Universidad Politécnica de Cartagena

industriales
etsii UPCT

Desarrollo de control domótico para acondicionamiento exterior.

Titulación: Ingeniero Técnico Industrial esp. Electrónica Industrial
Intensificación: Automática
Alumno/a: Mª Teresa Mato Martínez
Director/a/s: Juan Suardíaz Muro

Cartagena, 23 de julio de 2014

ÍNDICE

CAPÍTULO 1

INTRODUCCIÓN

En este capítulo se describen los objetivos del proyecto, así como las fases que se han llevado a cabo para su desarrollo.

En referencia a los objetivos, se describirá con detalle las características del proyecto que se pretende desarrollar y se justificará el porqué de la elaboración en general del mismo, así como de aspectos más específicos. Se analizará la complejidad que puede llevar la elaboración del proyecto, el fin del mismo.

En cuanto a las fases, se describirán en cada una de ellas de manera clara y concisa los distintos aspectos que serán objeto de estudio y desarrollo. Desde la primera fase sobre el estudio del arte hasta la última sobre conclusiones y trabajos futuros, se expondrán los distintos objetivos a alcanzar y mostrar en cada una de ellas, de manera que sirva tanto al autor en la realización del proyecto, como a cualquier persona que lea esta memoria, como guion de cómo se pretende llevar a cabo todo el proceso de realización del proyecto.

1.1 OBJETIVOS DEL PROYECTO

El presente proyecto se enfoca en la domótica para la eficientización de la gestión energética en edificaciones, haciendo uso de microcontroladores y protocolos de comunicación inalámbrica para el desarrollo de soluciones en dicho sector.

El objetivo del proyecto es el desarrollo de una interfaz de control de uso domótico de dos luminarias externas basadas en tecnología LED a partir de un sensor de presencia, estudiando a fondo tanto el proceso de creación hardware como el software de control.

En este proyecto fin de carrera, se presenta un adelanto de estas soluciones planteadas, específicamente el diseño e implementación de un dispositivo para el control inteligente de iluminación de tecnología LED.

1.2 ESTRUCTURA DE LA MEMORIA DEL PROYECTO

Aparte de este primer capítulo de introducción, donde se detallan los objetivos del proyecto, la memoria se ha estructurado en los siguientes capítulos:

Capítulo 2. Estado del arte de los sistemas domóticos

En una primera fase, se realizará un estudio sobre los aspectos más significativos que engloba el mundo de la domótica. Para comenzar, se hará un repaso histórico de la domótica a nivel internacional, para centrarse en la evolución que ha sufrido a nivel nacional. Después, se estudiarán los principales aspectos técnicos de los elementos que intervienen, como son los sensores y las luminarias.

Así mismo, se verá los tipos de arquitecturas en las que se clasifica la domótica y los medios de transmisión más usados. Por último, se realizará un análisis de los distintos fabricantes y empresas que gozan de prestigio comercial a nivel internacional.

Capítulo 3. Arquitectura Hardware de Control

El presente capítulo, uno de los más extensos debido a su gran importancia, se va a profundizar en el diseño de la arquitectura hardware encargada del control del sistema. Para comenzar, se analizará todo lo relacionado con el microcontrolador, para llegar seguidamente al universo Arduino y finalizar con la tecnología inalámbrica ZigBee, que se usará en el presente proyecto.

Se realizará un estudio teórico pormenorizado de cada uno de los bloques presentados, atendiendo a sus diferentes principios, características y tipologías disponibles para la futura implementación en el proyecto.

En cada sección abordada, se presentará una visión global sobre cada uno de los dispositivos que se pueden emplear, para después argumentar el uso de los modelos concretos elegidos de entre la variedad disponible en el mercado.

Capítulo 4. Implementación Hardware

Tras el análisis de las características la estructura hardware de control, se estará ya en disposición de realizar la construcción de los módulos, que supone uno de los objetivos principales a llevar a cabo en este proyecto.

En este capítulo, se abordará el montaje e implementación de los diferentes bloques funcionales que forman el control domótico, detallando todo lo relacionado a sus componentes, esquemas de montaje y proceso de fabricación.

Para finalizar este capítulo se ofrecerá una visión de los resultados finales, acabando con una pequeña referencia al software que ha sido utilizado durante el diseño a lo largo del capítulo.

Capítulo 5. Implementación Software

En este penúltimo capítulo se realiza un análisis del software de control del que se ha dotado al microcontrolador que incorpora la plataforma Arduino.

En este análisis se explica el software del módulo emisor y del módulo receptor por separado, comentando sus principales características, así como las similitudes y diferencias existentes entre ambos módulos.

Capítulo 6. Conclusión y Trabajos Futuros

En este último capítulo se expondrán las conclusiones que se obtienen tras la elaboración del presente proyecto. A continuación se realizará una valoración a título personal sobre lo que supone al autor el desarrollo y elaboración del proyecto. Por último se trazarán aquellas posibles líneas en las que se puede orientar la elaboración de futuros trabajos y proyectos relacionados con el mundo de la domótica y en particular con el proyecto expuesto en esta memoria.

Anexo I. Software Desarrollado

Este anexo abarcará las líneas completas del programa que forman el software del proyecto. Se encuentra dividido en dos bloques que contendrán el código del módulo emisor y del módulo receptor por separado y de manera ininterrumpida.

CAPÍTULO 2

ESTADO DEL ARTE DE LOS SISTEMAS DOMOTICOS

En este segundo capítulo se tratará sobre el estado del arte de la domótica, donde se realizará un estudio sobre los aspectos más importantes que engloban a éste.

Para comenzar, se hará un repaso histórico a nivel internacional, para centrarse en la evolución que ha sufrido en España.

También se analizarán aspectos técnicos de funcionamiento además de profundizar en los principios físicos de mayor importancia.

Posteriormente se estudiarán los principales elementos que intervienen en el mundo del control electrónico domótico, como son las luminarias y las plataformas de mando o controladores analógicos y digitales.

Por último, se realizará un análisis de los distintos fabricantes que gozan de prestigio comercial a nivel internacional.

2.1 INTRODUCCIÓN

La domótica es el conjunto de tecnologías aplicadas al control y la automatización inteligente de la vivienda, que permite una gestión eficiente del uso de la energía, además de aportar seguridad, confort, y comunicación entre el usuario y el sistema. Un sistema domótico es capaz de recoger información proveniente de unos sensores o entradas, procesarla y emitir órdenes a unos actuadores o salidas. El sistema puede acceder a redes exteriores de comunicación o información.

La domótica aplicada a edificios no destinados a vivienda, es decir oficinas, hoteles, centros comerciales, de formación, hospitales y terciario, se denomina, inmótica.

La domótica permite dar respuesta a los requerimientos que plantean estos cambios sociales y las nuevas tendencias de nuestra forma de vida, facilitando el diseño de casas y hogares más humanos, más personales, polifuncionales y flexibles.

Según el destinatario podemos hacer distintas definiciones:

- Para el usuario: Aquella que permite una mayor calidad de vida a través de la tecnología, ofreciéndole un aumento del bienestar y la seguridad de los habitantes, a la vez que una reducción de las tareas domésticas y una racionalización de los distintos consumos.

- Para el técnico: Aquella que incluye agrupaciones automatizadas de equipos normalmente asociadas por funciones, que disponen de la capacidad de

comunicarse interactivamente entre ellas a través de un soporte de comunicaciones que las integra.

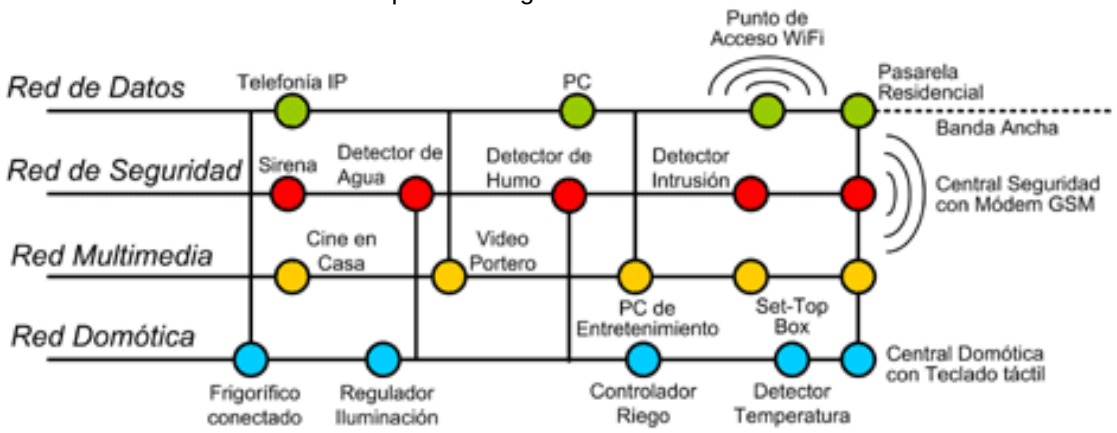


Fig. 2.1 *Conexiones de red domésticas*

El sector de la domótica ha evolucionado considerablemente en los últimos años, y en la actualidad ofrece una oferta más consolidada. Hoy en día, la domótica aporta soluciones dirigidas a todo tipo de viviendas, incluidas las construcciones de vivienda oficial protegida. Además, se ofrecen más funcionalidades por menos dinero, más variedad de producto, y gracias a la evolución tecnológica, son más fáciles de usar y de instalar. En definitiva, la oferta es mejor y de mayor calidad, y su utilización es ahora más intuitiva y perfectamente manejable por cualquier usuario. Paralelamente, los instaladores de domótica han incrementado su nivel de formación y los modelos de implantación se han perfeccionado. Asimismo, los servicios posventa garantizan el perfecto mantenimiento de todos los sistemas. Por tanto, la domótica de hoy contribuye a aumentar la calidad de vida, hace más versátil la distribución de la casa, cambia las condiciones ambientales creando diferentes escenas predefinidas, y consigue que la vivienda sea más funcional al permitir desarrollar facetas domésticas, profesionales, y de ocio bajo un mismo techo.

