SISTEMA DE MONITOREO Y CONTROL DE VARIABLES REMOTAS A TRAVÉS DE INTERNET.

DIEGO HERNÁN MEJÍA MELO
GUSTAVO LÓPEZ SANABRIA

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA, FÍSICA Y SISTEMAS Y COMPUTACIÓN
PROGRAMA DE INGENIERÍA ELECTRÓNICA
PEREIRA
2011

SISTEMA DE MONITOREO Y CONTROL DE VARIABLES REMOTAS A TRAVÉS DE INTERNET.

DIEGO HERNÁN MEJÍA MELO
GUSTAVO LÓPEZ SANABRIA

Proyecto de grado

Profesor José Alfredo Jaramillo

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA, FÍSICA Y SISTEMAS Y COMPUTACIÓN
PROGRAMA DE INGENIERÍA ELECTRÓNICA
PEREIRA
2011

CONTENIDO

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

LISTA DE ANEXOS

pág.

RESUMEN

Este proyecto se realiza como requisito para la obtención de grado de Ingeniero Electrónico en la Universidad Tecnológica De Pereira, y mediante el cual se realiza el desarrollo de un Sistema De Monitoreo Y Control De Variables Remotas A Traves De Internet.

En este documento se presentan los conceptos básicos relacionados en el ámbito de los protocolos de comunicaciones aplicativos para el sector hogar y oficina necesarios para el diseño del sistema, en el desarrollo de este proyecto se consideran variables críticas a tener en cuenta para el diseño de circuitos electrónicos y se definen los parámetros necesarios para la selección de un controlador general para el dispositivo.

La concepción de la idea surge como un dispositivo que permita a las personas tener acceso remoto a lugares sobre los cuales deseen realizar monitoreo y control de variables de modo tal que la distancia deje de ser una limitante a través de una herramienta de comunicación y acceso a la información por excelencia como el internet.

En la última etapa y luego de realizar completamente el proceso de diseño y simulación el proyecto se materializa mediante la fabricación y posterior ensamble de un prototipo que valida el correcto funcionamiento del sistema y su aplicabilidad para la solución que fue pensado.

INTRODUCCIÓN

El presente proyecto se centra en el área de la Domótica, motivo por el cual es importante la definición de este concepto. La Domótica tiene origen en la palabra latina "domus" y telemática, haciendo referencia a la incorporación de elementos tecnológicos al hogar. Esta introducción de las nuevas tecnologías en el hogar fue dando lugar a nuevos sistemas de información conocidos como sistemas domóticos, cuyo fin es el control de los recursos presentes en las casas o edificios de forma remota y automatizada, de tal forma que podemos definirla como la tecnología de la automatización aplicada al control de las casas y edificios.

Durante varios años las aplicaciones domoticas se han visto relegadas a sectores de la sociedad muy exclusivos, en los cuales los fabricantes y comercializadores no han encontrado un nicho de mercado que les permita garantizarse unos volúmenes atractivos para el negocio debido a los altos costos de los dispositivos y su complicada y también costosa instalación, pero actualmente y como resultado del abaratamiento de las materias primas y el fácil acceso a ellas producto de la globalización se pueden diseñar y fabricar diferentes aplicaciones en el área de la domotica con precios mucho mas atractivos y asequibles a los usuarios promedio, los cuales son los que convierten o no en rentable la comercialización de muchos productos en especial en el área de la electrónica y la tecnología.

Recientemente y debido a las tecnologías verdes y la concientización de los seres humanos como actores principales y determinantes en el futuro de nuestro planeta han tomado fuerza los conceptos de aprovechamiento eficiente de la energía y el ahorro de la misma, estos factores sumados a la necesidad constante de ideas novedosas y prácticas que le proporcionen confort a los usuarios hacen que la domotica sea un campo de trabajo atractivo para el desarrollo de nuevos productos y la incursión de estos en el mercado nacional e internacional.

1 FORMULACIÓN DEL PROBLEMA

El agitado ritmo de vida actual hace que en muchas ocasiones se necesite tener acceso remoto a lugares en los cuales se desea conocer el estado de algunas variables tales como la temperatura, la presencia o no de personas, sensores asociados a puertas y ventanas, etc. De igual manera existe la necesidad de realizar ajustes con base en dicha información, activar o desactivar dispositivos, acudir de inmediato al lugar, realizar llamadas de emergencia, todas estas medidas encaminadas a mejorar las condiciones de seguridad y confort de dicho lugar y de las personas y objetos que se encuentren allí. Es por esto que existe la necesidad de un dispositivo que brinde al usuario la posibilidad de acceder a estas prestaciones de tal forma que no exista la necesidad de estar físicamente en el sitio.

2 JUSTIFICACIÓN

El tiempo es un factor altamente valioso sobre todo en épocas modernas, lo que hace ineficiente el hecho de que las personas tengan que desplazarse largas distancias para realizar tareas sencillas como encender una luz, un equipo, abrir una puerta o enterarse de los eventos ocurridos en un lugar determinado. Es por esto que es de gran ayuda para los usuarios poder tener acceso remoto a otros lugares para realizar estas acciones, lo cual se ve reflejado en un aumento sustancial de la seguridad, el confort, el ahorro de tiempo y energía, entre otros factores. Dado que Internet es el medio de comunicación que se encuentra más a la vanguardia de las tecnologías de la información, se convierte también en el medio de información más eficiente para las aplicaciones en el campo de la Domótica.

3 OBJETIVOS

3.1 OBJETIVO GENERAL

Diseñar y Fabricar un dispositivo multi-entrada, multi-salida con conexión wifi a Internet por medio del cual se pueda monitorear variables, realizar control de acceso y encender y apagar dispositivos en lugares remotos.

3.1.1 Objetivos Específicos

✓ Identificar y recolectar información sobre las diferentes técnicas de diseño de circuitos impresos, tarjetas de adquisición de datos y diseño de páginas web.

✓ Identificar y recolectar información sobre la arquitectura de las redes LAN inalámbricas.

✓ Desarrollar un programa capaz de controlar un micro controlador que será el dispositivo programable encargado del manejo de la información arrojada por los sensores del sistema.

✓ Simular el funcionamiento del dispositivo por medio del software existente destinado a este propósito.

✓ Utilizar el software existente para el diseño de circuitos impresos para el diseño de la tarjeta electrónica necesaria para la fabricación del dispositivo.

4 MARCO REFERENCIAL

La Domótica tiene como objetivo general mejorar la calidad de vida de las personas que habitan un determinado lugar mediante la optimización y el control de variables asociadas a la seguridad, confort y ahorro energético.

Los sistemas domóticos constan de varios elementos comunicados entre sí dentro del mismo lugar a través de redes y recursos electrónicos que les permiten interactuar de tal forma que sumados al lugar geográfico y las instalaciones físicas se pueden catalogar como una casa, una oficina o un edificio inteligente.

La Historia de la Domótica está ligada al desarrollo de la computación y la electrónica, por cuanto hablar de su historia es hablar del desarrollo de las nuevas tecnologías y de la adaptación de estas a las necesidades de los seres humanos por cuanto es difícil precisar una fecha concreta para el nacimiento de la Domótica. Sin embargo, hay fechas importantes qué destacar en la breve historia de la Domótica.

En el año 1978 salió al mercado el sistema X-10 que era un lenguaje de comunicación utilizado por los productos X-10 compatibles para comunicarse entre sí y el cual les permite controlar las luces y los electrodomésticos de un hogar aprovechando para ello la instalación eléctrica existente, evitando tener que instalar cableado adicional. ***(Carlos alvial, 2011)***

En los años 90 en Estados unidos, Japón y algunos países del norte de Europa fueron desarrollados sistemas integrados a nivel comercial coincidiendo con el inicio y el despliegue de Internet; para este tiempo se iniciaban además los desarrollos de las pasarelas residenciales y nuevos métodos de acceso.

La introducción de la tecnología al hogar se ha venido realizando de forma gradual en cada uno de los electrodomésticos, aunque de forma individual y limitando la comunicación entre dispositivos, relegando la Domótica a un mercado bastante limitado. Recientemente, con la plena irrupción de Internet en el hogar y en general las denominadas TIC (Tecnologías de La Información y las Comunicaciones), se ha forjado una nueva forma de entender la aplicación de la tecnología en el hogar mucho más positiva y realista.

5 MARCO TEÓRICO

Este proyecto busca establecer comunicación de dispositivos domésticos a través de internet por medio de un dispositivo mediante el cual se gestione, controle y establezca dicha comunicación, por lo que se hace importante entonces conocer los protocolos de comunicación con los cuales se trabaja actualmente para, de acuerdo a las necesidades y las ventajas que ofrezcan dichos protocolos realizar la selección del más adecuado.

En vista de que esta comunicación entre dispositivos deberá presentar un ambiente amigable al usuario cobran importancia conocimientos acerca del protocolo de transferencia de texto y algunos conceptos y lenguajes de programación útiles en el diseño de páginas web.

5.1 PROTOCOLOS DE RED TIPOS Y UTILIDADES

Existen dos tipos de tecnología para la comunicación entre los dispositivos presentes en una casa, oficina o edificio inteligente: la tecnología cableada que utiliza un medio físico para transmitir la señal y la tecnología inalámbrica que utiliza como medio de transmisión de información el aire.

La tecnología cableada se subdivide en dos tipos, el cable dedicado para el cual existe la necesidad de colocar un cable nuevo para la comunicación con el dispositivo y el cable compartido en el que se utiliza el cable que ya está distribuido por la vivienda.

5.1.1 Cable Dedicado

Este tipo de tecnología ofrece tres protocolos.

5.1.1.1 Interfaz IEEE 1394. Es conocido con nombres diferentes dependiendo el fabricante, Apple Inc lo denomino FireWire, Sony lo llamo i.LINK y Texas Instruments decidió nombrarlo como Lynx este protocolo es un estándar multiplataforma para entrada y salida de datos en serie a gran velocidad, por su velocidad suele utilizarse para la interconexión de dispositivos digitales como cámaras, videocámaras, discos duros, impresoras, reproductores de vídeo digital, sistemas domésticos para el ocio, sintetizadores de música y escáneres. ***(Domoprac)***

Su última versión fue superada en popularidad por USB 3.0 por lo que esta interfaz se está viendo reducida a ser una interfaz para aplicaciones muy particulares en productos de audio y video cada vez más pocos y los ordenadores de Apple.

5.1.1.2 USB (Universal Serial Bus). Este protocolo fue creado en 1996 por siete empresas (IBM, Intel, Northern Telecom, Compaq, Microsoft, Digital Equipment Corporation y NEC). Este estándar incluye la transmisión de energía eléctrica al dispositivo conectado, para el caso en que los dispositivos requieren una potencia mínima se pueden conectar varios sin necesitar fuentes de alimentación extra. ***(USB, 2011)***

Los dispositivos USB se clasifican en cuatro tipos según su velocidad de transferencia de datos:

✓ **Baja Velocidad (1.0).** cuya tasa de transferencia de hasta 1,5 Mbps (192 KB/s) y es utilizada en su mayor parte por dispositivos de interfaz humana como los teclados, los ratones (mouse), las cámaras web, etc.

✓ **Velocidad Completa (1.1).** Su tasa de transferencia de hasta 12 Mbps (1,5 Mbps) fue la más rápida antes de la especificación USB 2.0. Muchos dispositivos fabricados en la actualidad trabajan a esta velocidad. Estos dispositivos dividen el ancho de banda de la conexión USB entre ellos basados en un algoritmo de impedancias.

✓ **Alta Velocidad (2.0).** Su tasa de transferencia es de hasta 480 Mbps (60 Mbps) y está presente casi en el 99% de los PC actuales. El cable USB 2.0 dispone de cuatro líneas, un par para datos, una línea de corriente y una línea de toma de tierra.

✓ **Súper Alta Velocidad (3.0).** Tiene una tasa de transferencia de hasta 4.8 Gbps (600 MB/s). Esta especificación es diez veces más veloz que la anterior (2.0) y se lanzó a mediados de 2009 es compatible con los estándares anteriores.

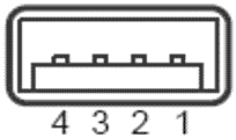
Las principales ventajas del USB (2.0) son su bajo costo, sencillo montaje y configuración ideal para la conexión de todo tipo de dispositivos a un PC o similar.

✓ **Generalidades Del USB 2.0** El USB es un sistema punto a punto en el cual el punto de partida es el host (Computador o hub) y el destino es un periférico u otro Hub. Dado que sólo hay un host, los periféricos comparten la banda de paso .

Este protocolo está basado en el paso de testigo (token) en el cual el computador proporciona el testigo al periférico seleccionado y seguidamente este le devuelve el testigo en respuesta.

✓ **Especificaciones Eléctricas.** En la Figura 1 se observa un diagrama de los pines presentes en el conector USB, la señal se transfiere mediante un par trenzado con una impedancia característica de 90Ω y la alimentación de 5 Voltios con su respectiva conexión a tierra.

Figura 1. Diagrama eléctrico del conector USB



1	+5 (Vbus)	Red
2	Data-	White
3	Data+	Green
4	Gnd	Black

✓ **Descripción Del Protocolo.** Una transacción sobre el bus requiere al menos tres paquetes. La transacción comienza por el envió por parte del host de un paquete que indica el tipo, la dirección de la transacción, la dirección del USB y su punto terminal (Endpoint), este paquete recibe el nombre de paquete testigo o (Token Paquet). ***(Domoprac)***

A continuación el transmisor envía un paquete de datos indicando que no tiene paquetes para transmitir y por último el destinatario envía un paquete indicando que ha recibido bien el paquete de datos.

El enlace entre el host y el periférico es conocido como (pipe), existen dos tipos el Stream y message en el caso del Stream es un flujo sin estructura y el Message que si posee una. Cada enlace es definido durante la inicialización del USB y se caracteriza por su banda de paso, su tipo de servicio, el número del Endpoint y el tamaño de los paquetes

Siempre existe un enlace de valor 0 que permite tener acceso a la información de configuración del periférico USB (estado, control e información).

En el ordenador el driver se encarga del encaminamiento de los paquetes sobre el bus.

✓ **Robustez.** El sistema es fiable y robusto, garantizando la integridad de los datos gracias a las siguientes características.

Señal diferencial, en donde el apantallamiento (blindaje) protege los hilos.

CRC (control) sobre la información de control y datos.

Detección de conexión y desconexión del periférico

Recuperación automática en cado de error, gestión del tiempo de recuperación (timeout).

Gestión del flujo de datos, de los buffers y del modo síncrono.

Construcción que le hace independiente de las funciones, de los datos y del control.

✓ **Tipos De Transferencia.** El pipe puede ser de cuatro tipos, y solo soporta un tipo de los descritos a continuación.

Control, utilizado para realizar configuraciones, no tiene perdida de datos puesto que lso dispositivos de detección de recuperación de errores están activos a nivel USB.

Bulk, utilizado para la transacción de datos voluminosos con pocas restricciones de duración de la transmisión, tampoco tiene perdida de datos. La banda de paso otorgada a este tipo de transmisión depende de los demás periféricos siendo esta la de menor prioridad.

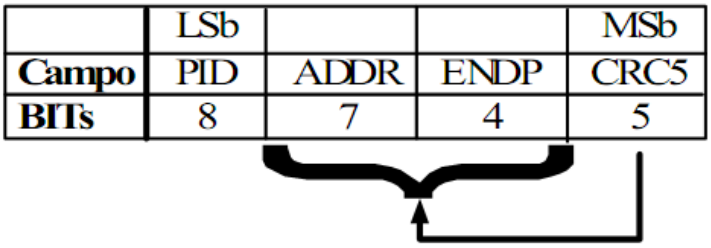
Interrupt, utilizado para transmisión de pequeños paquetes con un tiempo no mayor al determinado por la interfaz, aplicaciones como el mouse en un computador son típicas de este tipo de transferencia.

Isochronus o flujo en tiempo real, utilizado para la transmisión de audio y video.

✓ **Asignación De La Banda De Paso.** La banda de paso se asigna en función de los pipes, los dispositivos cuentan con memoria intermedia de datos (buffers) de acuerdo con el tamaño de esta memoria es asignada la banda de paso de tal forma que a pesar de los buffers el retardo de encaminamiento sea de unos pocos milisegundos.

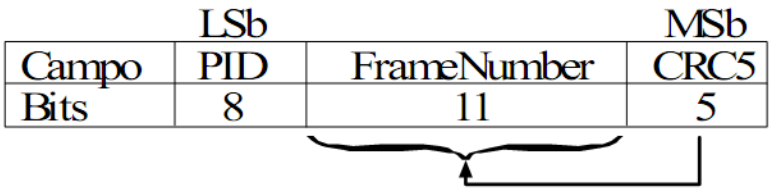
✓ **Formato De Los Paquetes.** La Figura 2 muestra la estructura de un token, el USB define un token especial (token de 4 bytes) para las transacciones split, este tipo de transacción permite el envió de información adicional y es utilizada para que soporte el intercambio de archivos entre el host y el hub en alta velocidad cuando el hub está conectado y transfiriendo archivos con dispositivos de baja y velocidad completa en uno de sus puertos.

Figura 2. Estructura de un Token



La Figura 3 muestra un esquema general de un paquete Start-Of-Frame, el host envía los SOF con una relación nominal cada 1,00ms±0,0005ms en un bus velocidad completa y 125µs±0,0625µs en uno de alta velocidad, los paquetes SOF consisten en un PID indicando el tipo de paquete seguido de un número de Frame de 11 bits.

Figura 3. Estructura de un paquete Start-Of-Frame



El Token SOF comprende la transacción Token-Only que distribuye un marcador SOF y acompaña el Frame Number con intervalos de tiempo precisos al comienzo de cada Frame, todas las funciones de alta y velocidad completa, incluyendo los hubs, reciben el paquete SOF. El Token SOF no provoca la función recepción para generar un paquete de retorno, por tanto, un envío SOF a una función no tiene garantías.

El paquete SOF entrega dos trozos de información de tiempo. Una función informa que ha ocurrido un SOF cuando este detecta el PID SOF.

✓ **Estados Del USB.** Los distintos pasos hasta que el dispositivo se pueda utilizar son.

Por defecto, el dispositivo una vez este alimentado puede no responder ante una transacción hasta que recibe el Reset por parte de bus a partir del este momento el dispositivo tendrá su dirección por defecto.

Direccionado, antes de utilizar el dispositivo debe ser configurado, dicha operación implica una respuesta al SetConfiguration distinta de 0.

Enumeración del bus, cuando se conecte un dispositivo USB a un puerto alimentado, transcurren las siguientes acciones.

El hub avisa al host el cambio de pipe y el dispositivo está en estado alimentado y el hub en estado desactivado.

El host determina la naturaleza exacta preguntando al hub.

El host conoce el puerto al cual está conectado el dispositivo, el host espera 100ms para que se complete el proceso de inserción y para que la alimentación del dispositivo se vuelva estable. El host activa el puerto y manda un reset al mismo.

El hub realiza el proceso requerido para el reset en este puerto, cuando la señal de Reset se ha completado, el puerto se activa. El dispositivo USB está en el estado “por

defecto” y no puede consumir más de 100 mA del V BUS. Todos estos registros y estados se han reiniciado y responden al estado “Por Defecto”.

El host asigna una dirección única al dispositivo USB, trasladando al dispositivo al estado “Direccionado”

Antes de que el dispositivo tenga una única dirección, se accede a la pipe de control por defecto por la dirección por defecto. El host lee el descriptor del dispositivo para determinar cuál es el máximo del tamaño de los datos útiles que utiliza el dispositivo en la pipe por defecto.

El host lee la información de configuración del dispositivo leyendo la configuración de 0 hasta n-1, donde n es el número de configuraciones. Este proceso puede tardar varios milisegundos.

Basándose en la información de configuración y cómo se debe usar el dispositivo USB, el host asigna un valor de configuración al dispositivo. El dispositivo está en el estado “Configurado” y todos los Endpoints de esta configuración están como se indica en sus características. El dispositivo USB puede que no tenga la energía que indica en sus características. Desde el punto de vista del dispositivo, está listo para usarse. (Gonzalez, 2011)

5.1.1.3 Ethernet (LAN). Ethernet es un estándar de redes de computadoras de área local con acceso al medio por CSMA/CD “Acceso Múltiple por Detección de Portadora con Detección de Colisiones" que es una técnica usada en redes Ethernet para mejorar sus prestaciones.

Ethernet se planteó en un principio como un protocolo destinado a cubrir las necesidades de las redes LAN pero a partir de 2001 alcanzó los 10 Gbps lo que le dio mucha más popularidad a la tecnología. Hace ya mucho tiempo que Ethernet consiguió situarse como el principal protocolo del nivel de enlace. Ethernet 10Base2 consiguió en la década de los noventas una gran aceptación en el sector y actualmente esta versión se considera como una "tecnología de legado" respecto a 10BaseT, 100BASE-TX y 100BASE-T que registran velocidades de 10 Mbps, 100 Mbps y 1 Gbps respectivamente y son las tecnologías más usadas en la actualidad.

