UCLV
Universidad Central
"Marta Abreu" de Las Villas

FIE
Facultad de
Ingeniería Eléctrica

Departamento de Control Automático

TRABAJO DE DIPLOMA

Título: Propuesta de automatización de una habitación inteligente

Autor: Rafael García Martínez

Tutores: Ing. Ailet Abreu López

Ing. Jorge Peña Martín

Ms. C. Delvis Garcia Garcia

Santa Clara, junio 2019

PENSAMIENTO

"la imaginación es más importante que el conocimiento. El conocimiento es limitado. La imaginación rodea el mundo."

Albert Einstein

DEDICATORIA

A mis hermanos por servirme de ejemplos para establecer todas mis metas …

AGRADECIMIENTOS

A mis tutores por ayudarme a entender el mundo de la domótica, por ser pacientes y persistentes en la preparación de mi trabajo de diploma …

A Leonardo Navarro por apoyarme con sus amplios conocimientos de Arduino y de electrónica, y por ser un estupendo amigo durante estos cinco años…

A Diony Castillo por estar siempre presente cuando lo he necesitado, por entenderme y ayudarme con mis estudios…

A todos los profesores de la Facultad de Ingeniería Eléctrica que contribuyeron de forma exitosa en mi formación como ingeniero, creando una persona más culta y con conocimientos suficientes para aportar soluciones en su futuro centro laboral, con mención especial a Robby Gustabello, Diony Castillo, Rubén Orozco, María del Carmen, Yunier Valeriano, Alain Martínez, Delvis Garcia, José Rafael Abreu, Miguel Borroto, Eduardo Izaguirre y Luis Hernández …

A mi padre por financiar y apoyar mis estudios durante estos cinco años …

A mis hermanos por motivarme cada día a realizarme como ingeniero automático …

A mi primo Samuel González por financiar los medios para la realización de mi trabajo de diploma …

A mis sobrinos por ser las personas que más amo en la vida y mi inspiración para esforzarme estudiando cada día …

A mis dos mejores amigos por ser mi apoyo y comprensión durante los cinco años de carrera, a decir Arturo Javier Cárdenas y Daniel Granado …

A mis compañeros de clases por formar parte de esta aventura universitaria …

A mi entrenador Luis Ernesto Pombo Domínguez por buscar siempre la forma de que mi deporte no afecte mis estudios, permitiéndome llevar dos carreras a la vez …

RESUMEN

La domótica es una tecnología que dota a los hogares de un nivel de conectividad e inteligencia tal que se adapta automáticamente a las necesidades de sus residentes. En Cuba se implementa este tipo de tecnología en el sector del turismo, específicamente en los hoteles, creando un sistema más personalizable, multifuncional y flexible. En este trabajo se propone el diseño de un sistema domótico basado en la plataforma Arduino y los sensores que la complementan. Todo ello no solo aporta confort, conectividad, sino también seguridad, accesibilidad y ahorro energético. Con este nuevo diseño se pretende lograr un sistema totalmente abierto, ampliable y personalizable, capaz de reducir costos con respecto al existente basado en tecnología PLC. Según las pruebas realizadas, la arquitectura de hardware y software libres propuesta, cumple con todas las expectativas para su futura aplicación, siendo una aplicación ventajosa e innovadora, representando un aumento significativo en la economía nacional y un avance en la protección del medioambiente.

TABLA DE CONTENIDOS

INTRODUCCIÓN

En los últimos años han ido apareciendo diversos dispositivos y servicios que han permitido el desarrollo de la tecnología innovadora denominada domótica. En ella se integra automatización, informática y nuevas tecnologías de comunicación; todas ellas dirigidas a mejorar la comodidad, la seguridad y, en definitiva, el bienestar dentro de los hoteles.

Actualmente existen varias alternativas para el diseño de un sistema domótico que cumplen con todas las condiciones vigentes en la normativa de los hoteles. Dichas alternativas están basadas en tecnología: Raspberry Pi, PIC, Odroid C2, PLC, y Arduino. Los medios de transmisión más utilizados para la conformación de este tipo de proyectos son: KNX, X10, DSL, Fibra Óptica, PLC, Cable Coaxial, Wifi, Bluetooth, y Zigbee (Leon, 2011). Todas estas opciones son utilizadas para automatizar hoteles en países con un gran desarrollo en el sector del turismo, ejemplo de ellos son: México, Estados Unidos, y España (Chipuxi, 2015).

La domótica da respuesta a los requerimientos que plantean los cambios sociales y las nuevas tendencias de vida, facilitando el diseño de habitaciones más personalizadas, multifuncionales y flexibles. Se centra en el ámbito de las tecnologías de la información y la comunicación (José Manuel Huidobro, 2007). Con este tipo de sistemas se obtienen ventajas en la programación y el ahorro energético con el propósito de gestionar de forma inteligente la iluminación, la climatización, el agua caliente, el riego, y los electrodomésticos, aprovechando mejor los recursos naturales, utilizando las tarifas horarias de menor coste, y reduciendo así, la factura energética. Aumenta el confort a través de la gestión de dispositivos y actividades domésticas, dado que permite encender, y regular los equipos eléctricos, aire acondicionado, ventilación, iluminación natural y artificial, persianas, toldos, puertas, regadío, suministro de agua, gas, electricidad y otras comodidades que ofrecen un hogar más confortable a sus inquilinos. Establece comunicación mediante el control y supervisión

remotos de la vivienda a través de teléfonos, ordenadores personales, etc. que permite la recepción de avisos de anomalías e información del funcionamiento de equipos e instalaciones. Promueve accesibilidad, facilitando el manejo de los elementos del hogar a las personas con discapacidad de la forma que más se ajuste a sus necesidades, además de ofrecer servicios de tele asistencia para aquellos que lo necesiten. Aporta seguridad por medio de la vigilancia automática de personas, bienes y animales, a través de cámaras de vigilancia, alarmas personales, cierre automático de todas las ranuras o aberturas (Sanchéz, 2016).

En Cuba, se viene desarrollando desde hace unos años la implementación de la domótica en los hoteles, sabiendo las ventajas que esto conlleva en el sector del turismo, fundamental para el crecimiento económico del país. En la actualidad dichos hoteles presentan sistemas domóticos basados en PLC (Programable Logic Controller, por sus siglas en inglés), la cual es muy extendida y robusta, y los sensores empleados son de altas prestaciones, pero sus precios son elevados. Por eso, en este proyecto se propone el diseño de un sistema domótico basado en la plataforma Arduino Mega 2560. Este tipo de hardware protegido por otros medios físicos puede ser una alternativa fiable y económica.

En esta investigación se plantea como **problema científico**: en la mayoría de las habitaciones de los hoteles cubanos que se encuentran ubicados en la Cayería Norte de Villa Clara están implementados sistemas domóticos para aumentar el confort de los turistas durante su estancia en el inmueble. Dichos sistemas están montados sobre la plataforma de PLCs y sensores de altas prestaciones y gran robustes, pero todos estos componentes utilizados para la domótica representan un gran gasto para el país, específicamente el sector del turismo, ya que sus precios en el mercado internacional son elevados. Existen otros factores que representan grandes inversiones como son: el remplazo de los componentes, el pago de la licencia para utilizar el software para programar los PLCs, la aparición de nuevas funcionalidades y aplicaciones. Por lo tanto, en aras de reducir costos y cumplir con las funciones de domótica más comunes, se hace necesario la creación de un nuevo diseño montado en la tecnología de Arduino Mega 2560.

Hipótesis: Con el diseño de un sistema domótico basado en la plataforma Arduino, se dota a la habitación de un nivel de conectividad e inteligencia tal que se adapte automáticamente

a las necesidades de sus ocupantes, aumentando así la calidad de las habitaciones hoteleras, el confort de los clientes, y la reducción de los gastos por parte del ministerio del turismo.

Se plantea como **objetivo general**: Proponer un sistema de hardware y software libres para la automatización de una habitación inteligente.

A partir del objetivo general, se plantean los siguientes **objetivos específicos**:

- Seleccionar las variables físicas a controlar.
- Seleccionar los componentes que conforman un sistema domótico de una habitación inteligente.
- Proponer una arquitectura de hardware para conformar el sistema domótico de una habitación inteligente.
- Proponer una arquitectura de software del sistema domótico de una habitación inteligente. para la programación de los sensores.
- Crear un modelo a escala para evaluar el funcionamiento del sistema domótico diseñado.

Para alcanzar los objetivos expuestos anteriormente se sugiere realizar las siguientes **tareas de investigación**:

- Consulta de la literatura especializada dentro el objeto de estudio.
- Análisis de las mejores alternativas de sistemas domóticos basados en Arduino y sus aplicaciones más comunes.
- Estudio de las herramientas para la automatización de habitaciones inteligentes, protocolos de comunicación, sensores y controladores, así como sus características, ventajas y desventajas.
- Selección de los elementos controladores, sensores, actuadores, y módulos de comunicación del sistema domótico.
- Propuesta de una arquitectura de hardware para conformar el sistema domótico.
- Propuesta de un sistema de comunicación para interconectar los componentes del sistema domótico.
- Propuesta de una arquitectura de software para conformar el sistema domótico.
- Confección de la maqueta a escala para la simulación del sistema domótico diseñado.

Organización del informe

La investigación incluye tres capítulos, además de las conclusiones, recomendaciones, referencias bibliográficas y anexos correspondientes. Los temas que se abordan en cada capítulo se encuentran estructurados de la forma siguiente:

CAPÍTULO I: Se realizará una descripción sobre los sistemas domóticos a partir de la revisión de literatura especializada en el tema. Se describirán las tecnologías y arquitecturas empleadas en la automatización de habitaciones hoteleras y se analizarán brevemente los conceptos de hardware y software libres. Se seleccionará el microcontrolador a utilizar para el diseño de la propuesta realizada.

CAPÍTULO II: Se describirán con detalle los componentes de hardware seleccionados para el diseño del proyecto, así como su interconexión. Se realizará el diseño de la arquitectura de hardware del dispositivo. Se diseñará la arquitectura de software que se encargará de establecer la lógica de control de cada módulo. Se describirá el tipo de conexiones utilizado para la transmisión y recepción de información en cada módulo.

CAPÍTULO III: Se presentarán el hardware y el software diseñados. Se dedicará además a la evaluación de la efectividad del sistema diseñado, a partir de los resultados de las simulaciones del funcionamiento en una maqueta a escala bajo diferentes condiciones. Se finalizará con el análisis económico y medioambiental de la propuesta realizada.

CAPÍTULO 1. SISTEMAS DOMÓTICOS BASADOS EN TECNOLOGÍA ARDUINO EN CUBA Y EL MUNDO

1.1 Introducción a los sistemas domóticos

El término domótica proviene de la unión de las palabras “domus” (que proviene del latín y significa “casa”) y tica (de automática, vocablo griego, que significa “que funciona por si sola”). Se entiende por domótica al conjunto de sistemas capaces de automatizar una vivienda, aportando servicios de gestión energética, seguridad, bienestar y comunicación, y que pueden estar integrados por medio de redes interiores y exteriores de comunicación; cableadas o inalámbricas, y cuyo control goza de cierta ubicuidad (capacidad de estar en todas partes al mismo tiempo) (Leon, 2011).

1.1.1 Servicios ofrecidos por la domótica

Las diferentes áreas de aplicación de la domótica se pueden agrupar en cinco funciones (Sanchéz, 2016):

- **Programación y ahorro energético**: gestiona inteligentemente la iluminación, climatización, agua caliente, el riego, los electrodomésticos, aprovechando mejor los recursos naturales, utilizando las tarifas horarias de menor coste, y reduciendo así, la factura energética.
- **Confort**: a través de la gestión de dispositivos y actividades domésticas. La domótica permite abrir, cerrar, apagar, encender, regular, los electrodomésticos, la climatización, ventilación, iluminación natural y artificial, persianas, toldos, puertas, riego, suministro de agua, gas, y la electricidad.
- **Seguridad**: vigilancia automática de personas, animales y bienes, así como de incidencias y averías. Mediante controles de intrusión, cierre automático de todas las

aberturas, simulación dinámica de presencia, cámaras de vigilancia, y a través de alarmas técnicas que permiten detectar incendios, fugas de gas, inundaciones de agua, y fallos del suministro eléctrico.

- **Comunicaciones**: mediante el control y supervisión remotos de la vivienda a través de su teléfono, ordenador personal, que permite la recepción de avisos de anomalías e información del funcionamiento de equipos e instalaciones.
- **Accesibilidad**: facilita el manejo de los elementos del hogar a las personas con discapacidad de la forma que más se ajuste a sus necesidades, además de ofrecer servicios de tele-asistencia para aquellos que lo necesiten.

1.1.2 Características de un sistema domótico

Desde el inicio de la domótica se han ido adhiriendo nuevas características, que son el resultado de las mejorías y del aumento de las aplicaciones que se crean a partir del estudio de las necesidades de los usuarios. Las características son las siguientes (Bustamante y Acosta, 2016):

- **Integración**: Control del sistema a través de un microcontrolador o unidad de cómputo, es decir que los usuarios no tienen que estar ahí o pendientes de los diversos acontecimientos ya que son autónomos, con su propio diseño de programación, indicadores situados a diferentes lugares de conexión entre componentes de distintos fabricantes.
- **Interrelación**: Son capaces para poder relacionar en desiguales elementos y así obtener una gran versatilidad y variedad en su toma de decisiones como por ejemplo la facilidad de relacionar en un funcionamiento de aire acondicionado con otros tipos de electrodomésticos, y como su apertura de ventanas o con detección de usuarios que estén en sus hogares.
- **Facilidad de uso**: Con solo mirar la pantalla de la computadora o dispositivo móvil, el usuario está completamente informado del estado de su casa. Y si desea modificar algo, solo necesitará pulsar un reducido número de teclas o solo hacer un clic con el mouse, por ejemplo, la temperatura dentro y fuera de la habitación.

- **Control remoto**: Posibilidades de supervisión y control disponibles localmente, mediante cualquier conexión a internet desde otro componente y en cualquier parte que se encuentren.
- **Fiabilidad**: El hardware utilizado actualmente es muy potente, fiable y rápido. A ocurrido un incremento del uso de sistemas de alimentación interrumpida, ventilación forzada de CPU, baterías de gran capacidad que alimenta a los periféricos. Se dispone de una plataforma ideal para que las aplicaciones domóticas sean capaces de funcionar a lo largos de los años sin problemas.
- **Actualización**: La actualización de sistemas de este tipo son muy fácil de realizar, al poder ver nuevas versiones y mejoras solo será preciso cargar el nuevo software en la PC.

1.1.3 Conceptos fundamentales relacionados con los sistemas domóticos

La diversidad de componentes que conforman un sistema domótico, lo que trae consigo la necesidad de definir conceptos que intervienen durante el diseño, con el objetivo de lograr mayor claridad en la elaboración de la propuesta realizada. En la bibliografía consultada varios autores definen los siguientes conceptos en sus libros (Chipuxi, 2015), (Giorgini, 2017), (Sánchez, 2012), y (Eizmendiz, 2016), aunque (Leon, 2011) destaca estas palabras en el área de la domótica logrando encerrar en su significado sus principales características:

Software: Se conoce como software al soporte lógico de una computadora digital; comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos, que son llamados hardware (Leon, 2011).

Hardware: se refiere a las partes físicas tangibles de un sistema informático; sus componentes eléctricos, electrónicos, electromecánicos y mecánicos (Leon, 2011).

Controlador: es un programa informático que permite al sistema operativo interactuar con un periférico, haciendo una abstracción del hardware y proporcionando una interfaz posiblemente estandarizada para su uso (Leon, 2011).

Sensor: es un dispositivo capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables eléctricas (Leon, 2011).

Actuador: es un dispositivo capaz de transformar energía hidráulica, neumática o eléctrica en la activación de un proceso con la finalidad de generar un efecto sobre un proceso automatizado. Este recibe la orden de un regulador o controlador y en función a ella genera la orden para activar un elemento final de control (Leon, 2011).

1.1.4 Ventajas y desventajas de un sistema domótico

La incorporación de un sistema domótico en una instalación hotelera es una moneda de dos caras, por lo que, es importante analizar si las ventajas son mayores que las desventajas para poder determinarse si es efectivo su diseño (Leon, 2011):

Ventajas:

1. Protección de la habitación y el cliente
2. Aumento del valor del total del hotel
3. Aumento de la calidad de vida y del servicio brindado
4. Incremento del ahorro energético
5. Extensión de la visión futurista
6. Extensión del respeto por el medio ambiente

Desventajas:

1. Ausencia de instaladores autorizados
2. Incremento del grado de complejidad
3. Baja existencia de centros de servicios

Un sistema domótico dota a las habitaciones de un grupo de aplicaciones brindando un elevado confort al cliente. A continuación, son mencionadas las más importantes (Leon, 2011):

1. Control de acceso
2. Control de presencia
3. Control del sistema de televisión
4. Control del clima
5. Control de servicios (limpieza)

1.2 Síntesis de trabajos relacionados con la domótica

La domótica es un tema innovador por lo que se han desarrollado varias investigaciones y trabajos en universidades y centros de investigación alrededor del mundo, entre los cuales se destacan los mencionados a continuación:

En el trabajo de A. M. A. Sanchéz se desarrolla un sistema de automatización para el hogar programable basado en el *protocolo X-10* que permite implementar una alternativa económica a las opciones existentes actualmente en el mercado. Para conseguirlo, se hace uso de dos microcontroladores de código abierto: *Arduino UNO* e *Intel Galileo*. Por una parte, Arduino se encarga de hacer de interfaz con los dispositivos que forman el sistema domótico (luces, electrodomésticos, etc.). Por la otra, Galileo alberga un pequeño servidor que es el verdadero encargado de gestionar el sistema, proporcionando una interfaz para interactuar con la vivienda, accesible a través de una *API REST*. Este sistema de automatización se completa con el desarrollo de una aplicación Android que proporciona una interfaz sencilla desde la que cualquier usuario podrá gestionar y controlar todo lo que ocurra en su vivienda y que se comunicará con el servidor a través de su API (Application Programming Interface, por sus siglas en inglés) (Sanchéz, 2016).

En el proyecto de E. Ll. Sánchez se realiza una propuesta para crear un sistema domótico simple utilizando las placas de bajo costo de Arduino y otros dispositivos, como sensores, actuadores y comunicadores. Igualmente, se dota al sistema de la lógica necesaria para que puedan comunicarse las placas que están controlando la habitación en la cual han sido instaladas. El hardware que se utiliza es la placa *Arduino UNO* y otra de *Seeeduino* para controlar las lecturas de los sensores. El sistema es repartido en varios módulos, a decir un sensor de gas que hace que la vivienda gane en seguridad, un PIR que interviene en la seguridad cuando no hay personas en la casa, una fotocelda capaz de detectar el nivel de intensidad de luz que hay en la habitación, un módulo de humedad y temperatura DHT11 capaz de representar digitalmente la humedad ambiental medida en porciento además de la temperatura en °C, un medidor de temperatura capaz de establecer un umbral que domina el apagado/encendido o modificación de la potencia de la calefacción o el aire acondicionado. El actuador utilizado para el diseño es un relé. Las interfaces empleadas para la comunicación entre el sistema y el ser humano son LEDs que avisan la ocurrencia de un evento que puede

requerir atención, un timbre para avisar al ser humano sobre un problema grave en la vivienda, y un push button para aplicaciones como apagar luces, calefacción y activar el sistema de seguridad al salir de casa, un dispositivo visualizador que permite conocer la situación de la vivienda y de la ocurrencia de eventos. Para la comunicación entre los electrodomésticos del hogar se utilizan el modelo Ethernet, WiFi, e infrarrojo. El servidor utilizado es una placa Raspberri Pi para controlar el sistema incluso fuera de casa (Sánchez, 2012).

En el artículo de Sánchez et al. Se desarrolla un prototipo de casa inteligente de bajo costo como una experiencia en el área de la domótica, en él se incluye fallos y aciertos claves que se tuvieron durante los diferentes procesos: planeación, diseño y ejecución del prototipo. Se definen cuatro módulos principales, un módulo para la luz que enciende las luces si hay alguna persona que las utiliza, un módulo de ventilador que se activa solo cuando se supera un nivel de temperatura definido por el usuario, un módulo de puerta que funciona automáticamente, un módulo de lógica y comunicación para revisar el estado de cada componente y de acuerdo con este modificar el actuador correspondiente. Como hardware se utiliza un microcontrolador Arduino, y como sensores análogos se emplea un sensor de temperatura, a partir del cual se determina cuando se rebasa el umbral establecido para el encendido del ventilador, un sensor infrarrojo, y un sensor de presencia para controlar el encendido y apagado de los LEDs de la casa, un sensor ultrasónico cuya misión es detectar presencia de personas y activar un servomotor para abrir y cerrar la puerta, y un transmisor inalámbrico Bluetooth que permite controlar diversos parámetros del prototipo de forma remota, ya sea desde un teléfono inteligente o desde un computador (Sánchez *et al.*, 2014).

En el trabajo de D. E. Ch. Gallego se diseña un sistema domótico que permite a los usuarios consultar y controlar en tiempo real la información y las acciones que tienen lugar en la casa. El funcionamiento de este montaje se basa en controlar vía *Ethernet*, los datos correspondientes de los sensores instalados, creando alertas y eventos de acuerdo a los datos que estos arrojan. El microcontrolador *Arduino UNO* se utiliza para el aviso y control remoto de eventos en el hogar, un sensor de movimiento PIR HC-SR501 que tiene como función ejecutar un evento ya sea una señal de luz, una alerta sonora y una visualización en la página web, un sensor de dióxido de carbono, un sensor de gas metano MQ-4 que activa una alerta de luz y una electro-válvula conectada a un canal del módulo relé de 4 posiciones, un sensor

de flama que arroja una señal que activa el sistema de riego interno, señal de luz, alarma sonora y una alerta en la página web, un sensor de temperatura LM 35 para activar y desactivar la ventilación del lugar o la calefacción, y un módulo relé de 4 canales (Gallego, 2016).

