC REGF,0
        GOTO OFFFE
        BSF PORTB,2
        MOVLW B'11111111'
        MOVWF REGF
        CALL RETARDO
        MOVLW 'F'
        MOVWF TXREG
        GOTO START
OFFFE
        BCF PORTB,2
        MOVLW B'00000000'
        MOVWF REGF
        CALL RETARDO
        MOVLW "f"
        MOVWF TXREG
        GOTO START

ONGE
        BTFSC REGG,0
        GOTO OFFGE
        BSF PORTB,3
        MOVLW B'11111111'
        MOVWF REGG
        CALL RETARDO
        MOVLW 'G'
        MOVWF TXREG
        GOTO START
OFFGE
        BCF PORTB,3
        MOVLW B'00000000'
        MOVWF REGG
        CALL RETARDO
        MOVLW "g"
        MOVWF TXREG
        GOTO START

ONHE
        BTFSC REGH,0
        GOTO OFFHE
        BSF PORTB,4
        MOVLW B'11111111'
        MOVWF REGH
        CALL RETARDO
        MOVLW 'H'
        MOVWF TXREG
        GOTO START
OFFHE
        BCF PORTB,4
        MOVLW B'00000000'
        MOVWF REGH
```

```
        CALL RETARDO
        MOVLW "h"
        MOVWF TXREG
        GOTO START

ONIE
        BTFSC REGI,0
        GOTO OFFIE
        BSF PORTB,5
        MOVLW B'11111111'
        MOVWF REGI
        CALL RETARDO
        MOVLW 'I'
        MOVWF TXREG
        GOTO START
OFFIE
        BCF PORTB,5
        MOVLW B'00000000'
        MOVWF REGI
        CALL RETARDO
        MOVLW "i"
        MOVWF TXREG
        GOTO START

ONJE
        BTFSC REGJ,0
        GOTO OFFJE
        BSF PORTB,6
        MOVLW B'11111111'
        MOVWF REGJ
        CALL RETARDO
        MOVLW 'J'
        MOVWF TXREG
        GOTO START
OFFJE
        BCF PORTB,6
        MOVLW B'00000000'
        MOVWF REGJ
        CALL RETARDO
        MOVLW "j"
        MOVWF TXREG
        GOTO START

ONKE
        BTFSC REGK,0
        GOTO OFFKE
        BSF PORTB,7
        MOVLW B'11111111'
        MOVWF REGK
        CALL RETARDO
        MOVLW 'K'
        MOVWF TXREG
        GOTO START
OFFKE
        BCF PORTB,7
        MOVLW B'00000000'
```

```
        MOVWF REGK
        CALL RETARDO
        MOVLW "k"
        MOVWF TXREG
        GOTO START

ONLE
        BTFSC REGL,0
        GOTO OFFLE
        BSF PORTD,1
        MOVLW B'11111111'
        MOVWF REGL
        CALL RETARDO
        MOVLW 'L'
        MOVWF TXREG
        GOTO START
OFFLE
        BCF PORTD,1
        MOVLW B'00000000'
        MOVWF REGL
        CALL RETARDO
        MOVLW "l"
        MOVWF TXREG
        GOTO START

ONME
        BTFSC REGM,0
        GOTO OFFME
        BSF PORTD,0
        MOVLW B'11111111'
        MOVWF REGM
        CALL RETARDO
        MOVLW 'M'
        MOVWF TXREG
        GOTO START
OFFME
        BCF PORTD,0
        MOVLW B'00000000'
        MOVWF REGM
        CALL RETARDO
        MOVLW "m"
        MOVWF TXREG
        GOTO START
;*****************************************************************************************************************
;
ONAS
       BTFSC REGA,0
       GOTO OFFAS
       BSF PORTD,5
       MOVLW B'11111111'
       MOVWF REGA
       MOVLW 'A'
       MOVWF TXREG
       CLRF DATO
       GOTO START
OFFAS
       BCF PORTD,5
```