Su principal ventaja es que es la tecnología de red doméstica más rápida, segura y fácil de mantener. Como principal desventaja tiene que su instalación de cableado y dispositivos de red puede resultar costosa. (Shoch, 1980)

5.1.2 Cable Compartido

5.1.2.1 PLC (Power Line Communications). Es un término inglés que puede traducirse como comunicaciones mediante cable eléctrico y que se refiere a diferentes tecnologías que utilizan las líneas de energía eléctrica convencional para transmitir señales de radio con propósitos de comunicación y para convertirla en una línea digital de alta velocidad de transmisión de datos, permitiendo la aplicación de diferentes tecnologías entre ellas las más usadas Homeplug y HomeplugAV.

HomePlug es una organización que componen cerca de 50 compañías, cuyo objetivo es usar el concepto de PLC para establecer una línea de comunicación entre los equipos conectados a la línea de potencia en un determinado lugar.

Este tipo de tecnología trata de cubrir el nicho de mercado donde el WIFI no puede llegar por distancias, muros gruesos entre habitaciones e interferencias. Su desarrollo está marcado por diferentes versiones ***(Domoprac)***

- ✓ **Homeplug 1.0+AV (In-Home Connectivity).** Se incluyen los aparatos electrónicos con aplicaciones dentro del hogar digital por ejemplo distribución de HDTV por la red eléctrica.

- ✓ **Homeplug 1.0.** Conecta computadores u otros dispositivos que utilizan Ethernet, USB, y 802.11.

HomePlug AV es la siguiente generación creada para dar soporte al ancho de banda necesario para implantar HDTV y VOIP en el hogar digital, llegando a velocidades de 200 Mbps (PHY layer) o 100 Mbps (MAC layer). Además soporta protocolos de encriptación de datos de 56 a 128 bits, pudiendo coexistir ambas tecnologías (1.0 y AV).

- ✓ **Homeplug BPL (To-The-Home Broadband).** Creada para desarrollar tecnologías de banda ancha, usando las redes de suministro eléctrico de baja tensión para dar servicios de Internet y telefonía como alternativa al xDSL.

- ✓ **Homeplug Home Automation (Command And Control Applications).** Para aplicaciones de domotica enfocada en diseñar dispositivos para el control de luces y climatización.

Existe una nueva versión en desarrollo llamado HomePlugAV2 se estima que su capacidad de transferencia sea de 600 Mbps y que sea totalmente compatible con HomePlug AV su salida al mercado con productos está planeada para finales de 2011.

Sus principales ventajas son el bajo costo de instalación, la ausencia de cableado adicional y un alto ancho de banda como principales desventajas están la oferta limitada de productos y la inexistencia de instaladores especializados. ***(Domoprac)***

5.1.2.2 Home PNA (Home phoneline networking alliance). Es una alianza de varias empresas que trabajan en el desarrollo de tecnología para implementar redes de área local a través de la instalación telefónica de una vivienda mediante la construcción de una red de área local sin nuevos cables ni obras que permita comunicar computadores, impresoras y otros recursos como HUBS, routers y xDSL.

Consiguiendo velocidades de transmisión de datos de hasta 320 Mbps, con calidad de servicio garantizada (QoS), la tecnología HomePNA está preparada para cubrir la fuerte demanda de servicios multimedia en el hogar digital tales como televisión IP (IPTV) y Telefonía IP (VoIP).

Sus principales ventajas son Instalación fácil, económica y no requiere equipos de red, como principales desventajas tiene velocidad limitada según aplicaciones, ruidos, muy pocos dispositivos en el mercado. ***(Domoprac)***

5.1.3 Tecnologías Inalámbricas

Este tipo de tecnología se divide en dos grandes grupos determinados por las distancias que puedan cubrir.

5.1.3.1 Corto Alcance. Tiene un cubrimiento de áreas pequeñas la WPAN (Wireless Personal Area Network) cuya potencia de transmisión es de 10 mW y tiene un bajo consumo de potencia está representada por 6 protocolos descritos a continuación.

- ✓ **Bluetooth IEEE 802.15.1.** Bluetooth es el nombre común de la especificación industrial IEEE 802.15.1 que define un estándar global de comunicación inalámbrica que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia segura y sin licencia de corto rango. Mediante este estándar se pretende facilitar las comunicaciones entre equipos móviles y fijos y eliminar cables y conectores entre éstos, ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre equipos personales.

Los dispositivos que con mayor frecuencia utilizan esta tecnología son los de los sectores de las telecomunicaciones y la informática personal, como PDAs, teléfonos celulares, computadoras portátiles, impresoras y cámaras digitales, las principales ventajas de este protocolo son la inexistencia de cables, bajo consumo en corriente y sus principales desventajas son la configuración y alto costo de instalación. (Vasquez, 2006)

- ✓ **Wimedia IEEE 802.15.3.** Difiere sustancialmente de las estrechas frecuencias de banda de radio (RF) y tecnologías spread spectrum, como el Bluetooth y el 802.11.Wimedia por que usa un ancho de banda muy alto del espectro de RF para transmitir información por lo tanto es capaz de transmitir más información en menos tiempo que las tecnologías anteriormente citadas.

Sus principales ventajas son que a diferencia de Bluetooth, WiFi, teléfonos inalámbricos y demás dispositivos de radiofrecuencia que están limitados a frecuencias sin licencia en los 900 MHz, 2.4 GHz y 5.1 GHz este protocolo hace uso de un espectro de frecuencia recientemente legalizado, como principal desventaja está el hecho de que al estar compartiendo bandas de frecuencia con otros dispositivos, ha hecho que aunque esto le permite tener una alta productividad los dispositivos han de estar a distancias relativamente cortas. ***(Vasquez, 2006)***

✓ **Zigbee IEEE 802.15.1.** Es un conjunto de protocolos de comunicación de alto nivel para su utilización con radios digitales de bajo consumo, basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (wireless personal area network, WPAN). Su objetivo son las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de sus baterías, su campo de aplicación principal es la domótica debido a su bajo consumo, su sistema de comunicaciones vía radio (con topología de red en malla) y su fácil integración (se pueden fabricar nodos con muy poca electrónica). Sus principales ventajas son su bajo costo, bajo consumo de energía, por lo que pueden funcionar con base en pilas ordinarias (y en intervalos de tiempo que alcanzan el orden de años). Su principal desventaja es la baja velocidad. ***(Gislason, 2011)***

✓ **Wave.** Tecnología propietaria desarrollada por la empresa Zensys (que vende chips y software a las firmas que deseen diseñar productos compatibles con Z-Wave). Es una tecnología de comunicación inalámbrica, que permite transmitir y recibir pequeñas instrucciones (señales de comando) las redes basadas en esta tecnología no dependen de un punto central de control (servidor), ya que la plataforma de conectividad se establece a partir de dispositivos compatibles que se enlazan entre sí. Los chips Z-Wave se utilizan para crear sistemas inalámbricos que controlan funciones de iluminación, seguridad, acceso, sensores, alarmas y comunicación entre dispositivos residenciales o industriales. Su velocidad de transmisión es hasta de 40 Kbps (en chips de segunda generación), soporta hasta 232 dispositivos. **(Domoprac)**

✓ **IRDA Infrared.** Este protocolo desarrollado en 1993 entre HP, IBM, Sharp y otros. Define un estándar físico en la forma de transmisión y recepción de datos por rayos infrarrojo. Esta tecnología está basada en rayos luminosos que se mueven en el espectro infrarrojo soporta una amplia gama de dispositivos eléctricos, informáticos y de comunicaciones, permite la comunicación bidireccional entre dos extremos a velocidades que oscilan entre los 9.6 Kbps y los 4 Mbps. Sus principales ventajas son la tecnología muy extendida, fácil implantación y uso. Su principal desventaja es la baja velocidad. ***(Domoprac)***

✓ **Home RF**. La idea de este estándar se basa en el teléfono inalámbrico digital mejorado (Digital Enhaced Cordless Telephone, DECT) transporta voz y datos por separado al contrario de los protocolos como el WiFi que transporta la voz como una forma de datos, los creadores de este estándar pretendían diseñar un aparato central en cada casa que conectara los dispositivos y además proporcionar un ancho de banda de datos entre ellos.

Cabe resaltar que el estándar HomeRF presta servicios como identificador de llamadas, llamadas en espera, regreso de llamadas e intercomunicación dentro del hogar. El HomeRF2 tiene velocidad de entre 5 y 10 Mbps, 15 canales de 5 MHz para voz. Su principal ventaja es que no requiere punto de acceso y es de fácil instalación como su principal desventaja tiene que El Home RF Working Group se disolvió en Enero de 2003.

5.1.3.2 Largo alcance. Este grupo cubre áreas mucho más grandes de hasta 100 metros WLAN (Wireless Local Area Network) presenta alta potencia de transmisión 100 mW, y por consiguiente un gran consumo de potencia

✓ **WLAN (Wireless Local Area Network).** Una LAN inalámbrica se define como una de red de alcance local que tiene como medio de transmisión el aire que cubre un entorno geográfico limitado con una velocidad de transferencia de datos relativamente alta (mayor o igual a 1 Mbps según normas IEEE 802.11x) con baja tasa de errores y administrada de forma privada.

En 1980 WLAN fue aprobada por la FCC (Federal Communication commission) para uso industrial, medico y científico, ya en 1997 se presento la primera estandarización IEEE 802.11 con tasas de datos superiores a los 2 Mbps utilizando modulación Spread Spectrum en las bandas ISM. Desde entonces han surgido varias versiones del estándar (Ver Tabla 1).

Tabla 1. Variaciones del estándar 802.11

	802.11	**802.11ª**	**802.11b**
Aplicación	WLAN	WLAN	WLAN
Banda de Frecuencia	2.4GHz	5GHz	2.4GHz
Máxima Velocidad	2Mbps	54Mbps	11Mbps

Los beneficios de utilizar WLAN son diversos se destacan entre ellos la movilidad que le permite a los usuarios acceder a información en tiempo real dentro del área de cobertura de la red, la simplicidad y economía en la instalación al eliminar por completo la ubicación de cable a través de paredes y techos lo que la convierte en la tecnología que puede ir donde la alambrada no puede ir, aunque la inversión inicial requerida puede ser mayor los beneficios a corto y largo plazo son mayores sobretodo en ambientes dinámicos en los cuales se requieren movimientos y cambios con bastante frecuencia, son completamente escalables lo que les permite ser configuradas de forma fácil en diferentes topologías además de permitir la incorporación de nuevos usuarios a la red sin mayores complicaciones. ***(Cruz, 2006)***

✓ **GSM.** Group Special Mobile es un estándar mundial para teléfonos móviles digitales, fue creado por la CEPT y posteriormente desarrollado por ETSI Europa, (alrededor del 70% de los usuarios de teléfonos móviles del mundo en 2001 usaban GSM), anteriormente conocida como un estándar para los teléfonos móviles europeos, fue desarrollado con la intención de desarrollar una normativa que fuera adoptada mundialmente.

Las implementaciones más veloces de GSM se denominan GPRS y EDGE, también denominadas generaciones intermedias o 2.5G, que conducen hacia la tercera generación 3G o UMTS.

✓ **GPRS.** General Packet Radio Service (GPRS) o servicio general de paquetes vía radio es una extensión del GSM para la transmisión de datos no conmutada (o por paquetes). Permite velocidades de transferencia de 56 a 144 kbps.

Con esta tecnología se ofrecen los servicios de Wireless, SMS, mensajería multimedia e internet, existen tres tipos de dispositivos de acuerdo con la posibilidad de utilizar simultáneamente GPRS Y GSM.

Clase A pueden utilizar servicios de GPRS y GSM simultáneamente, B solo pueden estar conectados a uno de los servicios, C se conectan alternativamente a uno u otro servicio.

5.2 HTTP PROTOCOLO DE TRANSFERENCIA DE TEXTO.

Es el protocolo usado en cada transacción del sistema de distribución de información World Wide Web. En 1999 se especifico la versión 1.1 HTTP que definió la sintaxis y la semántica que utilizan los elementos de software de la arquitectura web para comunicarse.

HTTP funciona bajo el esquema de petición-respuesta entre un cliente y un servidor, la información transmitida se le llama recurso y se la identifica mediante un localizador uniforme de recursos URL.

Este es un protocolo sin estado es decir no guarda ningún tipo de información sobre conexiones anteriores para tal efecto se utilizan los cookies que es información que el servidor puede almacenar en un sistema cliente.

Una transacción HTTP es formada por un encabezado que especifica generalmente el tipo de acción requerida del servidor, el tipo de dato retornado o el código de estado, el encabezado es un bloque de datos que precede a la información propia de la transacción por lo que suelen referirse a él como un metadato ya que contiene datos sobre los datos.

Existen múltiples versiones de este protocolo HTTP 0.9, HTTP 1.0, HTTP 1.1 Y HTTP 1.2. ***(W3C, Hypertext Transfer Protocol, 2011)***

5.2.1 Métodos De Petición.

✓ **Head:** Pide una respuesta idéntica a la que correspondería a una petición GET, pero sin el cuerpo de la respuesta.

✓ **Get:** Pide una representación del recurso especificado.

✓ **Post:** Somete los datos a que sean procesados para el recurso identificado. Los datos se incluirán en el cuerpo de la petición. Esto puede resultar en la creación de un nuevo recurso o de las actualizaciones de los recursos existentes o ambas cosas.

✓ **Put:** Sube, carga o realiza un upload de un recurso especificado (archivo).

✓ **Delete:** Borra el recurso especificado.

✓ **Trace:** Este método solicita al servidor que envíe de vuelta en un mensaje de respuesta.

✓ **Options:** Devuelve los métodos HTTP que el servidor soporta para un URL especifico.

✓ **Connect:** Convierte la solicitud de conexión a un sistema transparente de TCP / IP del túnel.

5.3 HTML (HYPERTEXT MARKUP LANGUAGE).

Es el lenguaje predomínate para la elaboración de páginas web, usado para describir la estructura y el contenido en forma de texto y complementar dicho texto con objetos tales como imágenes.

El HTML se escribe en forma de etiquetas rodeadas por corchetes angulares, sus principales componentes son los elementos, atributos, tipos de data y declaración de tipo de documento.

El lenguaje HTML puede ser creado y editado por cualquier editor de textos básico incluyendo a aquellos que admitan texto sin formato***. (W3C, HyperText Markup Language)***

5.4 JAVASCRIPT

JavaScript es un lenguaje de programación interpretado que surgió para extender las capacidades del lenguaje HTML. Con Javascript es posible incorporar efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario sin de sobrecargar el servidor.

JavaScript no es un lenguaje de programación propiamente dicho como C, C++, Delphi, etc. Es un lenguaje script u orientado a documento, como pueden ser los lenguajes de macros que tienen muchos procesadores de texto y planillas de cálculo. No se puede desarrollar un programa con JavaScript que se ejecute fuera de un Navegador. ***(Basics, 2011)***

Javascript no guarda ninguna relación con el lenguaje de programación java.

5.5 AJAX

Ajax, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

Ajax es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores dado que está basado en estándares abiertos como JavaScript y Document Object Model (DOM). ***(Uberbin, 2011)***

6 DISEÑO Y CONSTRUCCIÓN DEL DISPOSITIVO

6.1 ESQUEMA GENERAL DE FUNCIONAMIENTO DEL SISTEMA

El sistema de monitoreo y control de variables remotas a través de internet es compatible con las tecnologías de comunicación típicas en el hogar moderno y permite una fácil configuración y manejo.

Actualmente es posible ingresar a internet desde una gran cantidad de dispositivos. Los teléfonos celulares y las consolas de videojuegos, por ejemplo, permiten navegar en internet sin necesidad de un computador y como lo que se busca es poder monitorear nuestro sistema desde internet lo mejor es hacerlo compatible con la mayoría de estos equipos electrónicos. La navegación en internet es posible gracias al protocolo de transferencia de hipertexto HTTP, en el cual están basadas todas las páginas web y que es interpretado por todos los navegadores sin importar el sistema operativo o tipo de dispositivo desde el que estamos navegando. Por esta razón, el monitoreo y control se hace por medio de una página web escrita en lenguaje HTML contenida en el cerebro del sistema.

6.1.1 Modos De Funcionamiento. Para el sistema se han establecido dos modos de funcionamiento.

6.1.1.1 Modo Normal (Monitoreo Y Control). En este modo de funcionamiento el usuario se conecta al dispositivo a través de internet o de la red local para conocer el estado de los sensores y controlar los puertos de salida. La comunicación se realiza por medio de una conexión inalámbrica Wifi. El factor determinante en la decisión de utilizar este protocolo es la facilidad de instalación, al no requerir de cableado de datos y la compatibilidad directa con el protocolo usado para las comunicaciones en internet. Además, la red inalámbrica Wifi está disponible en la mayoría de hogares que cuentan con internet de banda ancha.

6.1.1.2 Modo De Configuración. Permite la configuración de los parámetros de red (Dirección IP, mascara de subred, puerta de enlace, nombre de la red y contraseña de acceso). La configuración se realiza desde un software de computador desarrollado para este fin, el cual realiza la conexión entre el sistema de monitoreo y el computador usando el protocolo USB 2.0.

6.1.2 Bloques Funcionales. El sistema está compuesto por varios bloques los cuales cumplen una función específica y que juntos interactuando hacen posible el correcto funcionamiento del sistema. Un diagrama de dichos bloques se puede observar en la **Figura 4**.

6.1.2.1 Conexión Inalámbrica Wifi. Es un módulo conversor de protocolo bidireccional que traduce los paquetes de datos recibidos desde la conexión inalámbrica Wifi al protocolo serial y viceversa, permitiendo que el controlador principal se conecte a una red inalámbrica.

Si bien existe la opción de desarrollar todo un sistema que realice la conexión a la red inalámbrica Wifi, se optó por utilizar un módulo conversor comercial para centrarnos en la aplicación puntual.

El uso de un módulo conversor comercial tiene las siguientes ventajas:

✓ Se reduce el tiempo de desarrollo.

✓ Se cuenta con un sistema probado y homologado para conexión inalámbrica.

✓ Posibilidad de migrar fácilmente la aplicación a otros tipos de red, para habilitar el sistema en una red LAN solo es necesario reemplazar el módulo conversor Wifi/Serial por un módulo Ethernet/Serial. ***(rovingnetworks)***

En el mercado existen diferentes fabricantes de módulos conversores Wifi/Serial, se eligió el módulo RN-131C Wifly GSX de Roving Networks por su bajo costo y su facilidad de configuración, este modulo se puede observar en la **Figura 5**.

Figura 4. Diagrama de bloques del dispositivo

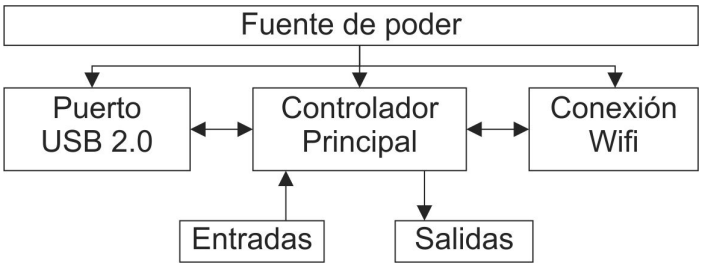


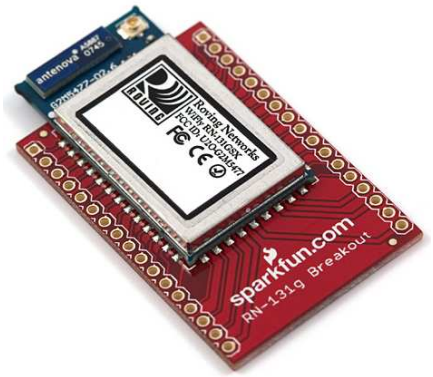
Figura 5. Modulo RN-131C

La tienda sparkfun tiene disponible el módulo montado en un circuito impreso que permite la conexión del conversor en un protoboard y facilita el desarrollo del prototipo como se observa en la Figura 6.

Las principales características del módulo seleccionado son:

- ✓ Transmisor/receptor para 2.4 Ghz IEEE 802.11b/g
- ✓ Transferencia de hasta 1 Mbps con TCP/IP y WPA2
- ✓ Ultra bajo consumo 4uA en reposo, 40mA en recepción, 210mA (máximo) en transmisión.
- ✓ Antena cerámica integrada y conector U.FL para antena externa
- ✓ Hardware para interface UART (Puerto serial RS-232)
- ✓ Certificado por la Wifi Allience
- ✓ Certificado por la FCC/CE/ICS
- ✓ Voltaje de alimentación 3.3 Vdc.
- ✓ Configuración mediante comandos en código ASCII por el puerto serial RS-232.
- ✓ Autentificación Wifi: WEP-128, WPA-PSK (TKIP), WPA2-PSK (AES)
- ✓ Temperatura de operación. 0 a 70 C
- ✓ Humedad relativa <= 90%

Figura 6. Modulo RN-131C ensamblado en un circuito impreso



6.1.2.2 Controlador Principal. Es el cerebro de todo el sistema y gestiona los modos de funcionamiento.

En el modo normal cumple la función de servidor web respondiendo a las peticiones recibidas desde el módulo conversor Wifi/Serial, ya sea enviando la página web que será visualizada por el navegador del usuario o ejecutando las diferentes acciones como activar o desactivar las salidas, también lee el estado de los sensores para mantener actualizada la información al usuario.