En el trabajo de X. Alaman; P. Haya; G. Montoro se realiza la conformación del proyecto ODISEA, el cual propone desarrollar un entorno inteligente que realice acciones automáticas para ayudar a los residentes con sus preferencias y las tareas que en ese momento están realizando, los usuarios interactúan con el sistema en lenguaje natural utilizando modos de interacción sobre visión artificial y otros tipos de sensores. Se implementa un entorno domótico y un entorno ofimático reales. La arquitectura básica de ODISEA descansa sobre dos capas: la capa de interacción con el entorno físico y la capa de contexto; la de interacción con el entorno físico incluye el conjunto de dispositivos que componen la infraestructura del entorno inteligente, por su parte la de contexto ofrece información sobre el estado del entorno, sus ocupantes, y las actividades que están realizando. Para el proceso de reconocimiento de la voz se emplea la herramienta comercial *ViaVoice* de *IBM*. La comunicación con el reconocedor se realiza utilizando la *Java Speech API*. En cuanto a la síntesis de voz se efectúa utilizando la capacidad *Text to Speech* de *ViaVoice*. Para que el módulo de diálogos pueda procesar la información que se recibe del reconocedor se utilizan un conjunto de analizadores y gramáticas integradas en las herramientas *MACO+ y Relax y Tacat* (Alaman, Haya and Montoro, 2010).

En el trabajo de O. S. Carretero se desarrolla un control de llenado de un tanque para un acuario, un servidor web, y un control de acceso a la vivienda. Complementariamente con el control de llenado del tanque, existe un control del nivel de sal de la descalcificadora, toda esa información es mostrada en una pantalla LCD en tiempo real, con la ayuda de varios sensores y de un Arduino UNO. La creación del servidor web se realiza con el propósito de controlar una serie de circuitos y componentes del hogar de forma automática a través de una interfaz, para ello se utiliza un Arduino Mega, y una plataforma Ethernet para comunicarse con el Arduino que es el que posee la información almacenada en su memoria EPROM. Para control de acceso a la vivienda se realiza una identificación de los usuarios para armar y desarmar la alarma con un tag de rfid de 13.5 MHz, para ello se utiliza también un módulo

de voz que permite determinar en qué estado se encuentra el control de acceso (Carretero, 2016).

R. Y. O. Chipuxi describe la construcción de un prototipo domótico vía *GSM*. Las alarmas se activan cuando alguno de los sensores envía una señal informando una posible presencia de intruso, de humo o fuga de agua; estas señales son extraídas de cualquier habitación del hotel San Miguel y notificadas mediante un SMS al teléfono móvil del administrador. Se utiliza un dispositivo Smartphone como base de datos para almacenar todos los incidentes y poder ser visualizados a través de una interfaz gráfica. En la realización del proyecto se seleccionó como opciones fiables a *Arduino Mega 2560* para el control de los sensores, y el módulo GSM (Sistema global de comunicaciones móviles), que unido a la tarjeta SIM900 GSM/GPRS se encargan del envío de SMS (mensajes de texto) (Chipuxi, 2015).

En el trabajo de O. G. Giorgini se desarrolla un sistema domótico mediante una Raspberry Pi e Internet. Para el diseño se utiliza una Raspberry Pi como un servidor de páginas web dinámicas y motor de bases de datos; y el protocolo X-10 es empleado para el control de los dispositivos de la casa, el cual envía y recibe datos a través de la línea eléctrica y placas electrónicas diseñadas para el proyecto. Además, se diseña una interfaz web para enviar los comandos que controlan a los dispositivos, en dicha página se implementa un sistema de autenticación, donde debe ser introducido el usuario y la contraseña para evitar el ingreso no autorizado a la interfaz de control, se usa para ello una base de datos MySQL. Una placa transceptora, cuyo componente más importante es un microcontrolador, recibe las señales de los puertos GPIO de la Raspberry Pi y las suministra a la red eléctrica, sincronizando el mensaje con los cruces por cero de la línea. La interfaz posee muchas posibilidades, dado que, permite agendar mediante el programa cron una tarea para determinado día y horario. Con el uso de un servicio de *DNS* dinámico y la configuración del router se logró hacer accesible la página web desde cualquier computadora conectada a internet (Giorgini, 2017).

1.3 Arquitectura y estructura de un sistema domótico

Existen dos tipos de arquitecturas básicas: la arquitectura centralizada y la distribuida (Eizmendiz, 2016).

a. **Arquitectura Centralizada**: Es aquella en la que únicamente existe una unidad de control que es la encargada de recepcionar la información de cada sensor y procesarla según los criterios e intereses del usuario, seguidamente generar las ordenes que son enviadas a los diversos actuadores distribuidos por el hogar. La unidad de control es el dispositivo fundamental, ya que de él depende todo el sistema para su funcionamiento.
b. **Arquitectura Descentralizada**: Es en la que hay diferentes unidades de control interconectadas por un bus, se envían informaciones entre ellos y a los actuadores e interfaces conectados correspondientes a cada una, según la información, programa o configuración que reciban de los sensores, sistemas interconectados y usuarios.
c. **Arquitectura Distribuida**: Es en la que el elemento de control se instala cerca de la magnitud que debe controlar y no necesita cablearse hasta el actuador al que debe accionar. En un sistema de domótica de arquitectura distribuida, cada sensor y actuador es también un controlador capaz de actuar y enviar información al sistema según el programa, la configuración, la información que capta por sí mismo y la que recibe de los otros dispositivos del sistema.
d. **Arquitectura Mixta**: En este tipo de arquitectura se combinan las arquitecturas de los sistemas centralizados y distribuidos. A la vez que puede disponer de un controlador central o varios controladores descentralizados, los dispositivos de interfaces, sensores y actuadores pueden también ser controladores.

Medios de transmisión o interconexión (Sanchéz, 2016):

a) Cableados
- Digital Subscriber Line (DSL, por sus siglas en inglés)
- Fibra óptica
- Power Line Conmunications (PLC, por sus siglas en inglés)
- X10
- Cable (coaxial y par trenzado)

b) Inalámbricos
- Wifi
- General Packet Radio Service (GPRS, por sus siglas en inglés)

- Bluetooth
- Radiofrecuencia
- Infrarrojos
- ZigBee

1.3.1 Tecnologías utilizadas para el diseño de un sistema domótico

Para la construcción de sistemas domóticos existen diferentes tecnologías dependiendo de los servicios requeridos y del nivel de complejidad. A continuación, se hará mención de las más utilizadas:

a. **Raspberry Pi**: es una computadora de dimensiones (85.60mm × 53.98mm), con un procesador ARM de 700 Mhz y 512 M de RAM, conexión ethernet, 2 puertos USB, 1 puerto HDMI, puerto RCA, 8 pines GPIO, SPI, I2C y UART, con consumo de 700 mA y 3,5 W. El diseño no incluye un disco duro ni unidad de estado sólido, ya que usa una tarjeta SD para el almacenamiento permanente, tampoco incluye fuente de alimentación ni carcasa. Se alimenta con una fuente de 5 volt y 2 ampere con puerto mini USB. Puede instalar varios sistemas operativos, la mayoría de ellos basados en núcleos Linux, entre ellos: GNU/Linux: Debian (Raspbian), Fedora (Pidora), Arch Linux (Arch Linux ARM), y Slackware Linux. No viene con reloj en tiempo real, por lo que el sistema operativo debe usar un servidor de hora en red, o pedir al usuario la hora en el momento de arrancar el ordenador; sin embargo, se podría añadir un reloj en tiempo real (como el DS1307) con una batería mediante el uso de la interface I²C. Es más potente que un micro controlador, permite correr un servidor web, bases de datos y un servidor de vídeo dentro de un sistema operativo robusto como los GNU/Linux, siendo fácilmente configurable y accesible, mediante escritorio remoto o protocolo SSH. Es equivalente a una computadora normal de uso doméstico, pero con la ventaja de muchos GPIO, pines configurables de entrada/salida, a través de los cuales se interactúa con el mundo físico. En la figura 1.1 se muestra la placa Raspberry Pi (Giorgini, 2017).

Figura 1.1. Circuito Integrado de la Raspberry Pi.

b. **Programable Integrated Circuit** (PIC, por sus siglas en inglés): es un circuito integrado programable fabricado por Microchip, posee en su interior CPU RISC, memoria RAM, periféricos de entrada y salida, unidad aritmética lógica, contadores de programa, temporizadores e interrupciones, bancos de datos, conversor análogo/digital. En el PIC se puede programar las instrucciones de funcionamiento, capaz de adaptarse a nuestras necesidades. Se caracterizan por su arquitectura Harvard, con memorias de programa y de datos independientes, lo que permite la accesibilidad simultánea, vienen en diferentes familias PIC8, PIC16, PIC32 y en diferentes encapsulados. Los PIC se clasifican de acuerdo al tamaño de sus instrucciones en: gama baja (instrucciones de 12 bits), gama media (instrucciones de 14 bits), gama alta (instrucciones de 16 bits). También se agrupan de acuerdo al número de sus terminales, por ejemplo: PIC10, PIC12, PIC16, PIC17, PIC18. En la figura 1.2 se muestra el microchip PIC (Chipuxi, 2015).

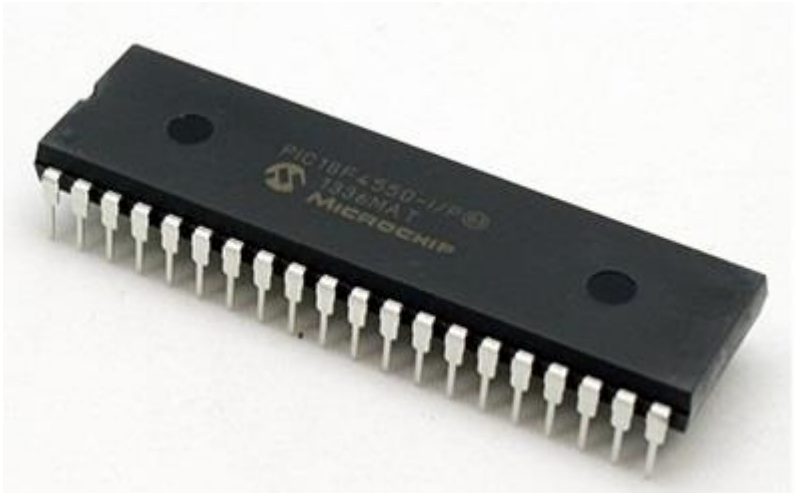


Figura 1.2. Microchip PIC18F4550.

c. **Odroid C2**: es una SBC (Single Board Computer, por sus siglas en inglés) de origen surcoreano creado por la empresa Hardkernel, cuyo nombre viene de Open Android, y en la búsqueda de una plataforma de desarrollo para aplicaciones Android se creó este sistema totalmente compatible con el sistema operativo Linux y Android de ese modo esta pequeña SBC tiene características superiores de procesador y memoria. Incorpora 4 puertos USB, un puerto OTG microUSB, un puerto Ethernet que soporta velocidades de transferencia Gigabit, un conector HDMI 2.0 para monitores que puedan soportar resolución de hasta 4k, un conector de alimentación de 5V/2A. Además de estas entradas convencionales, el C2 también incluye un puerto GPIO de 40 pines, un puerto de consola serie USB-UART, un conector para módulos eMMC y una ranura para tarjetas microSD. Odroid es una plataforma muy completa para el desarrollo de prototipos y aplicaciones en la domótica. En la figura 1.3 se muestra la placa de Odroid C2 (Rojas, 2017).

Figura 1.3. Circuito Integrado de Odroid C2.

d. **Controlador Lógico Programable** (PLC, por sus siglas en inglés): es una computadora utilizada en la automatización de procesos. Utiliza memoria programable dominada por una CPU para guardar instrucciones sobre la implementación de determinadas funciones, como operaciones lógicas, secuencia de acciones, especificaciones temporales, contadores y cálculos para el control mediante módulos de E/S analógicos. Están diseñados para múltiples señales de entrada y salida, amplios rangos de temperatura, inmunidad al ruido eléctrico y resistencia a la vibración y al impacto. Es capaz de realizar operaciones en tiempo real debido a su disminuido tiempo de reacción. Son dispositivos que se adaptan fácilmente a nuevas tareas debido a su flexibilidad a la hora de programarlos. Permite también una comunicación inmediata con otro tipo de controladores y ordenadores e incluso permite realizar las operaciones en red. Para instalarlos es necesario utilizar un carril DIN, y en su interior también dispone de convertidores digital analógico. También poseen memoria RAM para guardar los datos y una memoria en la que se puedan borrar programas, y el sistema operativo del sistema es de carácter permanente sin que se pierda al apagar el sistema, esta memoria solo se puede borrar eléctricamente. Los voltajes de operación más frecuentes son ±5V, ±12V y ±24V para las entradas/salidas. Los módulos más importantes son: módulo de interfaz (IM), módulo funcional (FM), regulador PID, procesador de la comunicación (CP), interfaz

hombre-máquina (HMI), entradas/salidas remotas, y módulos de señal de alta velocidad. En la figura 1.4 se muestra el PLC (Valdivia, 2014).

Figura 1.4. PLC S7-200 de Siemens.

e. **Arduino**: se inició en el año 2005 como un proyecto para estudiantes en el Instituto IVREA, Italia. Hoy en día existe una variedad de placas y versiones prácticamente adaptables a todo modelo de necesidades de dichos requisitos para poder llevar a cabo un determinado proyecto. El avance tecnológico llevo a la evolución del Duemilanove al Arduino "UNO", esta se conecta al ordenador a través de un cable de USB estándar que contiene todo lo que se necesita para poder trabajar y programar para hacer uso de la placa, que a diferencia del antiguo Duemilanove, integrará un nuevo chip USB- serie que cuenta con un nuevo modelo de etiquetando, facilitando así la identificación de las distintas salidas y entradas. La siguiente evolución es "Leonardo", muy similar a Arduino "UNO", esta placa tiene mejores prestaciones. Si se habla de Arduino MEGA es porque es el más adecuado por así decirlo para proyectos que requieren un gran número de salidas como de entradas, o para soportar la carga de códigos de programación pesados que no pueden almacenarse en las memorias de menor capacidad que ofrecen otras placas, como por ejemplo el modelo UNO (Bustamante and Acosta, 2016).

Arduino es una placa programable con entradas y salidas digitales y analógicas, cuyo bajo coste la hace ideal para iniciarse en automatización o realizar pequeños proyectos domésticos

en electrónica y robótica. Esto significa que disponemos de un pequeño «autómata», capaz de recibir información del entorno (sensores) y realizar acciones (actuadores, motores). Arduino es en realidad tres cosas (Sanchéz, 2016):

1. Hardware libre, consistente en un circuito impreso con un microcontrolador reprogramable, usualmente Atmel AVR, y puertos digitales y analógicos de entrada/salida, los cuales pueden conectarse a placas de expansión (llamadas shields) que amplían las características de funcionamiento del Arduino. Asimismo, posee un puerto de conexión USB desde donde se puede alimentar y establecer comunicación serie con un ordenador.
2. Software gratis, libre y multiplataforma que permite escribir, compilar y guardar en la memoria del microcontrolador de la placa Arduino el conjunto de instrucciones que se desea ejecutar.
3. Lenguaje de programación libre: tanto el entorno de desarrollo (o IDE, del inglés Integrated Development Kit) como el lenguaje de programación de Arduino están inspirados en otro entorno y lenguaje libre existente: Processing,

El microcontrolador incluye un entorno interactivo de desarrollo (IDE) que permite programar fácilmente la tarjeta Arduino, se basa en C/C++, y se simplifica con el uso de la biblioteca Arduino. El IDE (Integrated Development Environment) se basa en Processing y Wiring (open source). El lenguaje de programación C/C++ se divide en tres partes principales: estructura, valores (variables y constantes) y funciones (Chipuxi, 2015).

Figura 1.5. Circuito Integrado de Arduino Mega 2560.

Para el presente trabajo se propone el uso de la plataforma Arduino dado que, estas son más asequibles comparadas con otras plataformas de microcontroladores; el software de Arduino es multiplataforma; y posee un entorno de programación simple y directo. Además, es ampliable y de código abierto (Chipuxi, 2015), y está publicado bajo una licencia libre y puede ser ampliado, el lenguaje puede extenderse a través de librerías de C++, y modificarlo a través de lenguaje de programación AVR C en el que está basado. También, disfruta de un hardware ampliable y de código abierto (Chipuxi, 2015), con posibilidades de realizar modificaciones o versiones personales a los módulos, ampliándolo u optimizándolo facilitando el ahorro. Arduino consta con una placa que incluye un microcontrolador con toda la circuitería necesaria para su correcto funcionamiento, cuenta con el hardware necesario para la conexión a las entradas/salidas correspondientes del procesador de los sensores y actuadores que forman parte del sistema que se propone. Asimismo, se puede programar por conexión USB directa con la computadora, se puede alimentar de varias formas (fuente de 7 a 12 V, por USB o por sus pines de entrada de 5V) y posee salidas reguladas de 5V y 3.3 V que se pueden emplear para la alimentación de sensores u otro hardware adicional. Por otro lado, la tecnología rival de Arduino a nivel mundial, la PIC de Microchip Technology Inc, no constituye hardware ni software libre y además la compañía solamente ofrece el chip del

microcontrolador, por lo que requiere de circuitería adicional para su puesta en marcha y su programación, aunque existen modelos con grandes prestaciones.

Para el desarrollo del proyecto se selecciona específicamente a Arduino Mega 2560, dado que, tiene una gran capacidad de memoria lo que le permite utilizarlo para códigos muy extensos o que requieren de una gran cantidad de variables.

1.4 Consideraciones finales del capítulo

Luego de analizar la bibliografía especializada y realizar un análisis de los sistemas domóticos en el mundo, se llegó a las siguientes consideraciones:

1. La domótica está caracterizada por ofrecer cuarto servicios fundamentales: ahorro energético, confort, seguridad, y comunicación. Estos servicios representan un gran beneficio económico para cualquier hotel donde sea implementado este tipo de tecnología.
2. Existen cuatro elementos fundamentales para lograr conformar un sistema domótico, estos son: controladores o microcontroladores, sensores, actuadores y medios de transmisión.
3. Los sistemas de este tipo poseen una alta flexibilidad, o sea que puede adaptarse a nuevos cambios dependiendo de los avances tecnológicos que vayan sucediendo sin afectar significativamente la infraestructura sobre la que fue montado.
4. Una ventaja especial de estos sistemas, es que puede ser creado basándose en hardware y software libre permitiéndole al proyecto disfrutar de independencia tecnológica.
5. Existen diversas plataformas sobre las cuales se puede diseñar un sistema domótico. Luego de comparar las características de cada una, para el desarrollo del presente proyecto se propone utilizar la placa Arduino Mega 2560, debido a que cumple con los requisitos necesarios para la automatización de una habitación inteligente.

CAPÍTULO 2. ARQUITECTURA DE HARDWARE Y SOFTWARE DEL SISTEMA DOMÓTICO

En el presente capítulo se describen los elementos de hardware y las herramientas de software utilizados para la confección de la propuesta domótica. Posteriormente se describen las arquitecturas de hardware y software. Se describe además el sistema de comunicación utilizado. El peso principal de este capítulo recae en la programación de eventos como parte de la propuesta domótica diseñada.

2.1 Requisitos para el diseño del sistema domótico

Para evaluar el nivel de calidad que debe poseer la propuesta domótica a realizar es necesario establecer que exigencias se deben cumplir. El sistema a diseñar cumple con los siguientes requisitos:

1. Sistema confiable, portable y de fácil manejo para los usuarios.
2. Monitoreo en tiempo real dado el reducido tiempo entre la lectura del sensor y la respuesta del sistema.
3. Uso de tecnologías de software actuales, buscando la eficiencia y comunicación integra.
4. Diseño modular en el sistema, que permite reutilizar el código y facilitar el mantenimiento posterior del sistema.
5. Adaptabilidad a los avances tecnológicos.
6. Capacidad de proveer un entorno ecológico interior y exterior respectivamente habitable y sustentable.
7. Diseño multifuncional y flexible.
8. Microcontrolador de código abierto basado en hardware y software libres.

2.2 Características de las habitaciones hoteleras

Los Edificios Inteligentes (Smart Buildings, por sus siglas en inglés) cuentan con instalaciones y sistemas (climatización, iluminación, electricidad, seguridad, telecomunicaciones, multimedia, informáticas, control de acceso, etc.) que permiten una gestión y control integrado y automatizado, con el fin de aumentar la eficiencia energética, la seguridad y la accesibilidad. El control y gestión de todas estas instalaciones y sistemas se llevan a cabo a través de la inmótica. En relación a esto, esta ofrece la posibilidad de monitorización del funcionamiento general del hotel. Asimismo, la centralización de los datos del complejo posibilita supervisar y controlar confortablemente los estados de funcionamiento o alarmas de los sistemas que componen la instalación, así como los principales parámetros de medida (Zennio, 2018b).

En la arquitectura se busca el confort, la habitabilidad y la sostenibilidad de un hotel, por ello, la inclusión de estos sistemas debe ir enfocada a cumplir las expectativas y necesidades del usuario. En una evolución desde el punto de vista arquitectónico. “La inteligencia en un hotel está directamente relacionada con la sostenibilidad y el aprovechamiento máximo de los recursos energéticos e intensificado del uso de aquellos que son naturales, como, por ejemplo, la utilización de la luz solar mediante el empleo de persianas inteligentes (S.A, 2017).

El diseño domótico propuesto en esta investigación viene desarrollado para la arquitectura de habitación conformada por un baño, un dormitorio, y un balcón. Dentro de la habitación son controlados varios dispositivos: las luces, la climatización, el ventilador de techo, la alarma del closet, y el motor de desplazamiento de las cortinas. A continuación, es mostrado el esquema del estilo de habitación descrito:

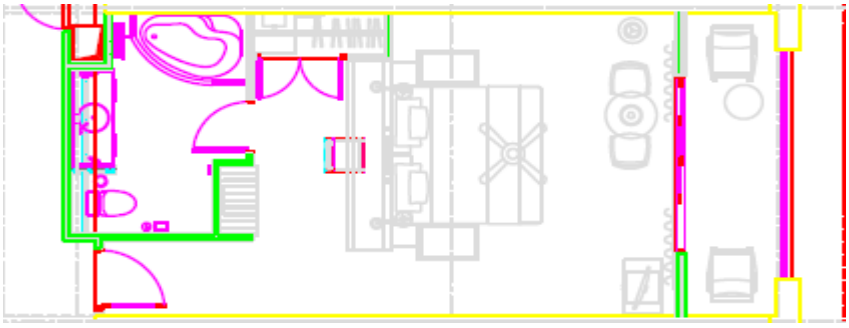


Figura 2.1. Modelo de habitación estándar.