```
    MOVLW B'00000000'
    MOVWF REGA
    MOVLW "a"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONBS
    BTFSC REGB,0
    GOTO OFFBS
    BSF PORTD,6
    MOVLW B'11111111'
    MOVWF REGB
    MOVLW 'B'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFBS
    BCF PORTD,6
    MOVLW B'00000000'
    MOVWF REGB
    MOVLW "b"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONCS
    BTFSC REGC,0
    GOTO OFFCS
    BSF PORTD,7
    MOVLW B'11111111'
    MOVWF REGC
    MOVLW 'C'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFCS
    BCF PORTD,7
    MOVLW B'00000000'
    MOVWF REGC
    MOVLW "c"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONDS
    BTFSC REGD,0
    GOTO OFFDS
    BSF PORTB,0
    MOVLW B'11111111'
    MOVWF REGD
    MOVLW 'D'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFDS
```

```
    BCF PORTB,0
    MOVLW B'00000000'
    MOVWF REGD
    MOVLW "d"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONES
    BTFSC REGE,0
    GOTO OFFES
    BSF PORTB,1
    MOVLW B'11111111'
    MOVWF REGE
    MOVLW 'E'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFES
    BCF PORTB,1
    MOVLW B'00000000'
    MOVWF REGE
    MOVLW "e"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONFS
    BTFSC REGF,0
    GOTO OFFFS
    BSF PORTB,2
    MOVLW B'11111111'
    MOVWF REGF
    MOVLW 'F'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFFS
    BCF PORTB,2
    MOVLW B'00000000'
    MOVWF REGF
    MOVLW "f"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONGS
    BTFSC REGG,0
    GOTO OFFGS
    BSF PORTB,3
    MOVLW B'11111111'
    MOVWF REGG
    MOVLW 'G'
    MOVWF TXREG
    CLRF DATO
    GOTO START
```

```
OFFGS
    BCF PORTB,3
    MOVLW B'00000000'
    MOVWF REGG
    MOVLW "g"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONHS
    BTFSC REGH,0
    GOTO OFFHS
    BSF PORTB,4
    MOVLW B'11111111'
    MOVWF REGH
    MOVLW 'H'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFHS
    BCF PORTB,4
    MOVLW B'00000000'
    MOVWF REGH
    MOVLW "h"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONIS
    BTFSC REGI,0
    GOTO OFFIS
    BSF PORTB,5
    MOVLW B'11111111'
    MOVWF REGI
    MOVLW 'I'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFIS
    BCF PORTB,5
    MOVLW B'00000000'
    MOVWF REGI
    MOVLW "i"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONJS
    BTFSC REGJ,0
    GOTO OFFJS
    BSF PORTB,6
    MOVLW B'11111111'
    MOVWF REGJ
    MOVLW 'J'
    MOVWF TXREG
    CLRF DATO
```

```
    GOTO START
OFFJS
    BCF PORTB,6
    MOVLW B'00000000'
    MOVWF REGJ
    MOVLW "j"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONKS
    BTFSC REGK,0
    GOTO OFFKS
    BSF PORTB,7
    MOVLW B'11111111'
    MOVWF REGK
    MOVLW 'K'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFKS
    BCF PORTB,7
    MOVLW B'00000000'
    MOVWF REGK
    MOVLW "k"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONLS
    BTFSC REGL,0
    GOTO OFFLS
    BSF PORTD,1
    MOVLW B'11111111'
    MOVWF REGL
    MOVLW 'L'
    MOVWF TXREG
    CLRF DATO
    GOTO START
OFFLS
    BCF PORTD,1
    MOVLW B'00000000'
    MOVWF REGL
    MOVLW "l"
    MOVWF TXREG
    CLRF DATO
    GOTO START

ONMS
    BTFSC REGM,0
    GOTO OFFMS
    BSF PORTD,0
    MOVLW B'11111111'
    MOVWF REGM
    MOVLW 'M'
    MOVWF TXREG
```