En el modo de configuración establece un puente entre el puerto USB del computador y el módulo conversor Wifi/Serial.

Antes de seleccionar el controlador principal es necesario limitar el sistema para determinar los requerimientos de hardware y capacidad de procesamiento necesario, en este caso particular el sistema de monitoreo y control tiene las siguientes características.

✓ 3 Entradas para sensores de contacto seco típicos en sistemas de alarmas para la puerta principal, el sensor de movimiento y un sensor adicional.

✓ 1 Entrada análoga para sensor de temperatura ambiente

✓ 6 salidas de potencia para cargas de 110/220 VAC 10 A para la iluminación general, ventilador con opción de funcionar automáticamente dependiendo de la temperatura ambiente y las 4 salidas restantes como auxiliares para que el usuario elija que quiere controlar.

✓ 1 Salida para control de intensidad de luz incandescente.

Con lo anterior las principales características que debe tener el controlador para diseñar un sistema de bajo costo y con pocos componentes son.

✓ Número de puertos de entrada y salida suficientes

- ✓ Conversor análogo a digital
- ✓ Memoria interna suficiente para almacenar una página web básica
- ✓ Hardware UART (Universal Asynchronous Receiver-Transmitter) para la comunicación serial con el módulo conversor Wifi/Serial
- ✓ Dispositivo comercial y de bajo costo
- ✓ Herramientas de desarrollo de bajo costo
- ✓ Entorno de desarrollo IDE y compilador gratuito o con versiones limitadas para estudiantes

Los microcontroladores además de ser la opción más económica cumplen con todos los requerimientos de hardware con una ventaja adicional y es que ya hay varias referencias disponibles con puerto USB incorporado, esto permite reducir el costo final y el número de componentes necesarios en el circuito.

La elección del microcontrolador empieza por la marca, hay diferentes fabricantes reconocidos, algunos de ellos: Atmel, Freescale (antes Motorola), Holtek, Intel, National Semiconductor, Microchip, NXP (antes Philips), Renesas (antes Hitachi, Mitsubishi y NEC), ST Microelectronics, Texas Instruments, Zilog, Silabs, entre otros.

La aplicación no requiere un gran poder de procesamiento ya que no hay que realizar cálculos matemáticos complejos, por lo tanto un microcontrolador de 8 bits es suficiente. En Colombia las marcas más usadas por los estudiantes y entusiastas son Atmel, Freescale y Microchip y en todas ellas es posible conseguir referencias que cumplan con los requerimientos de la aplicación. La elección final es más un tema de gustos o experiencia previa con alguno de ellos.

Finalmente, el microcontrolador elegido fue el PIC18F2550 de Microchip cuyas características son:

- ✓ Memoria Flash de 32 KB con hasta 100000 ciclos de escritura
- ✓ 12 MIPS (millones de instrucciones por segundo)
- ✓ Memoria RAM 2048 bytes
- ✓ Memoria EEPROM 256 bytes
- ✓ 1 puerto serial USART
- ✓ 2 Módulos de captura/comparación/PWM

- ✓ 1 Temporizador de 8 bits
- ✓ 3 Temporizadores de 16 bits
- ✓ 10 Canales de conversor análogo a digital de 10 bits
- ✓ 2 Comparadores análogos
- ✓ 1 Puerto USB 2.0
- ✓ Rango de temperatura: -40 a 85 C
- ✓ Voltaje de operación 2 a 5.5 Vdc
- ✓ Encapsulado de 28 pines
- ✓ Corriente máxima de 25 mA en los pines de entrada y salida
- ✓ Corriente máxima de todos los puertos 200 mA
- ✓ Corriente máxima en el pin de alimentación VDD 250 mA

Además de cumplir de sobra los requerimientos de hardware, incluyendo el puerto USB, tiene las siguientes ventajas:

- ✓ Económico, el precio de venta al público es de solo $17500
- ✓ Muy comercial, está disponible en la mayoría de tiendas de electrónica en Pereira y en las tiendas más populares en Colombia.
- ✓ Hardware económico para la programación del dispositivo, Microchip incluso facilita toda la información para construir un programador compatible con la mayoría de microcontroladores basado en este mismo PIC. (microchip)
- ✓ Entorno de desarrollo MPLAB gratuito suministrado por el fabricante.
- ✓ Optimizado para código en lenguaje C.
- ✓ Compilador gratuito en versión de estudiante MPLAB C18
- ✓ Herramientas avanzadas de simulación con soporte de simulación animada de puerto USB y depuración de código paso a paso: Software de simulación Proteus.
- ✓ Librerías completas y gratuitas para manejo de los periféricos avanzados, como el puerto USB 2.0

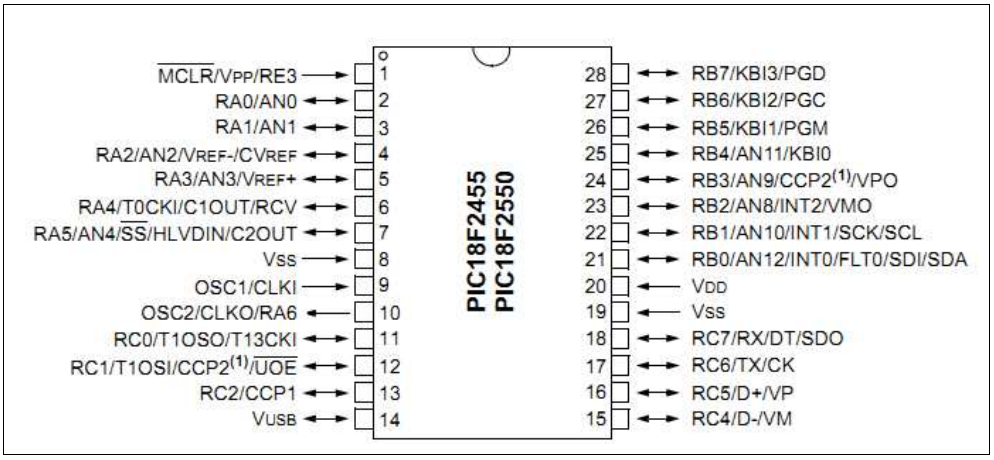
✓ Múltiples foros de ayuda y abundante información en internet, además del soporte oficial del fabricante con códigos de ejemplo y notas de aplicación.

✓ Experiencia previa con la familia de microcontroladores de Microchip.

Un diagrama de pines del microcontrolador utilizado puede observarse en la

Figura 7.

Figura 7. Diagrama del Microcontrolador



6.1.2.3 Puerto USB 2.0. Permite la conexión del sistema de monitoreo con el computador a través de puerto USB, está incluido en el controlador principal.

6.1.2.4 -Entradas. Entradas análogas y digitales que corresponden a los sensores que deben ser monitoreados por el controlador principal.

Las entradas del sistema se han definido de la siguiente forma.

✓ 3 Entradas para sensores de contacto seco típicos en sistemas de alarmas para la puerta principal, el sensor de movimiento y un sensor adicional.

✓ 1 Entrada análoga para sensor de temperatura ambiente

Los sistemas de alarmas trabajan a 12 Vdc, por lo tanto las entradas digitales deben ser compatibles con este voltaje, la conexión al microcontrolador se debe hacer a través de un circuito de adecuación que convierta las señales de 12 Vdc en señales de 5 Vdc compatibles con el voltaje del microcontrolador.

Para el sensor de temperatura ambiente se utiliza el circuito integrado LM35 el cual tiene las siguientes características:

- ✓ Calibrado en fábrica en grados centígrados
- ✓ Factor de escala lineal de 10 mV/˚C
- ✓ Precisión de 0.5 ˚C
- ✓ Rango de temperatura de -55 ˚C a 150 ˚C
- ✓ Voltaje de operación de 4 a 30 Vdc
- ✓ Requiere un máximo de 130 uA para su funcionamiento
- ✓ Corriente de salida máxima de 10 mA

El LM35 tiene la ventaja de que da una salida de voltaje proporcional a la temperatura y no requiere calibración, con 10 mV por grado centígrado en un rango de temperatura de 0 a 100 ^{o}C el voltaje máximo de salida del sensor es de 1 Vdc. El microcontrolador cuenta con un conversor análogo a digital de 10 bits el cual en un rango de entrada de 0 a 5 Vdc permite detectar cambios de voltaje de menos de 5 mV suficientes para leer la temperatura directamente del sensor con una precisión de 0.5 ˚C.

Una imagen del sensor utilizado se puede observa en la Figura 8.

Figura 8. Sensor LM35

6.1.2.5 Salidas. Salidas de potencia que permiten el manejo de cargas de 110 voltios en corriente alterna por parte del controlador principal.

Las salidas del sistema se han definido de la siguiente forma.

- ✓ 6 salidas de potencia para cargas de 110/220 Vac 10 A para la iluminación general, ventilador con opción de funcionar automáticamente dependiendo de la temperatura ambiente y las 4 salidas restantes como auxiliares para que el usuario elija que quiere controlar.

- ✓ 1 Salida para control de intensidad de luz incandescente, máximo 500 W.

Para las 6 salidas de 110/220 Vac 10A se han usado relés de 12Vdc/10A los cuales son controlados por medio del microcontrolador a través de un transistor, ya que la salida del microcontrolador no tiene el voltaje ni la corriente suficiente para manejar directamente los relés.

El relé es un dispositivo electromecánico el cual por medio de un electroimán acciona uno o varios contactos que permiten abrir o cerrar circuitos eléctricos independientes, de esta manera es posible accionar un circuito de 110 o 220 Vac con una señal de 12 Vdc.

La principal desventaja del relé electromecánico es el desgaste de los contactos debido a los arcos eléctricos que se presentan al momento de la conexión y desconexión de los circuitos de potencia por lo cual su vida útil es limitada, además de esto son lentos y requieren una gran cantidad de energía para su accionamiento comparados con los relés de estado sólido los cuales están basados en semiconductores. Sin embargo siguen vigentes y son muy usados en etapas de potencia de sistemas de domótica y accionamiento básico principalmente por su bajo costo y su facilidad de uso, además en estas aplicaciones no se requiere altas velocidades de conmutación y los ciclos de encendido/apagado son bajos.

La salida de control de intensidad de luz incandescente (dimmer) es mucho más compleja que las salidas simples de encendido/apagado, el control de intensidad de luz incandescente básicamente es un control de la potencia entregada a una carga que en este caso se trata de un bombillo de luz incandescente.

En una conexión directa a la red de corriente alterna el bombillo recibe todo el voltaje de la red o en otras palabras toda la onda seno de voltaje está pasando al bombillo y este enciende a su máxima intensidad.

El voltaje de la red entregado al bombillo en una conexión directa se puede observa en la
.

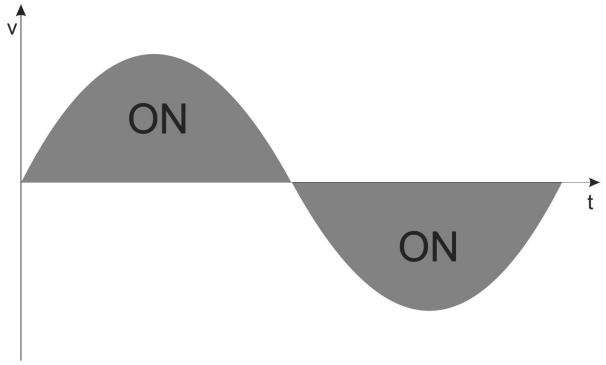
Si se deja pasar solo una parte de la onda seno de voltaje como se observa en la Figura 10 se está limitando también la potencia entregada al bombillo y por lo tanto este enciende a menor intensidad.

Para el control de potencia se debe usar en elemento semiconductor, ya que se requiere una alta velocidad de respuesta tanto de encendido como de apagado y un relé electromecánico no cumple con esta característica. Los Triacs son los elementos más usados para esta aplicación, trabajan en corriente alterna, son rápidos, de fácil manejo,

requieren una pequeña corriente en el terminal de puerta para pasar al modo de conducción y permanecen en ese estado hasta que la corriente a través de él sea menor a la corriente de sostenimiento.

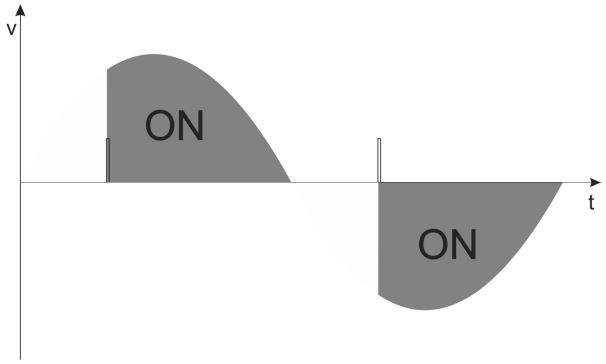
Por sus características el Triac requiere de un pulso de encendido de pequeña duración y se apaga automáticamente en los cruces por cero de la señal de voltaje, ya que con una carga resistiva en ese mismo instante la corriente a través del Triac se hace cero.

Figura 9. Voltaje entregado al bombillo en una red directa



El control de potencia debe ser simétrico para evitar que la onda de voltaje tenga componente dc, por lo tanto se debe detectar el cruce por cero de la señal de voltaje para sincronizar los encendidos del Triac en los dos semi-ciclos de manera que estén en conducción durante el mismo tiempo.

Figura 10. Voltaje entregado al bombillo por el Triac



En el sistema de monitoreo se usa el Triac Q4040J7 de marca Littelfuse, es un Triac de gran capacidad que no requiere el uso de disipador de calor para manejar cargas de hasta 500 W. Las principales características del Triac Q4040J7 son:

- ✓ Corriente máxima en régimen: 40 amperios
- ✓ Máxima corriente de puerta: 4 amperios
- ✓ Temperatura de operación: -40 a 125 ˚C
- ✓ Terminales aislados del encapsulado
- ✓ Voltaje de ruptura: 400 voltios

6.1.2.6 Fuente De Poder. La fuente de poder es la encargada convertir el voltaje de corriente alterna de la red eléctrica en voltaje de corriente continua con los niveles necesarios que requiere todo el sistema para funcionar, debido a los diferentes componentes del circuito la fuente de poder debe tener las siguientes características.

- ✓ Voltaje de alimentación 110 Vac
- ✓ Voltajes de salida:
  - 12 Vdc al menos 2 A, este voltaje se usa para los relés electromecánicos y para los sensores de entrada.
  - 5 Vdc al menos 500 mA, usados para alimentar el microcontrolador y el sensor de temperatura.
  - 3.3 Vdc al menos 500 mA, usado para alimentar el módulo conversor wifi/RS-232

Los 12V y 5V se obtienen directamente de una fuente conmutada que se puede observar en la **Figura 11** con salidas de 12 V 3A y 5V 2A, los 3.3V se obtienen a partir de los 5V por medio del circuito integrado regulador de voltaje LD117-3.3 el cual se puede observar en la **Figura 12** y tiene las siguientes características.

Figura 11. Fuente de poder

✓ Regulador LDO (Low-Dropout): Funciona con una pequeña diferencia entre voltaje de entrada y salida. Requiere de solo 1.2V por encima del voltaje de salida para entregar una corriente de 800 mA.

✓ Disponible con salidas fijas de 1.8V, 2.5V, 2.85V, 3.3V, 5V y ajustable, en este caso se usa la versión con salida fija de 3.3V.

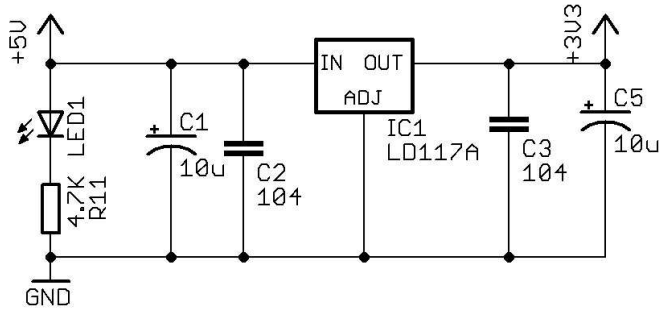
✓ Limitación de corriente y protección térmica incluida

Figura 12. Regulador LDO

6.2 DIAGRAMA ESQUEMÁTICO E INTEGRACIÓN DE LOS BLOQUES.

Después de definidos los bloques funcionales es necesario conectarlos en un único circuito que conforma el sistema de monitoreo y control. La fuente de poder principal es independiente del circuito, solo es necesario obtener los 3.3V para el módulo conversor a partir de la entrada de 5V.

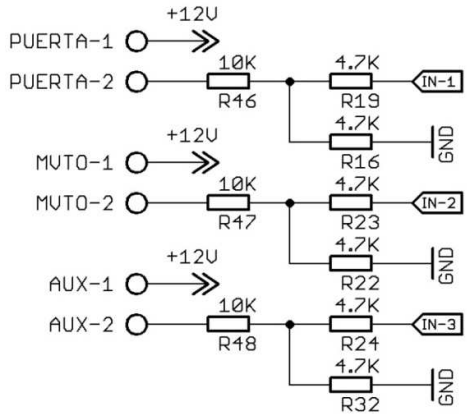
Figura 13. Esquemático de etapa de poder y filtrado



Como se observa en la **Figura 13**, el LED1 se usa como indicador de 5 V, los condensadores C1, C2, C3 y C5 ayudan a eliminar el rizado y las componentes de alta frecuencia que puedan estar en el voltaje.

6.2.1 Adecuación De Entradas. La hoja de datos del microcontrolador indica que a partir de 2.05V el microcontrolador lee la entrada como un uno lógico, por lo tanto se usa un divisor de tensión con resistencias de 10 KΩ y 4.7 KΩ como se observa en la **Figura 14**, calculado para que con 12V de entrada el microcontrolador tenga un voltaje de aproximadamente 3.8V. La resistencia adicional de 4.7 KΩ que está entre el pin de entrada del microcontrolador y el divisor de tensión es una protección contra altos voltajes. La activación de las entradas digitales se hace uniendo ambos terminales de cada entrada, si los terminales no están unidos el pin del microcontrolador queda conectado a 0V a través de las resistencias de 4.7 KΩ.

Figura 14. Esquemático de entradas



6.2.2 Salidas A Relé. El control de las bobinas de los relés se hace por medio de transistores PN2222 conectados como interruptores digitales. Para activar una salida el microcontrolador debe poner el pin correspondiente en estado lógico alto que para este caso corresponde aproximadamente 5V, con este voltaje los transistores se polarizan y permiten el paso de corriente entre los terminales colector y emisor energizando las bobinas de los relés. Las resistencias de 4.7 KΩ limitan la corriente de base de los transistores, los Led's en serie con las resistencias de 4.7 KΩ se usan como indicadores de activación de los relés, los diodos conectados en paralelo a los relés son una protección de los transistores contra el pico de voltaje inverso que generan las bobinas de los relés al ser desactivadas. Los contactos normalmente abiertos de los relés quedan disponibles en la regleta de conexión para controlar cargas en corriente alterna de hasta 10 A.

Un esquema de esta parte del circuito se puede observar en la **Figura 15**.

6.2.3 Conversión De Niveles De Voltaje. La comunicación entre el microcontrolador y el módulo conversor Wifi/Serial requiere de la conversión de niveles de voltaje de 5V a 2.8V y viceversa, aunque existen circuitos integrados que cumplen esta función es fácil implementar un circuito de conversión de nivel de voltaje con unos pocos componentes como se observa en la **Figura 16**.

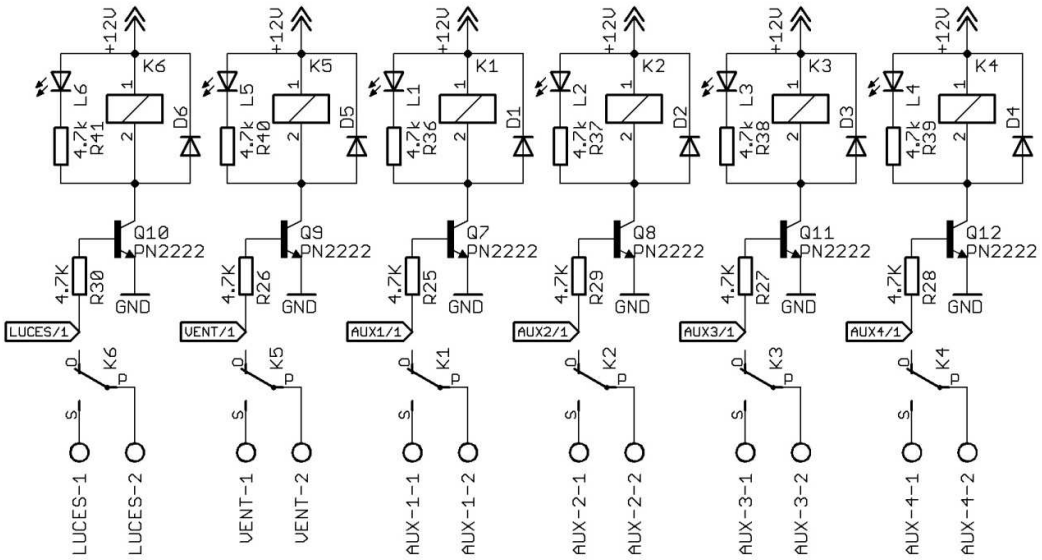
La comunicación serial se realiza principalmente por las líneas de datos TX (línea de transmisión) y RX (línea de recepción), las señales CTS (clear to send) y RTS (request to send) se usan para control de flujo de los datos seriales, no son obligatorias pero permiten un mejor control de la comunicación y previenen la pérdida de datos.