La red de control del sistema domótico se integra con la red de energía eléctrica y se coordina con el resto de redes con las que tenga relación: telefonía, televisión, y tecnologías de la información, cumpliendo con las reglas de instalación aplicables a cada una de ellas. Las distintas redes coexisten en la instalación de una vivienda o edificio. La instalación interior eléctrica y la red de control del sistema domótico están reguladas por el Reglamento Electrotécnico para Baja Tensión (REBT). En particular, la red de control del sistema domótico está regulada por la instrucción ITC-BT-51 Instalaciones de sistemas de automatización, gestión técnica de la energía y seguridad para viviendas y edificios.

2.2 UTILIDADES Y APLICACIONES DE SISTEMAS DOMÓTICOS

En este apartado se detallan ciertos sectores y aplicaciones en los que la domótica se está convirtiendo en una parte muy importante. A continuación, enumeramos algunos de ellos:

1. **La domótica contribuye a mejorar la calidad de vida del usuario:** Facilitando el ahorro energético ya que gestiona inteligentemente la iluminación, climatización, agua caliente sanitaria, el riego, los electrodomésticos, etc., aprovechando mejor los recursos naturales, utilizando las tarifas horarias de menor coste, y reduce de esta manera la factura energética. Además, mediante la monitorización de consumos, se obtiene la información necesaria para modificar los hábitos y aumentar el ahorro y la eficiencia.

2. **Fomentando la accesibilidad:** facilita el manejo de los elementos del hogar a las personas con discapacidades de la forma que más se ajuste a sus necesidades, además de ofrecer servicios de teleasistencia para aquellos que lo necesiten.

3. **Aportando seguridad de personas, animales y bienes:** controles de intrusión y alarmas técnicas que permiten detectar incendios, fugas de gas o inundaciones de agua, etc.
4. **Convirtiendo la vivienda en un hogar más confortable**: gestión de electrodomésticos, climatización, ventilación, iluminación natural y artificial…

5. **Garantizando las comunicaciones**: recepción de avisos de anomalías e información del funcionamiento de equipos e instalaciones, gestión remota del hogar, etc.

6. **Comunicaciones:** Transmisión de voz y datos, incluyendo textos, imágenes, sonidos (multimedia) con redes locales (LAN) compartiendo acceso a Internet, recursos e intercambio entre todos los dispositivos, acceso a nuevos servicios de telefonía sobre IP, televisión digital, televisión por cable, diagnóstico remoto, videoconferencias, etc.

7. **Mantenimiento:** Con capacidad de incorporar el telemantenimiento de los equipos.

8. **Ocio y tiempo libre**: Descansar y divertirse con radio, televisión, multi-room, cine en casa, videojuegos, captura, tratamiento y distribución de imágenes fijas (foto) y dinámicas (vídeo) y de sonido (música) dentro y fuera de la casa, a través de Internet, etc.

9. **Salud:** Actuar en la sanidad mediante asistencia sanitaria, consultoría sobre alimentación y dieta, telecontrol y alarmas de salud, medicina monitorizada, cuidado médico, etc.

10. **Compra:** Comprar y vender mediante la telecompra, televenta, telereserva, desde la casa, etc.

11. **Finanzas:** Gestión del dinero y las cuentas bancarias mediante la telebanca, consultoría financiera….

12. **Aprendizaje:** Aprender y reciclarse mediante la tele-enseñanza, cursos a distancia…

13. **Actividad profesional:** Trabajar total o parcialmente desde el hogar, posibilidad viable para ciertas profesiones (teletrabajo), etc.

14. **Ciudadanía:** Gestiones múltiples con la Administración del Estado, la Comunidad Autónoma y el Municipio, voto electrónico, etc.

15. **Acceso a información:** Museos, bibliotecas, libros, periódicos, información meteorológica, etc.

Y todas las posibles ideas que la creatividad y la innovación puedan aportar. Lo indicado hasta aquí es sólo una muestra del actual estado de conocimiento y progreso.

2.3 RESEÑA HISTÓRICA DE LA DOMÓTICA

La evolución marca el ritmo de la vida y las casas tampoco pueden escapar a ella. De la cueva con fuego, para calentar e iluminar, a las antorchas, las velas, el candil y por último: la electricidad.

La electricidad nos ha permitido elevar el nivel de confort en nuestras casas y ha dado paso a la entrada de los electrodomésticos: lavadora, frigorífico, lavavajillas, horno, placas vitrocerámicas,… máquinas capaces de realizar tareas cotidianas de forma casi autónoma, elevando nuestro nivel de confort a cotas en otro tiempo inimaginables.

Estas máquinas no existirían sin el desarrollo de una nueva evolución: la electrónica, permitiendo realizar programaciones (rutinas), que regulan cada proceso.

La siguiente evolución que ha llegado es la: Domótica, que se encarga de la integración y regulación de ambos sistemas (eléctricos y electrónicos), de tal manera que "la casa" es capaz de "sentir" (detectar la presencia de personas, la temperatura, el nivel de luz,…) y reaccionar por sí sola, a estos estímulos (regulando el clima, la iluminación, conectando la alarma,…), al mismo tiempo que es capaz de comunicarse e interactuar con nosotros (telecontrol) por multitud de medios(pantalla táctil, PC, móvil,...), llegando a elevadas cotas de confort, seguridad y sobretodo: ahorro energético.

La Historia de la domótica comprende una serie de etapas, desde los primeros protocolos orientados al "control remoto", hasta los grandes protocolos capaces de realizar "funciones lógicas complejas", para satisfacer las más exigentes programaciones de regulación y preparados para la verdadera Revolución Domótica: La autorregulación.

El origen de la domótica se remonta a los años setenta, cuando, tras muchas investigaciones aparecieron los primeros dispositivos de automatización de edificios basados en la aún exitosa tecnología X-10. Durante los años siguientes la comunidad internacional mostró un creciente interés por la búsqueda de la casa ideal, comenzando diversos ensayos con avanzados electrodomésticos y dispositivos automáticos para el hogar. Los primeros sistemas comerciales fueron instalados, sobre todo, en Estados Unidos y se limitaban a la regulación de la temperatura ambiente de los edificios de oficinas y poco más.

Más tarde, con el auge de los PC's (Personal Computer) a finales de la década de los 80 y principios de los 90, se empezaron a incorporar en estos edificios los Sistemas de Cableado Estructurado (SCE) para facilitar la conexión de todo tipo de terminales y periféricos entre sí, utilizando un cableado estándar y tomas repartidas por todo el edificio. Además de los datos, estos sistemas de cableado permitían el transporte de la voz y la conexión de algunos dispositivos de control y de seguridad, por lo que a aquellos edificios, que disponían de un SCE, se les empezaron a llamar edificios inteligentes.

Posteriormente, todos estos automatismos destinados a edificios de oficinas, se han ido aplicando también a las viviendas de particulares u otro tipo de edificios donde el número de necesidades que hay que cubrir es mucho más amplio, dando origen a la vivienda domótica.

Tras una etapa de introducción lenta de la tecnología digital, ahora estamos en los comienzos de una revolución de los servicios para el hogar, donde las pasarelas residenciales, apoyadas con conexiones de banda ancha, conectarán inteligentemente todos los dispositivos del hogar, soportando una gran diversidad de servicios interactivos.

Fig. 2.2 *Vivienda controlada domóticamente*

2.3.1 La domótica en España

Los orígenes de la domótica a España se pueden situar alrededor del año 1990, fecha en la que se empiezan a llevar a cabo las primeras iniciativas e investigaciones principalmente por el Institut Cerdà. Al principio, el mercado se caracterizaba por un gran desconocimiento de la domótica tanto en el ámbito tecnológico como de posibilidades y aplicaciones por lo que el interés que suscitaba este adelanto tecnológico era muy limitado y su investigación mínima.

Los primeros sistemas estaban poco integrados y las áreas de gestión que se cubrían eran, a duras penas, el aspecto de la confortabilidad y la seguridad, aunque también cabe destacar que había otras aplicaciones más aisladas tales como la gestión de las comunicaciones y la energía.

Las características de aquel mercado al que se enfrentaban los sistemas domóticos españoles solían ser que, generalmente, los productos estaban fabricados atendiéndose a las normativas europeas y destinados a mercados extranjeros más desarrollados, había ciertas dificultades a la hora de diseñar e instalar dispositivos, al carecer de suficiente personal cualificado ya que era una novedad en el campo tecnológico.

El coste de las instalaciones era muy elevado y estas resultaban poco productivas y no había entidades públicas o privadas especializadas en instalaciones de este tipo ni interés por abrir un mercado ya que, al existir, relativamente, poca demanda, se verían avocados a pérdidas gananciales. Había desconfianza y reticencia por parte de los usuarios al encontrarse delante de algo que podía poner en riesgo la seguridad de los edificios o viviendas (o incluso los mismos usuarios) debido a una excesiva automatización.

A pesar de que actualmente la situación se diferencia notablemente respecto otros países, no hay lugar a dudas de que en los próximos diez años las instalaciones automatizadas serán un valor añadido de las construcciones, ya que el mercado actual se caracteriza por los aspectos como el de la creación de nuevas empresas dedicadas a la fabricación e instalación de sistemas automatizados, avances en la normalización y homologación de determinados productos así como el rechazo a otros que no cumplen la normativa tecnológica española y/o europea, desarrollo de nuevos sistemas por parte de las empresas del sector electrónico, creación de organismos de investigación y desarrollo (I+D), la apuesta por el progreso y la innovación mediante sistemas domóticos y la financiación por parte de las instituciones públicas junto con la Comunidad Económica Europea de proyectos I+D entre otros.