Este sistema domótico puede ser actualizado y ampliado si se desea cambiar la estructura interna de la habitación, posibilitando que los recursos sean reaprovechables y así evitar gastos mayores que afectarían la inversión inicial realizada.

2.2.1 Beneficios de las habitaciones hoteleras con sistemas domóticos

El proceso para lograr la automatización exitosa de un hotel se basa en un amplio equipamiento tecnológico que se encarga de apoyar y ejecutar con éxito el cumplimento de los requerimientos de cada habitación (Zennio, 2018a).

1. Pantallas táctiles a medida y diseños personalizados.
2. Diseños modernos y funcionales, abiertos a cambios y ampliables
3. Tecnología fiable y protección de la inversión
4. Ahorro y disminución de costes (iluminación, calefacción, refrigeración)
5. Máximo rendimiento de la instalación técnica de la habitación
6. Confort memorable y mejora de la experiencia del huésped
7. Bienestar y satisfacción cuando se utilizan las instalaciones de la habitación
8. Optimización del consumo energético 25-45 % del ahorro medio de energía
9. Conciencia medioambiental
10. Retorno de la inversión media de 3 años.

2.3 Elementos de hardware que componen al sistema domótico

El sistema domótico diseñado requiere para su funcionamiento un conjunto elementos de hardware, siendo el principal el microcontrolador que permite controlar el resto con el propósito de cumplir con cada una de las especificaciones domóticas trazadas. A continuación, será descrito cada sensor y dispositivo utilizado durante la elaboración de la propuesta:

2.3.1 Placa Arduino Mega 2560 R3

El Arduino Mega 2560 es una placa electrónica basada en el microprocesador Atmega 2560, tiene 54 entradas/salidas digitales (de los cuales 14 pueden utilizarse para salidas PWM), 16 entradas analógicas, cuatro puertos seriales (UARTs), un oscilador de 16 MHz, una conexión USB, un conector de alimentación, un cabezal ICSP y un botón de reseteo (Chipuxi, 2015).

Tabla 2.1. Características de Arduino Mega 2560 (Chipuxi, 2015)

Elementos	Valores
Microcontrolador	Atmega 2560
Voltaje de Operación	5V
Voltaje de Entrada (recomendado)	7-12V
Voltaje de Entrada (límite)	6-20V
Pines de E/S Digitales	54 (14 salidas PWM)
Pines de Entrada Analógica	16
Corriente DC por el pin E/S	40mA
Corriente DC para el pin 3.3V	50mA
Memoria Flash	256KB
SRAM	8KB
EEPROM	4KB
Velocidad del Reloj	16MHz

Descripción de los elementos que componen a la placa Arduino Mega 2560 (Chipuxi, 2015):

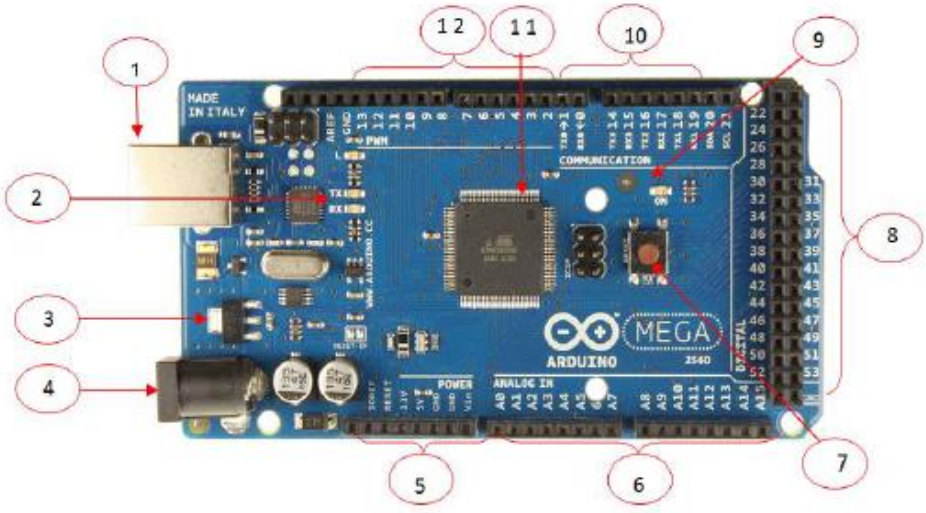


Figura 2.2. Componentes de la placa Arduino Mega 2560.

1. Conector USB
2. LEDs de Rx y Tx
3. Regulador de voltaje 5V

4. Plug de conexión para fuente de alimentación externa, el voltaje suministrado es de 6-20V
5. Puerto de conexiones: Reset
   Pin 3.3V
   Pin 5V
   GND
   Pin Vin
6. Puertos de entrada analógicas
7. Botón reset
8. Puertos de entradas/salidas digitales
9. LED On
10. Pines de Tx y Rx
11. Microcontrolador Atmega 2560
12. Puertos de entradas/salidas digitales, los pines digitales se pueden utilizar como una entrada o salida, usando funiciones (pinMode(), digitalWrite(), y digitalRead()). Operan a 5 V, y cada pin puede proporcionar o recibir un máximo de 40 mA y tiene una resistencia de pull-up (desconectado por defecto) de 20-50 kΩ.

Interrupciones externas: 2 (interrupción 0), 3 (alarma 1), 18 (interrumpir 5), 19 (interrumpir 4), 20 (interrumpir 3), y 21 (interrumpir 2). Estos pines pueden ser configurados para activar una interrupción en un valor bajo, un flanco ascendente o descendente, o un cambio en el valor, para el manejo de interrupciones se utiliza attachInterrupt ().

PWM: 2 a 13 y 44 a 46 para salidas PWM de 8 bits con la función () analogWrite.

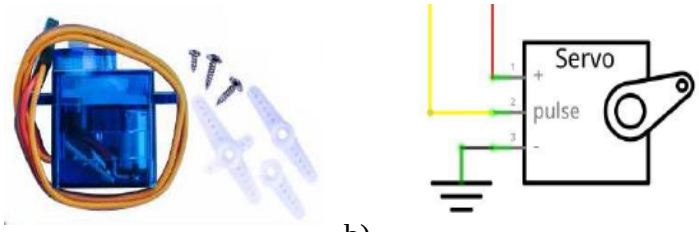
SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). Estos pines soportan comunicación SPI. Estos pines también se desglosan en la cabecera ICSP.

TWI: 20 (SDA) y 21 (SCL). Estos pines soportan comunicación TWI utilizando la librería Wire.

El Arduino Mega 2560 R3 tiene 16 entradas analógicas, cada una de las cuales proporcionan 10 bits de resolución (1024 valores diferentes).

2.3.2 Servo Motor SG90

Servo es un tipo de motor reductor que sólo puede girar 180 grados, y se controla mediante el envío de impulsos eléctricos de la placa Arduino Mega 2560 R3. Estos pulsos le indican al servo a que posición se debe mover. Contiene tres cables: GND, 5V, y señal (Inc. Elegoo, 2011).



a) b)

Figura 2.3. a) Servo Motor SG90. b) Conexiones del Servo Motor SG90.

Tabla 2.2. Características del SG90 (Inc. Elegoo, 2011)

Elementos	Valores
Longitud del cable	25 cm
Velocidad sin carga	0.12 s/60° (4.8 V) ó 0.10 s/60° (6.0 V)
Torque impuesto (4.8 V)	1.6 kg/cm
Temperatura	30~60ºC
Ancho de banda muerta	5 µs
Voltaje de operación	3.5~6 V
Dimensiones	3.2 cm x 3 cm x 1.2 cm
Peso	134 g

2.3.2 Módulo de receptor IR

Los detectores infrarrojos (IR) son pequeños microchips con una célula fotoeléctrica que están configurados para recibir luz infrarroja. Se utilizan para detectar la señal de un control remoto conectado al Arduino Mega 2560. Dentro del control remoto existe un LED IR, que emite pulsos IR con el objetivo de crear una acción sobre el sistema. Son especialmente

filtrados para IR ligero y poseen un demodulador para IR modulada a 38 kHz (Inc. Elegoo, 2011).

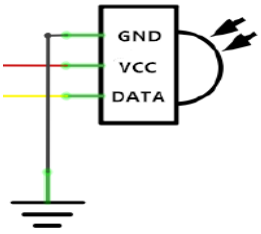
a)  b)  c) 

Figura 2.4. a) Control remoto IR. b) Módulo de receptor IR. c) Conexiones del módulo receptor IR.

Tabla 2.3. Características del control remoto IR (Inc. Elegoo, 2011)

Elementos	Valores
Alcance	>8m
Longitud de onda de la señal IR emitida	940Nm
Frecuencia de oscilación del cristal	455 kHz
Frecuencia de la señal infrarroja	38 kHz
Codificador	formato de codificación de la NEC, esquema de codificación upd612216
Dimensiones	86 x 40 x 6 mm
Batería	CR2025/160mAH
Botón	altura < 3mm, fuerza de 200-350g

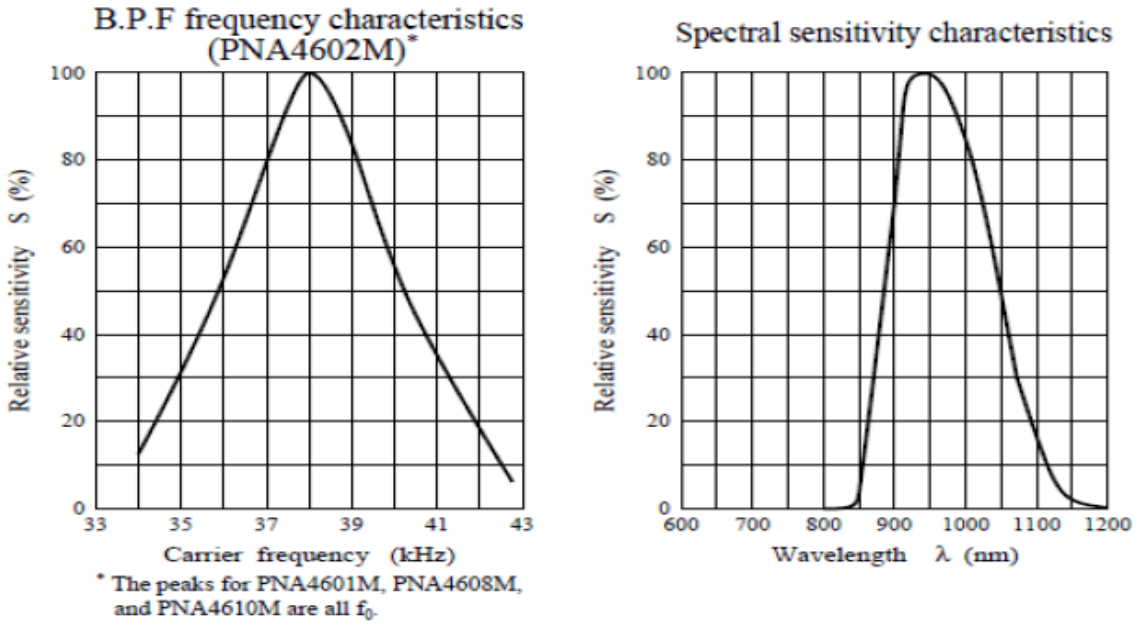


Figura 2.5. Gráficos de características para la medición de los receptores IR.

En los gráficos anteriores se observa que el pico de frecuencia ocurre en 38 kHz y el pico de la sensibilidad espectral del LED es de 940 nm. Se puede utilizar Arduino Mega 2560 con frecuencias de un rango de 35 kHz hasta 41 kHz, pero la sensibilidad se atenuará para que el LED no detecte en tan amplio rango. Asimismo, se pueden utilizar LEDs de 850 a 1100 nm, pero se logra un mejor funcionamiento con LEDs con sensibilidad espectral de 900 a 1000 nm (Inc. Elegoo, 2011).

2.3.3 Resistencia

Son elementos que se resisten al flujo de electricidad y son representadas por la unidad Ohm (Ω). Los valores más comunes de resistencia utilizados durante la realización de la propuesta son: 220 Ω, 1 kΩ, y 10 kΩ. Los valores de resistencia son determinadas por el código de colores en ellas, y son capaces de conectarse de cualquier manera dado que no poseen polaridad (Inc. Elegoo, 2011).

a) b)

Figura 2.6. a) Resistencias con código de colores. b) Símbolo eléctrico de la resistencia.

2.3.4 LED

El Diodo Emisor de Luz (LED) se utiliza para indicar acciones o simular eventos. Es un elemento que requiere de poca electricidad y que posee una gran vida útil. Los utilizados para la confección de la propuesta poseen un diámetro de 5 mm y su color depende de la acción que se quiere ejecutar. Estos son conectados con una resistencia para limitar la cantidad de corriente que fluye a través de ellos, y así evitar su ruptura; de igual manera según el valor de la resistencia utilizada así será la intensidad de la luz del LED (Inc. Elegoo, 2011).

a) b)

Figura 2.7. a) LEDs de distintos espectros de luz. b) Símbolo eléctrico del LED.

2.3.5 Potenciómetro

El potenciómetro permite variar la resistencia de 0-10 KΩ y así cambiar el valor del voltaje de entrada en el pin V0 de la pantalla LCD, con el objetivo de lograr un contraste adecuado para la visualización de los caracteres mostrados. Los pines de entrada al potenciómetro son VCC y GND. (Inc. Elegoo, 2011).

Figura 2.8. Potenciómetro de 1-10 kΩ.

2.3.6 Módulo de Reloj Digital DS3231 RTC

El DS3231 es un módulo de reloj que muestra el año, mes, día, minuto, segundo, y semana. Es un simple chip de tiempo que tiene una batería integrada, por lo que el reloj puede seguir teniendo tiempo incluso cuando esta desconectada del transformador (Inc. Elegoo, 2011).

a) 

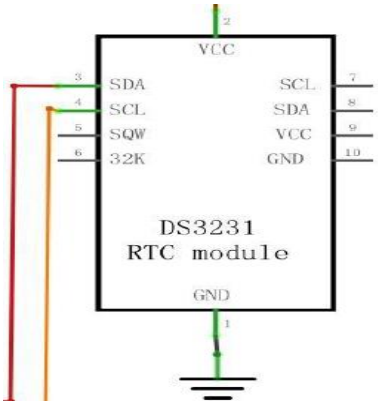
b) 

Figura 2.9. a) DS3231 RTC. b) Conexiones del DS3231 RTC.

2.3.7 Fotocelda

La fotocelda utilizada es un resistor dependiente de luz (LDR). La resistencia cambia en respuesta a cuanta luz está incidiendo sobre ella. Esta tiene una resistencia cerca de 50 kΩ en oscuridad y 500 Ω en luz brillante. Para convertir el valor variable de la resistencia en voltaje, se utiliza una resistencia fija que permita medir cuanto ha variado el valor de resistencia con la intensidad de la luz. La resistencia y la fotocelda se comportan como una sola; cuando la luz es muy brillante la resistencia de la fotocelda es muy baja en comparación con la resistencia de valor fijo, y es como si se girara a los 5V; de modo contario, cuando la fotocelda está en luz apagada, la resistencia es mayor que la resistencia fija de 1kΩ y es como si el conjunto girara hacia a GND (Inc. Elegoo, 2011).

a) 

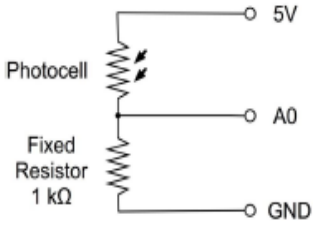
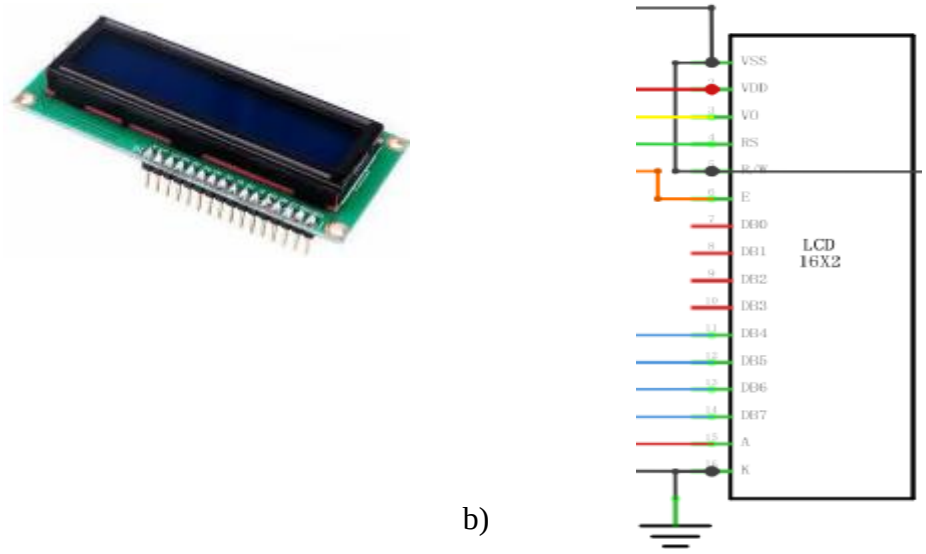
b) 

Figura 2.10. a) Fotocelda. b) Conexiones de la fotocelda con la resistencia fija.

2.3.8 Módulo de pantalla LCD 1602

El Cristal Líquido Digital (LCD) alfanumérico tiene una retroalimentación de LED y puede mostrar dos filas con hasta 16 caracteres en cada fila. En la pantalla se puede observar los rectángulos para cada carácter y los píxeles que la componen. La pantalla LCD utilizada para la conformación del proyecto está diseñada para mostrar texto (Inc. Elegoo, 2011).



a) b)

Figura 2.11. a) Pantalla LCD 1602. b) Conexiones de la pantalla LCD 1602.

Pines de la pantalla LCD 1602:

VSS: conexión a GND

VDD: conexión a +5V

V0: pasador para el ajuste del contraste de la pantalla

RS: Un pin de registro que controla en qué posición de memoria de la pantalla LCD se está escribiendo los datos. Se pueden seleccionar los datos que se registran, lo que pasa en la pantalla, o un registro de instrucción, que es donde el controlador busca para sostener instrucciones sobre qué acción tomar.

R/W: pin de Lectura/Escritura para seleccionar el modo de lectura o escritura

E: pin que, abastecido de un nivel bajo de energía, permite al módulo LCD ejecutar instrucciones adecuadas.

D0-D7: pines para leer y escribir datos

A y K: pines que controlan la retroiluminación de los LED

2.3.9 Módulo conmutador de membrana

El teclado conmutador de membrana le permite determinar al Arduino Mega 2560 R3 las teclas que están siendo presionadas. Es utilizado para la entrada de caracteres por el usuario. Para este proyecto se utiliza el teclado matricial, este sigue un esquema de codificación que le permite tener muchos menos pines de salida que llaves, por lo tanto, menos conexiones a hacer para que el teclado funcione. El teclado de matriz se caracteriza por tener 16 teclas, pero sólo 8 pines de salida (Inc. Elegoo, 2011).

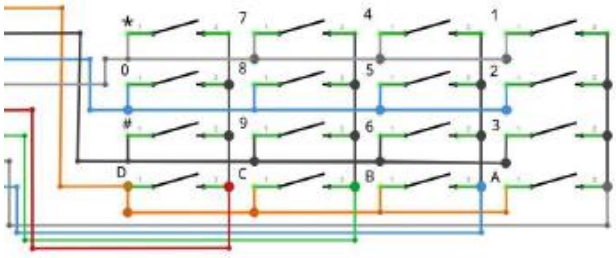
a)  b) 

Figura 2.12. a) Módulo conmutador de membrana. b) Conexiones del teclado de matriz.

2.3.10 Sensor de movimiento HC-SR501 PIR

El sensor PIR dispone de dos ranuras, cada ranura está hecha de un material especial que es sensible al IR. Cuando el sensor está inactivo, ambas ranuras detectan la misma cantidad de IR, la cantidad ambiente irradiada desde la habitación o las paredes o al aire libre. Cuando pasa un cuerpo cálido como un humano o un animal, primero intercepta la mitad del sensor PIR, lo que provoca un cambio diferencial positivo entre las dos mitades. Cuando el cuerpo caliente abandona el área de detección, ocurre lo contrario, por lo que el sensor genera un cambio diferencial negativo. Los pulsos de cambio son los que se detectan. Cuando el SR501 detecta cambios infrarrojos, se interpreta como movimiento, y se establece su salida en bajo. El dispositivo requiere casi un minuto para inicializar, y puede detectar movimiento dentro de un cono de 110° con un rango de 3 a 7 metros (Inc. Elegoo, 2011).

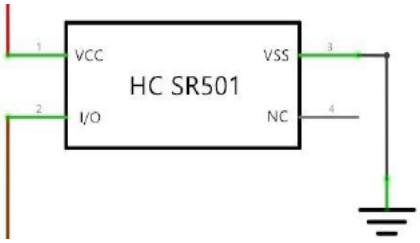
a)  b) 

Figura 2.13. a) Sensor de movimiento HC-SR501 PIR. b) Conexiones del HC-SR501 PIR.

Tabla 2.4. Características técnicas del HC-SR501 PIR (Inc. Elegoo, 2011)

Pin o Control	Función
Ajustar el tiempo de retardo	Establece cuánto tiempo la salida permanece alta después de detectar movimiento a partir de 5 s a 5 min
Ajuste de sensibilidad	Establece el rango de detección de 3-7m
Jumper de selección de disparo	Establece uno o varios disparos
Conector de tierra	Entrada de tierra
Pin de salida	Bajo cuando no se detecta movimiento. Alta (3.3 V) cuando se detecta movimiento
Pin de energía	Entrada de 5~20V_{DC}

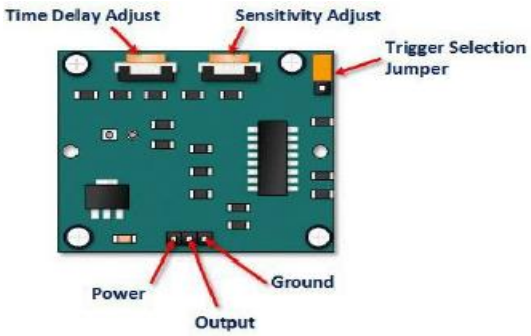


Figura 2.14. Pines de configuración del módulo HC-SR501 PIR.