Las señales CTS_5V, TX y RESET/PGC son salidas del microcontrolador hacia el módulo conversor, los divisores de tensión con las resistencias de 1.5 KΩ y 2 KΩ bajan el voltaje de 5V del microcontrolador a 2.8V que requieren los pines del módulo conversor Wifi/Serial el cual aunque se energiza a 3.3V está limitado a 2.8V en los pines de entrada y salida.

Las señales TXD y RTS son salidas del módulo conversor Wifi/Serial hacia el microcontrolador, la conversión de 2.8V a 5V se realiza con transistores PN2222, en la señal TXD se usan dos transistores para corregir la polaridad de la señal, en el caso de RTS no es necesario ya que solo se trata de una bandera de estado del puerto que se verifica fácilmente en el programa del microcontrolador.

Las salidas GPIO-4 y GPIO-6 son usadas como monitores de estado de la conexión TCP del módulo conversor Wifi/Serial. El LED IP_OK indica que el módulo se encuentra conectado a una red inalámbrica y el LED TCP indica conexión del cliente por medio del protocolo TCP/IP.

Figura 15. Esquemático de salidas



6.2.4 Control De Intensidad.

El control de intensidad mostrado en la **Figura 17** tiene 2 partes, ambas aisladas de la red eléctrica, detección de cruce por cero y control de disparo del Triac.

La detección de cruce por cero se hace por medio de un optocoplador 4N25 que se muestra en la
Figura 18 conectado a la red eléctrica a través de las dos resistencias de 4.7 KΩ, se usan 2 resistencias en serie para que el voltaje no sobrepase el aislamiento de las resistencias y para reducir la potencia disipada por cada una de ellas. Los optocopladores son circuitos integrados que contienen un Led en su entrada y un elemento de conmutación en la salida, el cual puede ser un fototransistor o un fototriac, al encender el Led se acciona la base o la puerta fotosensible del elemento de salida y se logra la conmutación sin contacto eléctrico.

Figura 16. Esquemático de conversión de niveles de voltaje

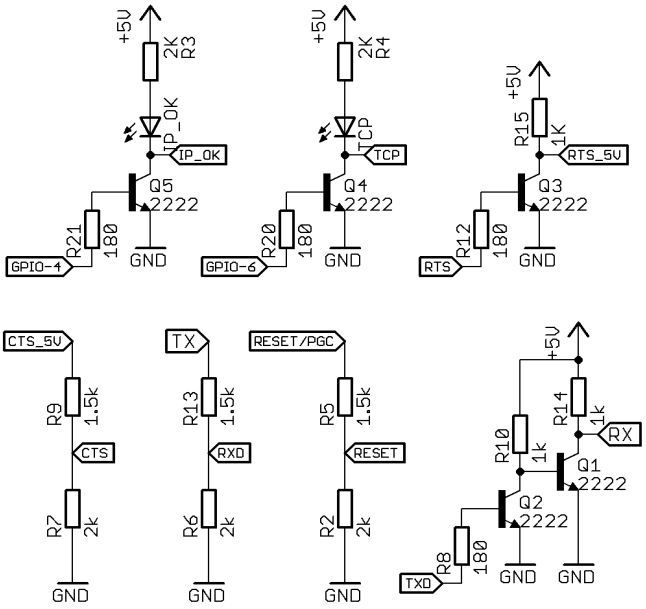
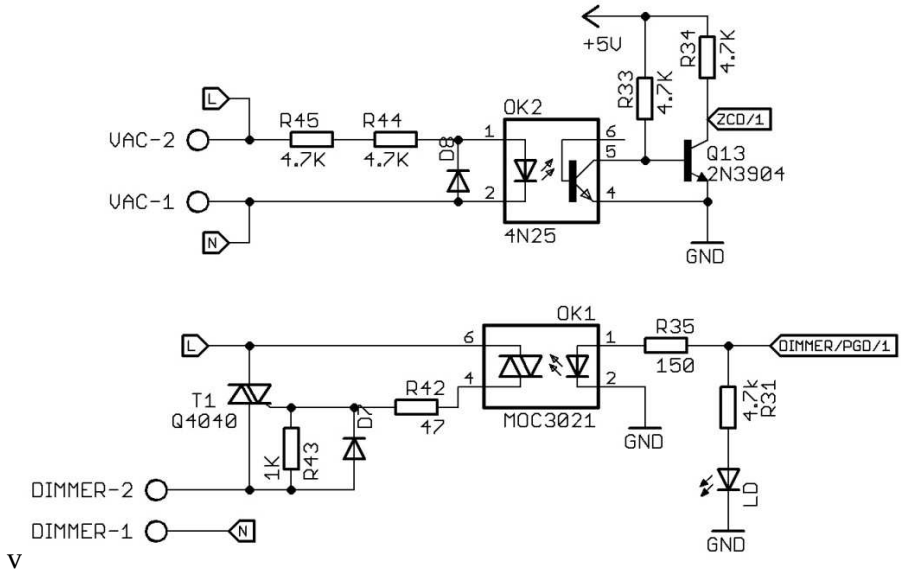


Figura 17. Esquemático del Dimmer



v

Durante el semi-ciclo positivo de la red eléctrica el Led de entrada del optocoplador está encendido, por lo tanto el transistor de salida está polarizado a través de la base fotosensible y el microcontrolador tiene un nivel lógico alto en el pin de entrada correspondiente.

En el semi-ciclo negativo de la red eléctrica el Led del optocoplador queda polarizado inversamente por lo tanto no conduce, el diodo da un camino para la corriente eléctrica y garantiza que entre terminales del Led el voltaje sea de máximo 0.7V que corresponden al voltaje umbral del diodo, de esta manera se protege el Led contra voltaje inverso. Con el Led apagado el fototransistor no está polarizado y el microcontrolador tiene un nivel lógico bajo en su pin de entrada.

Figura 18. Optoacoplador

Con este circuito la señal de corriente alterna se convierte en una señal cuadrada de amplitud 5V como se observa en la **Figura 19**, las transiciones de 0V a 5V y viceversa corresponden a los pasos por cero.

El accionamiento del Triac se realiza por medio de un optocoplador con salida de fototriac, el accionamiento del fototriac por medio del LED de entrada permite el paso de corriente a la puerta del Triac de potencia Q4040J7. El diodo D7 y la resistencia de 1 KΩ evitan encendidos no deseados del Triac.

6.2.5 Controlador. Como se observa en la **Figura 20** el microcontrolador PIC18F2550 funciona con un reloj externo de 20 MHz, el conector ICSP se usa para la programación en circuito, lo cual facilita el desarrollo y depuración de la aplicación. El sensor de temperatura se conecta al microcontrolador por medio del conector LM35. Para el puerto USB solo se requiere un condensador en el pin 14 del microcontrolador para la fuente interna del puerto USB, las resistencias de 10 KΩ se usan para detectar la conexión del puerto USB.

Figura 19. Señal cuadrada producida por el optoacoplador

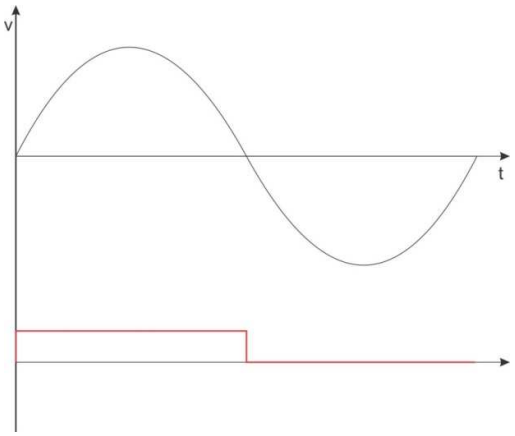
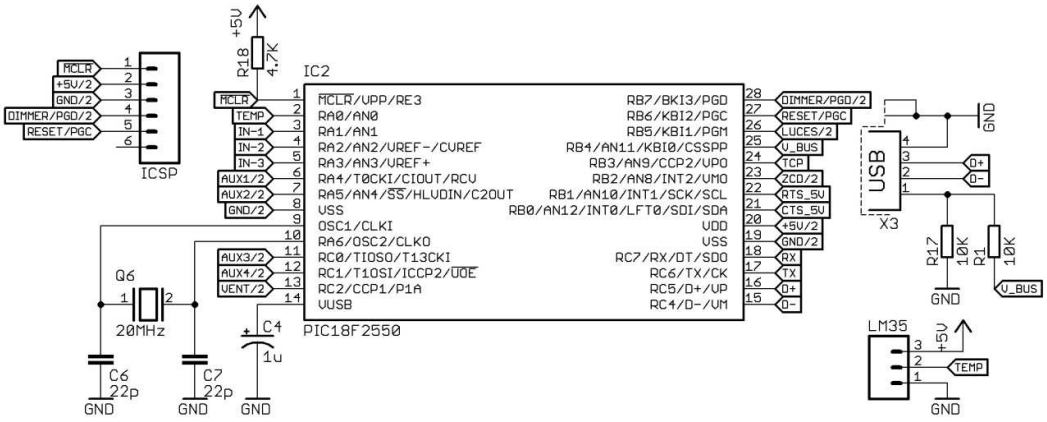


Figura 20. Esquemático Microcontrolador

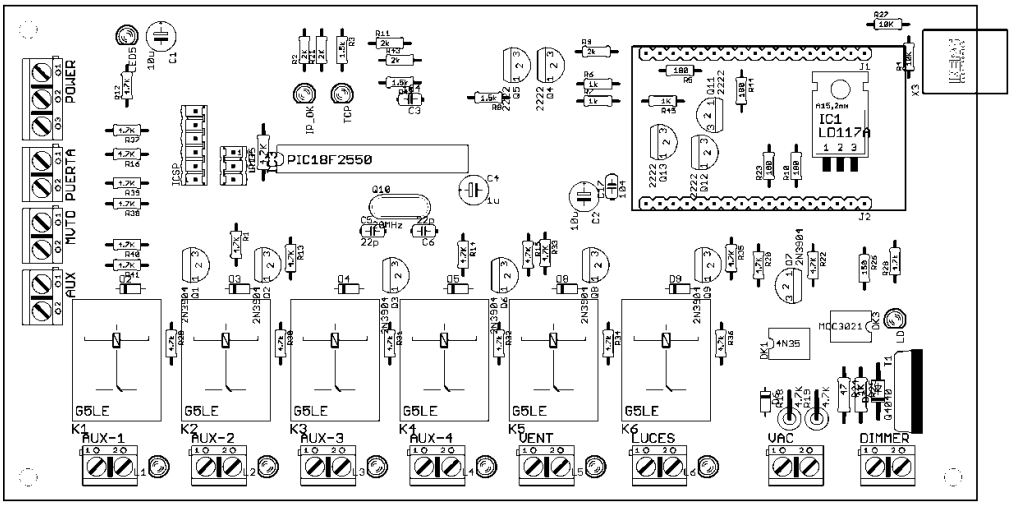


6.3 DISEÑO DEL CIRCUITO IMPRESO

El circuito impreso se diseño en el software Eagle versión 5.11, se trabajó en este software por su facilidad de uso y por su amplia librería de componentes. Para el circuito impreso se prestó especial atención en la separación de las líneas de alimentación de los componentes electrónicos y de los relevos, ya que estos últimos generan ruido eléctrico que puede afectar el correcto funcionamiento de elementos como el microcontrolador o el módulo conversor Wifi/Serial, aunque el Eagle posee una potente herramienta de autoruta todo el trabajo se realizó de manera manual para ajustar el circuito impreso a una sola cara. ***(Cadsoft)***

Una vista superior (Cara de componentes) se observa en la **Figura 21**.

Figura 21. Cara de componentes



Usando componentes de montaje de superficie se puede reducir el tamaño del circuito y mejorar presentación de la tarjeta electrónica. En la Figura 23 se muestran los planos del sistema de monitoreo y control con componentes de montaje en superficie, también se han realizado algunos cambios en el circuito.

Se han eliminado las salidas de potencia, este diseño requiere la adición de una tarjeta de potencia para el manejo de cargas en corriente alterna. La ventaja de hacerlo de esta forma es que se puede elegir el dispositivo de potencia a usar (relé o triac) y no hay limitación de corriente máxima en la tarjeta de control.

Aún se requiere la conexión de la red eléctrica a la tarjeta para la detección de cruce por cero.

El circuito queda diseñado en doble cara.

En este caso se uso la herramienta de auto-ruta del Eagle.

Una vista inferior (Cara de soldaduras) se observa en la **Figura 22**.

Los diagramas esquemáticos y las caras superior e inferior de las pistas del circuito impreso de ambas versiones se podrán encontrar en los anexos al final del documento.

Figura 22. Cara de soldaduras

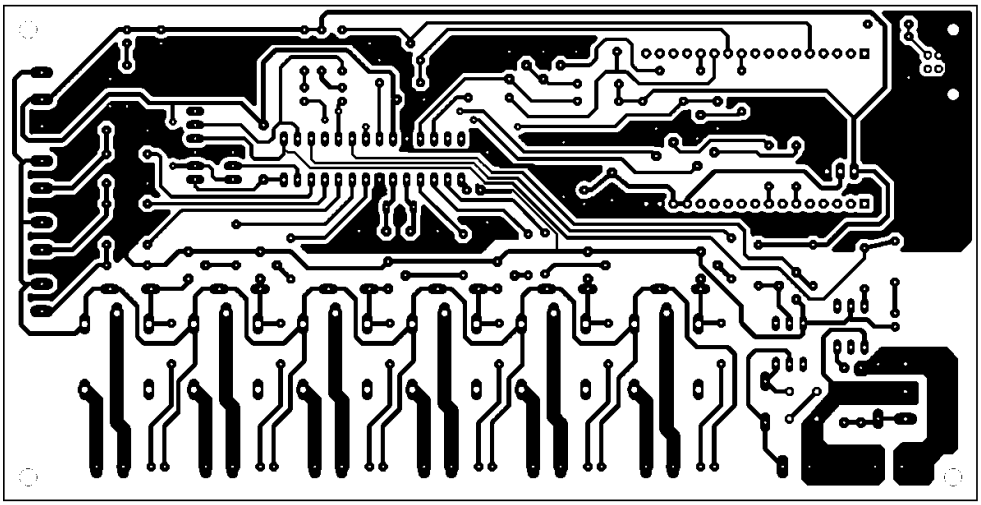
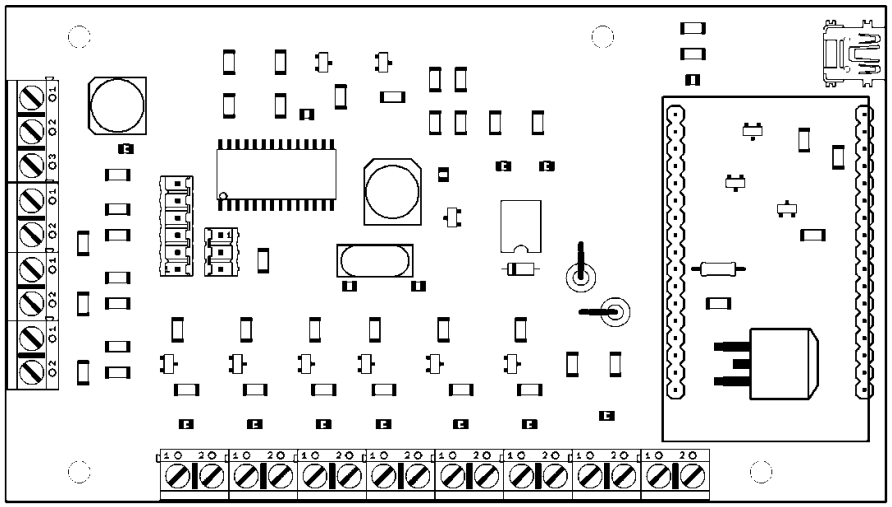


Figura 23. Montaje de superficie



6.4 ELABORACIÓN DEL PROGRAMA PARA EL MICROCONTROLADOR

El programa del microcontrolador está escrito en lenguaje C para el compilador MPLAB C18 de Microchip en la versión 3.38. Para el desarrollo del programa se deben considerar las siguientes funciones que cumple el microcontrolador en el sistema de monitoreo.

Identifica si en el momento de encender el sistema hay una conexión válida del puerto USB a un computador y si es así se inicia en modo configuración, de lo contrario se inicia en modo normal.

En modo normal:

Detecta el cruce por cero de la red eléctrica y genera los pulsos de encendido del Triac para el control de intensidad de luz incandescente según lo configurado por el usuario.

Aproximadamente cada 500 ms lee el dato de temperatura ambiente del sensor LM35.

Aproximadamente cada 1 segundo verifica la configuración del funcionamiento del ventilador y si esta en modo automático realiza la activación/desactivación de la salida correspondiente dependiendo de la temperatura.

Responde a las peticiones del protocolo HTTP recibidas desde el módulo conversor Wifi/Serial, las cuales pueden ser el envío de la página web del sistema de monitoreo y control, cambiar el estado de una salida o actualizar el estado de una entrada.

En modo de configuración actúa simplemente como conversor USB a Serial, permitiendo que los datos de configuración enviados por el computador a través del puerto USB lleguen hasta el módulo conversor Wifi/Serial por el puerto serial, o en otras palabras le agrega comunicación USB al módulo conversor Wifi/Serial.

El voltaje del puerto USB se usa para detectar la conexión USB del sistema de monitoreo y control a través del pin RB4 del microcontrolador, si este pin esta en 5V se inicia el sistema en modo de configuración, de lo contrario se inicia en modo normal.

En el modo de configuración no se realizan tareas de lectura de entradas ni control de salidas, el microcontrolador solo envía por el puerto serial los datos recibidos por el puerto USB y viceversa.

En el siguiente segmento de código se muestra la identificación por medio del microcontrolador de los modos Normal o configuración.

```
if(USB_BUS_SENSE)
{
        modo_usb = 1;
        INTCONbits.GIEL = 0;
        PIE1bits.RCIE = 0;
        USBDeviceInit();
}
else
{
        modo_usb = 0;
        INTCONbits.GIEL = 1;
        PIR1bits.RCIF = 0;
        PIE1bits.RCIE = 1;
}
```

En el modo normal el funcionamiento es más complejo, ya que se deben verificar los datos recibidos por el puerto serial para determinar el tipo de petición del protocolo HTTP y responder de la manera adecuada.

El protocolo HTTP trabaja con cadenas de caracteres, por lo tanto la página web almacenada en la memoria del microcontrolador no es más que un arreglo de variables tipo carácter que corresponden con el código html de la página. Las respuestas a solicitudes para actualizar el estado de las entradas y la confirmación a las órdenes ejecutadas también se realizan con cadenas de caracteres.

La petición del protocolo HTTP para el envío del código de la página web se produce cuando el usuario introduce en el navegador la dirección IP del módulo conversor Wifi/Serial y el puerto de conexión, el microcontrolador responde con la información que el usuario debe ver.

Un ejemplo de un Fragmento de código HTML almacenado en el microcontrolador es:

```
"<p>Puerta principal<label><input
type=\"text\"name=\"tx1\"id=\"tx1\"
value=\"*D\"readonly=\"readonly\"/></label></p>\r\n"
```

La conexión serial entre el microcontrolador y el módulo conversor Wifi/Serial se establece a una velocidad de 115200 bps, el hardware serial del microcontrolador permite la configuración para generar una interrupción cuando hay un byte disponible en el buffer de entrada, de esta manera el programa no tiene que estar leyendo continuamente el puerto sino que atiende la interrupción en el momento en que se produzca.

La interrupción del puerto serial está configurada como de alta prioridad, ya que la respuesta a las peticiones HTTP deben ser inmediatas, esto quiere decir que cuando se están verificando los datos del puerto serial todos los posibles eventos del microcontrolador quedan en espera hasta que se termine de ejecutar la tarea.

El código de interrupción de alta prioridad realiza las tareas del puerto USB en el modo configuración y atienden las interrupciones del puerto serial en el modo normal.