El progreso de la domótica española no habría podido llevarse a cabo sin los organismos e instituciones dedicadas a su óptimo desarrollo que, asumiendo un riesgo económico considerable han apostado por esta nueva tecnología. Las principales entidades comprometidas en este aspecto son las siguientes:

- ✓ CEDOM (Comité Español para la Gestión Técnica de Edificios y Viviendas).
- ✓ AFME (Asociación de Fabricantes de Materiales Eléctricos).
- ✓ EIBA (Asociación del Bus de Instalaciones Europeas).
- ✓ Institut Cerdà.
- ✓ Ministerio de Fomento.
- ✓ IDEA (Instituto para la Diversificación y Ahorro de Energía).
- ✓ El Mundo de la Domótica (revista de sistemas de control y gestión técnica de edificios).

El mercado español cuenta con más de 25 sistemas (casas equipadas con componentes automatizados) domóticos y un gran número de productos con prestaciones cada vez más atractivas y asequibles para los usuarios no familiarizados con el sector. Por otro lado, dichos sistemas se están implantando en el 60% de hogares de nueva construcción y en el 40% de hogares ya existentes, la cual cosa implica una paulatina normalización (aun estando en estado embrionario) de los dispositivos automatizados a nuestras casas.

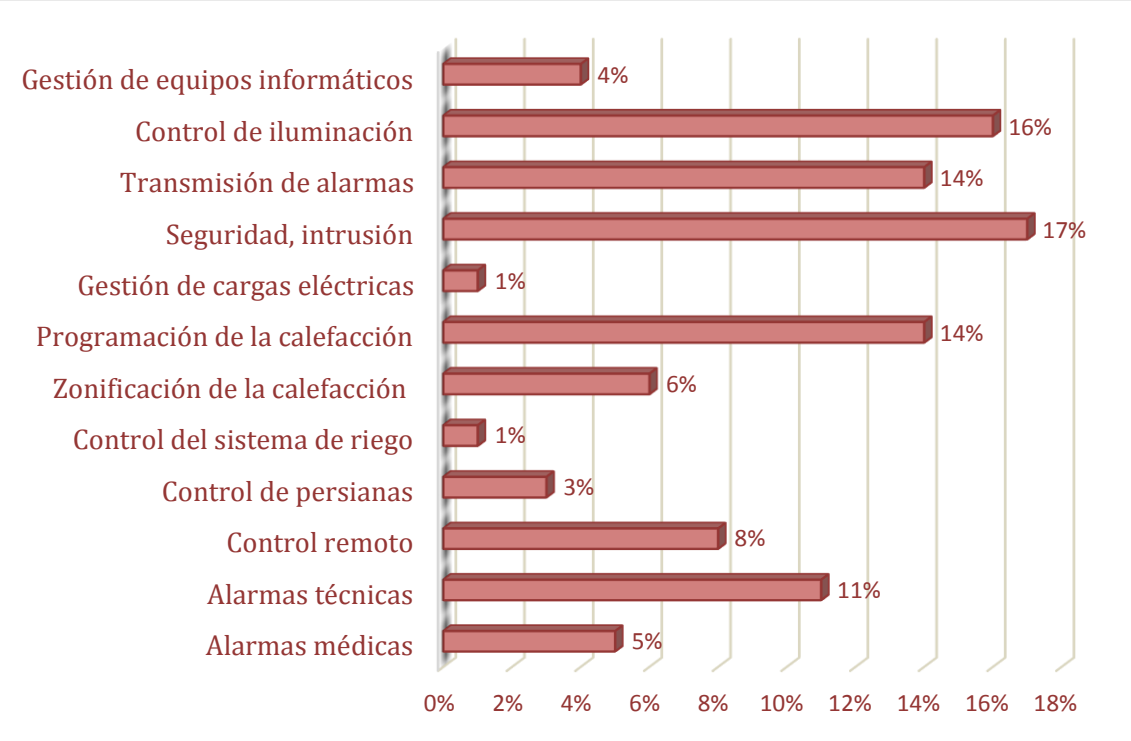


Fig. 2.3 *Gráfica de las aplicaciones domóticas más solicitadas por el cliente, España, 2000 (%)*

2.4 ARQUITECTURAS

La arquitectura de un sistema domótico, como la de cualquier sistema de control, especifica el modo en que los diferentes elementos de control del sistema se van a ubicar. El uso de diferentes tipos de cableado o de red implica diferencias notables en parámetros como la complejidad del cableado, velocidad de transmisión, vulnerabilidad, gestión de la red, tasa de fallos, etc. (Aguirre & Zapata, 2006)

Existen dos arquitecturas básicas: la arquitectura centralizada y la distribuida.

2.4.1 Arquitectura centralizada

Es aquella en la que los elementos a controlar y supervisar han de cablearse hasta el sistema de control de la vivienda (computador). El controlador centralizado recibe información de múltiples sensores y, una vez procesada, genera las órdenes oportunas para los actuadores. El sistema de control es el corazón de la vivienda, en cuya falta todo deja de funcionar, y su instalación no es compatible con la instalación eléctrica convencional en cuanto que en la fase de construcción hay que elegir esta topología de cableado.

Fig. 2.13 *Esquema arquitectura centralizada*

Entre sus ventajas destacan:

- Bajo coste ya que ningún elemento necesita módulos especiales de direccionamiento ni interfaces para distintos buses.
- Instalación sencilla y posibilidad de utilizar una gran variedad de elementos comerciales.
- Requerimientos mínimos.

Y entre sus inconvenientes:

- Flexibilidad limitada ya que las reconfiguraciones son costosas.
- Poca robustez puesto que si cae el módulo central cae todo el sistema.
- Mayor longitud de cableado dada la topología, lo que incrementa el coste de la instalación y limita su uso en grandes instalaciones.

2.4.2 Arquitectura distribuida

En este caso no existe la figura del controlador centralizado sino que toda la inteligencia del sistema está distribuida por todos los módulos, sean sensores o actuadores. En este tipo de arquitectura el elemento de control se sitúa próximo al elemento a controlar. Hay sistemas que son de arquitectura distribuida en cuanto a la capacidad de proceso, pero no lo son en cuanto a la ubicación física de los diferentes elementos de control y viceversa, sistemas que son de arquitectura distribuida en cuanto a su capacidad para ubicar elementos de control físicamente distribuidos, pero no en cuanto a los procesos de control, que son ejecutados en uno o varios procesadores físicamente centralizados.

En los sistemas de arquitectura distribuida que utilizan como medio de transmisión el cable, existe un concepto a tener en cuenta que es la topología de la red de comunicaciones. La topología de la red se define como la distribución física de los elementos de control respecto al medio de comunicación.

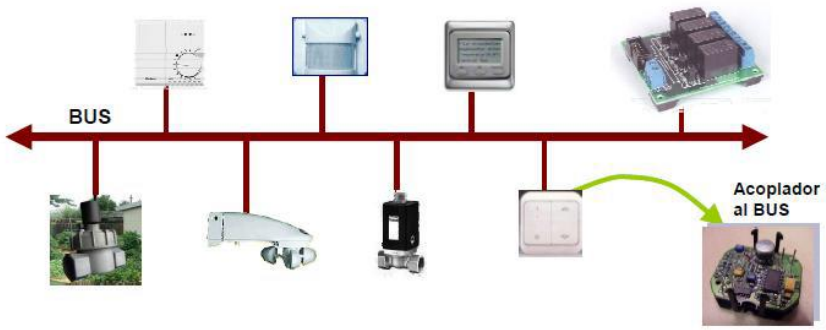


Fig. 2.14 *Esquema arquitectura distribuida*

La arquitectura distribuida es típica de los sistemas con topología en bus y se requiere un protocolo de comunicaciones. Todos los elementos disponen de un acoplador al bus con una interfaz de acceso compartido y técnicas de direccionamiento para que la recepción y el envío de información quede definida y el diálogo entre elementos asegurado. Es habitual, además, que se permitan cableados de topología libre, de manera que se facilita su instalación en la vivienda o edificio.

Las principales ventajas de los sistemas distribuidos son:

- Alta flexibilidad y una gran facilidad para reconfiguraciones.
- Escalabilidad. Suelen ser adaptables a cualquier tamaño de instalación y las ampliaciones resultan sencillas.
- Posibilidad de tecnologías plug & play que simplifican mucho las instalaciones.
- Ahorro de cableado en la instalación, lo que reduce los costes, sobre todo en instalaciones y proyectos a gran escala.

Sus inconvenientes:

- Mayor precio de los componentes, dado el incremento de complejidad que conllevan por la necesidad de incluir los protocolos y técnicas de direccionamiento utilizados.
- Necesidad de compatibilidad entre los equipos y componentes.
- Oferta de productos restringida al protocolo que emplean para garantizar la compatibilidad entre ellos.

2.5 MEDIOS DE TRANSMISIÓN

En todo sistema domótico los diferentes elementos de control deben intercambiar información unos con otros a través de un soporte físico (par trenzado, línea de potencia o red eléctrica, radio−frecuencia, infrarrojos, etc.).
Cada protocolo utiliza un medio de transmisión específico. A continuación enumeramos los siguientes tipos de medios:

2.5.1 Líneas de distribución de energía eléctrica (corrientes portadoras)

Si bien no es el medio más adecuado para la transmisión de datos, sí es una alternativa a tener en cuenta para las comunicaciones domésticas dado el bajo coste que implica su uso, ya que se trata de una instalación existente.

Para aquellos casos en los que las necesidades del sistema no impongan requerimientos muy exigentes en cuanto a la velocidad de transmisión, la línea de distribución de energía eléctrica puede ser suficiente como soporte de dicha transmisión. Dada las especiales características de este medio y, sobretodo, su idoneidad para las instalaciones domesticas lo hacen poseedor de una serie de ventajas e inconvenientes tales como el nulo coste de instalación, la facilidad del conexionado y la poca fiabilidad además de la baja transmisión de datos. (Arias, 2004)

2.5.2 Soportes metálicos

La infraestructura de las redes de comunicación actuales, tanto públicas como privadas, tiene en un porcentaje muy elevado de cables metálicos de cobre como soporte de transmisión de las señales eléctricas que procesa. En general se pueden distinguir dos tipos de cables metálicos: el par metálico y el coaxial.