```
     CLRF DATO
     GOTO START
OFFMS
     BCF PORTD,0
     MOVLW B'00000000'
     MOVWF REGM
     MOVLW "m"
     MOVWF TXREG
     CLRF DATO
     GOTO START
;*****************************************************************************************************************
**********************
RETARDO
          cblock
          d1
          d2
          d3
          endc

                              ;999997 cycles
          movlw  0x08
          movwf  d1
          movlw  0x2F
          movwf  d2
          movlw  0x03
          movwf  d3
Delay_0
          decfsz  d1, f
          goto    $+2
          decfsz  d2, f
          goto    $+2
          decfsz  d3, f
          goto    Delay_0

                              ;3 cycles
          goto    $+1
          nop
;**********RETARDO1*************************************************************************************

RETARDO1
          cblock
          F1
          F2
          F3
          endc

                              ;499994 cycles
          movlw  0x03
          movwf  F1
          movlw  0x18
          movwf  F2
          movlw  0x02
          movwf  F3
DELAY1_0
          decfsz  F1, f
          goto    $+2
```

```
        decfsz  F2, f
        goto    $+2
        decfsz  F3, f
        goto    DELAY1_0

                        ;6 cycles
        goto    $+1
        goto    $+1
        goto    $+1


;**********Rutina                                                                                                                de
usart*************************************************************************************************************************
*
USARTINI    BSF STATUS,RP0
        MOVLW 0X24
        MOVWF TXSTA
        MOVLW D'25'
        MOVWF SPBRG
        BSF PIE1,RCIE
        MOVLW 0XC0
        MOVWF INTCON
        BCF STATUS,RP0 ;BANCO 0
        MOVLW 0X90
        MOVWF RCSTA
        RETURN