Fragmento inicial del código de interrupción de alta prioridad:

```
#pragma interrupt YourHighPriorityISRCode

void YourHighPriorityISRCode()
{
        #if defined(USB_INTERRUPT)
        if(modo_usb)
                USBDeviceTasks();
        if (RCSTAbits.OERR)         {
```

```
            RCSTAbits.CREN = 0;  // reset the port
            c = RCREG;
            RCSTAbits.CREN = 1;  // and keep going.
        }
        #endif

        /*Interrupciòn serial: solo cuando no hay USB*/
        if((!modo_usb) && (PIE1bits.RCIE) && (PIR1bits.RCIF))
        {
```

El control de intensidad de luz incandescente parte del cruce por cero de la red eléctrica, el cual gracias al circuito de detección de cruce por cero no es más que el cambio de nivel o flanco de subida y bajada de una señal cuadrada. El microcontrolador detecta los flancos de esta señal y dependiendo de la intensidad de luz configurada determina el tiempo que debe transcurrir antes de la activación del Triac. La detección de flancos se realiza con una interrupción de baja prioridad la cual puede ser puesta en espera si se deben atender los datos del puerto serial

El código de detección de cruce por cero se muestra a continuación.

```
#pragma interruptlow YourLowPriorityISRCode
void YourLowPriorityISRCode()
{
        /*cruce por cero: Inicia conteo para activacion del Triac*/
        if((INTCON3bits.INT2IE) && (INTCON3bits.INT2IF))
        {
                INTCON2bits.INTEDG2 = !INTCON2bits.INTEDG2;       //Cambia    el
flanco de interrupción
                /*Habilita interrupción para el tiempo del dimmer*/
                if(intensidad == 10)    //máxima intensidad
                {
                        DIMMER = 1;
                }
                else if(intensidad == 0)          //mínima intensidad = apagado
                {
                        DIMMER = 0;
                }
                else     //niveles intermedios
                {
                        PIR1bits.TMR1IF = 0;
                        WriteTimer1(dimmer[intensidad]);
                        PIE1bits.TMR1IE = 1;
                }
                /*Borra bandera de interrupción*/
                INTCON3bits.INT2IF = 0;
        }
```

La lectura de temperatura se realiza mediante el conversor análogo a digital interno del microcontrolador, el cual con una resolución de 10 bits representa el voltaje de entrada en el rango de 0V a 5V como una variable de tipo entero que puede tomar valores entre 0 y 1023.

Para calcular el valor de salida del conversor análogo a digital se usa la siguiente ecuación:

$$Valor\ ADC = 2)$$

Con:

$Vmax = 5V$ $Vmin = 0V$ $Vin = Voltaje\ de\ entrada\ del\ conversor$

El sensor de temperatura da una salida lineal de 10 mV/˚C, por lo tanto para una temperatura de 100˚C el voltaje de salida del sensor es de 1V, usando la ecuación anterior este voltaje es representado por el conversor análogo a digital con el número 204 por lo tanto se debe dividir el valor del conversor A/D por aproximadamente 2.048 para obtener el dato de la temperatura. Finalmente se debe convertir el dato decimal de la temperatura leída en caracteres alfanuméricos para ser enviados en la trama de datos serial.

Código para lectura de temperatura:

```
/*lectura de temperatura*/
if(t_temperatura)
{
        t_temperatura = 0;
        temperatura = (unsigned char)((float)((LeerADC(0)+1)/2.046));
        btoa(temperatura, s_temperatura);
}
```

El ventilador tiene la opción de funcionar automáticamente dependiendo de la temperatura, en este modo se verifica la temperatura ambiente leída del sensor LM35 contra una temperatura de referencia programada, si la temperatura ambiente sobrepasa el valor de referencia se activa la salida del ventilador, en caso contrario la salida del ventilador permanece apagada.

```
/*Control del ventilador*/
if(t_ventilador)
{
        t_ventilador = 0;
        switch(control_ventilador)
        {
```

```
                case 0: //apagado
                        VENTILADOR = 0;
                        break;
                case 1: //prendido
                        VENTILADOR = 1;
                        break;
                case 2: //automatico
                        if(temperatura > set_temperatura)
                                VENTILADOR = 1;
                        else if(temperatura < set_temperatura)
                                VENTILADOR = 0;
                        break;
                default://en caso de dato no valido     se deja en modo automatico
                        control_ventilador = 2;
                        break;
        }
}
```

6.5 SIMULACIÓN

La simulación del circuito se realizó con el software Proteus de Labcenter Electronics como se observa en la **Figura 24** el cual permite la simulación del código fuente de algunas familias de microcontroladores incluyendo fabricantes como Microchip, Atmel, Intel y otros. El Proteus se aprovechó para comprobar el funcionamiento del programa especialmente en la comunicación serial con el módulo conversor Wifi/Serial y en las respuestas a las peticiones del protocolo HTTP.

En la

Figura 25 se observa una captura de pantalla de la simulación del circuito.

Para realizar la simulación se usaron otras herramientas de software como Virtual serial ports emulator que permite crear puertos virtuales, tanto seriales como puerto TCP.

Esta herramienta se uso para crear un puerto serial virtual conectado al puerto TCP 5555, de esta manera al ingresar en el navegador la dirección http://localhost:5555 se simulaba el comportamiento del conversor Wifi/Serial. El puerto serial virtual se conectó al microcontrolador en la simulación en Proteus.

El programa se puede descargar desde la siguiente dirección: http://www.eterlogic.com/Downloads.html

Docklight es un software de terminal serial avanzado que permite almacenar múltiples cadenas de caracteres para ser enviadas posteriormente al dispositivo conectado al puerto. Con esta herramienta se realizaron las primeras pruebas de funcionamiento y configuración del conversor Wifi/Serial.

Figura 24. Simulación en Proteus

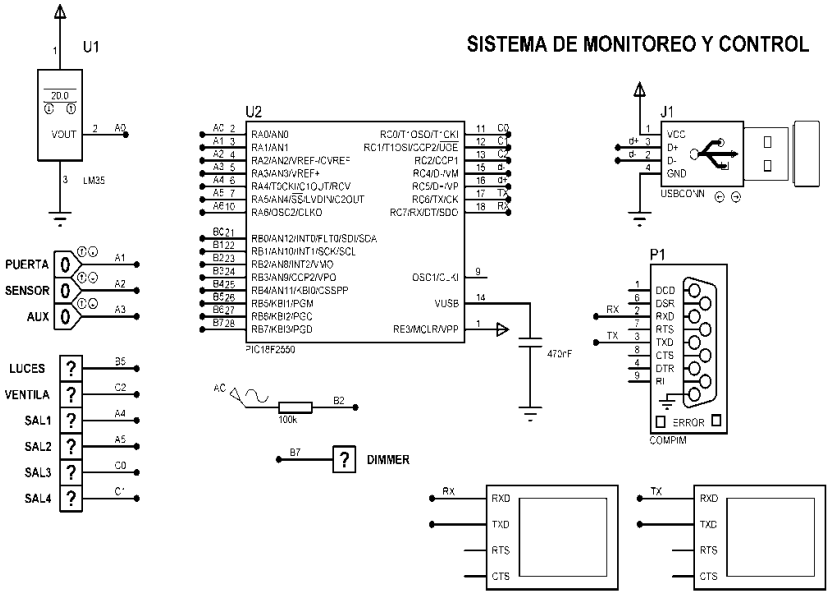
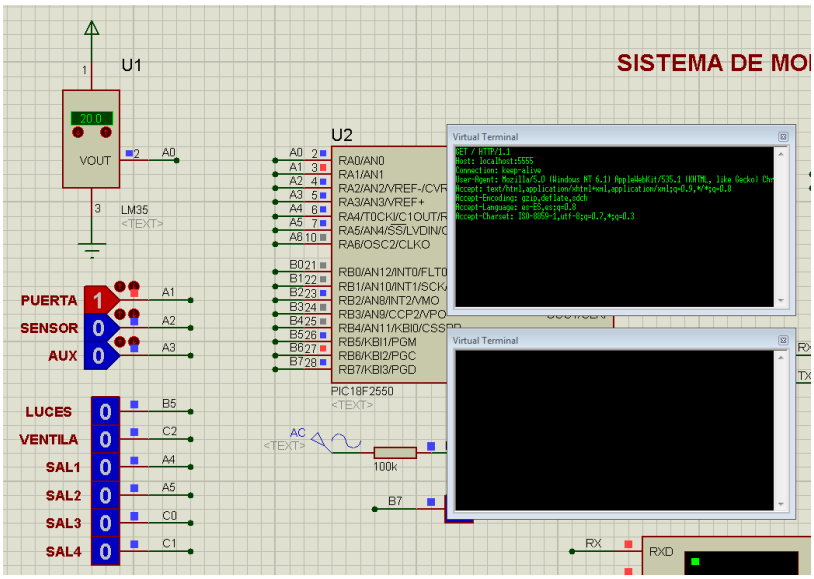


Figura 25. Simulación del circuito



La versión de demostración del programa se puede descargar desde el enlace: http://www.docklight.de/download_en.html

6.6 INTERFACE HUMANA

6.6.1 Página Web. El desarrollo de la página web se realizó teniendo en cuenta que debía ser una página que permitiera de manera amigable para el usuario la lectura de los sensores conectados a las entradas del sistema de monitoreo y el control de las cargas conectados a las salidas a relé, pero además de esto la página debía se lo suficientemente simple para poder ser incluida en la memoria del microcontrolador.

La página está escrita en lenguaje HTML y Javascript, HTML se usa principalmente para dar formato a la página y en general para crear todo lo que el usuario ve, Javascript es un lenguaje de programación que permite incrustar código en las páginas web para que sea ejecutado en el navegador del cliente, de esta manera se libera trabajo al servidor.

Un ejemplo de un fragmento de código Javascript se observa a continuación.

```
<script type="text/javascript">
function enviar(control){
	var peticion;
	if(window.XMLHttpRequest){
		peticion = new XMLHttpRequest();
	}
	else if(window.ActiveXObject){
		peticion = new ActiveXObject('Microsoft.XMLHTTP');
	}
	peticion.onreadystatechange = leerDatos;
	peticion.open('GET',control);
	peticion.send();
}

function leerDatos(){
	if(peticion.readyState==4){
		if(peticion.status==200){
			switch(control){
				case '1': alert(peticion.responseText);break
				default: alert(peticion.responseText);break
			}
		}
	}
}
</script>
```

Anteriormente para cambiar la información de una parte de la página web era necesario cargar nuevamente toda la página, lo que hacia el proceso lento y molesto para el usuario, ahora gracias a AJAX (Asynchronous JavaScript And XML) es posible actualizar información sin necesidad de cargar nuevamente toda la página, esta tecnología es usada por ejemplo para mostrar las fotos en páginas como facebook en las cuales se actualiza la imagen sin cambiar siquiera la dirección de la página web.

La implementación de AJAX se hace mediante el objeto XMLHttpRequest() el cual hace las peticiones al microcontrolador en los siguientes casos:

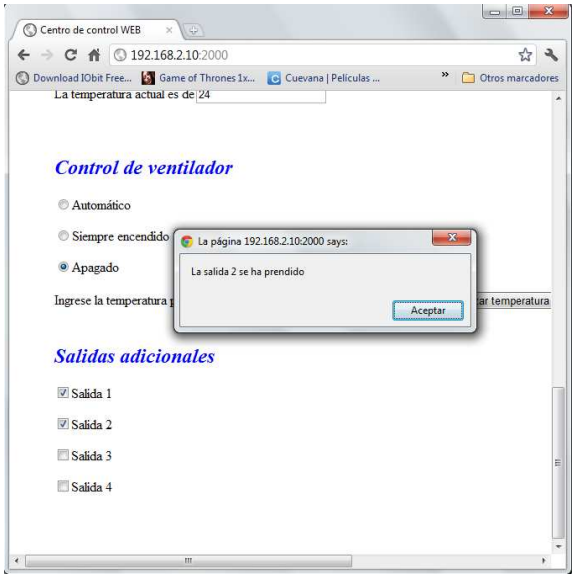
En respuesta a las acciones del usuario, por ejemplo dar click en un botón o activar una salida. Las acciones del usuario son comunicadas al microcontrolador a través de las peticiones realizadas con AJAX.

Periódicamente cada 20 segundos aún sin que el usuario realice ninguna acción. Esta petición es de actualización general de las entradas y salidas. El tiempo se fijo en 20 segundos para no generar mucho tráfico entre cliente y servidor.

Las respuestas del microcontrolador son mostradas al usuario con mensajes emergentes en la página web como se observa en la
Figura 26.

El código completo de la página Web se adjunta en los anexos al final del documento.

Figura 26. Pagina Web



6.6.2 Utilidad De Configuración Wifi. La configuración del módulo Wifi/Serial se realiza por medio de comandos de texto, si bien es posible ajustar todos los parámetros de configuración desde un programa de terminal serial no es lo más práctico para el usuario. Por esta razón se ha desarrollado un programa de utilidad de configuración que permite de una manera amigable ajustar el sistema de monitoreo y control para que se conecte automáticamente a la red inalámbrica del usuario como se observa en la **Figura 27.**

En el modo de configuración el sistema de monitoreo y control se identifica en el sistema operativo como un puerto serial virtual COMX (la X corresponde al número del puerto asignado por el sistema operativo), la primera vez que se conecta se debe instalar el driver del dispositivo suministrado por microchip. La utilidad de configuración permite escanear en busca de redes inalámbricas y configurar la contraseña de red y los parámetros de dirección IP.

El programa de utilidad de configuración se desarrollo en lenguaje C# en el entorno Visual Studio 2010 Express el cual puede ser descargado de manera gratuita directamente desde Microsoft. La comunicación con los puertos seriales se realiza mediante el objeto SerialPort, del cual la propiedad ReadExisting() lee los datos disponibles en el buffer de entrada del puerto y la propiedad Write() escribe en el buffer de salida. ***(Microsoft)***

La utilidad de configuración Wifi traduce las acciones del usuario a los comandos de texto que deben enviarse al módulo conversor Wifi/Serial.

Se adjunta el código fuente del programa de utilidad de configuración Wifi al final del documento en los anexos.

6.6.3 Instalación Del Sistema De Monitoreo Y Control.

El sistema de monitoreo y control se instaló en una caja plástica con la fuente de poder, interruptores de control de encendido y luces piloto para verificar la activación de las salidas, también se instaló un sensor tipo Reed switch para detectar la apertura de la puerta de la caja, tal como se observa en la **Figura 28** y la **Figura 30.**

Figura 27. Utilidad de configuración Wifi

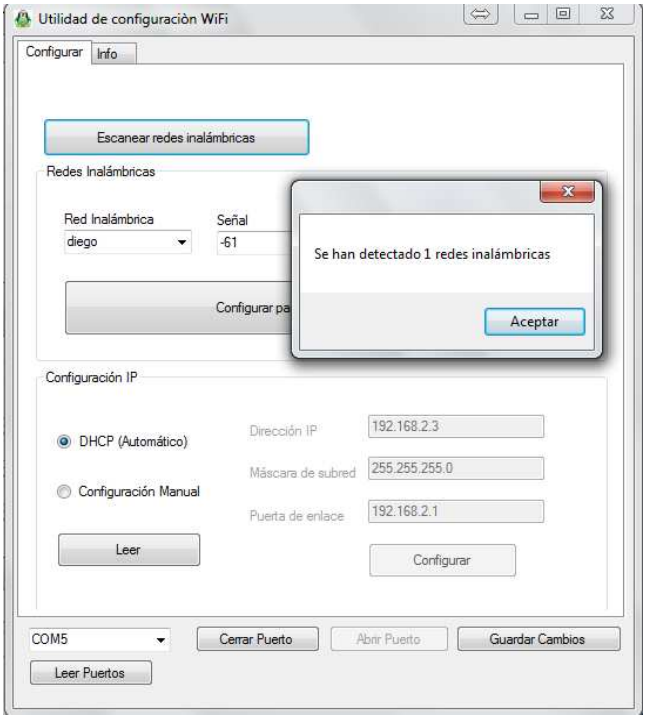


Figura 28. Sistema de monitoreo y control de variables remotas a través de internet

En la **Figura 29** se observa el sensor tipo Reed switch.

Figura 29. Sensor tipo Reed Switch

Los interruptores tienen las siguientes funciones.

El Interruptor Encendido controla la alimentación general del sistema.

El interruptor Salidas controla la alimentación de las salidas a 110 Vac, el apagar este interruptor solo apaga las cargas y el sistema sigue funcionando normalmente.

Cada interruptor tiene asignada una luz piloto de color amarillo que indica la correcta activación del interruptor.

La activación y desactivación de las salidas a relé se puede verificar por medio de los pilotos azules.

Para la configuración de la red se dejo una perforación que permite la conexión del cable USB y una perforación adicional para los cables de entrada de potencia.

Figura 30. Vista interior del prototipo

7 CONCLUSIONES

✓ Una de las principales limitaciones que han hecho que la domotica no haya tenido un crecimiento acelerado como algunas otras tecnologías de su época radica en el elevado costo de los equipos y su posterior instalación, como resultado de este proyecto se ha obtenido un dispositivo con un costo de materia prima de alrededor de $190.000 el cual aplicando una rentabilidad adecuada no superaría un precio de $500.000, valor que no es alto en comparación con los beneficios que el dispositivo presta.

Esto nos lleva a concluir que actualmente el sector de desarrollo electrónico colombiano está en capacidad de ofrecer alternativas tecnológicas a bajo costo y alta funcionalidad.

✓ El resultado de este proyecto además permite determinar la viabilidad de elaboración y desarrollo de aplicaciones domoticas con protocolos de comunicación y tecnologías que en un principio no fueron pensadas para estas aplicaciones especificas.

✓ El desarrollo tecnológico alcanzado en los dispositivos programables (microcontroladores) permite que al día de hoy se puedan realizar aplicaciones de adquisición de datos por página Web con dispositivos de un valor comercial que oscila entre los $15.000 y los $20.000.

✓ La implementación presentada en este proyecto representa una de las muchas alternativas que ofrecen las nuevas tecnologías de la información y la comunicación lo permite a los usuarios de dispositivos como este tener acceso a la información casi que en tiempo real generando estrategias de toma de decisiones mucho más oportunas y acertadas.

✓ Posterior al diseño y fabricación del dispositivo se pudo determinar que la instalación de este equipo en un lugar geográfico apartado con acceso a una red inalámbrica de internet permite el manejo y control de variables sin necesidad de trasladarse hasta dicho, lugar lo que pudiese mejorar significativamente las condiciones de calidad de vida y seguridad de un sin número de personas quienes por su ritmo de vida tienen que dejar completamente solos sus hogares durante el día o sus oficinas durante la noche.

8 BIBLIOGRAFIA

Basics Internet and Web Web Developers Notes [Online]. - 2011. - http://www.webdevelopersnotes.com/basics/languages_on_the_internet.php3.

Cadsoft Eagle [Online]. - www.cadsoftusa.com/downloads/documentation/?language=en.

Carlos alvial Udec [Online]. - 2011. - http://www2.udec.cl/~carlosalvial/domotica/pages/que_es.htm#up_que es.

Cruz Omar Esteban Siguencia Normas 802.11a, 802.11b, 802.11g [Book]. - Cuenca : Universidad Politecnica Salesiana, 2006.

Domoprac Domoprac [Online]. - http://www.domoprac.com/protocolos-de-comunicación-y-sistemas-domoticos/protocolos-de-red-tipos-y-utilidades.html.

Gislason Drew ZIGBEE WIRELESS NETWORKING [Book]. - 2011.

Gonzalez Guillermo David Herrero Conexion al puerto USB mediante un microcontrolador [Report]. - Bejar : Universidad de Salamanca, 2011.

microchip microchip [Online]. - www.microchip.com/en_US/family/8bit/index.html.

microchip microchip [Online]. - www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010280.

Microsoft Visual Studio Express [Online]. - msdn.microsoft.com/es-es/ff380143.

rovingnetworks rovingnetworks [Online]. - www.rovingnetworks.com/wifly-gsx.php.

Shoch Jhon F. Communications of the ACM [Journal] // Magazine Communications of the ACM. - 1980. - p. 12.

Uberbin AJAX un nuevo acercamiento a Aplicaciones Web [Online]. - 2011. - http://www.uberbin.net/archivos/internet/ajax-un-nuevo-acercamiento-a-aplicaciones-web.php.