- **Par metálico:** son cables formados por varios conductores y pueden dar un soporte a un amplio rango de aplicaciones en el entorno doméstico. Están diseñados para transportar señales de voz, datos y alimentación de corriente continua.

- **Coaxial:** Un par coaxial es un circuito físico asimétrico, constituido por un conductor filiforme que ocupa el eje longitudinal del otro conductor en forma de tubo. Este tipo de cables permite el transporte de las señales de video y señales de datos a alta velocidad. Dentro del ámbito de la vivienda, el cable coaxial puede ser utilizado como soporte de transmisión para señales de tele−difusión que provienen de las antenas de televisión, televisión por cable, radio y señales de control y datos a baja y media velocidad.

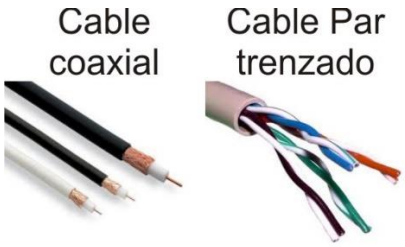


Fig. 2.15 *Principales soportes metálicos con hilos*

2.5.3 Otras alternativas

Además de los soportes metálicos descritos anteriormente, existen otros tipos de medios de transmisión que suponen un avance en el campo de la domótica:

- **Fibra óptica**: La fibra óptica es el resultado de combinar dos disciplinas no relacionadas, como son la tecnología de semiconductores (que proporciona los materiales necesarios para las fuentes y los detectores de luz), y la tecnología de guiado de ondas ópticas (que proporciona el medio de transmisión, el cable de fibra óptica).
  La fibra óptica está constituida por un material dieléctrico transparente, conductor de luz, compuesto por un núcleo con un índice de refracción menor que el del revestimiento, que envuelve a dicho núcleo. Estos dos elementos forman una guía para que la luz se desplace por la fibra. La luz transportada es generalmente infrarroja, y por lo tanto no es visible por el ojo humano.
  Algunas de sus múltiples ventajas y escasos inconvenientes son la gran fiabilidad en la transferencia de datos, la inmunidad frente a interferencias electromagnéticas y de radiofrecuencias, la elevada seguridad en la transmisión de datos y el elevado coste del cableado y las conexiones.

- **Conexiones Wireless (sin hilos):** El uso de mandos a distancia basados en transmisión por infrarrojos está ampliamente extendido en el mercado residencial para tele−comandar equipos de audio y vídeo. La comunicación se realiza entre un diodo emisor que emite una luz en la banda de IR, sobre la que se superpone una señal, convenientemente modulada con la información de control, y un fotodiodo receptor cuya misión consiste en extraer de la señal recibida la información de control.

Los controladores de equipos domésticos basados en la transmisión de ondas en la banda de los infrarrojos tienen como principales ventajas: comodidad y flexibilidad, que admiten un gran número de aplicaciones, y Al tratarse de un medio de transmisión óptico es inmune a las radiaciones electromagnéticas producidas por los equipos domésticos. Mientras que sus principales desventajas e inconvenientes son: la alta sensibilidad a las interferencias, la fácil intervención de las comunicaciones y su dificultad para la integración de las funciones de control y comunicación, en su modalidad de transmisión análoga.

2.6 FABRICANTES

2.6.1 KNX

KNX es un protocolo de comunicaciones abierto para el control de la vivienda y el edificio nacido a partir de la convergencia de los sistemas Batibus, EIB y EHS, y aprobado como:

- Estándar Europeo (CENELEC EN 50090 y CEN EN 13321-1)
- Estándar Internacional (ISO/IEC 14543-3)
- Estándar Chino (GB/Z 20965)
- Estándar Norteamericano (ANSI/ASHRAE 135)

A diferencia de X10, que utiliza la red eléctrica, y otros sistemas actuales por RF, el KNX utiliza su propio cableado, con lo cual se ha de proceder a instalar las conducciones adecuadas en el hogar y para el sistema.

El KNX, a través de pasarelas, puede ser utilizado en sistemas inalámbricos como los infrarrojos, radiofrecuencia o incluso empaquetado para enviar información por internet u otra red TCP/IP.

Originariamente conocido por Instabus, ingeniería de donde salieron los primeros esbozos, está abrazado por un conjunto de empresas (en su mayoría alemanas) y lleva más de 20 años en el mercado de la automatización penetrando lentamente en un mercado reticente como es la construcción, a pesar de que, es un sistema muy robusto y fiable.

2.6.1.1 Principales ventajas

Las principales ventajas de este protocolo son:

1. Estándar Internacional que garantiza su continuidad en el futuro.

   KNX es el único ESTANDAR Abierto para el Control de Casas y Edificios
   a. ISO/IEC
      Aprobó la tecnología KNX como el Estándar Internacional IS/IEC 14543-3 en 2006
   b. CENELEC
      Aprobó la tecnología KNX como el Estándar Europeo EN 50090 en 2003
   c. CEN
      Aprobó la tecnología NKX como EN13321-1 y EN1332-2 en 2006
   d. SAC
      Aprobó la tecnología KNX como Estándar GB/T 20965 en China en 2013
   e. ANSI/A SHRAE
      Aprobó la tecnología KNX como el Estándar Estadounidense ANSI/ASHRAE en 2005

2. Gracias a la certificación de producto, KNX garantiza Interoperabilidad Y Interworking de productos

   El proceso de certificación KNX asegura que funcionarán y se comunicarán diferentes productos de diferentes fabricantes usados en diferentes aplicaciones. Esto asegura un alto grado de flexibilidad en la extensión y modificaciones de las instalaciones. Laboratorios neutrales (terceras compañías) son quienes analizan la conformidad del producto.
   KNX es el único estándar para el control de casas y edificios que lleva a cabo un plan de certificación para productos, centros de formación (instituciones profesionales y privadas) e incluso personas (electricistas, proyectista).

3. KNX representa alta calidad de producto

   La KNX Association exige un alto nivel de producción y control de calidad durante todas las etapas de la vida del producto. Por lo que todos los miembros fabricantes tienen que mostrar conformidad a la norma ISO 9001 incluso antes de que soliciten la certificación para productos KNX.
   Además de la conformidad del fabricante a la norma ISO 9001, los productos tienen que cumplir con los estándares tanto Europeos como Internacionales para el control de Casas y Edificios. En caso de duda, la KNX Association tiene el derecho de volver a analizar el producto o puede exigir al fabricante el informe de conformidad de dicho hardware.

4. Un único software independiente del fabricante ETS (Engineering Tool Software)

   La herramienta software ETS permite proyectar, diseñar y configurar todos los productos certificados KXN. Dicha herramienta es además independiente del fabricante: el integrador de sistemas podrá combinar los productos de varios fabricantes en una instalación.

5. KNX puede ser usado para todas las aplicaciones en el control de casas y edificios

   KNX puede ser usado para el control de todas las posibles funciones/aplicaciones en casas y edificios desde iluminación, contraventanas, control de seguridad y alarmas, calefacción, ventilación, aire acondicionado, control de agua y dirección de energía, medición, hasta aplicaciones para el hogar, audio y mucho más.

   KNX mejora el confort y la seguridad con sus instalaciones así como contribuye al ahorro energético y la protección del clima (se puede conseguir hasta un 50% de ahorro en iluminación y calefacción)

6. KNX soporta diferentes modos de configuración

   KNX ofrece diferentes niveles de configuración para la realización de sus proyectos: vía E-mode, diseñadores no cualificados podrán diseñar proyectos. Vía S-mode, los integradores podrán realizar instalaciones sofisticadas. Los diferentes modos de configuración son:

   - Easy installation (E-mode):
     La configuración es hecha sin el uso de un ordenador pero sí necesita algún tipo de controlador central. La compatibilidad de los productos E-mode

normalmente tiene limitada su funcionalidad y ha sido pensada para instalaciones de tamaño medio.

- System installation (S-Mode):
  El diseño de la instalación y la configuración está hecha a través de un ordenador con el software ETS, a través del cual se usa la base de datos del producto de cada fabricante. S-mode está pensado para integradores de sistemas certificados y para grandes instalaciones.

7. KNX soporta diferentes medios de comunicación

   Cada medio de comunicación puede ser usado en combinación con uno o más modos d configuración, lo que permite a cada fabricante elegir la combinación perfecta para su segmento de mercado y aplicaciones.

   - Par trenzado (KNX TP):
     KNX es transmitido a través de un cable bus separado, con una estructura jerarquizada en líneas y áreas.
   - Corrientes portadoras (KNX PL):
     KNX es transmitido sobre la red eléctrica existente.
   - Radio frecuencia (KNX RF):
     KNX es transmitido por señales de radio. Estos dispositivos pueden ser unidireccionales o bidireccionales.
   - IP/Ethernet (IP KNX):
     Este conocido medio de comunicación puede ser usado en conjunción con las especificaciones 'KNXnet/IP', que permiten enviar tramas KNX encapsuladas en tramas IP.

8. KNX es independiente de cualquier plataforma hardware o software

   KNX puede ser llevada a cabo bajo cualquier plataforma de microprocesador. KNX puede ser implementada desde el principio, pero para una entrada más sencilla en el mercado, los fabricantes KNX también pueden recurrir a los proveedores de componentes KNX

2.6.2 LonWorks

LonWorks es una plataforma de control creada por la compañía norteamericana Echelon. Las redes LonWorks describen de una manera efectiva una solución completa a los problemas de sistemas de control. Al igual que la industria informática, la industria del control fue creada, y en muchos casos todavía lo es, basada en soluciones centralizadas de control punto-a-punto. Esto significa que existe un "maestro" o controlador principal similar a un ordenador, físicamente cableado a cada punto de control particular, como actuadores o sensores, denominados "esclavos". El resultado final es funcional, pero es caro y difícil para mantener, ampliar y gestionar. Igualmente, es menos fiable frente a fallos, ya que la caída del controlador principal provoca la caída de todo el sistema. (Aguirre & Zapata, 2006)

El comienzo de las redes LonWorks se basó en conceptos muy simples:

1) Los sistemas de control son fundamentalmente idénticos, independientemente de la aplicación final.