2.3.11 Interruptor

Los interruptores tienen como función permitir o interrumpir el paso de la corriente eléctrica de manera momentánea. Disponen de enclavamiento por cambio de posición y los contactos se cierran o abren según el estado del interruptor (Inc. Elegoo, 2011).

a)

b)

Figura 2.15. a) Interruptor mini deslizable de 2 pasos y 3 pines. b) Diagrama estructural del interruptor de 2 pasos y 3 pines.

2.3.12 Módulo de suministro de energía

El módulo de alimentación se utiliza para proporcionar la energía que necesita el motor de corriente continua, y así evitar dañar al Arduino Mega 2560 R3 (Inc. Elegoo, 2011).

Figura 2.16. Módulo de alimentación.

Las principales especificaciones del módulo de alimentación son:

Interruptor de bloqueo On/Off

LED indicador de energía

Entrada de voltaje: 6.5~9V (CC)

Salida de voltaje: 3.3V/5V

Máxima salida de corriente: 700mA

Alimentación independiente de salida por el riel de control:0, 3.3, 5V

Pines principales de salida para usos extremos

Tamaño: 2.1 pulgadas x 1.4 pulgadas

Conector USB sobre la placa para la alimentación de dispositivos externos

2.3.13 L293D

Este chip se utiliza para el control de dos motores independientes. El L293D está diseñado para proporcionar corrientes de impulsión bidireccional de hasta 600 mA en voltajes de 4.5~36V. Todas las entradas son TTL compatible, y cada salida es un circuito conductor completo polo-totem, con un transistor Darlington y una fuente de pseudo-Darlington (Inc. Elegoo, 2011).

a) b)

Figura 2.17. a) Chip L293D. b) Conexiones del L239D

Las principales especificaciones del chip L239D son:

Tensión de alimentación: 4.5V a 36V

Alimentación de entrada lógica separada

Protección interna ESD

Apagado térmico

Alta inmunidad de ruidos de entrada

Salida de corriente: 1 A por canal (600mA)

Máxima salida de corriente: 2 A por canal (1.2)

2.3.14 Motor DC

El motor DC utilizado es el estándar de tamaño 130, y vienen con un amplio rango de operación de 4.5 a 9 VDC (Inc. Elegoo, 2011).

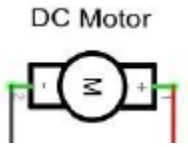
a)  b) 

Figura 2.18. a) Motor DC. b) Conexiones del motor DC.

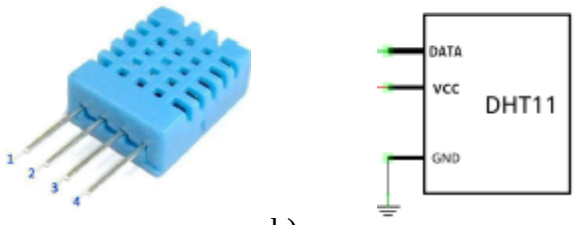
Tabla 2.5. Características del motor DC (Inc. Elegoo, 2011)

Elementos	Valores
Temperatura de operación	-10°C a +60°C
Voltaje recomendado	6 V_{DC}
Carga recomendada	10 g*cm
Corriente sin carga	70 mA máx.
Velocidad sin carga	9100 ± 1800 rpm
Corriente con carga	250 mA máx.
Velocidad con carga	4500 ± 1500 rpm
Torque de arranque	20 g*cm
Voltaje de arranque	2V
Corriente de parada	500 mA máx.
Dimensiones del eje	8mm x 2mm de diámetro
Peso	17.5 g

2.3.15 Módulo de temperatura y humedad DHT11

El sensor digital de temperatura y humedad DHT11 contiene la salida de la señal digital calibrada de la temperatura y la humedad. Incluye un sentido resistente de componentes

mojados, un dispositivo de medición de temperatura , y conexiones con un microcontrolador de 8 bits de alto rendimiento (Inc. Elegoo, 2011).



a) b)

Figura 2.19. a) Módulo de temperatura y humedad DHT11. b) Conexiones del DHT11.

Tabla 2.6. Características del módulo DHT11 (Inc. Elegoo, 2011)

Elementos	Valores
(Humedad relativa)	
Resolución	16 bits
Repetibilidad	±1% H.R.
Precisión	25°C ±5% hr
(Intercambiabilidad)	intercambiables
Tiempo de respuesta	1/e (63%) de 25° 6s
Alimentación	3.5 ~5.5V_{DC}
Corriente	medición: 0.3mA, espera: 60µA
Período de muestreo	>2s

Pines del módulo DHT11:

V_{DD}: alimentación

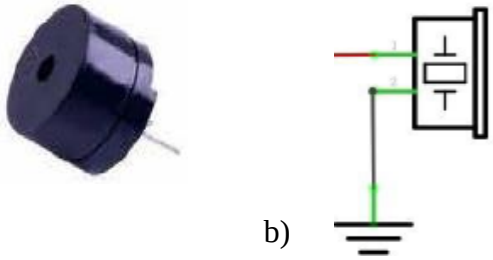
Data: serie de datos

NC: pin vacío

GND: tierra

2.3.16 Zumbador

El funcionamiento del zumbador pasivo se basa en utilizar el modulador de ancho de pulso (PWM, por sus siglas en inglés) para generar audio para hacer que el aire vibre. Cambiando la frecuencia de vibración se pueden generar diferentes sonidos. Las frecuencias a utilizar para generar un pulso de timbre son mayores a 500 Hz (Inc. Elegoo, 2011).



a) b)

Figura 2.20. a) Zumbador pasivo. b) Conexiones del zumbador pasivo.

2.4 Herramientas de software utilizadas en el diseño de la propuesta domótica

El software es un elemento fundamental en la conformación de un sistema domótico, dado que, es necesario para la programación de eventos en las habitaciones del hotel. A continuación, se describirá la plataforma utilizada:

2.4.1 Arduino 1.8.7

El entorno de desarrollo integrado (IDE, por sus siglas en inglés) de Arduino en su versión 1.8.7 es el software utilizado para programar las funciones a controlar por el microcontrolador. En esta plataforma de programación se crean todas las aplicaciones que conforman el sistema domótico, y se comunica con el hardware usando el protocolo STK500 original (Inc. Elegoo, 2011). Posee una interfaz sencilla y el lenguaje de programación tiene la misma sintaxis que C++, excepto en la estructura del programa, puesto que no cuenta con una función *main()*. En cambio, cada programa cuenta con dos funciones principales que aparecen en el área de trabajo al iniciar cada proyecto: *setup()*, que se ejecuta una sola vez al iniciar el programa y *loop()* que constituye un ciclo infinito (Chipuxi, 2015). El lenguaje incluye las funciones necesarias para realizar las acciones básicas con el microcontrolador. Además de las bibliotecas estándares de C y C++, cuenta con una serie de bibliotecas que brindan una gran cantidad de opciones, por ejemplo, *Servo.h*, *IRremote.h*, *Keypad.h*, *LiquidCrystal.h, y Wire.h*. Una gran ventaja es que dada la gran comunidad internacional de

Arduino y la posibilidad de ser software libre y extensible, existe gran cantidad de bibliotecas creadas por miembros de la comunidad, accesibles gratuitamente, entre ellas, las que brindan facilidades de comunicación y/o control sobre gran cantidad de dispositivos, y sensores (Sánchez, 2012).

2.5 Aplicaciones domóticas diseñadas

El sistema domótico que se diseña en este proyecto está estructurado a partir de varias aplicaciones, que serán la que agregarán funcionalidad a la habitación. A continuación, se describe cada una de ellas.

2.5.1 Módulo de apertura y cierre de cortinas

Esta aplicación consiste en el control automático de la apertura y cierre de las cortinas de la habitación. Se han creado tres modos según las preferencias del cliente de la habitación. Para cambiar de modos se hace presionando el botón "*power*" del control remoto, dichos modos serán explicados a continuación:

Modo Manual: Está activo cuando se enciende el LED amarillo. Este modo es de prioridad, dado que el usuario será el responsable de las acciones que el sistema realizará a partir de sus decisiones personales. A partir del control remoto el usuario controlará el nivel de apertura o cierre de las cortinas de la habitación. Presionando el botón "*FAST FORWARD*" el servomotor será capaz de abrir completamente la cortina; de manera contraria, si se presiona el botón "*FAST BACK*" el servomotor se moverá 180° de manera que cerrará completamente las cortinas. Ahora, en el caso de presionar la tecla "*VOL+*" el servomotor solo se moverá de manera incremental 10°, dándole al usuario la opción de solamente abrir la cortina una distancia deseada por el mismo; en el caso contrario, si se presiona la tecla "*VOL-*" el servomotor se moverá de manera que se decrementará 10°, permitiendo al usuario cerrar la cortina según desee.

Modo Horario: Será activado cuando se enciende el LED rojo. Este modo es un caso especial, dado que le permitirá al usuario introducir los horarios en que desea que se abran y se cierre las cortinas. El horario introducido por el cliente puede ser modificado por el propio cliente en caso de que cambie de opinión acerca del estado de las cortinas. Los horarios establecidos son introducidos a través de un teclado de membrana y son mostrados en una pantalla LCD

para que el huésped tenga noción del momento en que se activará el actuador. El horario mostrado en la pantalla LCD se encuentra en formato de 24 horas.

Modo Automático: Está activo cuando está encendido el LED azul. El modo automático está diseñado para que el controlador se encargue por si solo de gestionar en que momentos la cortina debe abrirse o cerrarse en dependencia de la intensidad de la luz natural. Para determinar qué nivel de intensidad tiene la luz solar se utiliza una fotocelda, y estableciendo un umbral el microcontrolador Arduino Mega 2560 acciona un actuador, en este caso el servomotor que será el encargado de abrir o cerrar las cortinas. En caso de que sea de día, las cortinas permanecerán abiertas para que pueda entrar la luz a la habitación, contrariamente, si es de noche las cortinas permanecerán cerradas para el mejor provecho de la luz artificial y lograr que el cliente disfrute de mayor privacidad en su horario de sueño.

Hardware utilizado:

- Microcontrolador Arduino Mega 2560
- 4 Resistencias de 1K
- LEDs amarillo, rojo, azul
- Control remoto IR
- Módulo de receptor IR
- Módulo conmutador de membrana
- Pantalla LCD 1602
- Potenciómetro
- Servomotor SG90
- Fotocelda
- Reloj Digital DS3231

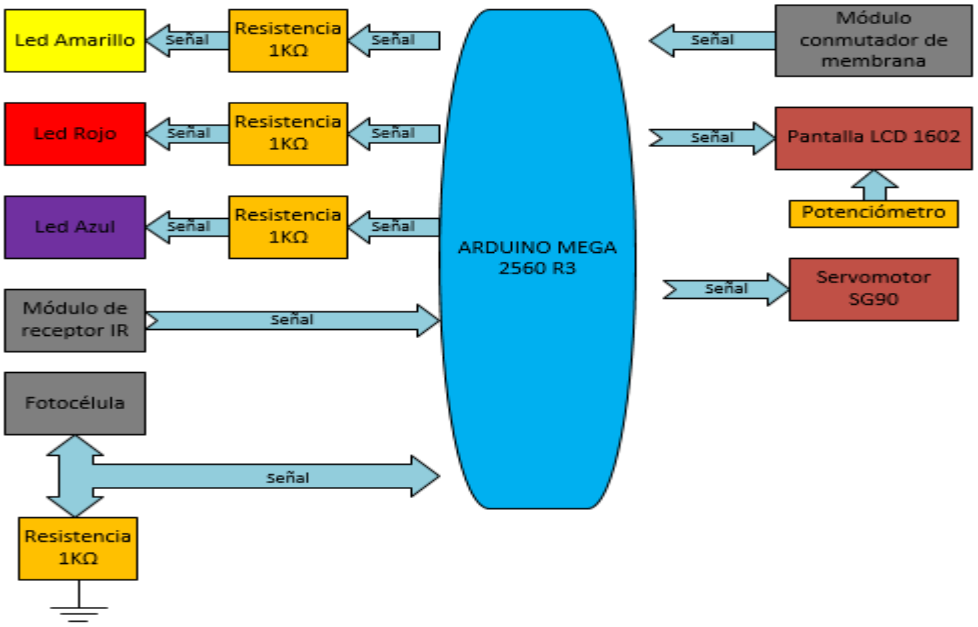


Figura 2.21. Arquitectura de hardware del módulo de apertura y cierre de las cortinas.

La arquitectura de hardware del módulo de apertura y cierre de las cortinas se muestra en la Figura 2.21, en ella se especifica el sentido de la señal, si es una medición o comando de control, o si es una orden dada por el microcontrolador a los actuadores y demás dispositivos que conforman el sistema. El componente principal lo constituye el Arduino Mega 2560, que representa la unidad de procesamiento que controlará que el cumplimiento de todas las funciones creadas. El Arduino se comunica con la computadora por el puerto serie a través de un cable USB, permitiendo la posibilidad de transmisión y recepción de grandes paquetes de datos que dirigen todo el funcionamiento del módulo. Las conexiones de todos los dispositivos al Arduino se realizan a través de cables "*Macho a Macho*" y "*Hembra a Macho*". Los pines GND y VCC (5V) del Arduino son utilizados para la alimentación de los dispositivos que conforman el módulo. Las conexiones se detallan continuación:

- Servomotor SG90: pin digital 9 PWN
- Módulo de receptor IR: pin digital 10 PWN
- LED amarillo/rojo/azul: pin digital 24/26/22
- Reloj Digital DS3231: pin SCL y SDA
- Fotocelda: pin analógico A0
- Pantalla LCD 1602: pines digitales 2,3,4,5,6,7

- Módulo conmutador de membrana filas/columnas: pines digitales [53,52,51,50]/ [49,48,47,46]

La programación del módulo creada para el control automático de la apertura y cierre de las cortinas de la habitación utiliza las bibliotecas de Arduino: *Servo.h*, *Wire.h*, DS3231.h, *LiquidCrystal.h*, *IRremote.h*, *Keypad.h*, dichas bibliotecas poseen una estructura propia que facilita la programación del hardware utilizado. Las variables *PIN_LED_ESTADO*, *PIN_LED_USUARIO*, y *PIN_LED_HORARIO* indican en qué estado se encuentra el sistema: modo automático, modo manual, y modo horario respectivamente. La variable *UMBRAL_LUZ 1* define el nivel de intensidad de la luz natural. Dentro de la función *setup ()* se definen todas las señales de entrada y salida del Arduino, en ella se establece que en la pantalla LCD se muestre en todo momento el horario de apertura y cierre de las cortinas para que el usuario tenga siempre noción del horario en que sucederá dicho evento. En la función *loop ()* se establece un chequeo para determinar cuando el módulo receptor IR recibe una señal y que valor posee, también se establecerá el estado de cada LED según el modo escogido por el usuario (automático, manual, horario). En la función *translatedIR ()* se establecen los comandos para seleccionar los modos; y los comandos para el caso del modo manual, dado que el usuario determinará qué acción realizará el actuador a través de los botones de su control remoto IR, en ella se incluye un comando para dar la posibilidad de cambiar el horario de apertura y cierre de cortinas si el usuario cambia de opinión desde su última modificación. Dentro de la función creada por el programador: *modoAutomatico ()*, se realiza una lectura de la señal enviada por la fotocelda al microcontrolador, para así comparar ese valor con el umbral establecido por el programador, y determinar qué acción debe realizar el actuador según el nivel de intensidad de la luz natural. En el caso de la función creada para el modo manual: *modoManual ()*, el usuario a través de los comandos de su control remoto decide que acción realizará el actuador. La próxima función denominada Reloj () permite que el tiempo y la fecha sean mostrados por el Monitor Serial del software ino de Arduino. Las funciones *ConfigHoraAbrir ()* y ConfigHoraCerrar () se emplean para que el usuario a través del teclado conmutador de membrana pueda introducir el horario de apertura y cierre de las cortinas, eliminando posibles errores que pueden ocurrir durante la configuración, y en donde queda establecido un formato de 24 horas. Por último, la

función *modoHorario ()* se encarga determinar qué acción realizará el actuador según el horario establecido por el usuario. En el Anexo VIII se encuentra el programa de control de este módulo.

2.5.2 Módulo de seguridad del closet

Esta aplicación es creada con el propósito de proteger los bienes de valor del cliente hospedado en la habitación. Consiste en asegurar mediante contraseña la apertura de los closets de las habitaciones. A continuación, se detallará como funciona dicha aplicación:

En esta aplicación se trata de controlar el movimiento de un servomotor para simular un llavín en la puerta del closet. Todas las acciones que se realizan en el sistema son ordenadas por el control remoto IR y procesadas por el Arduino Mega 2560. El primer paso que se realiza es guardar una nueva contraseña de cuatro números en el sistema, este paso se ejecuta cuando un nuevo cliente es registrado en la habitación, el encargado de la entrega de las habitaciones presiona el botón "*UP*" de su control remoto y hace que se muestre en una pantalla LCD el texto: 'Save Password', y en ese momento el cliente introducirá la nueva contraseña a través del teclado conmutador de membrana. Presionando el botón "*DOWN*" del control remoto del usuario, el mismo podrá abrir la puerta del closet cada vez que lo desee, dado que en la pantalla LCD será mostrado el texto 'Intro Password'. Cuando la contraseña introducida es correcta el servomotor se mueve 180° y la puerta del closet puede ser abierta; en el caso de que el usuario presione la tecla para introducir la contraseña estando la puerta abierta se mostrará en pantalla 'Closet abierto', para que el usuario este informado del estado de la puerta y pueda saber por qué no lo deja introducir la contraseña, para simular el estado de la puerta se utiliza un interruptor minideslizable de 2 pasos y 3pines. Cuando la puerta es cerrada el servomotor se mueve 180° en sentido inverso para dejar el closet asegurado.

Un caso extraordinario ocurre cuando es tres veces introducida incorrectamente la contraseña por el usuario. En este caso se activará un zumbador pasivo, que sonará 5 veces, seguido por un LED azul que encenderá de manera intermitentemente, ambos ubicados en el local de seguridad y harán la función de alarma para avisar de la violación de seguridad del closet de la habitación al responsable de seguridad del hotel. Cuando el usuario introduce incorrectamente tres veces la contraseña el sistema se bloquea hasta que el jefe de seguridad

acude a la habitación para asegurarse que todo se encuentra en orden, en caso de no existir problemas de violación de la seguridad, entonces este encargado de la seguridad presionará en su control remoto el botón "*PAUSE*" para apagar la alarma y permitir al húesped volver a introducir la contraseña correctamente. Si sucede que el cliente de la habitación olvidara la contraseña, el encargado de la entrega de la habitación a través del botón "*UP*" de su control remoto, le aprobará al cliente la posibilidad de volver a guardar la contraseña que desee.

Hardware utilizado:

- Microcontrolador Arduino Mega 2560
- 1 Resistencia de 1K
- LED azul
- Control remoto IR
- Módulo de receptor IR
- Módulo conmutador de membrana
- Pantalla LCD 1602
- Potenciómetro
- Interruptor minideslizable de 2 pasos y 3 pines
- Servomotor SG90
- Zumbador Pasivo

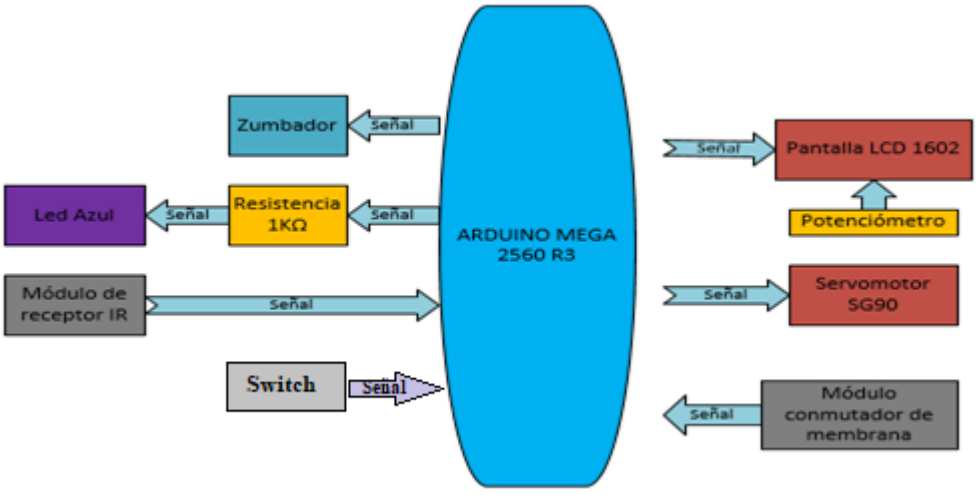


Figura 2.22. Arquitectura de hardware del módulo de seguridad del closet.