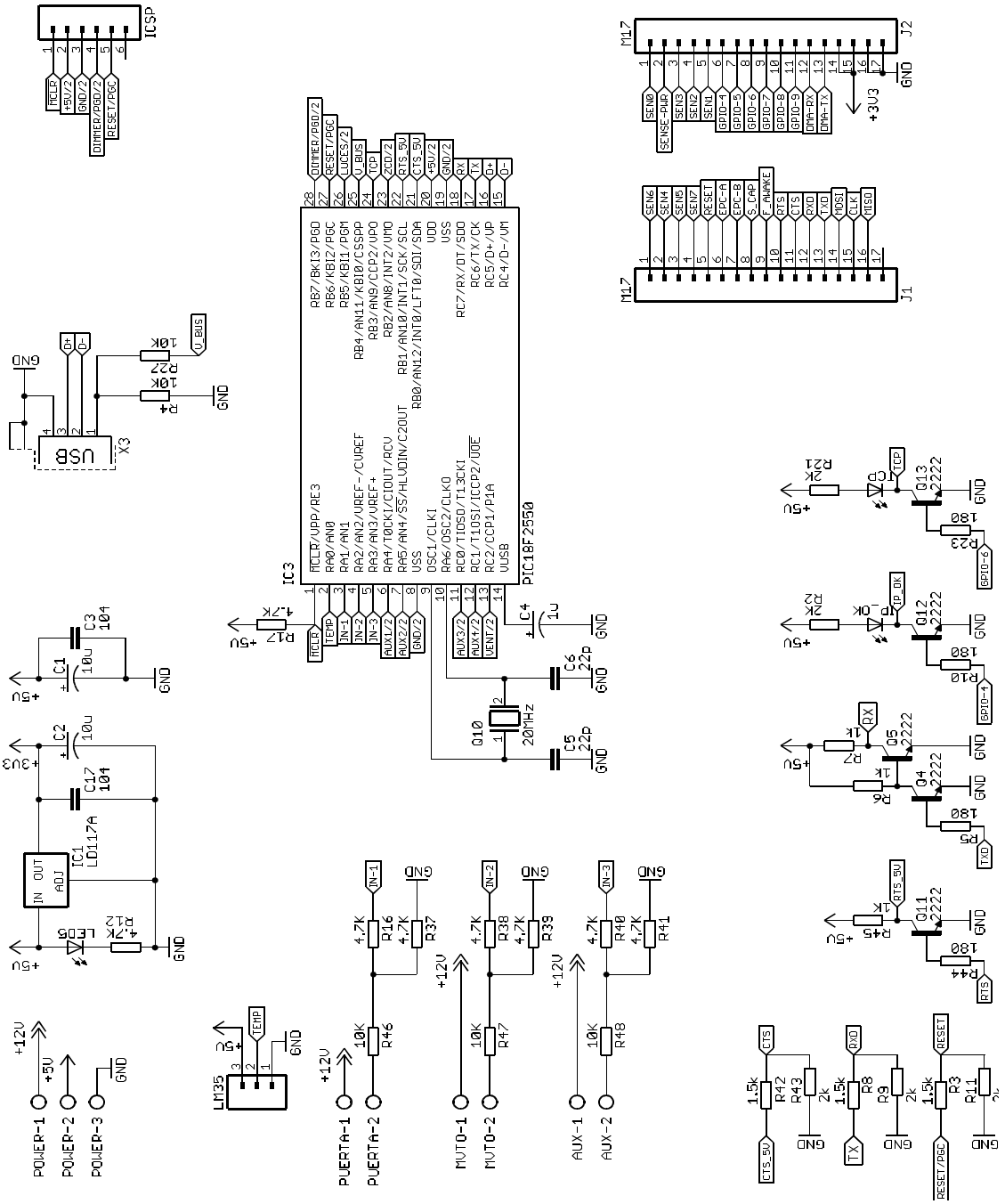
USB Universal Serial Bus [Online]. - 2011. - http://www.usb.org.

Vasquez Walter Edwin Barriga Tecnologias inalambricas de corto alcance [Book]. - Cuenca : Universidad Politecnica Salesiana, 2006.

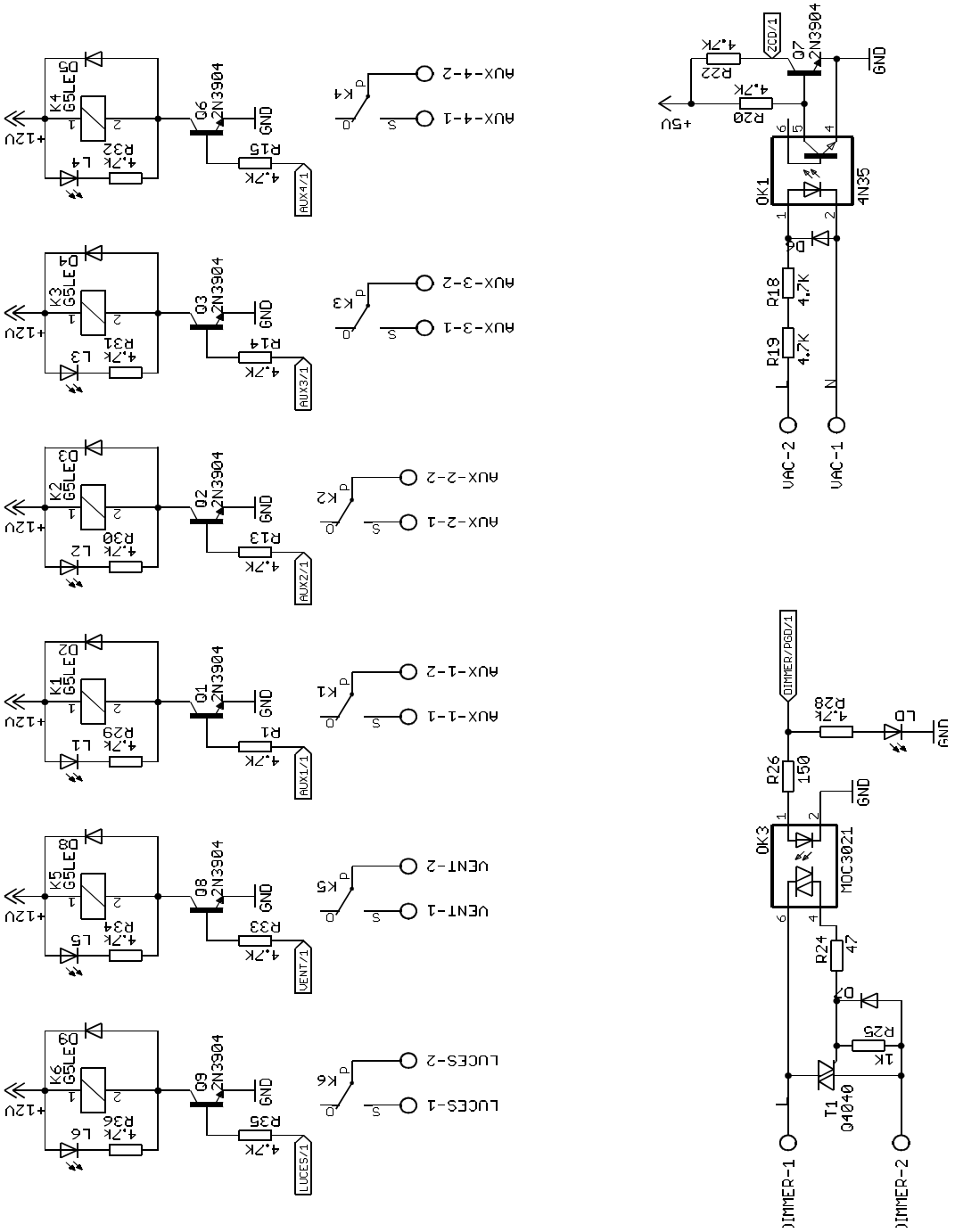
W3C HyperText Markup Language [Online]. - http://www.w3.org/html/.

W3C Hypertext Transfer Protocol [Online]. - 2011. - 2011. - http://www.w3.org/Protocols/.

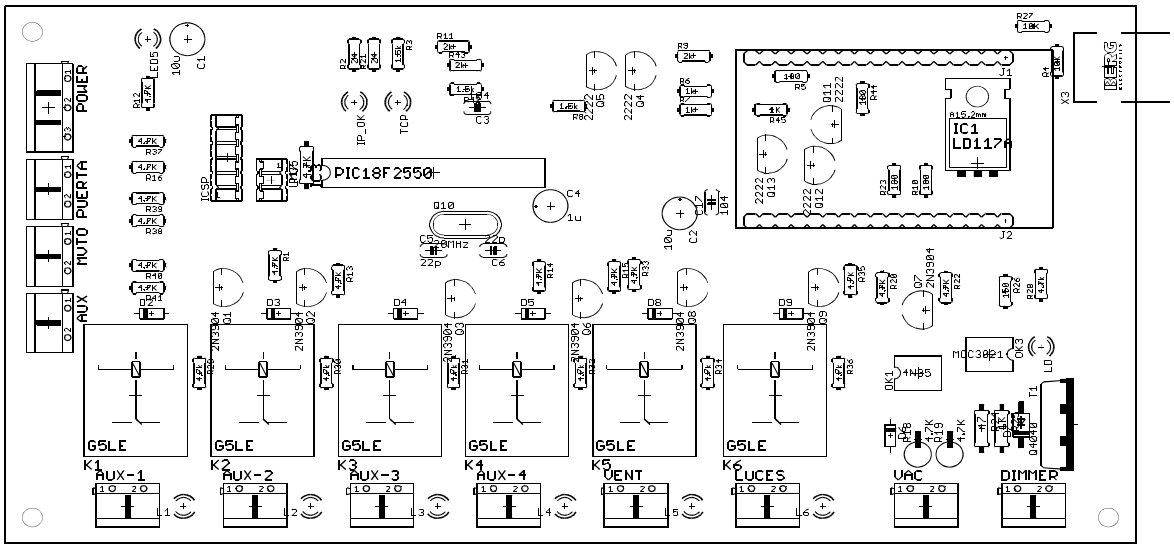
ANEXO A: Diagrama esquemático del circuito

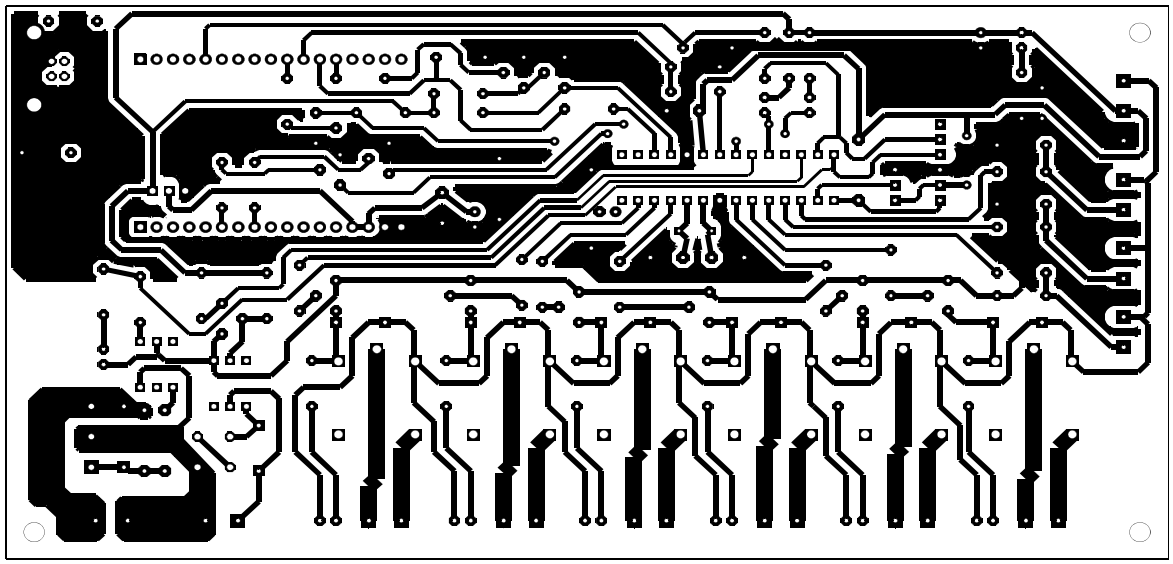


ANEXO A: Diagrama esquemático del circuito

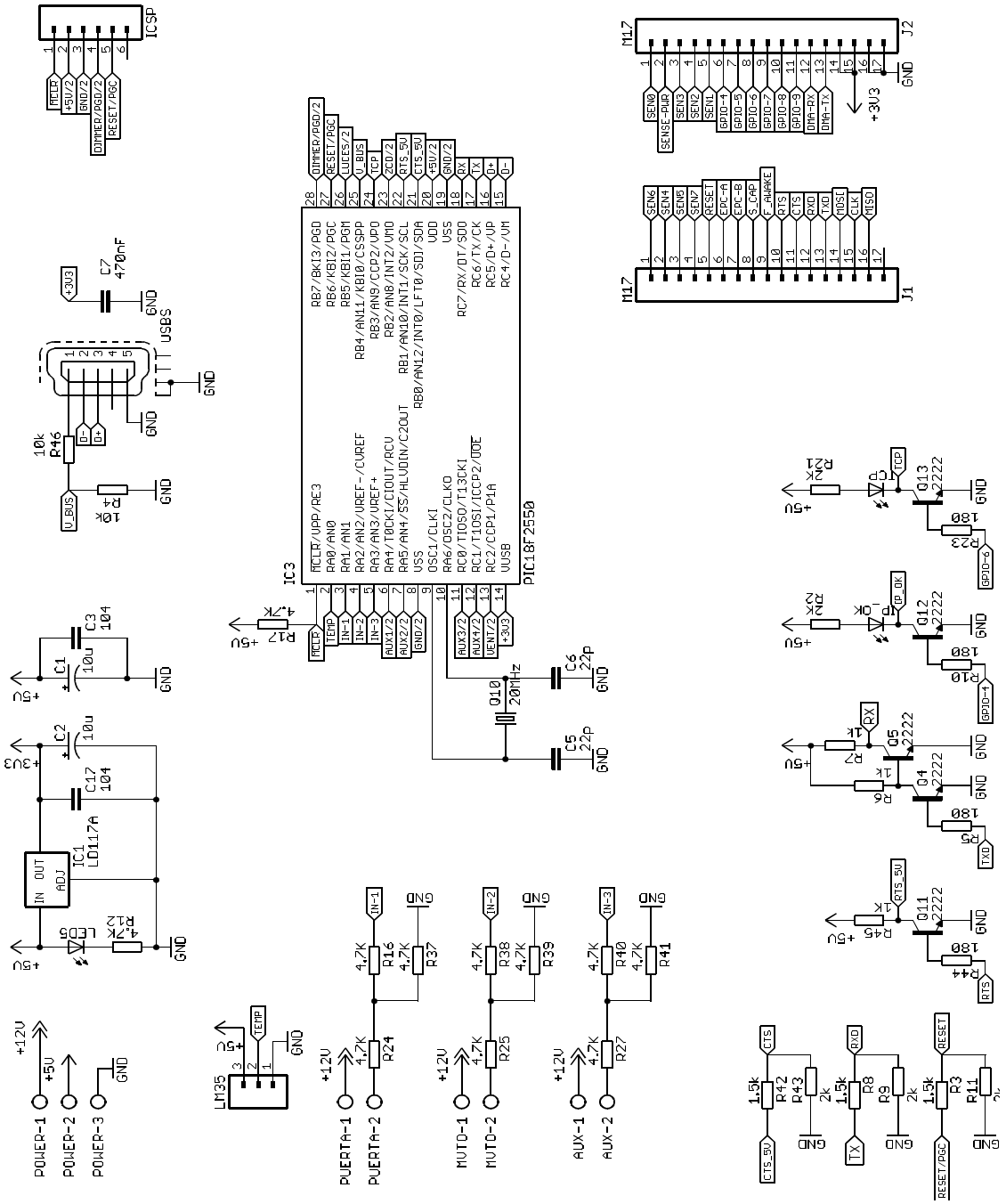


ANEXO B: Circuito impreso

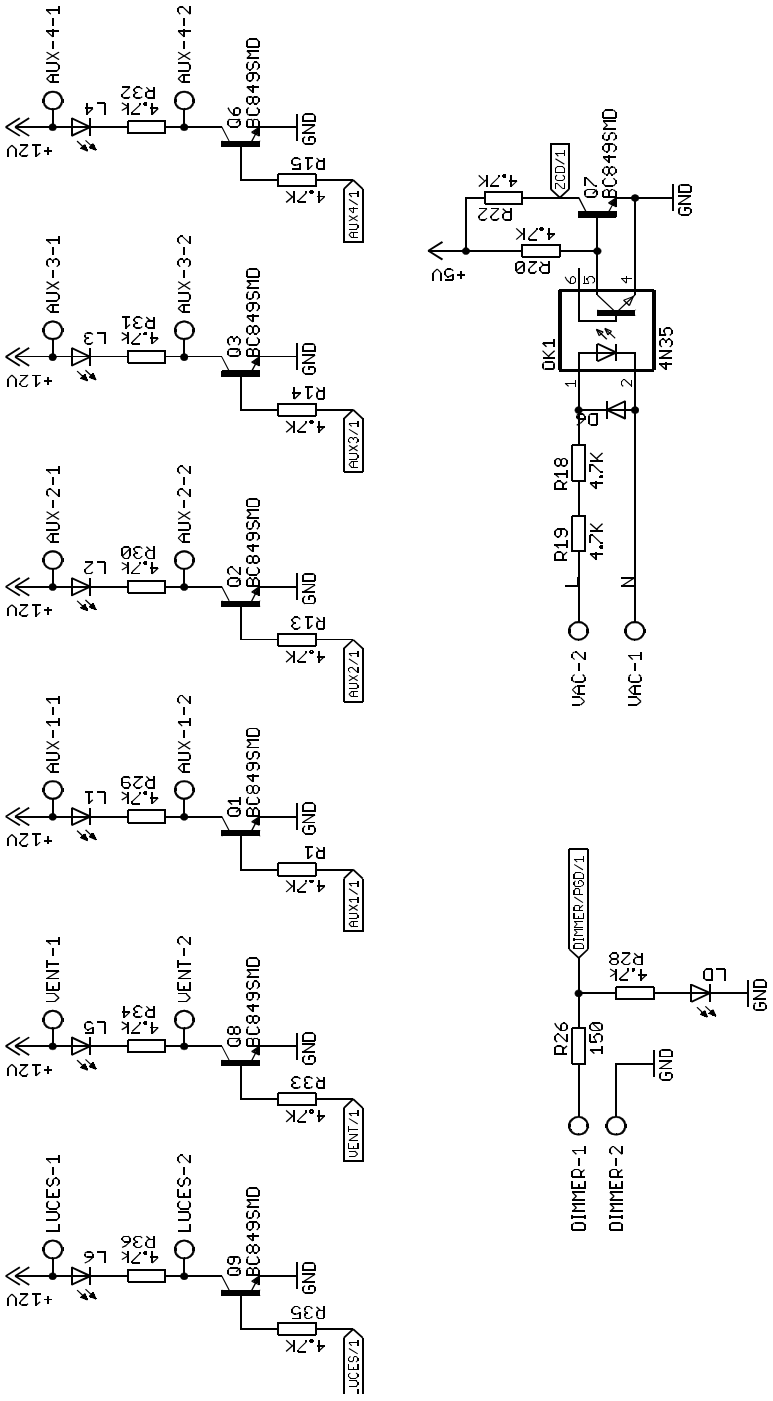




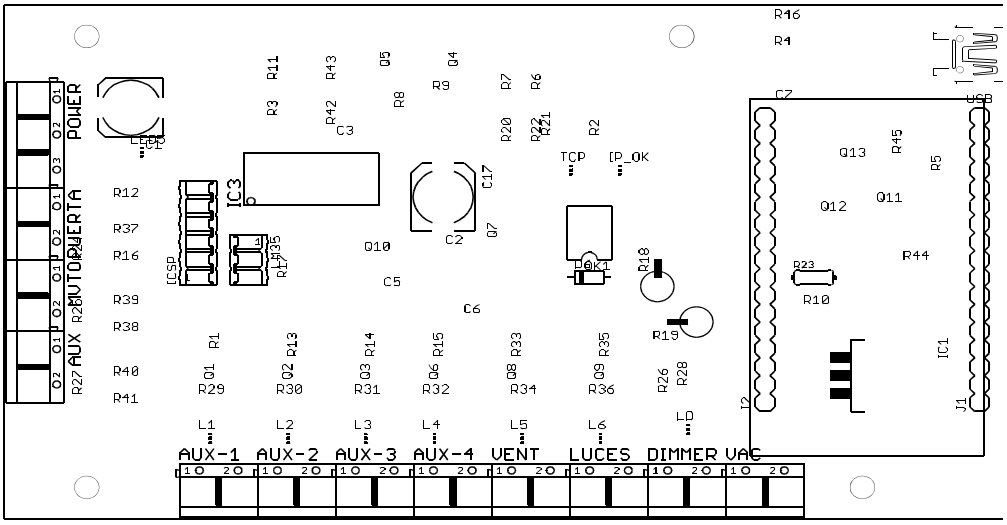
ANEXO C: Diagrama esquemático con componentes de montaje de superficie