2) Un sistema de control distribuido es significativamente más potente, flexible, y ampliable que un sistema de control centralizado.

3) Las empresas ahorran más dinero a largo plazo instalando redes distribuidas que instalando redes centralizadas.

La tecnología LonWorks proporciona una solución a los múltiples problemas de diseño, construcción, instalación, y mantenimiento de redes de control; redes que pueden variar en tamaño desde dos a 32,000 dispositivos y se pueden usar en cualquier aplicación desde supermercados a plantas de petrolíferas, desde aviones hasta ferrocarriles, desde edición por láser a máquinas de mecanizado, desde rascacielos a viviendas particulares o los esquemas de control basados en sistemas centralizados.

Los fabricantes están utilizando sistemas abiertos, chips estándar, sistemas operativos estándar y componentes para construir productos que mejoren la flexibilidad, el costo del sistema y su instalación. La tecnología LonWorks está acelerando la tendencia a evitar los sistemas propietarios o los sistemas centralizados, proporcionando interoperabilidad, una tecnología robusta, desarrollos más rápidos y ahorro económico. (Echelon)

En términos de interoperabilidad y compatibilidad, Lonworks es a las redes de domótica lo que WINDOWS es a los sistemas informáticos, en el sentido que sirve de protocolo común para la compatibilidad entre múltiples fabricantes.

LonWorks es un estándar abierto para cualquier fabricante, está basado en una arquitectura distribuida y multimedia tanto de proceso como de ubicación, es decir, cada elemento del sistema tiene propia capacidad de proceso y puede ser ubicado en cualquier parte de la vivienda. Esta característica proporciona al instalador domótico una libertad de diseño que le posibilita adaptarse a las características físicas de cada vivienda en particular.

Cada sistema de control está compuesto básicamente por los siguientes componentes: sensores, actuadores, programas de aplicación, redes de comunicación, interfaces hombre-máquina y herramientas para el manejo de red.

En esta arquitectura los nodos se conectan entre si utilizando el protocolo LonWorks y por cualquier medio que sea el más conveniente, a esta característica se le denomina multimodal. Los medios disponibles son:

- ✓ par trenzado
- ✓ línea directa
- ✓ radio frecuencia
- ✓ fibra óptica
- ✓ infrarrojo

La tecnología LonWorks tiene los siguientes elementos fundamentales:

- Neuron Chip Control processor y transceivers
- Protocolo de Comunicación LonTalk

2.6.2.1 Neuron Chip Control Processor y Transceivers

Los transceivers son dispositivos emisores-receptores que se encargan de conectar las neuronas con el medio de transmisión. Mientras que los neuron chip son circuitos integrados que contienen tres microprocesadores y memoria, memorias ROM y RAM, interfaces de comunicación y puertos de entrada y salida (I/O).

La ROM del chip contiene el sistema operativo, el protocolo de comunicación LonTalk y una librería de funciones de aplicación. Cada Neuron Chip contiene un único código de 48 bits llamado “Neuron ID”, estos chips están disponibles en una serie de velocidades, capacidades de

memoria e interfaces diferentes y actualmente son fabricados por Motorola y Toshiba. (Jiménez Buendía, 2009)

Todos los dispositivos presentes en una red Lonworks precisan de un chip Neuron. Dos de los microprocesadores están optimizados para ejecutar el protocolo de comunicaciones, mientras que el tercero está dedicado a ejecutar el programa de control del nodo. Hay por tanto dos procesadores de comunicación y un procesador para la aplicación.

Disponer de dos procesadores dedicados a tareas de comunicación en red y uno dedicado a la aplicación asegura que la complejidad del programa no afecta negativamente a la respuesta de la red y viceversa. Adicionalmente, el hecho de encapsular ambas funciones en un solo chip ahorra tiempos de diseño y producción.

Ventajas Técnicas:

- El uso del chip Neuron garantiza un entorno de ejecución hardware para el protocolo. Para asegurar suficiente potencia de proceso, el protocolo se implementa como una mezcla de hardware y firmware.
- Diseñado para un amplio rango de aplicaciones, y fabricados en masa por dos de los mayores fabricantes de semiconductores del mundo, el chip Neuron ofrece una implementación del protocolo LonTalk más económica que cualquier otra solución propietaria. El resultado neto se traduce en que el chip Neuron es el mejor y más económico procesador Lonworks para cualquier aplicación que precise potencia de proceso de 8 bits.

2.6.2.2 Protocolo de Comunicación LonTalk

LonTalk es un protocolo por niveles orientado al envío de paquetes, está totalmente de acuerdo con la arquitectura de niveles propuesta por la ISO (Internacional Standard Organization). Cada uno de los terminales conectados al canal tiene un turno para trasmisión de paquetes. Cada paquete tiene un largo variable de bytes y contiene la información sobre el nivel de aplicación a la de direccionamiento y a otra información.

El protocolo LonTalk proporciona principalmente dos técnicas para asegurar el correcto envío y recepción de las transmisiones. La fiabilidad de las transmisiones se asegura mediante una confirmación entre emisor y receptor (la mayoría de los protocolos pueden asegurar que un paquete fue transmitido con éxito, pero no que fue recibido por el destinatario). Todos los terminales conectados miran cada uno de los paquetes para ver si corresponde a ellos, si así lo es, procede a ver el paquete y si no, no pasa nada. Este protocolo es independiente del medio utilizado. La integridad de los datos se garantiza por el hecho que todas las transmisiones disponen de un control de errores basado en códigos polinómicos de 16 bits.

El programa de implementación del protocolo LonTalk firmware está dentro de la RPM de Neuron Chip, dando una serie de parámetros de seguridad, rendimiento y calidad. Todas las operaciones en la red de control se realizan usando un sistema de “autentificación de remitente” como una capa de nivel 4 (Nivel de Servicio del modelo OSI). Esta capa proporciona una garantía de autenticidad del remitente, que no puede ser violada por piratas informáticos (“hackers”).

Cada transmisión de paquete proporciona autentificación del remitente. Dado que la implementación de esta característica se encuentra a nivel de chip, por una parte no puede ser modificada y por otra está garantizada en todos los productos, independientemente del fabricante del mismo.

2.6.3 X-10

Entre 1976 y 1978 se desarrolló la tecnología X−10 en Glenrothes, Escocia, por ingenieros de la empresa *Pico Electronics Ltd.*; en la actualidad se distribuye X−10 en los cinco continentes, siendo su principal mercado los Estados Unidos. Durante los últimos 15 años se han vendido más de 150 millones de equipos X−10. Desde que empezó su comercialización en 1978, millones de instalaciones en todo el mundo avalan este sistema técnicamente conocido por *Power Line Carrier* (corrientes portadoras), su funcionamiento se basa en la utilización de la red eléctrica existente en cualquier tipo de edificio, ya sea casa u oficina, como medio físico para la comunicación interna de los distintos componentes del sistema domótico. (Aguirre & Zapata, 2006)

Sus más de 25 años de experiencia, con millares de instalaciones realizadas en España, la multitud de fabricantes que asegura una amplia gama de productos, continuidad de la tecnología y el importante hecho de no tener que realizar obras de infraestructura para cableados especiales, son suficientes motivos para recomendar este sistema domótico domótica destinado para apartamentos, oficinas y locales, tanto de nueva como de antigua construcción.

Pero además, combinando múltiples productos de dilatada y probada experiencia, se puede lograr un sistema domótico de altas prestaciones y baja inversión. Su instalación y configuración es tan sencilla que el propio usuario puede configurar las aplicaciones que desee en cada momento entre una amplio abanico de funciones.

Gracias a la flexibilidad que supone el ser un sistema escalable, resulta todo un interesante y nuevo mundo de bricolaje tanto en seguridad doméstica como en confort, ahorro energético, comunicación e incluso ocio, pudiendo manejar a distancia el DVD, las fotos, vídeos y canciones mp3 almacenadas en nuestro PC para visionarlas en el home cinema de nuestro salón.

La red eléctrica para X−10 sería el equivalente al Bus de otros sistemas como EIB o LonWorks, claro está, salvando las distancias. X−10 es el estándar con mayor implantación en el mercado domótico de corrientes portadoras. La filosofía fundamental de diseño X−10 es que los productos pueden interrelacionarse entre ellos y la compatibilidad con los productos anteriores de la misma gama, es decir, equipos instalados de hace 20 años siguen funcionando con la gama actual.

El sistema X−10 ha sido desarrollado para ser flexible. Se puede empezar con un producto en particular, por ejemplo, un mando a distancia, y expandir luego el sistema para incluir la seguridad o el control con el ordenador, siempre que desee, con componentes fáciles de instalar y no requieren cableados especiales.

X−10 es el lenguaje de comunicación que utilizan los productos compatibles X−10 para hablarse entre ellos y que le permiten controlar las luces y los electrodomésticos de su hogar, aprovechando para ello la instalación eléctrica existente de 220V de su casa, y evitando tener que instalar cables. Los productos de automatización del hogar X−10 están diseñados para que puedan ser instalados fácilmente por cualquier persona sin necesidad de conocimientos especiales.

El sistema X−10 proporciona a los usuarios funcionalidades como:

- Conectar y funcionar (Plug & Play).
- Facilidad de manejo.
- Confort y diversión.

A los instaladores:

- Soluciona problemas economizando proyectos.
- Flexibilidad, modularidad, capacidad de crecimiento.
- Rehabilitación de casas, optimizando recursos con X−10.