La arquitectura de hardware del módulo de seguridad del closet se muestra en la Figura 2.22. La unidad de procesamiento o microcontrolador (Arduino Mega 2560 R3), se encarga de analizar todas las señales de entrada, para determinar la respuesta que le corresponde. El Arduino utiliza un cable USB para conectarse con el puerto serie de la computadora. El actuador principal es el servomotor SG90, que simula el funcionamiento de un llavín que asegura la puerta del closet. Es importante destacar, que cada operario del hotel poseerá un Control Remoto IR según su función en el hotel, con el objetivo de asegurar el correcto funcionamiento del sistema, pues cada uno de sus botones está representado por un código único, que se encuentra implementado en el programa almacenado en la memoria del Arduino, lo que permitirá que solo la persona responsable de ese control pueda realizar la acción que corresponde a ese botón. El potenciómetro conectado a la pantalla LCD se utiliza para ajustar el contraste, pues dependiendo del medio donde sea ubicada dependerá su nivel de visualización. Las conexiones de todos los dispositivos al Arduino se realizan a través de cables "*Macho a Macho*" y "*Hembra a Macho*". Los pines GND y VCC (5V) del Arduino son utilizados para la alimentación de los dispositivos que conforman el módulo. Las conexiones se detallan continuación:

- Servomotor SG90: pin digital 9 PWN
- Módulo de receptor IR: pin digital 11 PWN
- Zumbador Pasivo: pin digital 10 PWN
- LED azul: pin digital 38
- Pantalla LCD 1602: pines digitales 44,45,42,43,40,41
- Módulo conmutador de membrana filas/columnas: pines digitales [52,53,50,51]/ [48,49,46,47]
- Interruptor minideslizable de 2 pasos y 3 pines: pin digital 39

En la programación del módulo para el control automático de la seguridad del closet de la habitación se destaca el uso de las bibliotecas de Arduino: *Servo.h*, *LiquidCrystal.h*, *IRremote.h*, y *Keypad.h*. Las variables que definen las señales de entrada y salida del Arduino son: PIN_LED_ALARMA para el LED indicador de alarma, PIN_ZUMBADOR_ALARMA para el zumbador para la parte sonora de la alarma, estas dos variables son utilizadas para cuando existen indicios de violación de la seguridad introducida en el closet, PIN_PUERTA

para el interruptor que define el estado en que se encuentra la puerta (abierta/cerrada). Dentro de la función *setup ()* del programa se inicializan las variables que representan las señales de entrada y salida del Arduino, y se define en qué estado se encontrarán al inicio. La función *loop ()* se utiliza para determinar si el módulo receptor de IR está recibiendo señal y que valor posee. En el caso de la función *translate IR ()* se establecen los comandos para modificar e introducir la contraseña, y para apagar la alarma en caso de una posible violación de la seguridad. Las funciones *Zumbador 1 ()* y *Zumbador ()* se encargan de hacer sonar el zumbador una vez cada vez que un usuario introduzca mal la clave, y cinco veces cuando sea introducida incorrectamente tres veces. La función *ConfigContrasena ()* es la encargada de guardar/modificar la contraseña de cuatro números que introduce el usuario para asegurar el closet. Mientras que la función *Comparacion ()* se encargará de permitir que el usuario introduzca una combinación para comparar con la guardada y determinar si la puerta del closet debe abrirse o activarse la alarma, además es agregada una interrupción para el estado abierto de la puerta, con el objetivo de no permitir que el usuario introduzca nuevamente la clave hasta que la puerta sea cerrada, y así evitar fallas en la interfaz de hardware y software creada por el programador. En el Anexo IX se encuentra el programa de control de este módulo.

2.5.3 Módulo de encendido de las luces y control del clima

El encendido de las luces y el control del clima es una aplicación conjunta dado que, dependen ambos de los sensores de presencia PIR y de los interruptores situados en las puertas de entrada y del balcón para determinar en qué estado están (abiertas o cerradas). Para simular el encendido de las luces se utilizan LEDs, mientras que el clima incluye el aire acondicionado y el ventilador de techo. El Modo Manual es representado por el LED rojo, y el Modo Manual por el LED azul. A continuación, se dará una mayor explicación de la aplicación:

Encendido de las luces: Cuando un nuevo cliente entra a la habitación se activa un interruptor ubicado en el llavín para determinar el estado de la puerta (abierta/cerrada), después de entrar el cliente, es cerrada la puerta, y el interruptor guarda un 1 por cada vez que este se haya estado activo en alto, y así poder saber que la puerta fue abierta. Cada vez que la puerta sea abierta los sensores PIR comprobarán durante 20 minutos si existe presencia de alguna

persona dentro de la habitación, en caso de estar ocupada por una persona, las luces correspondientes al sensor PIR que detecta presencia permanecerán encendidas; pero si a los 20 min se determina que la habitación se encuentra vacía las luces se apagarán. Ahora, en caso de activarse el interruptor de la puerta del balcón, y estar en estado de puerta abierta, se encenderá la luz del balcón y las luces internas serán apagadas instantáneamente, a menos que los sensores PIR dentro de la habitación detecten presencia de otra persona. Para que no se enciendan innecesariamente las luces durante el día se utiliza una fotocelda para comprobar si la intensidad de la luz es mínima o máxima. También se le dará prioridad al mando manual, en caso específico de que se necesite del uso de la luminaria para alguna funcionalidad.

Control del clima: En la puerta de la habitación se encuentra ubicado un interruptor para saber cuándo el cliente sale o entra. En el momento en que se activa el interruptor los sensores de movimiento chequean durante 20 minutos si existe presencia de personas, en caso de estar ocupada la temperatura de la habitación se ira regulando hasta alcanzar el valor mínimo del umbral de temperatura fresca establecido por la jefatura del hotel con el objetivo de aumentar la eficiencia energética. Igualmente, si el PIR ubicado en el dormitorio detecta presencia se encenderá el ventilador de techo hasta que se active nuevamente el interruptor de la puerta y los sensores de movimiento vuelvan a hacer el chequeo. Ahora, en el caso de que después de pasados los 20 minutos se determine que la habitación está vacía el aire acondicionado subirá su temperatura hasta 2°C menos que la temperatura exterior. Esta será medida a través de un sensor de temperatura ubicado en el exterior de la misma, y utilizado para comparar la temperatura dentro y fuera, con el objetivo de mantener la interna fresca durante la ausencia del usuario. Si el usuario permanece fuera de la habitación más de 2 horas entonces será suspendido el flujo de alimentación del aire acondicionado para hacer un apagado automático del mismo y será encendido nuevamente cuando el cliente vuelva a ocupar la habitación. La temperatura actual del aire acondicionado será mostrada en todo momento en la pantalla LCD.

Hardware utilizado:

- Microcontrolador Arduino Mega 2560
- 4 Resistencia de 1K

- 3 LEDs blancos
- Control remoto IR
- Módulo de receptor IR
- Pantalla LCD 1602
- Potenciómetro
- 2 Interruptores minideslizables de 2 pasos y 3 pines
- 2 Motores DC
- 2 sensores HC-SR501 PIR
- Fotocelda
- Reloj Digital DS3231
- 2 sensores DHT11

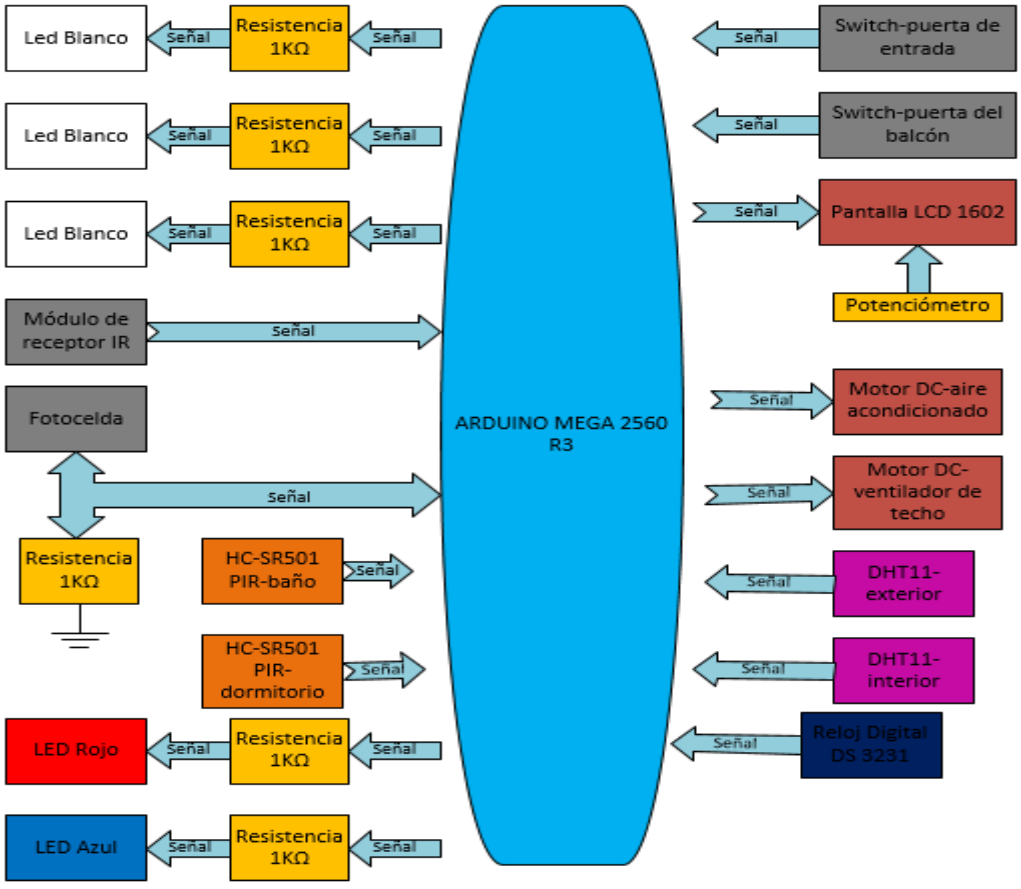


Figura 2.23. Arquitectura de hardware del módulo de encendido de las luces y control del clima.

La arquitectura de hardware del módulo de encendido de las luces y control del clima se muestra en la Figura 2.23. El cerebro que controla el funcionamiento del módulo lo representa el microcontrolador Arduino Mega 2560 R3, donde dependiendo de las señales de entrada de los sensores, dará una respuesta que activará los actuadores utilizados para simular el funcionamiento del módulo. La transmisión y recepción de información del Arduino con la computadora se realiza a través del puerto serie utilizando para ello un cable USB. Para simular el encendido de las luces, se recurre a los interruptores, los sensores de presencia HC-SR501, la fotocelda para medir la intensidad de la luz, y los LEDs. En el caso de la simulación del funcionamiento del clima en la habitación, se utilizan los interruptores, los sensores de presencia HC-SR501 PIR, los sensores de temperatura DHT11, la pantalla LCD 1602, y los motores DC. El módulo completo podrá ser controlado también en modo manual, modo que le permite al usuario seleccionar sus preferencias en el encendido de las luces y la temperatura de la habitación. Las conexiones de todos los dispositivos al Arduino se realizan a través de cables "*Macho a Macho*" y "*Hembra a Macho*". Los pines GND y VCC (5V) del Arduino son utilizados para la alimentación de los dispositivos que conforman el módulo. Las conexiones se detallan continuación:

- Módulo de receptor IR: pin digital 10 PWN
- LED Blanco (baño): pin digital 38
- LED Blanco (dormitorio): pin digital 39
- LED Blanco (balcón): pin digital 51
- LED Rojo: pin digital 52
- LED Azul: pin digital 53
- HC-SR501 PIR (baño): pin digital 8
- HC-SR501 PIR (dormitorio): pin digital 9
- Fotocelda: pin analógico A0
- Interruptor minideslizable de 2 pasos y 3 pines (entrada): pin digital 19
- Interruptor minideslizable de 2 pasos y 3 pines (balcón): pin digital 18
- Pantalla LCD 1602: pines digitales 22, 23, 24, 25, 26, 27
- Motor DC (ventilador de techo): pin digital 7
- Motor DC (aire acondicionado): pin digital 6

- DHT11 (interior): pin digital 4
- DHT11 (exterior): pin digital 2
- Reloj Digital DS3231: pin SCL y SDA

En la programación del módulo para el encendido de las luces y el control del clima, se destaca el uso de las bibliotecas de Arduino: *SimpleDHT.h*, *LiquidCrystal.h*, *IRremote.h*, y *Wire.h*. Las variables utilizadas para denominar el control de la luminaria son: LEDBano, LEDDormitorio, y LEDBalcon. Mientras que las variables para el control del clima son: PinVentilador y PinAire. Dentro de la función *setup ()* del programa se inicializan las variables que representan las señales de entrada y salida del Arduino, y se define en qué estado se encontrarán al inicio. La función *loop ()* se utiliza para determinar si el módulo receptor de IR está recibiendo señal y que valor posee, y en ella son llamadas las dos funciones principales del programa, a decir *modoManual ()* y *Automatico ()*, encargadas de controlar el desempeño del resto de las funciones, para lograr un control organizado del módulo. En el caso de la función *Automatico ()* se desarrollan todas las consideraciones que realiza el sistema para un correcto funcionamieno del módulo, estos son: chequeo del nivel de luminosidad, chequeo de los sensores de movimiento, y control del clima a través de las mediciones de temperatura realizada por los sensores DHT11. Dentro de la función translateIR () se definen los comandos utilizados para el control independiente de cada dispositivo por el control remoto IR, dichos dispositivos son: los LEDs del baño, dormitorio, balcón, el aire acondicionado y el ventilador de techo. Para una mejor claridad y desempeño del programa se utilizan interrupciones para el chequeo de la apertura y cierre de la puerta principal. El objetivo del programa es hacer una distribución inteligente del consumo energético por los dispositivos dentro de la habitación a través del control on/off de los mismos. En el Anexo X se encuentra el programa de control de este módulo.

2.6 Consideraciones finales del capítulo

En este capítulo se platearon inicialmente los requisitos de diseño en cuanto a funcionamiento, configuración, flexibilidad y seguridad con los que debe cumplir el dispositivo a diseñar, a partir del análisis de las características de los sistemas domóticos realizado en el primer capítulo. En correspondencia con estos requisitos se propuso la arquitectura de hardware y software a partir de la selección de los elementos más adecuados

que la componen. Luego del estudio y diseño realizado durante este capítulo, se determinaron las siguientes consideraciones:

La realización del diseño cumple con todos los requisitos expuestos al inicio del capítulo para así poder sobrepasar el nivel de automatización mínimo para estos sistemas.

Cada módulo diseñado es independiente del otro, por lo que cada uno posee su propio microcontrolador Arduino Mega 2560 R3.

Todos los módulos fueron programados a prueba de errores.

La arquitectura de hardware garantiza la interconexión exitosa entre todos los componentes del dispositivo para que este funcione correctamente.

La arquitectura de software está diseñada de manera tal que se cumplan todos los requisitos planteados al inicio del capítulo.

CAPÍTULO 3. VALIDACIÓN DEL SISTEMA

En el presente capítulo se presentan y discuten los resultados obtenidos tras la simulación del sistema domótico montado sobre la maqueta a escala. Posteriormente se realiza el análisis económico y medioambiental.

3.1 Resultados de la simulación

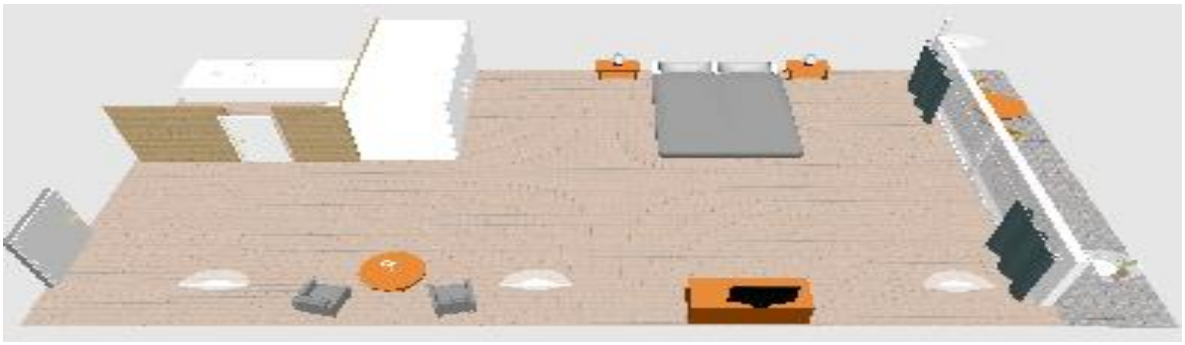
La realización de las pruebas experimentales se realiza a través de la implementación del sistema domótico propuesto en una maqueta a escala. El control de cada funcionalidad es separado por módulos con el objetivo de lograr una mayor organización y rendimiento. Cada módulo posee su propio Arduino Mega 2560 R3, los cuáles, a través de los programas almacenados en su memoria RAM son capaces de controlar todas las aplicaciones que fueron elegidas en el proyecto para la automatización de una habitación. Los Arduinos son energizados a través de un Adaptador con entrada de (100-240V, 50/60Hz, 0.35A) y salida de (9V, 1A) conectado a la red eléctrica nacional. En la figura 3.1 se muestra el boceto ideal de la maqueta a escala:

a)

c)

b)

d)



e)

Figura 3.1. a) Vista del balcón. b) Vista del cuarto. c) Vista del frente. d) Vista del baño. e) Vista superior.

En el Anexo I se muestra la maqueta a escala diseñada para la realización de las pruebas experimentales del sistema domótico propuesto para una habitación estándar de hotel.

3.1.1 Primer experimento

Objetivo: Evaluar el funcionamiento del módulo de apertura y cierre de cortinas.

Descripción: El módulo se conecta a la computadora a través del puerto serie utilizando un cable USB para cargar el programa en la memoria RAM del Arduino Mega 2560 R3. Luego, este dispositivo es energizado utilizando un adaptador de 9V y 1 A conectado a la red eléctrica. Seguidamente el módulo demora pocos segundos en arrancar el funcionamiento (el tiempo de retardo para el arranque depende de la cantidad de dispositivos conectados al microcontrolador, el nivel de complejidad de cada uno, y el tamaño de almacenamiento del programa) (5-6 segundos).

Este módulo es utilizado para el control automático de la apertura y cierre de las cortinas de la habitación. Al empezar a funcionar, por defecto se inicia en Modo Automático

representado por el LED azul, por tanto, la fotocelda empieza midiendo la intensidad de la luz solar (1V representa baja intensidad y 5 V alta intensidad), en dependencia de esos valores el servomotor SG90 se mueve hasta 10° o hasta 170°, o sea, abre o cierra las cortinas (si el valor de la medición es de 1V se cierran las cortinas en caso de que estén abiertas, de igual manera, si es 5V se abren las cortinas en caso de que estén cerradas). En Anexo II se muestran las imágenes de los componentes de hardware seleccionados para el funcionamiento del Modo Automático con las correspondientes conexiones.

Presionando una vez el botón *"Power"* se activa el Modo Horario representado por el LED rojo. Al activarse este modo, primeramente, se enciende la pantalla LCD mostrando en pantalla: "Hora de abrir: _: _", para que el cliente introduzca el horario en que desea que se abra la cortina, una vez introducido correctamente el horario se presiona la tecla *"A"* para pasar a escribir el horario de cerrar , luego se muestra durante dos minutos el texto: "Abrir: hora: minutos", para que el usuario sepa que introdujo bien el tiempo deseado; igualmente sucede con el horario de cerrado de las cortinas, pero con los comentarios: "Hora de cerrar: _: _" y "Cerrar: hora: minutos" . Al finalizar todo este proceso, se muestra permanentemente en la pantalla LCD la nota: "Abrir: hora: minutos" en la primera fila y "Cerrar: hora: minutos" en la segunda fila, para que el usuario tenga noción en todo momento del horario establecido. El formato de hora utilizado es de 24 horas. Este modo está diseñado a prueba de errores para que no se puedan introducir caracteres que no sean números, o horas y minutos errados, posee la posibilidad de borrar un caracter en caso de que el usuario se equivoque escribiendo (presionando la tecla *"D"*), y la función de modificar los horarios por si el huésped desea cambiarlos. En el Anexo III se muestran las imágenes de los componentes de hardware seleccionados para el funcionamiento del Modo Horario con las correspondientes conexiones.

Presionando nuevamente el botón *"Power"* se activa el Modo Manual, este modo funciona a través de los botones del Control Remoto IR. Presionando el botón "*VOL+*" el servomotor se incrementa en 10° cada vez que es presionado hasta el límite de 170°. Presionando el botón "*VOL-*" el servomotor se decrementa en 10° cada vez que es presionado hasta el límite de 10°. Presionando el botón "*FAST BACK*" el servomotor se mueve hasta 10°, de manera contraria, cuando se presiona el botón "*FAST FORWARD*" el servomotor se mueve hasta

170°. Para cuando el usuario desea cambiar el horario de apertura y cierre de las cortinas en el Modo Horario, se presiona el botón *"ST/REPT"*. En el Anexo IV se muestran las imágenes de los componentes de hardware seleccionados para el funcionamiento del Modo Manual con las correspondientes conexiones.

Conclusiones:

No existen fallas en el funcionamiento del sistema, cumpliendo con los requerimientos establecidos para este módulo. Se destaca el alto tiempo de respuesta que posee el sistema para realizar cada una de las funciones.

3.1.2. Segundo experimento

Objetivo: Evaluar el funcionamiento del módulo de seguridad del closet

Descripción: El módulo se conecta a la computadora a través del puerto serie utilizando un cable USB para cargar el programa en la memoria RAM del Arduino Mega 2560 R3. Luego, este dispositivo es energizado utilizando un adaptador de 9V y 1 A conectado a la red eléctrica. Seguidamente el módulo demora pocos segundos en arrancar el funcionamiento (2-4 segundos).

Este módulo es utilizado para el control automático de la apertura de la puerta del closet de la habitación. Al encender el módulo primeramente se presiona el botón "*UP*" del Control Remoto IR para guardar la contraseña, mostrándose por la pantalla LCD el texto: "Save Password" en la primera fila, y en la segunda "####" para introducir una clave de cuatro números, luego de introducida la clave almacenada en la memoria del Arduino, y la es limpiada la pantalla LCD. Cuando se desea entrar la contraseña para abrir el closet se presiona el botón "*DOWN*" y se muestra el cartel: "Intro Password" en la primera fila, y en la segunda "####" para introducir los dígitos; cuando coinciden los caracteres de la contraseña introducida y la guardada se muestra en la LCD el argumento: "Codigo Correcto" y el servomotor se desplaza 170 ° para abrir la puerta del closet, y no podrá ser introducida nuevamente hasta que el interruptor de la puerta demuestre que está cerrada, cuando se trata de introducir la contraseña con la puerta abierta se muestra en pantalla: "Closet abierto". Cada vez que se introduce el código incorrecto por el huésped suena el zumbador pasivo una vez, en el caso de introducido incorrectamente el tres veces el zumbador suena cinco veces

y se enciende un LED intermitente para alertar de la posibilidad de una violación de la seguridad. Hasta que la alarma no es apagada no se puede modificar nada en el sistema, o sea, que está bloqueado para evitar un posible robo. En ese caso a través del botón *"PAUSE"* se apaga el LED. En la configuración del sistema esta implementada la posibilidad de borrar un caracter durante la escritura de la clave si el cliente se equivoca. También está diseñado a prueba de errores porque no permite la entrada de caracteres no válidos. Además, los textos de guardar e introducir contraseña mostrados en la pantalla LCD están en inglés porque la misma solo posee la capacidad de mostrar dos filas de 16 caracteres cada una. En el Anexo V se muestran las imágenes de los componentes de hardware seleccionados para el funcionamiento del módulo de seguridad del closet con las correspondientes conexiones.