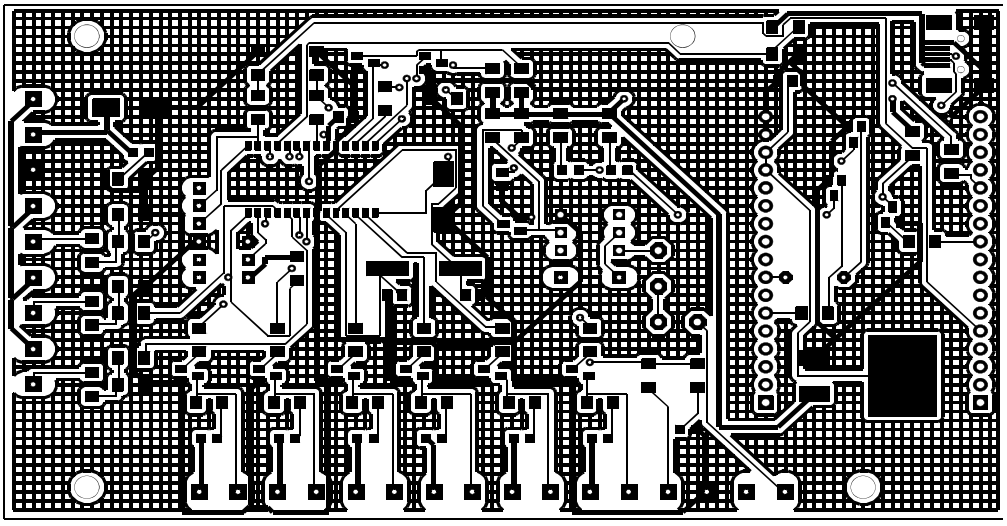


ANEXO C: Diagrama esquemático con componentes de montaje de superficie

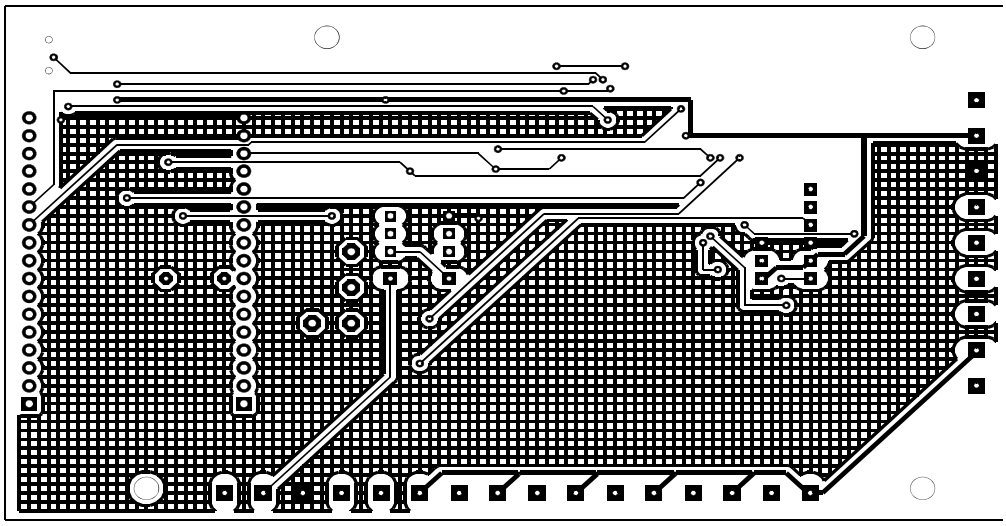


ANEXO D: Circuito impreso con componentes de montaje de superficie





ANEXO D: Circuito impreso con componentes de montaje de superficie



ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
/**USB I N C L U D E S **********************************************************/
#include "./usb/HardwareProfile.h"
#include "./usb/GenericTypeDefs.h"
#include "./usb/usb.h"
#include "./usb/usb_function_cdc.h"
#include "./usb/Compiler.h"
#include "./usb/usb_config.h"
#include "./usb/usb_device.h"
#include "./usb/usb.h"
#include "./usb/usb_callback.h"
#include "stdio.h"
#include "stdio.h"

/** V A R I A B L E S  N O  I N I C I A L I Z A D A S****/
#pragma udata
char USB_In_Buffer[64];
char USB_Out_Buffer[64];

//Variables de tx serial
unsigned char numBytesRead, NextUSBOut, LastRS232Out, RS232cp;
char s_temperatura[4];
char c, accion_servidor, pulsador;

BOOL RS232_Out_Data_Rdy, RS232_Out_Data;

#pragma romdata
const rom char get[] = "GET /";
const rom char post[] = "POST";
const rom char fin[] = "\r\n\r\n";

/** V A R I A B L E S  I N I C I A L I Z A D A S****/
#pragma idata

unsigned char cont_config = 0;
char temp_config[10];
char enter = "\r\n";

unsigned char intensidad = 0, cont_rx = 0;
unsigned char cont_tiempo = 0;
unsigned char temperatura = 0;
unsigned char set_temperatura = 28;
char s_set_temperatura[4] = "28";
unsigned char temp = 0;
unsigned char control_ventilador = 0;
unsigned int dimmer[11] = {Dimmer0, Dimmer10, Dimmer20, Dimmer30, Dimmer40, Dimmer50,
Dimmer60, Dimmer70, Dimmer80, Dimmer90, Dimmer100};
/*Cadena de respesta para refrescar info*/
char s_enviar[] = "CNNA000A00AAAA\r";
/* Las letras corresponde a:
0. Estado de la puerta: C = Cerrada, A = abierta
1. Estado del sensor de movimiento: N = Normal, A = Activado
2. Estado del sensor adicional: N = Normal, A = Activado
3. Estado de la salida de iluminacion on/off: A = Apagada, P = Prendida
4. Intensidad de la luz incandescente: 0 = Minima .... X = maxima
5 y 6. Temperatura en grados
7. Ventilador: A = Auto, E = Encendido, O = Apagado
8 y 9. Temperatura para activar ventilador en grados
10. Salida 1: A = Apagada, P = Prendida
11. Salida 2: A = Apagada, P = Prendida
12. Salida 3: A = Apagada, P = Prendida
13. Salida 4: A = Apagada, P = Prendida
*/

/*banderas de estado*/
UINT8_BITS flags1 = 0, flags2 = 0;
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
#define modo_usb                    flags1.bits.b0 //define el modo de funcionamiento
#define iniciando                   flags1.bits.b1 //
#define t_100ms                     flags1.bits.b2 //tiempo base para las tareas varias
#define t_temperatura         flags1.bits.b3 //1 segundo para lectura de temperatura
#define cliente                     flags1.bits.b4
#define fin_rx                      flags1.bits.b5
#define pulsador_remoto             flags1.bits.b6
#define accion                      flags1.bits.b7
#define t_ventilador          flags2.bits.b0
#define tx_usb                      flags2.bits.b1
#define set_temperatura_ok    flags2.bits.b2

/** P R I V A T E  P R O T O T Y P E S ***************************************/
static void InitializeSystem(void);
void ProcessIO(void);
void USBDeviceTasks(void);
void USBCBSendResume(void);
void BlinkUSBStatus(void);
void UserInit(void);
void YourHighPriorityISRCode();
void YourLowPriorityISRCode();
unsigned int LeerADC(unsigned char Canal);

//user includes
#include "config.h"
#include "user/UserDelays.h"
#include "user/pagina.h"
#include "user/UserUsart.h"
#include "user/ScriptReloj.h"
#include <usart.h>
#include <timers.h>
#include <portb.h>
#include <adc.h>


/*I N T E R R U P C I O N E S*/
#pragma code
//Alta prioridad: Puerto serial y tareas USB
#pragma interrupt YourHighPriorityISRCode
void YourHighPriorityISRCode()
{
        #if defined(USB_INTERRUPT)
        if(modo_usb)
                USBDeviceTasks();
        if (RCSTAbits.OERR)  // in case of overrun error
        {                    // we should never see an overrun error, but if we do,
                RCSTAbits.CREN = 0;  // reset the port
                c = RCREG;
                RCSTAbits.CREN = 1;  // and keep going.
        }
        #endif

        /*Interrupciòn serial: solo cuando no hay USB*/
        if((!modo_usb) && (PIE1bits.RCIE) && (PIR1bits.RCIF))
        {
                /*Error del puerto serial*/
                if (RCSTAbits.OERR)  // in case of overrun error
                {                    // we should never see an overrun error, but if we do,
                        RCSTAbits.CREN = 0;  // reset the port
                        c = RCREG;
                        RCSTAbits.CREN = 1;  // and keep going.
                }
                c = ReadUSART();
                /*Petición GET*/
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
		if(!cliente && c == get[cont_rx])
		{
			cont_rx++;
			if(cont_rx >= 5)
			{
				cont_rx = 0;
				while(!DataRdyUSART()); c = ReadUSART();
				if(c == ' ')
				{
					cliente = 1;
				}
				else
				{
					accion = 1;
					accion_servidor = c;
					/*Espera por los datos de temperatura para el
ventilador*/
					if(accion_servidor == 't')
					{
						while(!DataRdyUSART()); s_set_temperatura[0] =
ReadUSART();
						while(!DataRdyUSART()); s_set_temperatura[1] =
ReadUSART();
						temp = atob(s_set_temperatura);
						if((temp >= 1) && (temp <= 80))
						{
							set_temperatura = temp;
							btoa(set_temperatura,
s_set_temperatura);
							set_temperatura_ok = 1;
						}
						else
						{
							set_temperatura_ok = 0;
						}
					}
				}
			}
		}

		/*Petición POST*/
		else if(!pulsador_remoto && c == post[cont_rx])
		{
			cont_rx++;
			if(cont_rx >= 4)
			{
				cont_rx = 0;
				pulsador_remoto = 1;
			}
		}

		/*Fin de transmisión del cliente*/
		else if((cliente || pulsador_remoto || accion) && !fin_rx && c ==
fin[cont_rx])
		{
			cont_rx++;
			if(cont_rx >= 4)
			{
				cont_rx = 0;
				fin_rx = 1;
				t_100ms = 0;
				cont_tiempo = 0;
				if(pulsador_remoto)
				{
					while(!DataRdyUSART());
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
                                        pulsador = ReadUSART();
                                }
                                else
                                {
                                        pulsador = 0;
                                }
                        }
                }
                else
                {
                        cont_rx = 0;
                }
        }
}

//Baja prioridad
#pragma interruptlow YourLowPriorityISRCode
void YourLowPriorityISRCode()
{
        /*cruce por cero: Inicia conteo para activacion del Triac*/
        if((INTCON3bits.INT2IE) && (INTCON3bits.INT2IF))
        {
                INTCON2bits.INTEDG2 = !INTCON2bits.INTEDG2; //Cambia el flanco de
interrupción
                /*Habilita interrupción para el tiempo del dimmer*/
                if(intensidad == 10)  //máxima intensidad
                {
                        DIMMER = 1;
                }
                else if(intensidad == 0)      //mínima intensidad = apagado
                {
                        DIMMER = 0;
                }
                else   //niveles intermedios
                {
                        PIR1bits.TMR1IF = 0;
                        WriteTimer1(dimmer[intensidad]);
                        PIE1bits.TMR1IE = 1;
                }
                /*Borra bandera de interrupción*/
                INTCON3bits.INT2IF = 0;
        }

        /*Timer para control de fase*/
        if((PIE1bits.TMR1IE) && (PIR1bits.TMR1IF))
        {
                DIMMER = 1;    /*Pulso de activación del gate del triac*/
                Delay50us();
                DIMMER = 0;
                PIR1bits.TMR1IF = 0;
                PIE1bits.TMR1IE = 0;
        }


        /*Timer para control de tareas cada 100 ms*/
        if((INTCONbits.TMR0IE) && (INTCONbits.TMR0IF))
        {
                WriteTimer0(initTimer0);
                INTCONbits.TMR0IF = 0;
                t_100ms = 1;
                cont_tiempo++;
                if(cont_tiempo >= 5)
                {
                        t_temperatura = 1;
                }
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
              if(cont_tiempo >= 10)
              {
                     cont_tiempo = 0;
                     t_ventilador = 1;
              }
       }
}


#pragma code
void main(void)
{
    InitializeSystem();
       #if defined(USB_INTERRUPT)
        USBDeviceAttach();
    #endif
    while(1)
    {
        #if defined(USB_POLLING)
               USBDeviceTasks();
        #endif
        ProcessIO();
    }
}

static void InitializeSystem(void)
{
       unsigned char i;

    #if defined(USE_USB_BUS_SENSE_IO)
    tris_usb_bus_sense = INPUT_PIN; // See HardwareProfile.h
    #endif

    #if defined(USE_SELF_POWER_SENSE_IO)
    tris_self_power = INPUT_PIN;     // See HardwareProfile.h
    #endif

       /*Puerto serial y Conrol de flujo*/
       InitPorts();
       //InitAnalog();
       Delay50ms();

       /*Habilita prioridad en las interrupciones*/
       RCONbits.IPEN = 1;

       /*Interrupcion serial con alta prioriodad*/
       IPR1bits.RCIP = 1;

       /*Las demas interrupciones son de baja prioridad*/
       IPR1bits.TMR1IP = 0;  //Timer1       DIMMER
       INTCON3bits.INT2IP = 0;       //INT2 DIMMER
       INTCON2bits.TMR0IP = 0;       //TIMER0 TAREAS

       /*ADC*/
       OpenADC( ADC_FOSC_64 &
                      ADC_RIGHT_JUST &
                      ADC_20_TAD,
                      ADC_CH0 &
                      ADC_INT_OFF &
                      ADC_VREFPLUS_VDD &
                      ADC_VREFMINUS_VSS,14);

       /*Usart*/
       OpenUSART(USART_TX_INT_OFF &
             USART_RX_INT_OFF &
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
            USART_ASYNCH_MODE &
            USART_EIGHT_BIT &
            USART_CONT_RX &  //USART_SINGLE_RX
            USART_BRGH_HIGH, BAUD_SPBRG);

        /*Configuración adicional usart*/
        BAUDCONbits.BRG16 = BAUD_BRG16;
        SPBRGH = BAUD_SPBRGH;

        /*Configura timer 1: usado para el control del dimmer*/
        OpenTimer1(    TIMER_INT_ON &
                               T1_16BIT_RW &
                               T1_SOURCE_INT &
                               T1_PS_1_8 &
                               T1_OSC1EN_OFF &
                               T1_SYNC_EXT_OFF);

        /*Configura timer 0: usado para el control de los periféricos*/
        OpenTimer0(    TIMER_INT_ON &
                               T0_16BIT &
                               T0_SOURCE_INT &
                               T0_PS_1_32);

        /*Configura puerto B*/
        /*Interrupciòn en INT2 para entrada de cruce por cero*/
        OpenRB2INT( PORTB_CHANGE_INT_ON &
                               FALLING_EDGE_INT &
                               PORTB_PULLUPS_OFF);


        INTCON3bits.INT2IF = 0;
        INTCON3bits.INT2IE = 1;

        /*Inicia el timer0 = tareas*/
        INTCONbits.TMR0IF = 0;
        WriteTimer0(initTimer0);
        INTCONbits.TMR0IE = 1;


        /* Enable all high priority interrupts */
        INTCONbits.GIEH = 1;

        /* Disable all low priority interrupts */
        INTCONbits.GIEL = 0;


        /*Inicializa variables de emulador serial*/
        //      Initialize the arrays
        for (i=0; i<sizeof(USB_Out_Buffer); i++)
    {
               USB_Out_Buffer[i] = 0;
    }
        NextUSBOut = 0;
        LastRS232Out = 0;
        numBytesRead = 0;
        RS232cp = 0;
        RS232_Out_Data_Rdy = 0;
        RS232_Out_Data = 0;
        RxReady();

        WifiRun();

        flags1.Val = 0;
        flags2.Val = 0;
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
	if(USB_BUS_SENSE)
	{
		modo_usb = 1;
		INTCONbits.GIEL = 0;
		PIE1bits.RCIE = 0;
	USBDeviceInit();
	}
	else
	{
		modo_usb = 0;
		INTCONbits.GIEL = 1;
		PIR1bits.RCIF = 0;
		PIE1bits.RCIE = 1;
	}

	c = ReadUSART();	//Limpia bandera de recepción serial
}


void ProcessIO(void)
{
	if(!modo_usb)
	{
		if(t_100ms)
		{
			t_100ms = 0;
			if(fin_rx)
			{
				if(cliente)
				{
					enviarWeb(web, s_temperatura, s_set_temperatura,
intensidad, control_ventilador);
					cliente = 0;
				}

				if(accion)
				{
					switch(accion_servidor)
					{
						case 'f':	break;
						/*Refrescar todas la variables*/
						case 'R':	//printf("Refrescar %c",
accion_servidor);
								if(!PUERTA)
									s_enviar[0] = 'A';
								else
									s_enviar[0] = 'C';
								if(SENSOR)
									s_enviar[1] = 'A';
								else
									s_enviar[1] = 'N';
								if(AUX)
									s_enviar[2] = 'A';
								else
									s_enviar[2] = 'N';
								if(LUCES)
									s_enviar[3] = 'P';
								else
									s_enviar[3] = 'A';
								//Intensidad
								if(intensidad == 10)
									s_enviar[4] = 'X';
								else
									s_enviar[4] =
intensidad + 48;
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
                                                        //Temperatura
                                                        if(s_temperatura[0] >=
'0')
                                                               s_enviar[5] =
s_temperatura[0];
                                                        else
                                                               s_enviar[5] = '0';
                                                        if(s_temperatura[1] >=
'0')
                                                               s_enviar[6] =
s_temperatura[1];
                                                        else
                                                               s_enviar[6] = '0';
                                                        //Ventilador
                                                        if(control_ventilador ==
2)
                                                               s_enviar[7] = 'A';
                                                        else
if(control_ventilador == 1)
                                                               s_enviar[7] = 'E';
                                                        else
                                                               s_enviar[7] = 'O';
                                                        //Set Point Temperatura
                                                        if(s_set_temperatura[0]
>= '0')
                                                               s_enviar[8] =
s_set_temperatura[0];
                                                        else
                                                               s_enviar[8] = '0';
                                                        if(s_set_temperatura[1]
>= '0')
                                                               s_enviar[9] =
s_set_temperatura[1];
                                                        else
                                                               s_enviar[9] = '0';
                                                        if(AUX1)
                                                               s_enviar[10] =
'P';
                                                        else
                                                               s_enviar[10] =
'A';
                                                        if(AUX2)
                                                               s_enviar[11] =
'P';
                                                        else
                                                               s_enviar[11] =
'A';
                                                        if(AUX3)
                                                               s_enviar[12] =
'P';
                                                        else
                                                               s_enviar[12] =
'A';
                                                        if(AUX4)
                                                               s_enviar[13] =
'P';
                                                        else
                                                               s_enviar[13] =
'A';


                                                        putsUSART(s_enviar);
                                                        break;
                                        /*Control de salidas auxiliares*/
                                        case 'P':      LUCES = 1;
                                                        printf("Se han encendido
```

```
las luces");
                                                        break;
                                                case 'A':       LUCES = 0;
                                                        printf("Se han apagado
las luces");
                                                        break;
                                                case '0':       intensidad = 0;
                                                        printf("Se ha apagado la
luz del dimmer");
                                                        break;
                                                case '1':       intensidad = 1;
                                                        printf("Se han
establecido la intensidad de luz en 10%");
                                                        break;
                                                case '2':       intensidad = 2;
                                                        printf("Se han
establecido la intensidad de luz en 20%");
                                                        break;
                                                case '3':       intensidad = 3;
                                                        printf("Se han
establecido la intensidad de luz en 30%");
                                                        break;
                                                case '4':       intensidad = 4;
                                                        printf("Se han
establecido la intensidad de luz en 40%");
                                                        break;
                                                case '5':       intensidad = 5;
                                                        printf("Se han
establecido la intensidad de luz en 50%");
                                                        break;
                                                case '6':       intensidad = 6;
                                                        printf("Se han
establecido la intensidad de luz en 60%");
                                                        break;
                                                case '7':       intensidad = 7;
                                                        printf("Se han
establecido la intensidad de luz en 70%");
                                                        break;
                                                case '8':       intensidad = 8;
                                                        printf("Se han
establecido la intensidad de luz en 80%");
                                                        break;
                                                case '9':       intensidad = 9;
                                                        printf("Se han
establecido la intensidad de luz en 90%");
                                                        break;
                                                case 'X':       intensidad = 10;
                                                        printf("Se han
establecido la intensidad de luz en 100%");
                                                        break;
                                                case 'v':       control_ventilador = 2;
                                                        printf("Ventilador
automatico");
                                                        break;
                                                case 'p':       control_ventilador = 1;
                                                        printf("Ventilador
siempre encendido");
                                                        break;
                                                case 'a':       control_ventilador = 0;
                                                        printf("Ventilador
apagado");
                                                        break;
                                                case 'B':       AUX1 = !AUX1;
                                                        if(AUX1)
                                                                printf("La salida
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
1 se ha prendido");
                                                                        else
                                                                                printf("La salida
1 se ha apagado");
                                                                        break;
                                                case 'C':       AUX2 = !AUX2;
                                                                        if(AUX2)
                                                                                printf("La salida
2 se ha prendido");
                                                                        else
                                                                                printf("La salida
2 se ha apagado");
                                                                        break;
                                                case 'D':       AUX3 = !AUX3;
                                                                        if(AUX3)
                                                                                printf("La salida
3 se ha prendido");
                                                                        else
                                                                                printf("La salida
3 se ha apagado");
                                                                        break;
                                                case 'E':       AUX4 = !AUX4;
                                                                        if(AUX4)
                                                                                printf("La salida
4 se ha prendido");
                                                                        else
                                                                                printf("La salida
4 se ha apagado");
                                                                        break;
                                                case 't':       if(set_temperatura_ok)
                                                                        {
                                                                                printf("Nuevo
valor asignado");
                                                                                set_temperatura_ok
= 0;
                                                                        }
                                                                        else

        printf("Temperatura fuera de rango");
                                                                        break;
                                                default:        printf("Accion %c",
accion_servidor);
                                                                        break;
                                        }
                                        accion = 0;
                                }
                                fin_rx = 0;
                        }
                }

                /*lectura de temperatura*/
                if(t_temperatura)
                {
                        t_temperatura = 0;
                        temperatura = (unsigned char)((float)((LeerADC(0)+1)/2.046));
                        btoa(temperatura, s_temperatura);

                 }

                /*Control del ventilador*/
                if(t_ventilador)
                {
                        t_ventilador = 0;
                        switch(control_ventilador)
                        {
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
                                case 0: //apagado
                                        VENTILADOR = 0;
                                        break;
                                case 1: //prendido
                                        VENTILADOR = 1;
                                        break;
                                case 2: //automatico
                                        if(temperatura > set_temperatura)
                                               VENTILADOR = 1;
                                        else if(temperatura < set_temperatura)
                                               VENTILADOR = 0;
                                        break;
                                default://en caso de dato no valido  se deja en modo
automatico
                                        control_ventilador = 2;
                                        break;
                        }
                }
                return;
        }
        //Si el USB no esta conectado retorna de la funcion
        if((USBDeviceState < CONFIGURED_STATE)||(USBSuspendControl==1))
        {
                return;
        }