;************FIN DEL PROGRAMA*****************************************************************************
   END
```

Anexo II. Memoria de cálculos.

Para el Disparo del Triac se utilizó el circuito que se muestra en la figura A1.

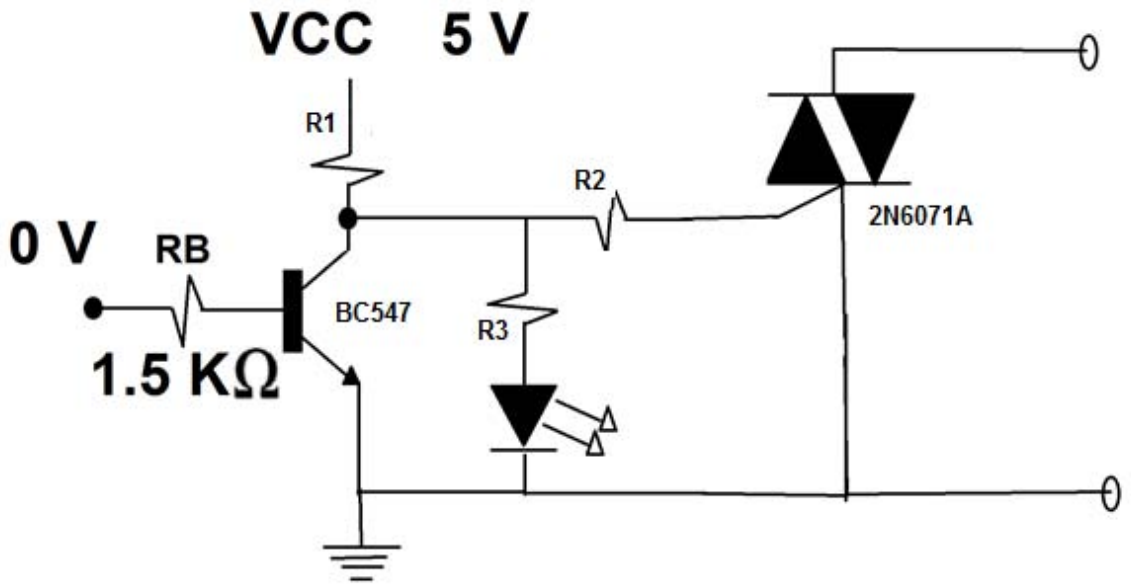

Figura A1 Diagrama del circuito de disparo.

El Triac utilizado fue el 2N6071A Sensitive Gate. Una de sus principales caracteristicas es que permite un acoplamiento directo a TTL como se muestra en la figura A2. El voltaje que manejan dichos dispositivos para un estado alto es de 2.5 a 3.5 V, tomando esto en consideración para el disparo, debe cumplir la siguiente condición.

$$5\,V < Voltaje\ de\ disparo < 3.5\,V$$

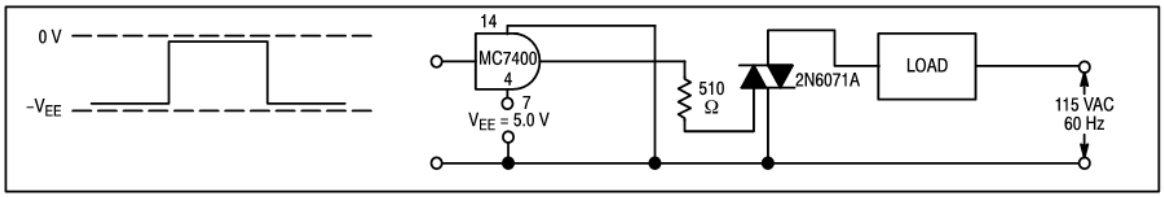

Figura A2 Diagrama recomendado por el fabricante para el disparo del triac 2N6071A.
Diagrama tomado de la datasheet del dispositivo 2N6071A.

Para continuar con el analisis se realizo un diagrama simplificado del circuito de disparo y se muestra en la figura A3.

Donde:

- R_T corresponde a la resistencia total.
- V_{D1} es el voltaje de operación del Led.
- V_{Triac} se refiere al voltaje de operación del Triac.

- Vce_{SAT} indica el voltaje colector emisor de saturación en el transistor.

Los valores fueron tomados de la ficha tecnica de los dispositivos utilizados, excepto RT cuyo valor fue calculado:

$$\boldsymbol{RT = R1 + R2\ //R3 = 100\Omega + 207\ \Omega = 307\Omega}$$

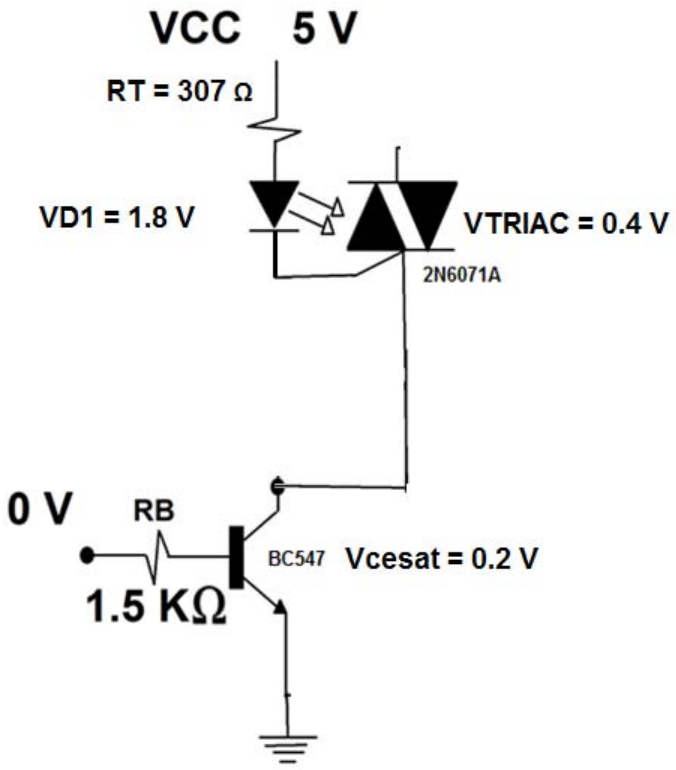

Figura A3 diagrama simplificado para el calculo de IT.

Por LKV tenemos:

$$\boldsymbol{-5\,V + (RT)(IT) + VD1 + VTRIAC + Vcesat = 0}$$

Despejando a IT:

$$\boldsymbol{IT = \frac{5V - VD1 - VTRIAC - Vcesat}{RT} = \frac{5V - 1.8V - 0.4V - 0.2V}{307\Omega} = 8.47\,mA}$$