Conclusiones:

El dispositivo funciona correctamente, puesto que no ocurrieron errores de hardware, de software o del sistema de comunicación durante la realización del experimento. Además, la velocidad de procesamiento de la información y de la ejecución de las funciones es rápida

3.1.3. Tercer experimento

Objetivo: Evaluar el funcionamiento del módulo del encendido de las luces y control del clima

Descripción: El módulo se conecta a la computadora a través del puerto serie utilizando un cable USB para cargar el programa en la memoria RAM del Arduino Mega 2560 R3. Luego, este dispositivo es energizado utilizando un adaptador de 9V y 1 A conectado a la red eléctrica. Seguidamente el módulo demora pocos segundos en arrancar el funcionamiento (9-11 segundos).

Este módulo es utilizado para el control automático de la luminaria y el clima de la habitación. Una vez iniciado el módulo en Modo Automático se activa el LED rojo, pero no sucede ninguna acción dentro de la habitación, simulando que está vacía. En el momento en que se activa el interruptor de la puerta de entrada los PIR se activan, y evalúan durante cinco minutos si existe presencia dentro del local. Cuando detectan presencia se enciende el led del lugar donde les varío su potencial, en caso de ser el ubicado en el dormitorio automáticamente se enciende el ventilador de techo hasta que no vuelva a activarse el interruptor de la puerta

de entrada y se vuelva a realizar el chequeo. Al detectarse presencia en cualquier lugar dentro, se activa el aire acondicionado; en el caso de que se active el interruptor de la puerta de entrada y se determine que la habitación se encuentra vacía el aire acondicionado se apaga, y se volverá a encender cuando la temperatura interna sea mayor a la externa, esos valores de temperatura son determinados a partir de la medición que realizan los sensores de temperatura y humedad DHT11 ubicados dentro y fuera del aposento. En el caso del LED del balcón se enciende cuando el interruptor de esa puerta está en bajo, o sea que mientras está abierta la puerta, el LED permanece encendido. Otra función que realiza este módulo es determinar si el domicilio permanece vacío un período mayor a dos horas, en ese caso se apaga el aire acondicionado permanentemente hasta que huésped vuelve a ocupar la instalación, este instante de tiempo se determina a través del reloj digital DS3231 que funciona con una batería de 3 V y que permite que sean almacenados en memoria el horario de salida del usuario y el actual. Hay una situación que se deben tener en cuenta para lograr un mayor nivel de calidad del módulo, consiste en que si hospeda más de una persona en la habitación, cada vez que se active el interruptor de la puerta principal se realizará una comprobación por los sensores de presencia, pero si el otro/s huésped está durmiendo durante esos cinco minutos todo el sistema trabajará como si la habitación estuviera vacía, por lo que, se recomienda para este caso que el usuario que abandone la habitación establezca el modo manual para que el sistema quede funcionando al gusto del cliente. En el Anexo VI se muestran las imágenes de los componentes de hardware seleccionados para el funcionamiento del Modo Automático con las correspondientes conexiones.

El módulo también funciona en Modo Manual para el control de eventos, para cambiar de modo se presiona el botón *"Power"* del Control Remoto IR y se activa el LED azul. Presionando el botón '1", se enciende o apaga el LED del baño. Presionando el botón "2", se enciende o apaga el LED del dormitorio. Presionando el botón "3", se enciende o apaga el LED del balcón. Apretando el botón "4", se enciende o apaga el aire acondicionado. Por último, apretando el botón "5", se enciende o apaga el ventilador de techo. En el Anexo VII se muestran las imágenes de los componentes de hardware seleccionados para el funcionamiento del Modo Manual con las correspondientes conexiones.

En ambos modos siempre se mostrará en la pantalla LCD la temperatura dentro de la habitación.

Conclusiones:

El módulo funciona correctamente, destacándose la precisión en la medición de los sensores. Aunque el tiempo de atraso del arranque del módulo es mayor que del resto, sigue siendo despreciable. Se recomienda que las conexiones permanezcan bien unidas, dado que, el módulo posee muchos dispositivos interconectados. Además, cumple con su objetivo fundamental de lograr reducir el consumo eléctrico de los dispositivos electrónicos de la habitación de hotel.

3.2 Análisis económico y medioambiental de la propuesta realizada

El precio en USD de todos los componentes del sistema diseñado se muestra en la Tabla 3.1. El total asciende a $ 397.35 USD.

El costo por pago de licencias de uso de los componentes de software o de hardware es nulo debido a que todos constituyen hardware y software libres.

Tabla 3.1. Precios de los componentes de sistema domótico.

Componentes	Precio (USD)
3 x Arduino Mega 2560 R3	3 x 38.50
2 x Servo SG90	2 x 8.99
3 x Receptor IR	3 x 9.95
Kit de resistencia (73 unidades) (10 KΩ, 1KΩ, 220Ω, 100Ω)	9.99
Reloj Digital DS3231 RTC	2 x 9.99
Potenciómetro	3 x 8.99
Fotocelda (20 unidades)	2 x 2.99

Pantalla LCD 1602	3 x 5.99
Teclado de membrana	2 x 6.99
HC-SR501 PIR	2 x 4.89
Interruptor (10 unidades)	5.99
Módulo de alimentación 3.3 V / 5V	5.69
L293D (10 unidades)	12.55
Motor DC	2 x 11.99
Sensor DHT11	2 x 10.49
Panel de conexiones	3 x 9.50
LED (10 unidades)	2.32
Zumbador Pasivo (10 unidades)	5.99
Cables de m-m y m-h (10 unidades)	3 x 3.84
Cable USB para Arduino	3 x 3.95

El sistema domótico diseñado aprovecha los recursos naturales con el objetivo de incrementar el ahorro energético. Para el caso de la utilización de la iluminación, en el día se le da prioridad a la luz natural para la iluminación de la habitación, mientras que en la noche funciona la luz artificial dependiendo de los sensores de movimiento, que se encargarán de determinar que luz se debe encender. Igualmente, el clima es controlado por detectores de presencia para disminuir el consumo energético, dado que, solo cuando el usuario permanece en la habitación el aire acondicionado y el ventilador de techo funcionarán a máximo rendimiento, de lo contrario permanecerán en perfil bajo. Al disminuir a través de todas estas condiciones la cantidad de energía que se consume en el hotel, será menor el impuesto a pagar a la empresa eléctrica.

Cuando el hotel decide implementar un sistema domótico en sus habitaciones, adquiere un valor adquisitivo que a largo plazo aumentará mientras se siga actualizando e incrementando las funcionalidades del sistema. Todo ello gracias a la mejora de la interfaz de la habitación al aparecer un sistema completamente automatizado que le permite al cliente disfrutar de

nuevas experiencias y facilidades. Además, el costo de una habitación asciende hasta un 30 % del precio que poseía anteriormente.

La inversión inicial realizada para la instalación y explotación de un sistema de este tipo puede ser elevada, pero en un período menor a 3 años es recuperado todo el valor de la inversión, dado el alto grado de confort, seguridad y efectividad energética logrado. Posteriormente, con el transcurrir de los años se empezarán a obtener ganancias notables que tendrán un efecto positivo sobre la economía nacional.

La propuesta realizada es económica y rentable, por lo que no requiere que se realicen grandes desajustes económicos en el hotel para la sustitución del sistema luego de pasado su tiempo de vida útil, ya que este tiempo es mayor a los 10 años, o sea, tiempo suficiente para recuperar el primer monto dado para la instalación y obtener valores de ganancias considerables.

A pesar del costo del sistema domótico diseñado, se ha demostrado en el mundo y en Cuba, que la utilización de un sistema de este tipo en un hotel tiene como resultado una reducción del consumo energético superior al 50 %, que se traduce en ahorro de combustible, y disminución de la emisión de gases de efecto invernadero al medioambiente.

Estos sistemas no solo nacen con la idea de optimar al máximo los recursos que se utilizan a diario, también tienen como finalidad conducir a un estilo de vida amigable con el entorno, reduciendo notoriamente los niveles de energía en los dispositivos de uso diario. La automatización de las habitaciones por medio de instalaciones domóticas contribuye al consumo responsable, y a cambiar los hábitos medioambientales menos sostenibles. También favorecen la explotación de los recursos naturales a partir de la reducción del uso de sistemas de climatización e iluminación artificial (National KNX, 2019).

Los dispositivos y medios propuestos para la automatización de una habitación inteligente son seguros, libres de contaminación radioactiva, y con bajos niveles de plomo (Pb). Igualmente, no emiten ondas o sonidos que afecten la salud de las personas que interactúen con el sistema automatizado.

3.3 Consideraciones finales del capitulo

En este capítulo se mostraron los resultados de las pruebas experimentales realizadas con el sistema domótico que responde a la propuesta de la arquitectura de hardware y software diseñada en el segundo capítulo. La efectividad del sistema propuesto quedó demostrada a través de los resultados de las pruebas realizadas en cuanto a:

- Velocidad de respuesta de los dispositivos que conforman los módulos.
- Funcionamiento de acuerdo a las funcionalidades deseadas.
- Coordinación de los eventos.
- Ahorro energético alcanzado.
- Facilidad de interacción del usuario con el sistema.
- Capacidad de almacenamiento de información del Arduino Mega 2560 R3.

La implementación de un sistema domótico trae consigo ventajas al sector del turismo, logrando reducir el consumo eléctrico, y apoyando la protección y disfrute de los recursos medioambientales.

La propuesta realizada para la automatización de una habitación de hotel constituye un producto autóctono, dado que, fue diseñado en una institución nacional permitiendo la sustitución de importaciones de sistemas demóticos similares.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Después de realizada la propuesta de automatización de una habitación es necesario evaluar su efectividad. Teniendo en cuenta estos resultados se pueden plantear las siguientes conclusiones generales:

1 A partir de los resultados del análisis de la literatura existente en torno al tema se seleccionan las variables físicas, siendo estas las más importantes teniendo en cuenta las características de la habitación, así como el confort y las necesidades de los clientes.

2 Teniendo en cuenta las tecnologías que constituyen hardware y software libres fue posible diseñar la arquitectura de un sistema domótico mediante la placa microcontroladora Arduino Mega 2560 R3, siendo la más adecuada, entre las variantes analizadas, garantizando la interconexión entre cada uno de los dispositivos empleados.

3 La arquitectura de software diseñada garantiza el funcionamiento de la propuesta domótica realizada, que permite establecer una lógica de control que define las acciones a realizar por cada dispositivo que conforma el sistema.

4 Las pruebas experimentales realizadas demostraron la efectividad de la arquitectura de hardware y software propuesta, al cumplir con los requisitos de funcionamiento, configuración, flexibilidad y seguridad que caracterizan a un sistema domótico permitiendo a los hoteles reducir el pago de la factura eléctrica, y una relación más provechosa y saludable con el medioambiente.

Recomendaciones

A partir del análisis de la propuesta de habitación inteligente realizada, se pueden determinar posibles mejoras que se le pueden hacer para aumentar su rendimiento:

1 Implementar puertos DIN con el objetivo de permitir múltiples conexiones y lograr organización en el cableado de cada módulo.

2 Separar los módulos por zonas en correspondencias con su funcionalidad para así evitar errores en la lectura de los sensores y facilitar el uso del Control Remoto IR por zonas.

3 Sustituir el sistemas de interruptores en las puertas de entrada y el balcón si se posee diodos receptores de luz láser, dado que, permiten determinar en que dirección se mueven los clientes, o sea, entran o salen, eliminando líneas de código inecesarias en el programa fuente.

REFERENCIAS BIBLIOGRÁFICAS

Alaman, X., Haya, P. and Montoro, G. (2010) *ODISEA : Hacia un entorno inteligente basado en un interfaz en lenguaje natural*. Universidad Autonóma de Madrid.

Bustamante, A. J. T. and Acosta, M. J. L. (2016) *'implementación de un sistema domótico con tecnología arduino en app inventor para mejorar el control de temperatura e iluminación del hotel san luis en amarilis'*. Universidad de Huánuco.

Carretero, O. S. (2016) *Casa domotica con arduino*. Universidad Politécnica de Valencia.

Chipuxi, R. Y. O. (2015) *Diseño e implementación de un sistema domótico remoto vía GSM para el hotel San Miguel*. Universidad Politécnica Salesiana.

Eizmendiz, M. D. (2016) *Diseño de sistema domótico de una vivienda mediante un controlador lógico programable (PLC) del tipo TM-251*. Universidad Politécnica de Cataluña.

Gallego, D. E. C. (2016) *Intelligent house v1.0 (IH 1.0)*. Universidad Católica de Manizales.

Giorgini, O. G. (2017) *Sistema de domotica mediante Raspberry Pi e Internet*. Universidad Nacional de Mar del Plata.

Inc. Elegoo (2011) *EL KIT MÁS COMPLETO TUTORIAL PARA MEGA2560*. Available at: www.elegoo.com.

José Manuel Huidobro (2007) 'Composiciin C M Y CM MY CY CMY K', in *La Domótica como Solución de Futuro*. Madrid, pp. 7–8. doi: 10.1017/CBO9781107415324.004.

Leon, I. D. C. (2011) *DOMÓTICA E INMÓTICA : VIVIENDAS Y EDIFICIOS INTELIGENTES*. Universidad Veracruzana.

National KNX (2019) *Beneficios de la domótica, domoticus-domótica para todos*. Available

at: www.Beneficios de la domótica - Beneficios - Domótica_Empresa domótica_Casas de lujo_Smart Home_Casa domótica_Casas modernas.com.

Rojas, J. H. (2017) *Control inteligente de sistemas de iluminacion en edificios*. Universidad de Piura.

S.A, C. (2017) *AUTOMATIZACIÓN HABITACIONAL HOTEL PLAYA CAYO SANTA MARÍA PROPUESTA TÉCNICO – COMERCIAL PARA 66 HABITACIONES*. Villa Clara.

Sanchéz, A. M. A. (2016) *Desarrollo de un controlador domótico programable basado en Arduino para un sistema X-10*. Universidad Politécnica de Valencia.

Sánchez, C. *et al.* (2014) 'Diseño e implementación de un prototipo de vivienda domótica basado en las plataformas arduino y android'.

Sánchez, E. Ll. (2012) *Diseño de un sistema de control domótico basado en la plataforma Arduino*. Universidad Politécnica de Valencia.

Valdivia, F. J. C. (2014) *Análisis y diseño de una red domótica para viviendas sociales*. Universidad Austral de Chile.

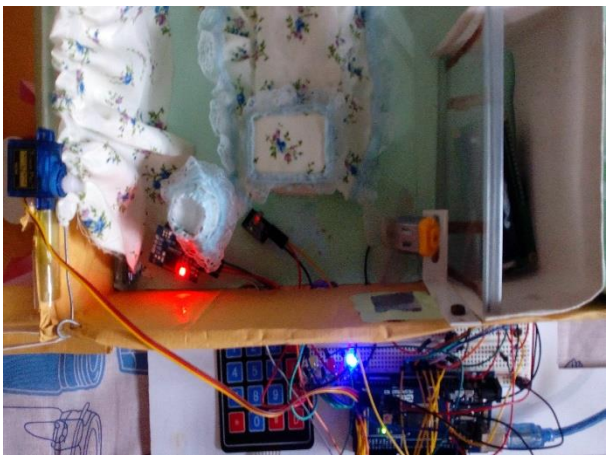
Zennio (2018a) 'Hoja de ruta para automatizar hoteles', *Zennio*, pp. 1–9. Available at: www.zennio.com.

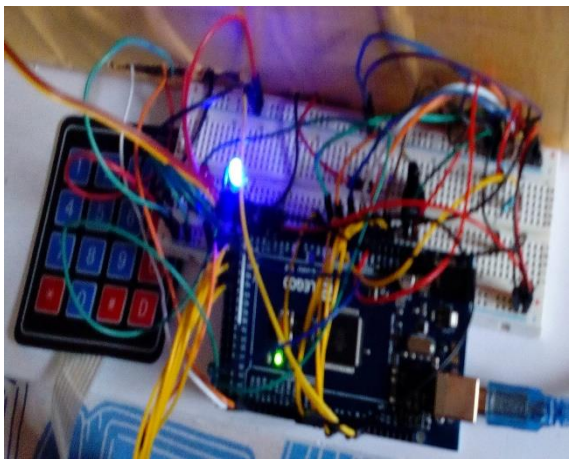
Zennio (2018b) 'Home, Building & Hotel Automation Solutions', *Zennio*, p. 55. Available at: www.zennio.com.

ANEXOS

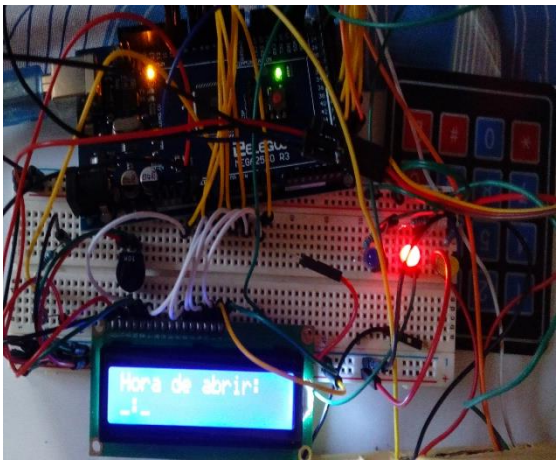
Anexo I Maqueta a escala diseñada para la realización de las pruebas a escala del sistema domótico propuesto para una habitación.

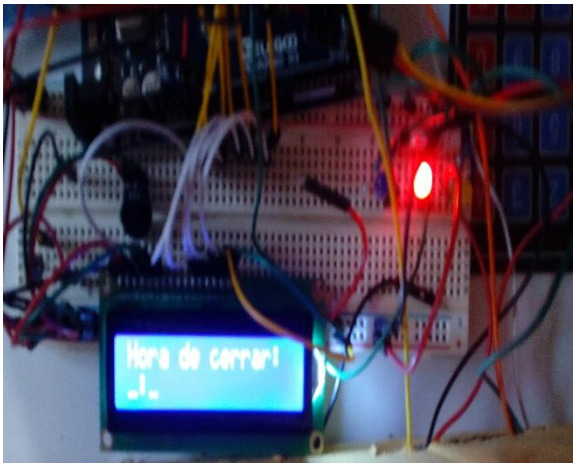
Anexo II Representación del modo automático del módulo de apertura y cierre de cortinas

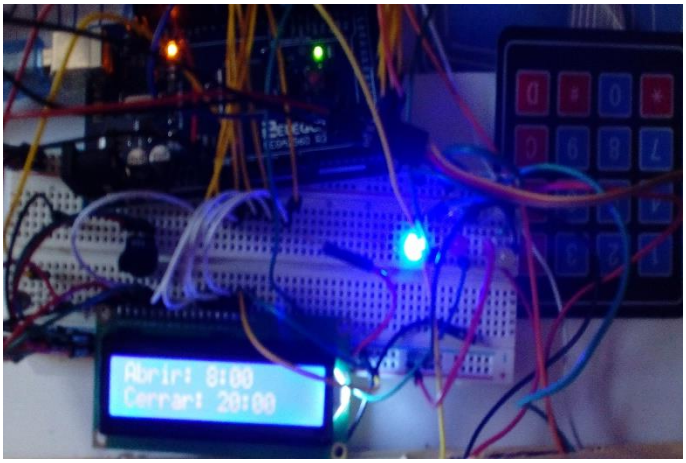




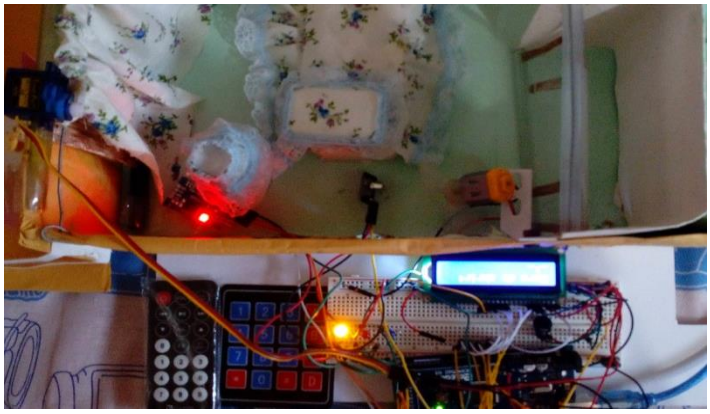
Anexo III Representación del modo horario del módulo de apertura y cierre de cortinas