        /*Con el puerto USB conectado el PIC hace puente entre el PC y el WiFly para
configuración*/

        /*Verifica el puerto usb solo si no hay datos pendientes por enviar*/
        if (RS232_Out_Data_Rdy == 0)
        {
                if(USBUSARTIsTxTrfReady())
                {
                        LastRS232Out = getsUSBUSART(USB_In_Buffer,64); //until the buffer is
free.
                        if(LastRS232Out > 0)
                        {
                                RS232_Out_Data_Rdy = 1;  // signal buffer full
                                RS232cp = 0;  // Reset the current position
                        }
                }
        }

    //Verifica si hay datos pendientes por enviar al puerto serial
        if(RS232_Out_Data_Rdy && mTxRdyUSART())
        {
                putcUSART(USB_In_Buffer[RS232cp]);
                ++RS232cp;
                if (RS232cp == LastRS232Out)
                        RS232_Out_Data_Rdy = 0;
        }

    //Verifica si hay datos en el bufer de entrada serial
        if(mDataRdyUSART() && (NextUSBOut < (CDC_DATA_OUT_EP_SIZE - 1)))
        {
                USB_Out_Buffer[NextUSBOut] = getcUSART();
                ++NextUSBOut;
                USB_Out_Buffer[NextUSBOut] = 0;
        }

    //Envia los datos recibidos por el puerto serial al puerto USB
        if((USBUSARTIsTxTrfReady()) && (NextUSBOut > 0))
        {
                putUSBUSART(&USB_Out_Buffer[0], NextUSBOut);
```

ANEXO E: Programa del microcontrolador. Solo archivo main.c

```
                NextUSBOut = 0;
        }

        //Tareas de puerto USB
    CDCTxService();
}

//Lectura de voltaje de entrada:
unsigned int LeerADC(unsigned char Canal)
{
        SetChanADC(Canal);
        Delay();
        ConvertADC();
        while(BusyADC());
        return ReadADC();
}
```

ANEXO F: Código de la página web

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>Centro de control WEB</title>
<style type="text/css">
.Estilo3{color:#FF0000;font-style:italic;font-weight: bold;}
.Estilo4{color:#0000FF;font-style:italic;}</style></head>

<script type="text/javascript">
var ocupado=false;
var pendiente=false;
var t_pendiente;
var contador=0;
function enviar(control){
var peticion;
if(ocupado){
t_pendiente = control;
pendiente=true;
}
else{
ocupado=true
if(window.XMLHttpRequest){
peticion = new XMLHttpRequest();
}
else if(window.ActiveXObject){
peticion = new ActiveXObject('Microsoft.XMLHTTP');
}
peticion.onreadystatechange = leerDatos;
peticion.open('GET',control);
peticion.send();
}

function leerDatos(){
var tx
if(peticion.readyState==4){
if(peticion.status==200){
switch(control){
case 'R': tx=peticion.responseText
if(tx.substring(0,1)=='C')
document.getElementById("tx1").value="Cerrada"
else
document.getElementById("tx1").value="Abierta"
if(tx.substring(1,2)=='N')
document.getElementById("tx2").value="Normal"
else
document.getElementById("tx2").value="Activado"
if(tx.substring(2,3)=='N')
document.getElementById("tx3").value="Normal"
else
document.getElementById("tx3").value="Activado"
document.getElementById("tx4").value=tx.substring(5,7)
break;
default: alert(peticion.responseText);break
}
ocupado=false
}
}
}
}

function t1s(){
if((!ocupado) && (pendiente)){
enviar(t_pendiente);
pendiente=false;}
```

ANEXO F: Código de la página web

```
contador++;
if(contador>19){
contador=0;
if(!ocupado)
enviar('R');
}
}
</script>

<body onLoad="setInterval('t1s()',1000);">
<body leftmargin="50" topmargin="50" marginwidth="50" marginheight="50">
<blockquote><blockquote><blockquote><blockquote><blockquote><blockquote><blockquote>
<h1 class="Estilo3"> Centro de control por Web</h1><blockquote>
<p><span class="Estilo3"><img src="http://www.hardomo.com/index/img/Casa.jpg" alt="casa"
width="259" height="194" align="left" /></span></p>
</blockquote></blockquote></blockquote></blockquote></blockquote></blockquote></blockquote><
/blockquote>
<h1 class="Estilo3"> </h1><p align="center"> </p><p align="center"> </p><p
align="center"> </p><p align="center"> </p>
<table width="955" border="0"><tr><td width="744"><h2 class="Estilo4">Sensores</h2>
<p>Puerta principal<label><input
type="text"name="tx1"id="tx1"value="Cerrada"readonly="readonly"/></label></p>
<p>Sensor de movimiento<label><input
type="text"name="tx2"id="tx2"value="Normal"readonly="readonly"/></label></p>
<p>Sensor adicional<label><input
type="text"name="tx3"id="tx3"value="Normal"readonly="readonly"/></label></p></td>
<td width="270"><img src="http://www.hardomo.com/index/img/Ladron2.jpg" width="124"
height="158" alt="ladron" /></td></tr><tr>
<td><h2 class="Estilo4">Control de iluminaci&#243n</h2>
<p><label><input
type="radio"name="rd1"id="rd1"onchange="enviar('P')"value="rd1"checked/></label>Prender
luces</p>
<p><label><input
type="radio"name="rd1"id="rd2"onchange="enviar('A')"value="rd2"A/></label>Apagar luces</p>
<h3><em>Intensidad de luz incandescente</em></h3>
0%<label><input
type="radio"name="rd2"id="rd3"onchange="enviar('0')"value="rd3"checked/></label>
10%<label><input type="radio"name="rd2"id="rd4"onchange="enviar('1')"value="rd3"/></label>
20%<label><input type="radio"name="rd2"id="rd5"onchange="enviar('2')"value="rd4"/></label>
30%<label><input type="radio"name="rd2"id="rd6"onchange="enviar('3')"value="rd5"/></label>
40%<label><input type="radio"name="rd2"id="rd7"onchange="enviar('4')"value="rd6"/></label>
50%<label><input type="radio"name="rd2"id="rd8"onchange="enviar('5')"value="rd7"/></label>
60%<label><input type="radio"name="rd2"id="rd9"onchange="enviar('6')"value="rd8"/></label>
70%<label><input type="radio"name="rd2"id="rd10"onchange="enviar('7')"value="rd9"/></label>
80%<label><input type="radio"name="rd2"id="rd11"onchange="enviar('8')"value="rd10"/></label>
90%<label><input type="radio"name="rd2"id="rd12"onchange="enviar('9')"value="rd11"/></label>
100%<label><input
type="radio"name="rd2"id="rd13"onchange="enviar('X')"value="rd12"/></label></td>
<td><img src="http://www.hardomo.com/index/img/bombillo.jpg" alt="luz" width="120"
height="77" align="left" /></td></tr>
<tr><td><h2 class="Estilo4"><em>Temperatura</em></h2><p>La temperatura actual es
de<label><input
type="text"name="tx4"id="tx4"value="23"readonly="readonly"/></label></p></td>
<td><img src="http://www.hardomo.com/index/img/termometro.jpg" width="76" height="150"
alt="termometro" /></td></tr>
<tr><td><h2 class="Estilo4"><em>Control de ventilador</em></h2>
<p><label><input
type="radio"name="rd3"id="rd14"onchange="enviar('v')"value="rd13"v/></label>Autom&#225tico</
p>
<p><label><input
type="radio"name="rd3"id="rd15"onchange="enviar('p')"value="rd14"checked/></label>Siempre
encendido</p>
<p><label><input
type="radio"name="rd3"id="rd16"onchange="enviar('a')"value="rd15"a/></label>Apagado</p>
<p>Ingrese la temperatura para activar el ventilador<label><input type="text"
```

ANEXO F: Código de la página web

```
name="tx5"id="tx5"value="28"/></label>
<label><input
type="submit"name="bt1"id="bt1"onclick="enviar('t'+document.getElementById('tx5').value)"val
ue="Actualizar temperatura"/></label>
</p></td><td><img src="http://www.hardomo.com/index/img/ventilador2.jpg" width="134"
height="144" alt="ventilador" /></td></tr>
<tr><td><h2 class="Estilo4">Salidas adicionales</h2>
<p><label><input
type="checkbox"name="ck1"id="ck1"onchange="enviar('B')"checked/></label>Salida 1</p>
<p><label><input
type="checkbox"name="ck2"id="ck2"onchange="enviar('C')"checked/></label>Salida 2</p>
<p><label><input
type="checkbox"name="ck3"id="ck3"onchange="enviar('D')"checked/></label>Salida 3</p>
<p><label><input
type="checkbox"name="ck4"id="ck4"onchange="enviar('E')"checked/></label>Salida 4</p></td>
<td><img src="http://www.hardomo.com/index/img/electrodomesticos.jpg" width="160"
height="167" alt="electrodomestico" /></td></tr>
</table>
</body>
</html>
```

ANEXO G: Código del programa de la Utilidad de Configuración

ANEXO G: Código del programa de la Utilidad de Configuración

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO.Ports;


namespace WindowsFormsApplication1
{

    public partial class Form1 : Form
    {
        /*Variables string para envío de comandos*/
        string buffer_in = null;
        string buffer_redes = null;
        /*general commands*/
        string wifly_command = "$$$";
        string wifly_exit = "exit\r";
        string wifly_factory_reset = "Factory RESET\r";
        string wifly_join = "set wlan join 1\r";
        string wifly_join_num = "join # ";
        string wifly_leave = "leave\r";
        string wifly_reset = "reboot\r";
        string wifly_scan = "scan\r";
        string wifly_save = "save\r";

        /*set commands*/
        string set_baudrate = "set uart b ";
        string set_dns_address = "set dns address ";
        string set_ip = "set ip address ";
        string set_port = "set ip port ";
        string set_ip_bk = "set ip backup ";
        string set_dhcp_on = "set ip dchp 1\r";
        string set_dhcp_off = "set ip dhcp 0\r";
        string set_gateway = "set ip gateway ";
        string set_netmask = "set ip netmask ";
        string set_wep = "set wlan auth 1\r";
        string set_wpa2 = "set wlan auth 2\r";
        string set_wpa1 = "set wlan auth 3\r";
        string set_wpa1_2 = "set wlan auth 4\r";   //recomendada
        string set_nopass = "set wlan auth 0\r";
        string set_wlan_pass = "set wlan key "; //solo en modo wep
        string set_wlan_phrase = "set wlan phrase "; //solo en modo wpa
        string set_wlan_ssid = "set wlan ssid ";
        //string scan = "set wlan channel 0\r";
        string scan = "scan\r";
        /*get commands*/
        string get_ip = "get ip\r";
        string get_wlan = "get wlan\r";
        string get_get = "show net\r";
        string get_signal = "show rssi\r";

        int confirmar = 0;
        int max_timer = 0;
        bool puntos = false;
        bool leyendo = false;
        bool modo_config = false;



        public Form1()
```

ANEXO G: Código del programa de la Utilidad de Configuración

```
        {
            InitializeComponent();
            GetPorts();
        }

        private void serialPort1_DataReceived(object sender,
System.IO.Ports.SerialDataReceivedEventArgs e)
        {
            /*Variables*/
            int redes, i, index1, index2, longitud;

            buffer_in = buffer_in + serialPort1.ReadExisting();
            if (buffer_in.EndsWith("\r\n"))
            {
                if (buffer_in.Contains("Suites"))
                {
                    listRed.Items.Clear();
                    listSenal.Items.Clear();
                    listSeguridad.Items.Clear();
                    if (buffer_in.Contains("Found 0"))
                    {
                        redes = 0;
                        MessageBox.Show("No se han encontrado redes inalámbricas");
                        timer1.Enabled = false;
                    }
                    else
                    {
                        System.Threading.Thread.Sleep(1000);    //retardo de 1 segundo
                        buffer_in = buffer_in + serialPort1.ReadExisting();
                        timer1.Enabled = false;
                        lbEscanear.Visible = false;
                        redes = Convert.ToInt32(buffer_in.Substring(buffer_in.IndexOf("Found
") + 6, 1));
                        buffer_redes = buffer_in.Substring((buffer_in.IndexOf("Suites")));
                        index2 = 0;
                        for (i = 0; i < redes; i++)
                        {
                            index1 = buffer_redes.IndexOf(Convert.ToString(i + 1), index2) +
1;
                            do
                            {
                                index1++;
                            }
                            while (buffer_redes.Substring(index1, 1).Equals(" "));
                            /*Nombre de la red*/
                            index2 = buffer_redes.IndexOf(" ", index1);
                            longitud = index2 - index1;
                            listRed.Items.Add(buffer_redes.Substring(index1, longitud));
                            /*Nivel de señal*/
                            do
                            {
                                index2++;
                            }
                            while (buffer_redes.Substring(index2, 1).Equals(" "));
                            index1 = buffer_redes.IndexOf(" ", index2);
                            do
                            {
                                index1++;
                            }
                            while (buffer_redes.Substring(index1, 1).Equals(" "));
                            index2 = buffer_redes.IndexOf(" ", index1);
                            longitud = index2 - index1;
                            listSenal.Items.Add(buffer_redes.Substring(index1, longitud));
                            /*Tipo de seguridad*/
                            index1 = index2;
```

```
                            do
                            {
                                index1++;
                            }
                            while (buffer_redes.Substring(index1, 1).Equals(" "));
                            index2 = buffer_redes.IndexOf(" ", index1);
                            longitud = index2 - index1;
                            listSeguridad.Items.Add(buffer_redes.Substring(index1,
longitud));
                            groupRedes.Enabled = true;
                        }
                        MessageBox.Show("Se han detectado " + redes.ToString() + " redes
inalámbricas");

                    }
                    buffer_in = null;
                }
                else if(buffer_in.Contains("AOK"))
                {
                    switch(confirmar)
                    {
                        case 0: break;
                        case 1: MessageBox.Show("Se ha configurado la red seleccionada");
                                confirmar = 0;
                                break;
                        case 2: MessageBox.Show("DHCP habilitado");
                                confirmar = 0;
                                break;
                        case 3: MessageBox.Show("DHCP desactivado, configure la red de forma
manual");
                                confirmar = 0;
                                break;
                        case 4: MessageBox.Show("La red se ha configurado correctamente");
                                confirmar = 0;
                                break;
                        default: confirmar = 0;
                                break;

                    }
                    buffer_in = null;
                }
                else if (buffer_in.Contains("Storing"))
                {
                    if (confirmar == 5)
                    {
                        MessageBox.Show("Configuración almacenada correctamente");
                        confirmar = 0;
                        buffer_in = null;
                    }
                }
                else if (buffer_in.Contains("BACKUP="))
                {
                    int inicial, final, datos;

                    if (buffer_in.Contains("DHCP=ON"))
                    {
                        radioButton1.Checked = true;
                    }
                    else
                    {
                        radioButton1.Checked = false;
                    }

                    //IP
                    inicial = buffer_in.IndexOf("IP") + 3;
```

ANEXO G: Código del programa de la Utilidad de Configuración

```
                    final = buffer_in.IndexOf(":2000", inicial);
                    datos = final - inicial;
                    textIp.Text = buffer_in.Substring(inicial, datos);
                    //Mascara de subred
                    inicial = buffer_in.IndexOf("NM") + 3;
                    final = buffer_in.IndexOf("\r", inicial);
                    datos = final - inicial;
                    textSubnet.Text = buffer_in.Substring(inicial, datos);
                    //Puerta de enlace
                    inicial = buffer_in.IndexOf("GW") + 3;
                    final = buffer_in.IndexOf("\r", inicial);
                    datos = final - inicial;
                    textGate.Text = buffer_in.Substring(inicial, datos);
                    buffer_in = null;
                }
                else if (buffer_in.Contains("DHCP: Start"))
                {
                    System.Threading.Thread.Sleep(1000);    //retardo de 1 segundo
                    MessageBox.Show("Dispositivo asociado a la red, presione Leer para ver
la configuracion de la red");
                    buffer_in = null;
                }
                else if (buffer_in.Contains("OK"))
                {
                    buffer_in = null;
                }
            }

        }

        private void button1_Click(object sender, EventArgs e)
        {
            serialPort1.PortName = listPorts.SelectedItem.ToString();

            try
            {
                serialPort1.Open();
                btAbrir.Enabled = false;
                btScan.Enabled = true;
                btCerrar.Enabled = true;
                /*Ingresa al modo comando*/
                serialPort1.Write(wifly_exit);          //garantiza que no se inicie en modo
comando
                System.Threading.Thread.Sleep(1000);    //retardo de 1 segundo
                serialPort1.Write(wifly_command);       //ingresa en el modo comando
                modo_config = true;
                panel1.Enabled = true;
                panelIp.Enabled = true;
            }
            catch
            {
                //throw new NotImplementedException();
                MessageBox.Show("El puerto no esta disponible");
            }
        }

        private void btCerrar_Click(object sender, EventArgs e)
        {
            /*Sale del modo comando*/
            serialPort1.Write(wifly_exit);
            System.Threading.Thread.Sleep(1000);    //retardo de 1 segundo
            serialPort1.Close();
            btAbrir.Enabled = true;
            btCerrar.Enabled = false;
            btScan.Enabled = false;
```

```
        panel1.Enabled = false;
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        CheckForIllegalCrossThreadCalls = false;
        textPassword.Text = null;
    }

    private void btEnviar_Click(object sender, EventArgs e)
    {
        if (!modo_config)
        {
            serialPort1.Write(wifly_command);       //ingresa en el modo comando
            System.Threading.Thread.Sleep(1000);    //retardo de 1 segundo
            modo_config = true;
        }
        groupRedes.Enabled = false;
        serialPort1.Write(scan);
        timer1.Enabled = true;
    }

    /*Obtiene la lista de puertos seriales disponibles*/
    private void GetPorts()
    {
        string[] ports = SerialPort.GetPortNames();
        // Display each port name to the console.
        listPorts.Items.Clear();
        foreach (string port in ports)
        {
            listPorts.Items.Add(port);

        }
    }

    private void label1_Click(object sender, EventArgs e)
    {

    }

    private void listRed_SelectedIndexChanged(object sender, EventArgs e)
    {
        listSeguridad.SelectedIndex = listRed.SelectedIndex;
        listSenal.SelectedIndex = listRed.SelectedIndex;
    }

    private void radioButton2_CheckedChanged(object sender, EventArgs e)
    {
        if (radioButton2.Checked == true)
        {
            panelIp.Enabled = true;
            serialPort1.Write(set_dhcp_off);
            confirmar = 3;
        }
        btGuardar.Enabled = true;
    }

    private void radioButton1_CheckedChanged(object sender, EventArgs e)
    {
        if (radioButton1.Checked == true)
        {
            panelIp.Enabled = false;
            serialPort1.Write(set_dhcp_on);
            confirmar = 2;
            System.Threading.Thread.Sleep(1000);
```

ANEXO G: Código del programa de la Utilidad de Configuración

```
                serialPort1.Write(wifly_join);
            }
            btGuardar.Enabled = true;
        }

        private void listSenal_SelectedIndexChanged(object sender, EventArgs e)
        {
        //    listRed.SelectedIndex = listRed.SelectedIndex;
        //    listSeguridad.SelectedIndex = listRed.SelectedIndex;
        }

        private void listSeguridad_SelectedIndexChanged(object sender, EventArgs e)
        {
        //    listRed.SelectedIndex = listRed.SelectedIndex;
        //    listSenal.SelectedIndex = listRed.SelectedIndex;
        }

        private void button1_Click_1(object sender, EventArgs e)
        {
            if (!modo_config)
            {
                serialPort1.Write(wifly_command);       //ingresa en el modo comando
                System.Threading.Thread.Sleep(1000);    //retardo de 1 segundo
                modo_config = true;
            }
            string seguridad;
            if (listRed.SelectedIndex >= 0)
            {
                seguridad = listSeguridad.SelectedItem.ToString();
                /*Clave WEP*/
                if (seguridad.Contains("WEP"))
                {
                    if (textPassword.Text.Length > 3)
                    {
                        serialPort1.Write(set_wep);
                        System.Threading.Thread.Sleep(1000);
                        serialPort1.Write(set_wlan_pass + textPassword.Text + "\r");
                        System.Threading.Thread.Sleep(1000);
                        serialPort1.Write(wifly_join);
                    }
                    else
                    {
                        MessageBox.Show("La red inalámbrica seleccionada requeriere el
ingreso de contraseña");
                    }
                }
                /*Clave WAP*/
                else if (seguridad.Contains("WPA"))
                {
                    /*WAP1*/
                    if (seguridad.Contains("1"))
                    {
                        serialPort1.Write(set_wpa1);
                    }
                    /*WAP 2*/
                    else if (seguridad.Contains("2"))
                    {
                        serialPort1.Write(set_wpa2);
                    }
                    /*WAP 1y2*/
                    else if (seguridad.Contains("Mix"))
                    {
                        serialPort1.Write(set_wpa1_2);
                    }
                    System.Threading.Thread.Sleep(1000);
```