Con la corriente total podemos calcular la caida de voltaje en R1 con la ley de ohm, para asegurar el disparo del triac.

$$\boldsymbol{V_{R1} = (R1)(IT) = (100\Omega)(8.47mA) = 0.847\,V}$$

Sabemos que la caida de voltaje en R1=0.847 V + Vcesat = 0.2 V = 1.047 V por lo tanto:

$$Vcc - V_{R1} + V_{CESAT} = 5V - 1.047 = 3.95V$$

Cumpliendo la condición antes mencionada:

$$5V < 3.95V < 3.5$$

La siguiente figura A4 muestra la simulación del circuito en Proteus. Donde el voltaje en la entrada de R2 = 3.93 V ≈ 3.95 V

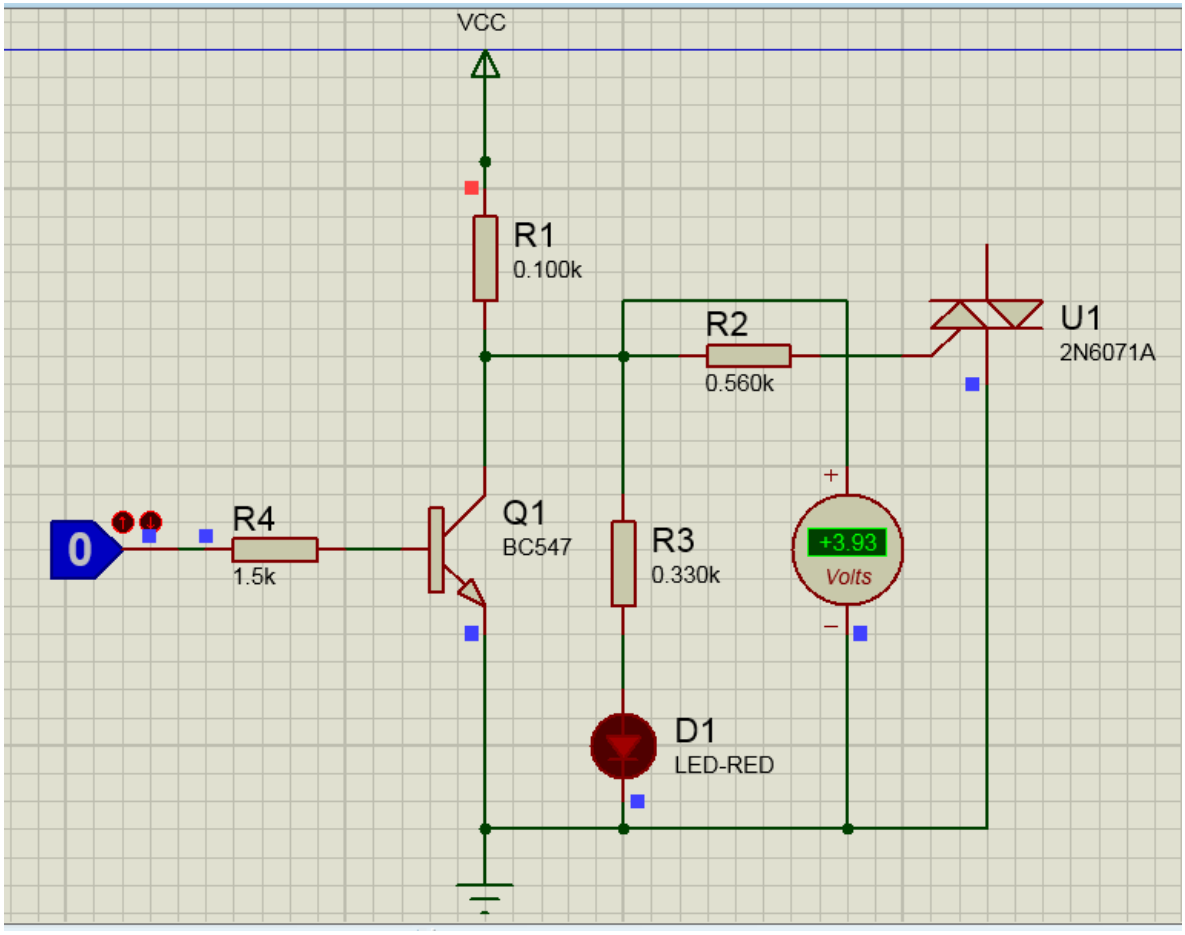

Figura A4 diagrama del circuito de disparo simulado en Proteus.

Bibliografía:

Bateman A (2003). ***Comunicaciones digitales: diseño para el mundo real.*** España: Marcombo

Catsoulis John (2005). ***Designing Embedded Hardware.*** E.U.A: O'Reilly Media

Gallager G, Robert (2008). ***Principles of Digital Communication***. New York: Cambridge University Press

Ilyas Mohammad, Ahson Syed A. (2006). ***Smartphones (Research Report series).*** Chicago, Ilinois E.U.A: ICE, Plubications

Mandado Pérez Enrique, Menéndez Fuertes Luis Manuel, Fernández Ferreira Luis López Matos Emilio (2007). ***Microcontroladores PIC: sistema integrado para el autoaprendizaje.*** España: Marcombo

Moreno Parra Rafael Alberto (2013). ***Desarrollo fácil y paso a paso de aplicaciones para Android usando MIT App Inventor 2013.*** Colombia: Autoedición

Prabhu C.S.R, Reddi Prathap A. (2004). ***Bluetooth Technology and Its Applications with JAVA and J2ME.*** Nueva Delhi: Prentice-Hall of India

Predko Myke (2008). ***Programming and Customizing the PIC Microcontroller.*** Toronto: McGraw-Hill