2.6.3.1. FUNCIONAMIENTO DE X-10

Los equipos X−10 poseen dos ruedas las cuales son utilizadas para la configuración en la red eléctrica, la primera es de color rojo y representa el código de la casa, está identificada con las letras de la *A* la *P* y la segunda marcada de color negro representa el número del módulo o numérico que corresponde a dicho dispositivo. Cada dispositivo tiene su propia dirección única que el usuario escoge rodando los dos diales en el dispositivo. Si dos actuadores tienen los mismos códigos de casa y numérico, ejecutarán simultáneamente las órdenes procedentes por la red eléctrica. Si a dos detectores de presencia X−10 se les asigna los mismos códigos, cosa que puede resultar útil para encender las luces de escalera desde dos plantas distintas por ejemplo, mandarán la misma orden. Hay 256 combinaciones, así que puede extender su instalación hasta 256 puntos de control X−10. (Jiménez Buendía, 2009)

Lleva sólo un par de segundos hacerlo: el usuario define un nombre para el dispositivo (usará este nombre al dirigir el sistema por voz), le pone el código correspondiente, prueba el dispositivo en tiempo real, y puede asignar el dispositivo a un grupo para que pueda operar un rango entero de dispositivos y luces incluso con una sola orden. Puede agregar nuevos dispositivos o puede renombrar los existentes todas las veces que el usuario quiera, fijar su funcionamiento a lo largo de las horas de un día, una semana, etc.
Las corrientes portadoras funcionan aprovechando la onda que genera la corriente alterna. Las transmisiones de datos se sincronizan en el paso del cero a la corriente continua. De esta forma se genera una serie de códigos formada por el 1 y el 0.

Debido a las características del medio físico utilizado se transmite dos veces cada uno de estos bloques de información para que conseguir reducir las probabilidades de error en la transmisión. Además, cada par de bloques de información debe estar precedido por seis pasos por cero (tres ciclos de red), tiempo de espera es necesario para que el receptor procese los datos de dirección recibidos. En la Fig. 2.8 se muestran los ciclos totales que necesita un transmisor para realizar una transmisión completa.

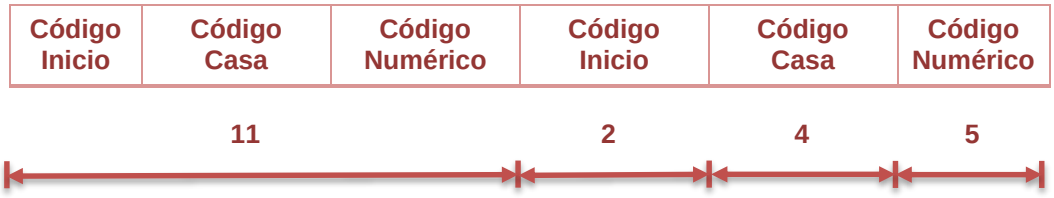


Fig. 2.15 *Ciclos totales para una transmisión completa*

Cada once ciclos de red se transmite un bloque de datos, y una transmisión estándar X-10 normal necesita 47 ciclos de la señal de red. A una frecuencia de 50 Hz esto supone un tiempo igual a 0,94 segundos en transmitir una orden completa.

2.6.3.2. TIPOS DE DISPOSITIVOS X-10

Los sensores de un sistema domótico transmiten órdenes mientras que los actuadores las reciben; por este motivo X-10 hace una clasificación y asigna a sus dispositivos unos logos para identificar su función, son los siguientes:

- Transmisores: Estos transmisores envían una señal especialmente codificada de bajo voltaje que es superpuesta sobre el voltaje del cableado. Un transmisor es capaz de

enviar información hasta 256 dispositivos sobre el cableado eléctrico. Múltiples transmisores pueden enviar señales al mismo módulo.

- Receptores: Como los receptores y transmisores pueden comunicarse con 256 direcciones distintas. Cuando se usan con algunos controladores de computadoras, estos dispositivos pueden reportar su estado.

- Bidireccionales: Estos dispositivos toman la seña enviada por los dispositivos transmisores. Una vez que la señal es recibida el dispositivo responde encendiéndose (ON) o apagándose (OFF). Los receptores generalmente tienen un código establecido por el usuario para indicar la dirección del dispositivo. Múltiples dispositivos con el mismo código pueden co−existir y responder al mismo tiempo dentro de una misma casa. Los dispositivos bidireccionales, tienen la capacidad de responder y confirmar la correcta realización de una orden, lo cual puede ser muy útil cuando el sistema X−10 está conectado a un programa de ordenador que muestre los estados en que se encuentra la instalación domótica de la vivienda. Este es el caso del programador para PC.

- Inalámbricos: Una unidad que permite conectarse a través de una antena y enviar señales de radio desde una unidad inalámbrica e inyectar la seña X−10 en el cableado eléctrico (como los controles remotos para abrir los portones de los garajes). Estas unidades no están habilitadas para controlar directamente a un receptor X−10, debe utilizarse un módulo transceptor.

2.7 ASPECTOS TÉCNICOS DE LOS SENSORES E ILUMINACIÓN UTILIZADOS

A continuación se describen los principios de funcionamiento de los sistemas de control básicos. Además, también se tratarán los aspectos físicos que intervienen en el mecanismo del encendido/apagado de los terminales así como la transferencia de la señal para el funcionamiento de los mismos.

2.7.1 Principios de funcionamiento en sensores de presencia

Este tipo de sensor es capaz de detectar la presencia de un objeto dentro de un radio de acción determinado. Esta detección puede hacerse con o sin contacto con el objeto. En el segundo caso se utilizan diferentes principios físicos para detectar la presencia, dando lugar a diferentes tipos de captadores. (A. Barrientos, 2007)

El sensor de presencia permite regular tanto la sensibilidad al movimiento (distancia de detección) como la intensidad de la luz, el tiempo que ha de permanecer encendida, su desconexión cuando una habitación está vacía y el umbral de iluminación a partir del cual debe activarse. Se pueden instalar tanto en el interior de la casa como en el exterior.

Los detectores de movimiento son muy útiles para iluminar zonas de paso, como corredores o accesos a la vivienda, pero también resultan prácticos en espacios exteriores (zonas residenciales o comerciales). Cada vez que detectan la presencia o ausencia de las personas, automáticamente, encienden o apagan la luz a la que están conectados.

Salir de una habitación sin preocuparse de apagar la luz. Entrar en una estancia a oscuras sin necesidad de buscar el interruptor para que se ilumine. Ambas situaciones son posibles gracias a la instalación de un sensor de movimiento. En espacios interiores o exteriores,

este mecanismo se encarga de encender y apagar automáticamente las luces ante la presencia o ausencia de personas. Son una manera eficaz de gestionar el consumo de energía y conseguir un ahorro en la factura de luz; evitan que una luz permanezca encendida cuando la estancia está vacía

Este sensor de movimiento se instala conectado directamente al sistema de iluminación (lámparas). Poseen un radio de acción que, cuando se invade, automáticamente emite una señal para encender la luz.

- Estas instalaciones ayudan a disuadir a extraños.
- Ahorran energía, ya que evitan que una luz permanezca encendida cuando la estancia está vacía.
- Fácil de instalar y adecuado como protección.
- Es más económico que permanecer con la luz encendida toda la noche.

2.7.2 Principios físicos – Radiación Infrarroja

La tecnología presente en los sensores de presencia es la infrarroja. En este apartado entenderemos el significado de esta radiación, así como su historia, usos y el papel fundamental que desempeña en los sensores de presencia.

La **radiación infrarroja**, o radiación **IR** es un tipo de radiación electromagnética y térmica, de mayor longitud de onda que la luz visible, pero menor que la de las microondas. Consecuentemente, tiene menor frecuencia que la luz visible y mayor que las microondas. Su rango de longitudes de onda va desde unos 0,7 hasta los 1000 micrómetros. La radiación infrarroja es emitida por cualquier cuerpo cuya temperatura sea mayor que 0 Kelvin, es decir, −273,15grados Celsius (cero absoluto).

Los infrarrojos son clasificados, de acuerdo a su longitud de onda, de este modo:

- Infrarrojo cercano (de 800 nm a 2500 nm)
- Infrarrojo medio (de 2.5 µm a 50 µm)
- Infrarrojo lejano (de 50 µm a 1000 µm)

La materia, por su caracterización energética (véase cuerpo negro) emite radiación. En general, la longitud de onda donde un cuerpo emite el máximo de radiación es inversamente proporcional a la temperatura de éste (Ley de Wien). De esta forma la mayoría de los objetos a temperaturas cotidianas tienen su máximo de emisión en el infrarrojo. Los seres, en especial los mamíferos, emiten una gran proporción de radiación en la parte del espectro infrarrojo, debido a su calor corporal.

La potencia emitida en forma de calor por un cuerpo humano, por ejemplo, se puede obtener a partir de la superficie de su piel (unos 2 metros cuadrados) y su temperatura corporal (unos 37 °C, es decir 310 K), por medio de la Ley de Stefan-Boltzmann, y resulta ser de alrededor de 100 vatios.

Esto está íntimamente relacionado con la llamada "sensación térmica", según la cual podemos sentir frío o calor independientemente de la temperatura ambiental, en función de la radiación que recibimos (por ejemplo del Sol u otros cuerpos calientes más cercanos): Si recibimos más de los 100 vatios que emitimos, tendremos calor, y si recibimos menos, tendremos frío. En ambos casos la temperatura de nuestro cuerpo es constante (37 °C) y la del aire que nos rodea también. Por lo tanto, la sensación térmica en aire quieto, sólo tiene que ver con la cantidad de radiación (por lo general infrarroja) que recibimos y su balance con la que emitimos constantemente como cuerpos calientes que somos. Si en cambio hay viento, la capa de aire en contacto con nuestra piel puede ser reemplazada por aire a otra temperatura, lo que también altera el equilibrio térmico y modifica la sensación térmica.