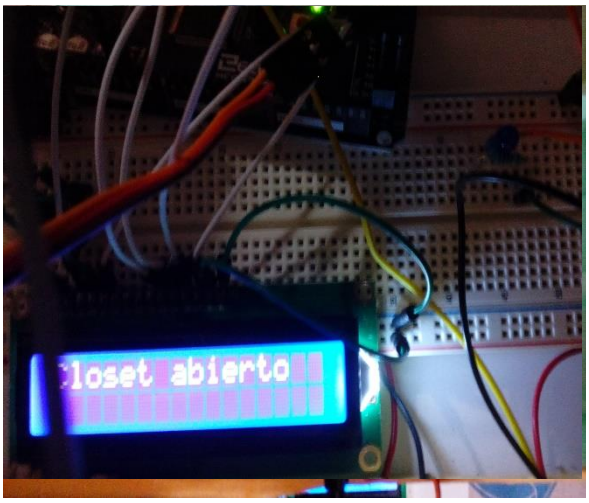


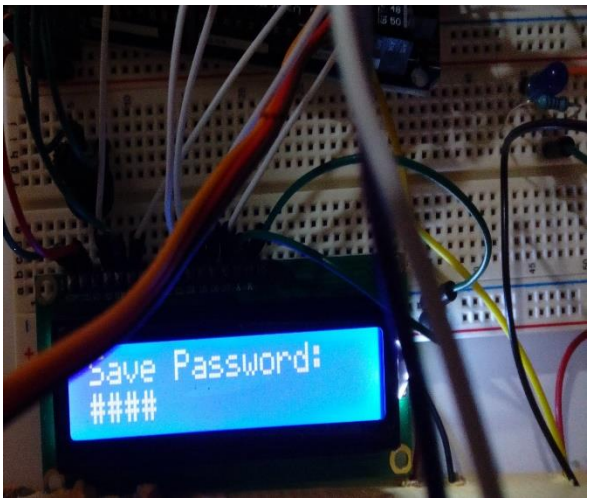


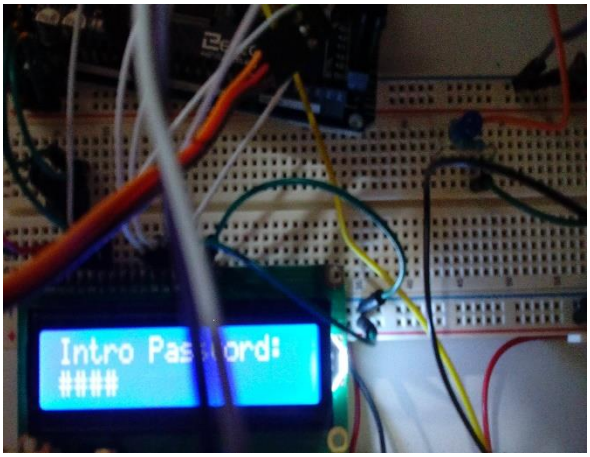
Anexo IV Representación del modo manual del módulo de apertura y cierre de cortinas

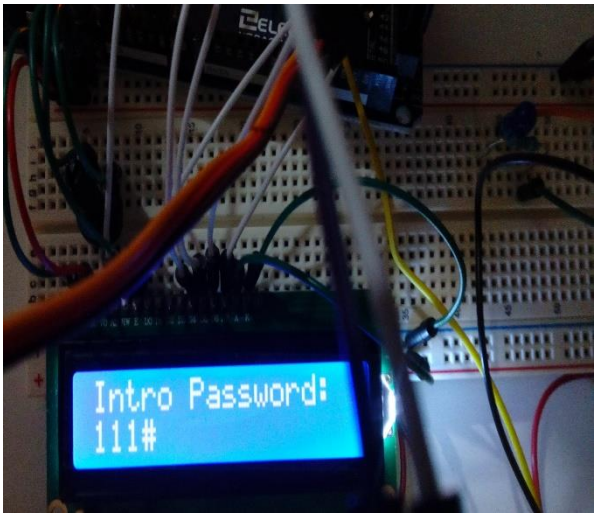


Anexo V Representación del módulo de seguridad del closet

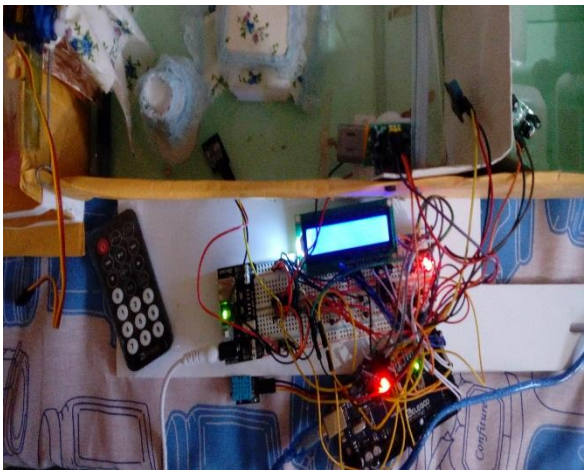


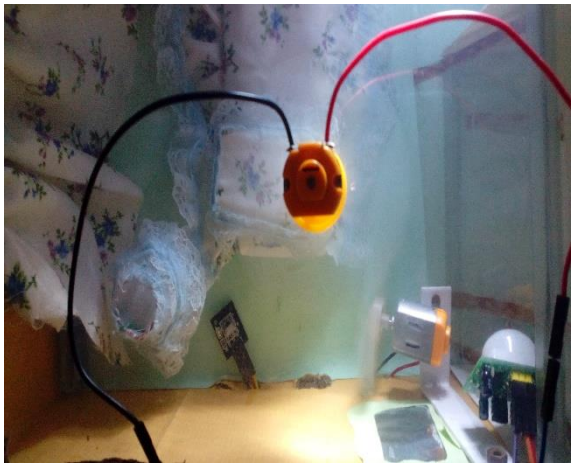


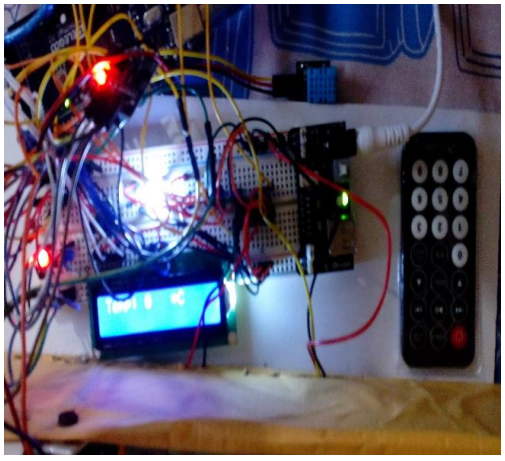




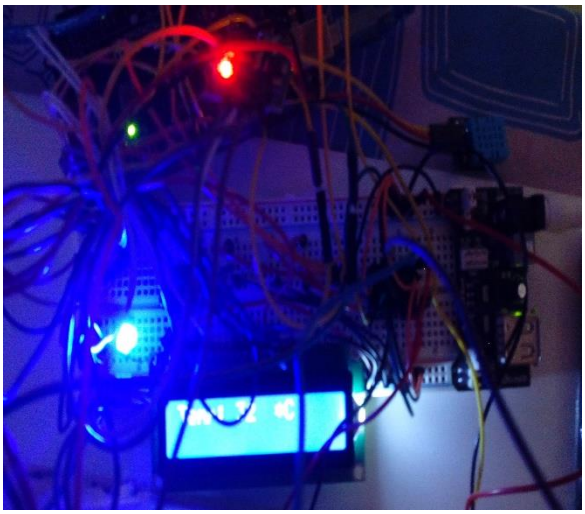
Anexo VI Representación del modo automático del módulo de encendido de luces y control del clima

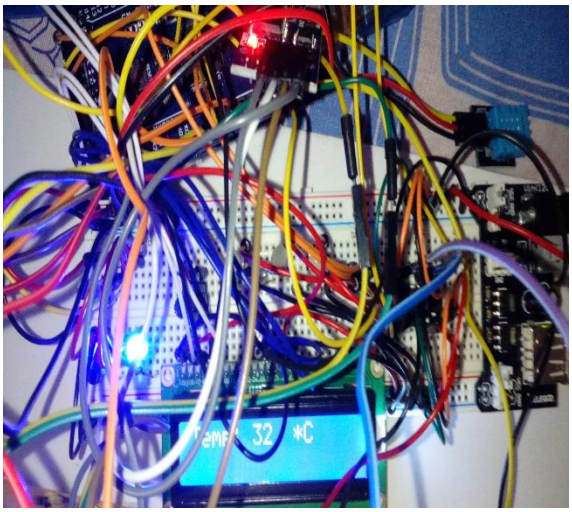






Anexo VII Representación del modo manual del módulo de encendido de las luces y control del clima





Anexo VIII Programa en C++ que dirige el funcionamiento del módulo de apertura y cierre de cortinas

```
/*servo motor_pin de señal 9PWN

  Control Remoto IR_pin de señal 10PWN

  Diodo de Luz azul_pin de señal 22 para el modo automatico

  Dodo de Luz amarillo_pin de señal 24 para el modo manual

  Diodo de Luz rojo_pin de señal 26 para el modo horario

  Reloj Pin SCL y Pin SDA con PWM

  Fotocelda pin_A0

  Potenciometro Vcc y GND

   LCD RS pin to digital pin 2

   LCD Enable pin to digital pin 3

   LCD D4 pin to digital pin 4

   LCD D5 pin to digital pin 5
```

```
  LCD D6 pin to digital pin 6
  LCD D7 pin to digital pin 7
  LCD R/W pin to ground
  LCD VSS pin to ground
  LCD VCC pin to 5V
  10K resistor:
  ends to +5V and ground
  wiper to LCD VO pin (pin 3)
  La LCD depende del reloj y la martix ,el  motor del reloj tambien */

#define UMBRAL_LUZ 1 // variable para determinar la intensidad de la luz
#define PIN_LED_ESTADO 22  //este pin indica que el modo automatico esta activo
#define PIN_LED_USUARIO 24 // ste led indica q esta en el modo manual
#define PIN_LED_HORARIO 26 //este led indica q esta en el modo horario
#include <Servo.h>
#include <Wire.h> // bibliotecas del I2C
#include <DS3231.h> //bibliotecas del reloj
#include <LiquidCrystal.h> //biblioteca de la LCD
#include "IRremote.h" // biblioteca del mando IR
#include <Keypad.h> //biblioteca del teclado

Servo myservo;  // crea un objeto llamado servo para el control del servomotor
DS3231 clock; //crea un objeto para el reloj
RTCDateTime dt;// aca estan alacenadas los valores del reloj
```

```
int receiver = 10; // Control Remoto IR Pin 10 PWM Signal
int angulo = 0; //se usa para el modo manual es el angulo/ domina el mov del servo
int estado = 0; // empieza en automatico = 0, manual =1 y modo horario =3
int HoraAbrir = 0, MinutoAbrir = 0; // variables utilizadas para mostrar en la pantalla LCD
int HoraCerrar = 0, MinutoCerrar = 0; // los horarios de las alarmas

const byte ROWS = 4; //four rows
const byte COLS = 4; //four columns
//define the symbols on the buttons of the keypads
char hexaKeys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
}; // crear un arreglo de 4x4

byte rowPins[ROWS] = {53, 52, 51, 50}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {49, 48, 47, 46}; //connect to the column pinouts of the keypad

//initialize an instance of class NewKeypad
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
```

```
LiquidCrystal lcd(2, 3, 4, 5, 6, 7); // pines del arduino que envían los pulsos de salida a la pantalla LCD


/*-----( Declare objects )-----*/
IRrecv irrecv(receiver);     // create instance of 'irrecv'
decode_results results;      // create instance of 'decode_results'
/*-----( Function )-----*/
void translateIR() // takes action based on IR code received/ describing Remote IR codes
{
  switch (results.value) // analiza que boton ha sido presionado
  {
    case 0xFFA25D: //codigo del boton power /utilizado para cuando sea presionado cambiar de modo
     estado++; // variable booleana para determinar el nivel de prioridad de los modos automatico,manual,horario
     if (estado > 2) {
       estado = 0; //se reinicia a modo automatico
     }
     break;

     if (estado == 1) { //modomanual

     case 0xFF629D: //codigo del boton VOL+
       angulo += 10; //se auto-incrementa en 10
```

```
    if (angulo > 170) {
      angulo = 170; //se limita la variable a el valor maximo
    }
    break;

  case 0xFF22DD://codigo del boton "FAST BACK";
    angulo = 10; //mov del servo 180 grados
    break;

  case 0xFFC23D://codigo del boton "FAST FORWARD";
    angulo = 170; //mov del servo 180 grados
    break;

  case 0xFFA857: //VOL-
    angulo -= 10;//se auto-decrementa en 10
    if (angulo < 10) {
      angulo = 10; //se limita la variable a el valor minimo
    }
    break;
  }
//*** opcion para si se desea cambiar los horarios de apertura y cierre de las cortinas***
case 0xFFB04F: //boton ST/REPT
  Reloj(); //Muestra por monitor serial la hora
  ConfigHoraAbrir(); // funcion para establecer el horario de apertua
```

```
    ConfigHoraCerrar(); // funcion para establecer el horario de cerrado
    lcd.setCursor(0, 0); // posiciona el cursor de la pantalla lcd en la primera fila, linea 1
    lcd.print("Abrir: " + String(HoraAbrir) + ":" + twoDigits(String(MinutoAbrir))); //muestra en la pantalla lcd la cadena de texto para la hora de apertura
    lcd.setCursor(0, 1); // posiciona el cursor de la pantalla lcd en la segunda fila, linea 1
    lcd.print("Cerrar: " + String(HoraCerrar) + ":" + twoDigits(String(MinutoCerrar))); //muestra en la pantalla lcd la cadena de texto para la hora de cerrado
    break;
  }// End Case
  delay(500); // Do not get immediate repeat
} //END translateIR

void modoAutomatico() { // funcion para el caso automatico /Lee la entrada de la señal al puerto A0

  int sensorValue = analogRead(A0);//esta entre 0 y 1023 / Guarda la lectura de la señal de entrada al puerto A0 en la variable sensorValue
  float voltajecell = sensorValue * ((double)5 / 1023); // Se convierte a rango de voltaje para la fotocelda /Divisor de voltaje

  if (voltajecell > UMBRAL_LUZ) { //caso de q este con alta luminosidad / Se abre la cortina
    myservo.write(170); // determina la apertura del servomotor
    angulo = 170;
  }
  else { //caso de q este con baja luminosidad / Se cierra la cortina
```

```
    myservo.write(10); // determinar l cerrado del servomotor
    angulo = 10;
  }
  // Muestra por el monitor serial el valor en voltaje que mide la fotocelda
  Serial.println(voltajecell, 4);
}



void modoManual() { // funcion para el caso manual
  myservo.write(angulo); // el mov del servo lo determina el valor de la variable angulo
}

void Reloj() { //funcion para mostrar la hora en tiempo real por el monitor serial
  dt = clock.getDateTime(); // variable donde se almacenan los valores del hora y el calendario
/ Original de la biblioteca del reloj

  //***Estos dt son utilizados para mostrar los valores del reloj en la pantalla LCD***
  Serial.print("Raw data: ");
  Serial.print(dt.year);   Serial.print("-");
  Serial.print(dt.month);  Serial.print("-");
  Serial.print(dt.day);    Serial.print(" ");
  Serial.print(dt.hour);   Serial.print(":");
  Serial.print(dt.minute); Serial.print(":");
  Serial.print(dt.second); Serial.println("");
  delay(1000);
```

```
}

void ConfigHoraAbrir() { //esta funcion es la encargada de configurar la hora apertura de la cortina /Aca tambien se muestran por el LCD lo que introduzca el usuario

HORA: //un go to para hacer un ciclo de la funcion
  lcd.clear(); //para limpiar la pantalla lcd
  lcd.setCursor(0, 0);
  lcd.print("Hora de abrir:"); //primer segmento de texto escrito en la lcd
  lcd.setCursor(0, 1);
  lcd.print("_:_");
  char customKey = customKeypad.getKey(); // funcion de la biblioteca del teclado para determinar si se ha presionado una tecla
  String HoraStr, MinutoStr; //variables para guardar la hora y los minutos como string
  boolean aceptado = false; //variable booleana para determinar la piroridad
  char KeyIntermedio;//esta variable es para guardar los valores de customKey y que no se pierdan para despues

  while (!aceptado) { //espera a que se introduzca las horas y luego se presione la letra A
    char customKey = customKeypad.getKey();
    if (customKey) { //se presiono una tecla
      KeyIntermedio = customKey; // no deja que se pierda el valor de customKey o sea que no dejara que se anule
      if (customKey == 'A' || customKey == 'B' || customKey == 'C' || customKey == 'D' || customKey == '*' || customKey == '#') {// evita que sean introducidos caracteres no validos
```

```
      aceptado = true; // accion para invalidar la entrada de caracteres no validos
     }
     else {
      HoraStr = HoraStr + customKey; // se le adiciona el siguiente caracter
      lcd.setCursor(0, 1);
      lcd.print(HoraStr + ":_");
     }
    }
  }//!While

  if (HoraStr != "00") { //caso especial en que se desea que sean las 12am
   HoraAbrir = HoraStr.toInt(); //llevar la de string a int para restringir los valores de la hora
   Serial.println(HoraAbrir);
   if (HoraAbrir == 0 || HoraAbrir > 23  || KeyIntermedio != 'A' ) { //HORA INCORRECTA
/Permitir que el caracter A sea aceptado
     lcd.setCursor(0, 0);
     lcd.print("                ");//para limpiar solo la primera linea
     lcd.setCursor(0, 0);
     lcd.print("Hora Incorrecta");
     delay(1000);
     goto HORA; //para volver al iniciar la funcion
    }
  }
  //en esta linea ya las horas estan bien
MINUTO:
```

```
  aceptado = false;
  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print(HoraStr + ":#");
  lcd.setCursor(0, 0);
  lcd.print("Hora de abrir:");
  MinutoStr = ""; //deja el espacio de los minutos vacio en la pantalla lcd
  while (!aceptado) { //espera a que se introduzca las horas y luego se presione la letra A


    customKey = customKeypad.getKey();


    if (customKey) { //se presiono una tecla
      KeyIntermedio = customKey;
      if (customKey == 'A' || customKey == 'B' || customKey == 'C' || customKey == 'D' || customKey == '*' || customKey == '#') {
        aceptado = true;
      }
      else {
        MinutoStr = MinutoStr + customKey; // se le adiciona el siguiente caracter
        lcd.setCursor(0, 1);
        lcd.print(HoraStr + ":" + MinutoStr); //se muestra en la lcd la cadena con los valores de la hora y los minutos
      }
    }
  }//!While
```

```
  if (MinutoStr != "00") { //caso especial en que se desea que sean las en punto
    MinutoAbrir = MinutoStr.toInt();

    if (MinutoAbrir == 0 || MinutoAbrir > 59 || KeyIntermedio != 'A') { //MINUTO INCORRECTO

      lcd.setCursor(0, 0);
      lcd.print("                ");//para limpiar solo la primera linea
      lcd.setCursor(0, 0);
      lcd.print("Minuto Incorrecto");
      delay(1000);
      goto MINUTO;
    }
  }
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Abrir: " + HoraStr + ":" + MinutoStr);
  delay(2000);
  lcd.clear();
  lcd.setCursor(0, 0);
}

void ConfigHoraCerrar() { //esta funcion es la encargada de configurar la hora de cierre de la cortina /Aca tambien se muestran por el LCD lo que introduzca el usuario
```

HORA:

```
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Hora de cerrar:");
  lcd.setCursor(0, 1);
  lcd.print("_:_");
  char customKey = customKeypad.getKey();
  String HoraStr, MinutoStr;
  int HoraInt = 0, MinutoInt = 0;
  boolean aceptado = false;
  char KeyIntermedio;//esta variable es para guardar los valores de customKey y que no se pierdan para despues
  while (!aceptado) { //espera a que se introduzca las horas y luego se presione la letra A
    char customKey = customKeypad.getKey();
    if (customKey) { //se presiono una tecla
      KeyIntermedio = customKey;
      if (customKey == 'A' || customKey == 'B' || customKey == 'C' || customKey == 'D' || customKey == '*' || customKey == '#') {
        aceptado = true;
      }
      else {
        HoraStr = HoraStr + customKey; // se le adiciona el siguiente caracter
        lcd.setCursor(0, 1);
```

```
      lcd.print(HoraStr + ":_");
    }
  }
 }//!While

 if (HoraStr != "00") { //caso especial en que se desea que sean las 12am
  HoraInt = HoraStr.toInt();

  if (HoraInt == 0 || HoraInt > 23 || KeyIntermedio != 'A') { //HORA INCORRECTA
   lcd.setCursor(0, 0);
   lcd.print("               ");//para limpiar solo la primera linea
   lcd.setCursor(0, 0);
   lcd.print("Hora Incorrecta");
   delay(1000);
   goto HORA;
  }
 }
 //en esta linea ya las horas estan bien

MINUTO:
 aceptado = false;
 lcd.clear();
 lcd.setCursor(0, 1);
 lcd.print(HoraStr + ":#");
```

```
  lcd.setCursor(0, 0);
  lcd.print("Hora de cerrar:");
  MinutoStr = "";
  while (!aceptado) { //espera a que se introduzca las horas y luego se presione la letra A

   customKey = customKeypad.getKey();

   if (customKey) { //se presiono una tecla
    KeyIntermedio = customKey;
    if (customKey == 'A' || customKey == 'B' || customKey == 'C' || customKey == 'D' || customKey == '*' || customKey == '#') {
     aceptado = true;
    }
    else {
     MinutoStr = MinutoStr + customKey; // se le adiciona el siguiente caracter
     lcd.setCursor(0, 1);
     lcd.print(HoraStr + ":" + MinutoStr);
    }
   }
  }//!While

  if (MinutoStr != "00") { //caso especial en que se desea que sean las 12am
   MinutoInt = MinutoStr.toInt();
   if (MinutoInt == 0 || MinutoInt > 59 || KeyIntermedio != 'A') { //MINUTO INCORRECTO
    lcd.setCursor(0, 0);
```

```
      lcd.print("                ");//para limpiar solo la primera linea
      lcd.setCursor(0, 0);
      lcd.print("Minuto Incorrecto");
      delay(2000);
      goto MINUTO;
    }
  }
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Cerrar: " + HoraStr + ":" + MinutoStr);
  HoraCerrar = HoraStr.toInt();
  MinutoCerrar = MinutoStr.toInt();
  delay(2000);
  lcd.clear();
  lcd.setCursor(0, 0);
}
```