Proakis G John, Salehi Masoud (2007). ***Digital Communication 5th edition***. Boston: McGraw-Hill

Sanchís Enrique (2002). ***Sistemas electrónicos digitales: Fundamentos y diseño de aplicaciones.*** España: UNIVERSITAT DE VALENCIA. SERVEI DE PUBLICACIONS

Wolber David, Abelson Hal, Spertus Ellen, Looney Liz (2011). ***App Inventor Book: Classic version (PDF)***. California: O´Reilly Media

WEBGRAFIA:

http://beta.appinventor.mit.edu
Pagina principal de MIT app inventor.
http://beta.appinventor.mit.edu/

http://bluehack.elhacker.net
El grupo de seguridad del bluetooth español.
http://bluehack.elhacker.net/proyectos/bluesec/bluesec.html

http://source.android.com
El código Fuente de Android.
http://source.android.com/source/index.html

http://www.bluetooth.com
Página principal de Bluetooth. Com.
http://www.bluetooth.com/Pages/Bluetooth-Home.aspx

http://www.datasheetcatalog.net
Fuente libre de fichas técnicas de componentes electrónicos
http://www.datasheetcatalog.net/es/

www.info-ab.uclm.es
Modos de direccionamiento
http://www.info-ab.uclm.es/labelec/solar/Microcontroladores/Modos_Direccionamiento.htm

http://www.neoteo.com
Mario Sacco. Modulo Bluetooth hc-06 Android.
http://www.neoteo.com/modulo-bluetooth-hc-06-android/

http://www.neoteo.com
Mario Sacco. Comandos AT modulo Bluetooth hc 06.
http://www.neoteo.com/comandos-at-modulo-bluetooth-hc-06/

http://www.neoteo.com
Mario Sacco. Ne555.
http://www.neoteo.com/ne555/

http://www.microchip.com
Página principal de Microchip.
http://www.microchip.com/developmenttools/

http://www.smart-gsm.com
SmartGSM
http://www.smart-gsm.com/moviles/htc-dream

http://www.wayerless.com
La historia del nacimiento del Bluetooth.
http://www.wayerless.com/2011/09/la-historia-del-nacimiento-de-bluetooth/