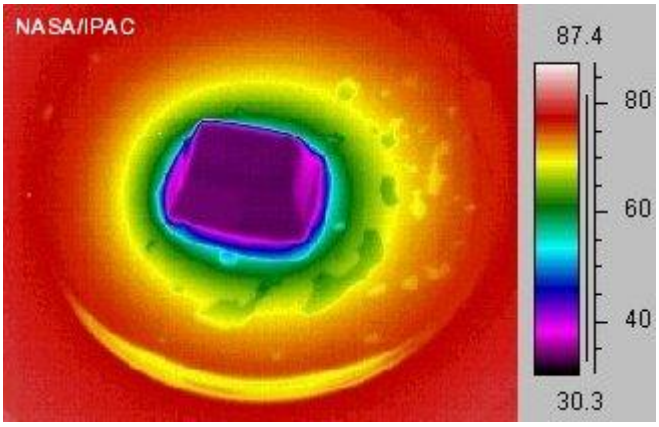


Fig. 2.4 *Imagen de un cubito de hielo derritiéndose bajo luz infrarroja*

2.7.2.1 Historia

Los infrarrojos fueron descubiertos en 1800 por William Herschel un astrónomo inglés de origen alemán. Herschel colocó un termómetro de mercurio en el espectro obtenido por un prisma de cristal con el fin de medir el calor emitido por cada color. Descubrió que el calor era más fuerte al lado del rojo del espectro y observó que allí no había luz. Esta es la primera experiencia que muestra que el calor puede transmitirse por una forma invisible de luz. Herschel denominó a esta radiación "rayos caloricos", denominación bastante popular a lo largo del siglo XIX que, finalmente, fue dando paso al más moderno de radiación infrarroja.

Los primeros detectores de radiación infrarroja eran bolómetros, instrumentos que captan la radiación por el aumento de temperatura producido en un detector absorbente.

2.7.2.2 Usos de los rayos infrarrojos

Los infrarrojos se utilizan en los equipos de visión nocturna cuando la cantidad de luz visible es insuficiente para ver los objetos. La radiación se recibe y después se refleja en una pantalla. Los objetos más calientes se convierten en los más luminosos.

Un uso muy común es el que hacen los mandos a distancia (o telecomandos) que generalmente utilizan los infrarrojos en vez de ondas de radio ya que no interfieren con otras señales como las señales de televisión. Los infrarrojos también se utilizan para comunicar a corta distancia los ordenadores con sus periféricos.

Los aparatos que utilizan este tipo de comunicación cumplen generalmente un estándar publicado por Infrared Data Association.

La luz utilizada en las fibras ópticas es generalmente de infrarrojos.

2.7.2.3 Emisores de infrarrojos industriales

Otra de las muchas aplicaciones de la radiación infrarroja es la del uso de equipos emisores de infrarrojo en el sector industrial. En este sector las aplicaciones ocupan una extensa lista pero se puede destacar su uso en aplicaciones como el secado de pinturas o barnices, secado de papel, termofijación de plásticos, precalentamiento de soldaduras, curvatura, templado y laminado del vidrio, entre otras. La irradiación sobre el material en cuestión puede ser prolongada o momentánea teniendo en cuenta aspectos como la distancia de los emisores al material, la velocidad de paso del material (en el caso de cadenas de producción) y la temperatura que se desee conseguir.

Generalmente, cuando se habla de equipos emisores de infrarrojo, se distinguen cuatro tipos en función de la longitud de onda que utilicen:

1. Emisores de infrarrojo de onda corta.
2. Emisores de infrarrojo de onda media rápida.
3. Emisores de infrarrojo de onda media.
4. Emisores de infrarrojo de onda larga.

2.7.2.4 Sensores Infrarrojos

El sensor infrarrojo es un dispositivo electrónico capaz de medir la radiación electromagnética infrarroja de los cuerpos en su campo de visión. Todos los cuerpos reflejan una cierta cantidad de radiación, ésta, resulta invisible para nuestros ojos pero no para estos aparatos electrónicos, ya que se encuentran en el rango del espectro justo por debajo de la luz visible.

Principio de funcionamiento

Los rayos infrarrojos (IR) entran dentro del fototransistor donde encontramos un material piroeléctrico, natural o artificial, normalmente formando una lámina delgada dentro del nitrato de galio [Ga (NO3)3], nitrato de cesio (CsNO3), derivados de la fenilpirazina, y ftalocianina de cobalto. Normalmente están integrados en diversas configuraciones (1, 2, 4 píxeles de material piroeléctrico). En el caso de parejas se acostumbra a dar polaridades opuestas para trabajar con un amplificador diferencial, provocando la autocancelación de los incrementos de energía de IR y el desacoplamiento del equipo.

- Sensores Pasivos

  Están formados únicamente por el fototransistor con el cometido de medir las radiaciones provenientes de los objetos.

- Sensores Activos
  Se basan en la combinación de un emisor y un receptor próximos entre ellos, normalmente forman parte de un mismo circuito integrado. El emisor es un diodo LED infrarrojo (IRED) y el componente receptor el fototransistor.

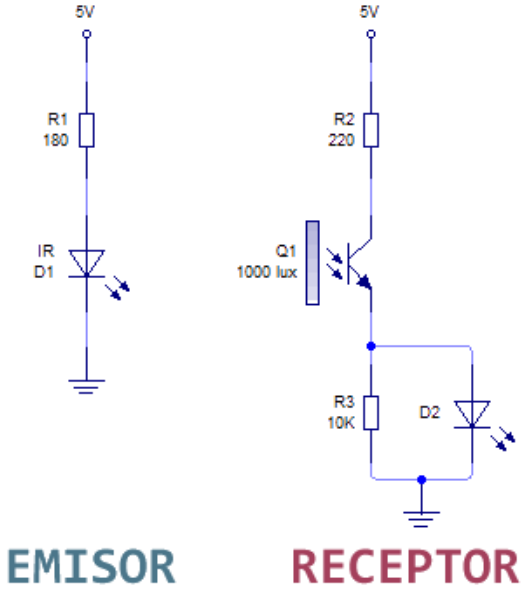


Fig. 2.5 *Esquema sensor infrarrojo de tipo activo*

Calificación según el tipo de señal emitida

1. Sensores reflexivos

Este tipo de sensor presenta una cara frontal en la que encontramos tanto al LED como al fototransistor. Debido a esta configuración el sistema tiene que medir la radiación proveniente del reflejo de la luz emitida por el LED. Se tiene que tener presente que esta configuración es sensible a la luz del ambiente perjudicando las medidas, pueden dar lugar a errores, es necesario la incorporación de circuitos de filtrado en términos de longitud de onda, así pues será importante que trabajen en ambientes de luz controlada. Otro aspecto a tener en cuenta es el coeficiente de reflectividad del objeto, el funcionamiento del sensor será diferente según el tipo de superficie.

2. *Sensores de ranura (Sensor Break-Beam*)

Este tipo de sensor sigue el mismo principio de funcionamiento pero la configuración de los componentes es diferente, ambos elementos se encuentran enfrontados a la misma altura, a banda y banda de una ranura normalmente estrecha, aunque encontramos dispositivos con ranuras más grandes. Este tipo se utiliza típicamente para control industrial. Otra aplicación podría ser el control de las vueltas de un volante.

3. Sensores modulados

Este tipo de sensor infrarrojo sigue el mismo principio que el de reflexión pero utilizando la emisión de una señal modulada, reduciendo mucho la influencia de la iluminación ambiental. Son sensores orientados a la detección de presencia, medición de distancias, detección de obstáculos teniendo una cierta independencia de la iluminación.

4. Sensores de barrido

La diferencia con los anteriores reside en que el sensor realiza el barrido horizontal de la superficie reflectante utilizando señales moduladas para mejorar la independencia de la luz, el color o reflectividad de los objetos. Normalmente estos sistemas forman parte de un dispositivo de desplazamiento perpendicular al eje de exploración del sensor, para poder conseguir las medidas de toda la superficie.

5. Configuración óptica

Esta configuración se basa en un único sensor enfrentado a un cristal, el cual genera la imagen de una sección de la región a medir. Dicho cristal solidario con un motor de rotación con el objetivo de lograr el barrido de toda el área. Tiene la ventaja que adquiere un secuencia continua de la región de barrido. Resulta un sistema lento en términos de exploración.

6. Configuración en array de sensores

En este caso la configuración del sistema de medida está formado por un array de sensores infrarrojos, por tanto no es necesario la utilización de ningún sistema de cristales, únicamente necesita un conjunto de lentes ópticas de enfoque (concentración de la radiación) a cada uno de los sensores. Esta configuración es más compleja pero permite mayor velocidad de translación y mejor protección contra errores de captación.

2.7.3 Principios de funcionamiento en diodos LED

2.7.3.1 La luz. Generalidades.

Se llama luz (del latín lux, lucis) a la parte de la radiación electromagnética que puede ser percibida por el ojo humano. En física, el término luz se usa en un sentido más amplio e incluye todo el campo de la radiación conocido como espectro electromagnético, mientras que la expresión luz visible señala específicamente la radiación en el espectro visible. La luz presenta una naturaleza compleja: depende de cómo la observemos se manifestará como una onda o como una partícula. Estos dos estados no se excluyen, sino que son complementarios. (Díaz Hernández, 2010)

A continuación se exponen los términos más relevantes sobre la luz:

- ***Flujo luminoso.*** Potencia (W) emitida en forma de radiación luminosa a la que el ojo humano es sensible. Su símbolo es Φ y su unidad es el lumen (lm).

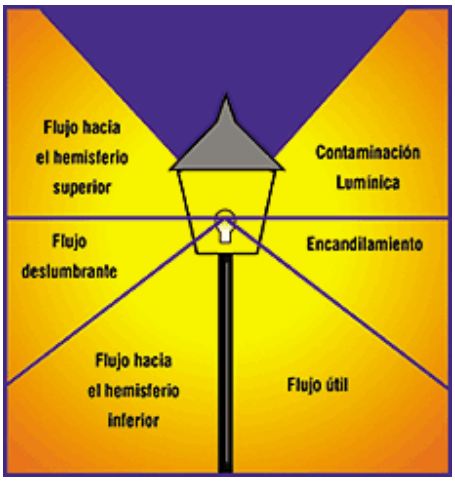


Fig.2.6 *Flujos luminosos procedentes de una lámpara*

- ***Intensidad Luminosa.*** Si pensamos en una lámpara con proyector, no en una bombilla colgada del techo, es fácil discernir que sólo iluminará en una dirección concreta. Por tanto, la Intensidad Luminosa será el Flujo Luminoso emitido por unidad de ángulo sólido (estereorradianes) en una dirección del espacio. Su símbolo es I y su unidad la **candela** (cd).