String twoDigits(String cadena) { //esto es para que los minutos si son menor q 10 se muestren con un cero delante ejm. 12:05 <-

```
  if (cadena.toInt() < 10) {
    cadena = '0' + cadena;
  }
  return cadena;
}
```

```
void modoHorario() {
  myservo.write(0); //para que cuando se ejecute la accion de horario el servomotor se encuentre en la posicion cero
  if (HoraAbrir == dt.hour && MinutoAbrir == dt.minute) { //se establece el caso de qe ocurra la concidencia de la hora establecida en el horario al guardado en el reloj DS3231
    myservo.write(170);//se abre la cortina /recorrido de 180 grados
  }
  if (HoraCerrar == dt.hour && MinutoCerrar == dt.minute) {
    myservo.write(0);//se cierra la cortina
  }
}

void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
  irrecv.enableIRIn(); // Start the receiver
  pinMode(PIN_LED_ESTADO, OUTPUT);//pin 22
  digitalWrite(PIN_LED_ESTADO, HIGH); // el primer led que se enciende
  pinMode(PIN_LED_USUARIO, OUTPUT); // pin 24
  digitalWrite(PIN_LED_USUARIO, LOW);
  pinMode(PIN_LED_HORARIO, OUTPUT);//pin 26
  digitalWrite(PIN_LED_HORARIO, LOW);

  // Initialize DS3231
```

```
  Serial.println("Initialize DS3231");
  clock.begin(); //empieza a actaulizarse la hora
  //***Con clock.setDataTime se carga la hora en el reloj DS3231 para que quede guardada en su memoria***
  // Set sketch compiling time
  //clock.setDateTime(__DATE__, __TIME__);//#####################
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  //******se llama funcion para introducir la hora en que se abre y se cierra la cortina
  Reloj(); //Muestra por monitor serial la hora
  ConfigHoraAbrir();
  ConfigHoraCerrar();
  lcd.setCursor(0, 0);
  lcd.print("Abrir: " + String(HoraAbrir) + ":" + twoDigits(String(MinutoAbrir)));
  lcd.setCursor(0, 1);
  lcd.print("Cerrar: " + String(HoraCerrar) + ":" + twoDigits(String(MinutoCerrar)));
}

// the loop routine runs over and over again forever:
void loop() { //se determina el nivel de prioridad de los modos
  Reloj();
  if (irrecv.decode(&results)) // have we received an IR signal?
  {
    translateIR();
```

```
    irrecv.resume(); // receive the next value
  }
  if (estado == 2) { //modo horario
    modoHorario();
    digitalWrite(PIN_LED_HORARIO, HIGH);
    digitalWrite(PIN_LED_ESTADO, LOW);
    digitalWrite(PIN_LED_USUARIO, LOW);
  }
  else if (estado == 1) { //modo manual activado
    modoManual();
    digitalWrite(PIN_LED_USUARIO, HIGH);
    digitalWrite(PIN_LED_ESTADO, LOW);
    digitalWrite(PIN_LED_HORARIO, LOW);
  }
  else if (estado == 0) { //modo automatico
    modoAutomatico();
    digitalWrite(PIN_LED_ESTADO, HIGH);
    digitalWrite(PIN_LED_HORARIO, LOW);
    digitalWrite(PIN_LED_USUARIO, LOW);
    delay(250);       // delay in between reads for stability
  }
}
```

Anexo IX Programa en C++ que dirige el funcionamiento del módulo de seguridad del closet

```
/*The circuit:
  servo motor_pin de señal 9PWN
  Control Remoto IR_pin de señal 11PWN
  ZUMBADOR_pin 10 y GND
  Diodo de Luz azul_pin de señal 38 para la alarma
  Switch para puerta pin_39
  Potenciometro Vcc y GND
  LCD RS pin to digital pin 44
  LCD Enable pin to digital pin 45
  LCD D4 pin to digital pin 42
  LCD D5 pin to digital pin 43
  LCD D6 pin to digital pin 40
  LCD D7 pin to digital pin 41
  LCD R/W pin to ground
  LCD VSS pin to ground
  LCD VCC pin to 5V
  10K resistor:
  ends to +5V and ground
  wiper to LCD VO pin (pin 3)
*/
#define PIN_LED_ALARMA 38//Led para indicar si se ha introducido tres veces la contrasena incorrecta
#define PIN_ZUMBADOR_ALARMA 10
```

```
#include <Servo.h> //bilioteca del motor
#include <LiquidCrystal.h>// biblioteca de la LCD
#include "IRremote.h" // biblioteca del mando IR
#include <Keypad.h>// biblioteca del teclado
#define ABIERTO 170
#define CERRADO 10
#define PIN_PUERTA 39//este es el fin de carrera que se activa cuando la puerta esta abierta

Servo myservo;

int receiver = 11; // Control Remoto IR Pin 11 PWM Signal
int angulo = 0;
String Codigo = "####";
String CodigoSave;
String Code;
int limite = 0;
bool equivocado = false;
boolean estado = false;
const byte ROWS = 4; //four rows
const byte COLS = 4; //four columns
//define the symbols on the buttons of the keypads
char hexaKeys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
```

```
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
}; // crear un arreglo de 4x4

//Con los numeros de frente de izquierda a derecha empieza a partir de pin 52
byte rowPins[ROWS] = {52, 53, 50, 51}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {48, 49, 46, 47}; //connect to the column pinouts of the keypad

//initialize an instance of class NewKeypad
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(44, 45, 42, 43, 40, 41);

IRrecv irrecv(receiver);     // create instance of 'irrecv'
decode_results results;      // create instance of 'decode_results'

void translateIR() // takes action based on IR code received
{ switch (results.value)
  {
    case 0xFF906F: // codigo del boton "UP";
      ConfigContrasena ();//para meter la contrasena/queda guardada en la variable Codigo
      break;
```

```
    case 0xFFE01F: //codigo del boton "DOWN"; /para meter la contrasena para comparar
      if (!estado)
      {
        Comparacion();
      }

      break;

    case 0xFF02FD: //codigo del boton "PAUSE" /para apagar la alarma
      digitalWrite(PIN_LED_ALARMA, LOW);
      //digitalWrite(PIN_ZUMBADOR_ALARMA, LOW);
      estado = false;
      break;

  }// End Case
  delay(500); // Do not get immediate repeat
} //END translateIR

void Zumbador1 () {

  //for (int i = 0; i < 200; i++) {
  //delayMicroseconds(956);
  //digitalWrite(PIN_ZUMBADOR_ALARMA, HIGH);
  //delayMicroseconds(956);
```

```
  digitalWrite(PIN_ZUMBADOR_ALARMA, LOW);
  //}
  delay(250);
}

void Zumbadormult () {

  for (int i = 0; i < 5; i++) {
    for (int i = 0; i < 200; i++) {
      delayMicroseconds(956);
      digitalWrite(PIN_ZUMBADOR_ALARMA, HIGH);
      delayMicroseconds(956);
      digitalWrite(PIN_ZUMBADOR_ALARMA, LOW);
    }
    delay(250);
  }
}



int numcont = 0;

void ConfigContrasena () {

  lcd.clear();
```

```
lcd.setCursor(0, 0);
lcd.print("Save Password:");
lcd.setCursor(0, 1);
Codigo = "####";
lcd.print(Codigo);//OJo debe ser Codigo = "####";

//char customKey = customKeypad.getKey();
boolean contrasena = false;
//  char KeyIntermedio;

while (!contrasena) {
  char customKey = customKeypad.getKey();
  if (customKey) { //se presiono una tecla

    if (numcont == 3) {
      contrasena = true;
    }

    if (customKey == 'B') {//para borrar, se resetea tambien el contador
      Codigo = "####";
      numcont = -1;
      lcd.setCursor(0, 1);
      lcd.print(Codigo);
    }
```

```
      else if (numcont <= 3) { //se admiten solo 4 caracteres de contraseña
        //Codigo = Codigo + customKey; // se le adiciona el siguiente caracter
        Codigo.setCharAt(numcont, customKey);
        lcd.setCursor(0, 1);
        lcd.print(Codigo);
      }

      numcont++;
    }
  }

  CodigoSave = Codigo;
  Serial.print("Contrasena a salvar"); Serial.println(Codigo);
  numcont = 0;
  delay(1000);
  lcd.clear();
  lcd.setCursor(0, 0);
}
int puerta = 0;



void Comparacion () {
```

```
puerta = digitalRead(PIN_PUERTA);
if (puerta == LOW) {
  myservo.write (CERRADO);

  do {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Intro Password:");
    lcd.setCursor(0, 1);
    Code = "####";
    lcd.print(Code);//ojo debe ser Codigo = "####";

    //char customKey = customKeypad.getKey();
    boolean contrasena = false;
    //  char KeyIntermedio;

    while (!contrasena) {
      char customKey = customKeypad.getKey();
      if (customKey) { //se presiono una tecla

        if (numcont == 3) {
          contrasena = true;
        }
```

```
    if (customKey == 'B') {//para borrar, se resetea tambien el contador
      Code = "####";
      numcont = -1;
      lcd.setCursor(0, 1);
      lcd.print(Code);
    }
    else if (numcont <= 3) { //se admiten solo 4 caracteres de contraseña
      //Codigo = Codigo + customKey; // se le adiciona el siguiente caracter
      Code.setCharAt(numcont, customKey);
      lcd.setCursor(0, 1);
      lcd.print(Code);

    }

    numcont++;
  }
}
delay(1000);
numcont = 0;

Serial.print("Contrasena real "); Serial.println(CodigoSave);
Serial.print("Contrasena introducida "); Serial.println(Code);

if ( Code == CodigoSave) {
```

```
  myservo.write(ABIERTO);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Codigo correcto!");
  limite = 0;
 }
 else {//CONTRASENA INCORRECTA
  Zumbador1();
  myservo.write(CERRADO);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Codigo mal!");
  delay(500);
  limite++;
  equivocado = true;

 }
 if (limite > 2) {
  //activar alarmas
  digitalWrite(PIN_LED_ALARMA, HIGH);
  Zumbadormult();
  estado = true;
  equivocado = false;
  limite = 0;
```

```
      }
    } while (equivocado);


  }
  else { //puerta abierta
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Closet abierto");
  }
}

void setup() {
  Serial.begin(9600);
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
  myservo.write(CERRADO);
  irrecv.enableIRIn(); // Start the receiver
  pinMode(PIN_LED_ALARMA, OUTPUT);
  pinMode(PIN_ZUMBADOR_ALARMA, OUTPUT);
  digitalWrite(PIN_LED_ALARMA, LOW);
  digitalWrite(PIN_ZUMBADOR_ALARMA, LOW);
  pinMode(PIN_PUERTA, INPUT);
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
}
```

```
void loop() {
  if (irrecv.decode(&results)) // have we received an IR signal?
  {
    translateIR();
    irrecv.resume(); // receive the next value
  }



  // char customKey = customKeypad.getKey();


  //if (customKey){
  // Serial.println(customKey);
  //}
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  // lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  // lcd.print(millis() / 1000);
}
```

Anexo X Programa en C++ que dirige el funcionamiento del módulo de encendido de las luces y control del clima

```
/*
  Sensor de presencia para activar el cambio de temperatura de la habitación
```

Debo utilizar el ventilador y un sensor de temperatura*

Motor de 12V por Pin (este motor lleva una fuente de 12V)

DHT11 exterior Pin 5**********

DHT11 interior Pin 3

Switch puerta de entrada Pin 19

Switch puerta de balcon Pin 18

LED Rojo_Pin 52 para modoAutomatico

LED Azul_Pin 53 modoManual

PIR HC-SR501 baño Pin 8

PIR dormitorio Pin 9

IR Remote Pin 10

L293D Pin_7 ventilador y 6 aire acondicionado

6 Resistencias de 1K

Led baño Pin 38

Led dormitorio Pin 39

led balcon Pin 51

Ds3231 Pin SDA SCL GND VCC

Fotocelda Pin A0

Potenciometro Vcc y GND

LCD RS pin to digital pin 22

LCD Enable pin to digital pin 23

LCD D4 pin to digital pin 24

LCD D5 pin to digital pin 25

LCD D6 pin to digital pin 26

```
  LCD D7 pin to digital pin 27
  LCD R/W pin to ground
  LCD VSS pin to ground
  LCD VCC pin to 5V
  10K resistor:
  ends to +5V and ground
  wiper to LCD VO pin (pin 3)
*/
#include "IRremote.h"
#include <SimpleDHT.h>
#include <DS3231.h> //bibliotecas del reloj
#include <Wire.h> // bibliotecas de
#include <LiquidCrystal.h> //biblioteca de la LCD

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(22, 23, 24, 25, 26, 27);

DS3231 clock; //crea un objeto para el reloj

SimpleDHT11 DHTInt, DHTExt;

RTCDateTime dt;// aca estan alacenadas los valores del reloj

#define UMBRAL_LUZ 1
```

```
int LEDPinA = 53; //LED para modoAutomatico
int LEDPinM = 52; //modoManual
int SwitchEntrada = 19; //switch para detectar si las puerta de entrada esta abierta
int SwitchBalcon = 18; //switch para detectar si las puerta del balcon esta abierta
int pirPinDormitorio = 9; // Input for HC-S501 PIR para el dormitorio
int pirPinBano = 8; //PIR para el balcon
int pirValueDormitorio; // Place to store read PIR Value
int pirValueBano;
int Fotocelda = A0;
int PinVentilador = 7;// control on/off del ventilador de techo
int PinAire = 6;// control on/off del aire acondicionado
int DHTPinInt = 3;// sensor de temperatura interior
int DHTPinExt = 5;// sensor de temperatura interior

#define LEDBano 38
#define LEDDormitorio 39
#define LEDBalcon 51
//MotorPin                                                                    =
//*************************************************************************
**********Falta agregar todo lo del motor//en las funciones manual y automatico
int i = 0; //contador
int receiver = 10; // Signal Pin of IR receiver to Arduino Digital Pin 10
// describing Remote IR codes
```

```
/*-----( Declare objects )-----*/
IRrecv irrecv(receiver);     // create instance of 'irrecv'
decode_results results;      // create instance of 'decode_results'

/*-----( Function )-----*/
boolean estado = true; //para activar/desactivar el modo automatico
boolean bano = true; //para activar/desactivar luz de bano
boolean dormitorio = true; //para activar/desactivar luz de dormitorio
boolean balcon = true; //para activar/desactivar luz de balcon
boolean aireacond = true; //para activar/desactivar el aire acondicionado
boolean ventilador = true; //para activar/desactivar el ventilador de techo


void translateIR()
{
  switch (results.value)
  {
    case 0xFFA25D: // Serial.println("POWER") para el modo automatico/manual
      estado = !estado; //para el cambio de modo manual a automatico por el boton del mando
Remoto
      digitalWrite(LEDPinA, estado); //el valor de estado activa o desactiva la luz
      digitalWrite(LEDPinM, !estado); //el valor de estado activa o desactiva la luz
      break;

      if (!estado) { //modo manual
```

```
  digitalWrite(LEDPinM, HIGH);
case 0xFF30CF: // codigo del boton 1 para encender la luz del bano
  bano = !bano;
  digitalWrite(LEDBano, bano);
  break;

case 0xFF18E7: // codigo del boton 2 para encender la luz del dormitorio
  dormitorio = !dormitorio;
  digitalWrite(LEDDormitorio, dormitorio);
  break;

case 0xFF7A85: // codigo del boton 3 para encender la luz del balcon
  balcon = !balcon;
  digitalWrite(LEDBalcon, balcon);
  break;

case 0xFF10EF: //codigo del boton 4 para encender el aire acondicionado
  aireacond = ! aireacond;
  digitalWrite(PinAire, aireacond);
  break;

case 0xFF38C7: //codigo del boton 5 para encender el ventilador de techo
  ventilador = ! ventilador;
  digitalWrite(PinVentilador, ventilador);
```

```
      break;


    }
  }
  delay(100);
}
//*********************MANEJO DE (INTERRUPT)*******************
volatile int estadoPuerta = 0; //cerrada
int cont = 0;
void interruptHandler() {
  cont++;
  if (cont % 2 == 0) { //si es par
    estadoPuerta = ! estadoPuerta; //cambia el estado entre 0 y 1
  }


}


//####################MANEJO DE (INTERRUPT)############
volatile int estadoPuertaBalc = 0; //cerrada


void interruptHandlerBalc() {
  estadoPuertaBalc = ! estadoPuertaBalc; //cambia el estado entre 0 y 1
}
```

```
void Automatico () {
  Balcon();

  int fotocelda = analogRead(A0);//esta entre 0 y 1023 / Guarda la lectura de la señal de entrada al puerto A0 en la variable sensorValue
  digitalWrite(LEDPinA, HIGH);
  digitalWrite(LEDPinM, LOW);
  while (estadoPuerta == 0) {
    delay(1);
  }
  if (estadoPuerta == 1) {
    dt = clock.getDateTime(); // variable donde se almacenan los valores del hora y el calendario / Original de la biblioteca del reloj
    int minInicial = dt.minute;
    float voltajecell = fotocelda * ((double)5 / 1023); // Se convierte a rango de voltaje para la fotocelda /Divisor de voltaje
    Serial.println(voltajecell, 4);     // Muestra por el monitor serial el valor en voltaje que mide la fotocelda

    if (voltajecell > UMBRAL_LUZ) { //caso de q este con alta luminosidad no se encienden las luces de la habitación
      digitalWrite(LEDDormitorio, LOW);
      digitalWrite(LEDBano, LOW);
    }
    else { //caso de q este con baja luminosidad los PIR chuequean movimiento para encender las luces
```

```
    dt = clock.getDateTime();
    while (((dt.minute) - minInicial) <= 5) { // espera 5 minutos en los que se van a chequear constantemente los PIR
      pirValueBano = digitalRead(pirPinBano);
      pirValueDormitorio = digitalRead(pirPinDormitorio);

      if (pirValueBano == 1) {
        digitalWrite(LEDBano, HIGH);
      }
      else {
        digitalWrite(LEDBano, LOW);
      }
      if (pirValueDormitorio == 1) {
        digitalWrite(LEDDormitorio, HIGH);
        VentTecho (true);//para el encendido del ventilador
      }
      else {
        digitalWrite(LEDDormitorio, LOW);
        VentTecho (false);//para el apagado del ventilador
      }
      TempControl(pirValueDormitorio, pirValueBano);

    }
  }
}
```

```
}

byte temperatureInt = 0;
byte humidityInt = 0;
byte temperatureExt = 0;
byte humidityExt = 0;

bool primera_vez = true;
int hourOut;

void TempControl(int pirDorm, int pirBano) {
  if (estadoPuerta == 0) { //salio de la habitacion
    if (pirDorm == 0 && pirBano == 0) {// no hay nadie dentro

      if (primera_vez) {
        dt = clock.getDateTime(); // variable donde se almacenan los valores del hora y el calendario / Original de la biblioteca del reloj
        hourOut = dt.hour;
      }
      primera_vez = false;
      dt = clock.getDateTime();
      if ((dt.hour) - hourOut >= 2) {
        digitalWrite(PinAire, LOW);
      }
      else {
```

```
      digitalWrite(PinAire, HIGH);
    }

    while (DHTInt.read(DHTPinInt, &temperatureInt, &humidityInt, NULL)) {
      //se queda dentro del while hasta que el sensor haga una lectura
    }
    while (DHTExt.read(DHTPinExt, &temperatureExt, &humidityExt, NULL)) {
      //se queda dentro del while hasta que el sensor haga una lectura
    }
    int deltaTemp = (int)temperatureExt - (int)temperatureInt;//diferencia entre las temperaturas
    //El control on/off de la temperatura se hace en base a deltatemp
    if (deltaTemp >= 2 ) {
      AireAcond(false);//se apaga el aire acondicionado
    } else {
      AireAcond(true);//se enciende el aire acondicionado
    }

   }
  }
  else {// entro a la habitacion
   primera_vez = true;
  }
}
```

```
void MostrarTempInt() {
  while (DHTInt.read(DHTPinInt, &temperatureInt, &humidityInt, NULL)) {
    //se queda dentro del while hasta que el sensor haga una lectura
  }
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Temp: ");
  lcd.setCursor(6, 0);
  lcd.print(String(temperatureInt));
  lcd.setCursor(10, 0);
  lcd.print("*C");
}

void VentTecho(bool estado) {// si estado es true-enciende el venitlador
  if (estado) { //encender
    analogWrite(PinVentilador, 255);

  }
  else { //apagar
    analogWrite(PinVentilador, 0);
  }
}
void AireAcond(bool estado) {
  if (estado) { //encender
```

```
    analogWrite(PinAire, 255);
  }
  else { //apagar
    analogWrite(PinAire, 0);
  }
}

void modoManual() {// takes action based on IR code received
  digitalWrite(LEDPinA, LOW); // para que se vea que esta activo el modo manual
  digitalWrite(LEDPinM, HIGH); //

}
void Balcon () {
  int fotocelda = analogRead(A0);//esta entre 0 y 1023 / Guarda la lectura de la señal de entrada al puerto A0 en la variable sensorValue
  digitalWrite(LEDPinA, HIGH);
  digitalWrite(LEDPinM, LOW);

  if (estadoPuertaBalc == 1) {//puerta abierta

    digitalWrite(PinAire, LOW);

    float voltajecell = fotocelda * ((double)5 / 1023); // Se convierte a rango de voltaje para la fotocelda /Divisor de voltaje
```

```
   Serial.println(voltajecell, 4);     // Muestra por el monitor serial el valor en voltaje que mide la fotocelda


   if (voltajecell > UMBRAL_LUZ) { //caso de q este con alta luminosidad no se encienden las luces de la habitación
     digitalWrite(LEDBalcon , LOW);
   } else {
     digitalWrite(LEDBalcon , HIGH);
   }



  }
}
void setup() {
  pinMode(LEDPinA, OUTPUT);
  pinMode(LEDPinM, OUTPUT);

  pinMode(pirPinDormitorio, INPUT_PULLUP);
  pinMode(pirPinBano, INPUT);
  pinMode(LEDBano, OUTPUT);
  pinMode(LEDDormitorio, OUTPUT);
  pinMode(LEDBalcon, OUTPUT);

  pinMode(SwitchEntrada, INPUT_PULLUP);
```

```
  attachInterrupt(digitalPinToInterrupt(SwitchEntrada), interruptHandler, CHANGE);//configuracion de interrupcion

  pinMode(SwitchBalcon, INPUT_PULLUP);// el PULLIP es para estar fijado en alto el pin mientras no esta activo
  attachInterrupt(digitalPinToInterrupt(SwitchBalcon), interruptHandlerBalc, CHANGE);//configuracion de interrupcion

  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver codes

  digitalWrite(LEDPinM, LOW);
  digitalWrite(LEDPinA, HIGH);
  digitalWrite(LEDBano, LOW);
  digitalWrite(LEDDormitorio, LOW);
  digitalWrite(LEDBalcon, LOW);

  clock.begin(); //empieza a actaulizarse la hora
}

void loop() {

  MostrarTempInt();
  if (irrecv.decode(&results)) // have we received an IR signal?
  {
```

```
    translateIR();
    irrecv.resume(); // receive the next value
  }
  if (estado == false) { //modo manual activado
    modoManual();
  }
  else { //modo automatico activado
    Automatico();

  }
}
```