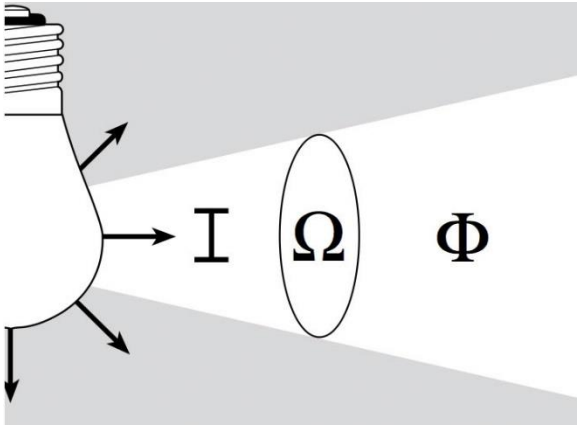


Fig. 2.7 *Intensidad Luminosa*

- ***Iluminancia.*** El flujo luminoso que llega a una superficie es función de la distancia a la que se encuentre ésta. Por tanto, se define iluminancia como el flujo luminoso recibido por una superficie. Su símbolo es E y su unidad el lux (lx) que es un lm/m2. La medida de este parámetro se realiza con el Luxómetro. Dependiendo de la distancia y forma de un objeto, la cantidad de luz que llegue desde un foco luminoso será diferente. Por tanto, se debe separar en sus componentes horizontal y vertical. Más tarde, cuando una persona perciba ese objeto, se deberá tener en cuenta la posición de éste, en lo que se denomina superficie aparente, que será función de la emisión luminosa reflejada por el objeto.

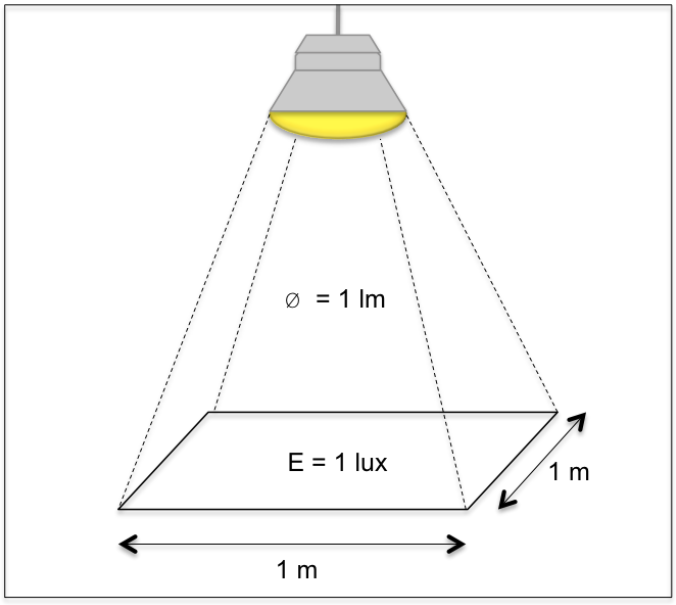


Fig. 2.8 *Iluminancia*

- ***Luminancia.*** Relación entre la intensidad luminosa y la superficie aparente vista por el ojo en una dirección determinada. Su símbolo es L y su unidad es la candela por metro cuadrado (cd/m2).

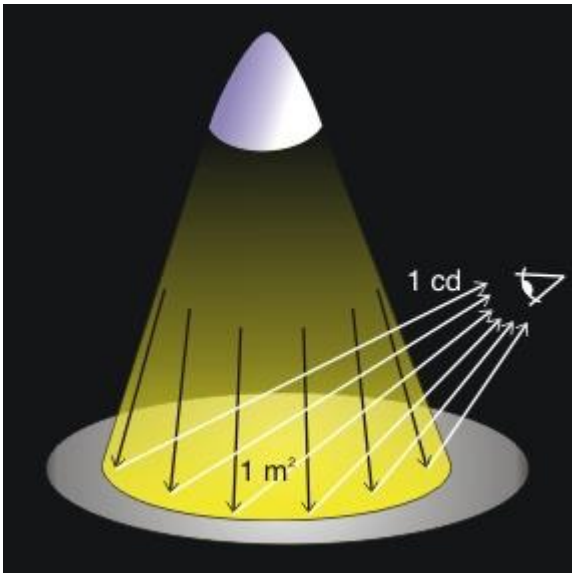


Fig. 2.9 *Luminancia*

2.7.3.2 Diodo LED

Un led (de la sigla inglesa LED: Light-Emitting Diode: 'diodo emisor de luz', también 'diodo luminoso') es un diodo semiconductor que emite luz. Se usan como indicadores en muchos dispositivos, y cada vez con mucha más frecuencia, en iluminación. Presentado como un componente electrónico en 1962, los primeros LEDs emitían luz roja de baja intensidad, pero los dispositivos actuales emiten luz de alto brillo en el espectro infrarrojo, visible y ultravioleta. (Hambley, 2001)

Cuando un LED se encuentra en polarización directa, los electrones pueden recombinarse con los huecos en el dispositivo, liberando energía en forma de fotones. Este efecto es llamado electroluminiscencia y el color de la luz (correspondiente a la energía del fotón) se determina a partir de la banda de energía del semiconductor. Por lo general, el área de un LED es muy pequeña (menor a 1 mm2), y se pueden usar componentes ópticos integrados para formar su patrón de radiación.

Fig.2.10 *Diodos LED*

Los LEDs presentan muchas ventajas sobre las fuentes de luz incandescente y fluorescente, principalmente con un consumo de energía mucho menor, mayor tiempo de vida, tamaño más pequeño, gran durabilidad, resistencia a las vibraciones, no es frágil, reduce considerablemente la emisión de calor que produce el efecto invernadero en nuestro planeta, no contienen mercurio (el cual al exponerse en el medio ambiente es altamente venenoso) a comparación de la tecnología fluorescente o de inducción magnética que si contienen mercurio, no crean campos magnéticos altos como la tecnología de inducción magnética con los cuales se crea mayor radiación hacia el ser humano, cuentan con un alto factor de CRI, reducen ruidos en las líneas eléctricas, son especiales para utilizarse con sistemas foto voltaicos (paneles solares) a comparación de cualquier otra tecnología actual, no les afecta el encendido intermitente (es decir pueden funcionar como luces estroboscópicas) y esto no reduce su vida promedio, son especiales para sistemas anti-explosión ya que no es fácil quebrar un diodo emisor de luz (LED) y cuentan con una alta fiabilidad. Los leds con la potencia suficiente para la iluminación de interiores son relativamente caros y requieren una corriente eléctrica más precisa, por su sistema electrónico para funcionar con voltaje alterno y requieren de disipadores de calor cada vez más eficientes a comparación de las bombillas fluorescentes de potencia equiparable.

2.7.4 Principios Físicos

El funcionamiento normal de un diodo led consiste en que, en los materiales conductores, un electrón al pasar de la banda de conducción a la de valencia, pierde energía; esta energía perdida se manifiesta en forma de un fotón desprendido, con una amplitud, una dirección y una fase aleatoria. El que esa energía perdida cuando pasa un electrón de la banda de conducción a la de valencia se manifieste como un fotón desprendido o como otra forma de energía (calor por ejemplo) depende principalmente del tipo de material semiconductor. Cuando un diodo semiconductor se polariza directamente, los huecos de la zona positiva se mueven hacia la zona negativa y los electrones se mueven de la zona negativa hacia la zona positiva; ambos desplazamientos de cargas constituyen la corriente que circula por el diodo.

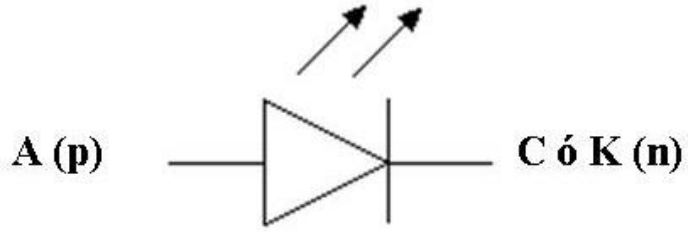


Fig. 2.11 *Representación electrónica del diodo LED*

Si los electrones y huecos están en la misma región, pueden recombinarse, es decir, los electrones pueden pasar a "ocupar" los huecos, "cayendo" desde un nivel energético superior a otro inferior más estable. Este proceso emite con frecuencia un fotón en semiconductores de banda prohibida directa [++* band gap]) con la energía correspondiente a su banda prohibida (véase semiconductor). Esto no quiere decir que en los demás semiconductores (semiconductores de banda prohibida indirecta [indirect bandgap]) no se produzcan emisiones en forma de fotones; sin embargo, estas emisiones son mucho más probables en los semiconductores de banda prohibida directa (como el nitruro de galio) que en los semiconductores de banda prohibida indirecta (como el silicio). (Hambley, 2001)

La emisión espontánea, por tanto, no se produce de forma notable en todos los diodos y solo es visible en diodos como los LEDs de luz visible, que tienen una disposición constructiva especial con el propósito de evitar que la radiación sea reabsorbida por el material circundante, y una energía de la banda prohibida coincidente con la correspondiente al espectro visible. En otros diodos, la energía se libera principalmente en forma de calor, radiación infrarroja o radiación ultravioleta. En el caso de que el diodo libere la energía en forma de radiación ultravioleta, se puede conseguir aprovechar esta radiación para producir radiación visible, mediante sustancias fluorescentes o fosforescentes que absorban la radiación ultravioleta emitida por el diodo y posteriormente emitan luz visible.

El dispositivo semiconductor está comúnmente encapsulado en una cubierta de plástico de mayor resistencia que las de vidrio que usualmente se emplean en las lámparas incandescentes. Aunque el plástico puede estar coloreado, es solo por razones estéticas, ya que ello no influye en el color de la luz emitida. Usualmente un led es una fuente de luz compuesta con diferentes partes, razón por la cual el patrón de intensidad de la luz emitida puede ser bastante complejo.

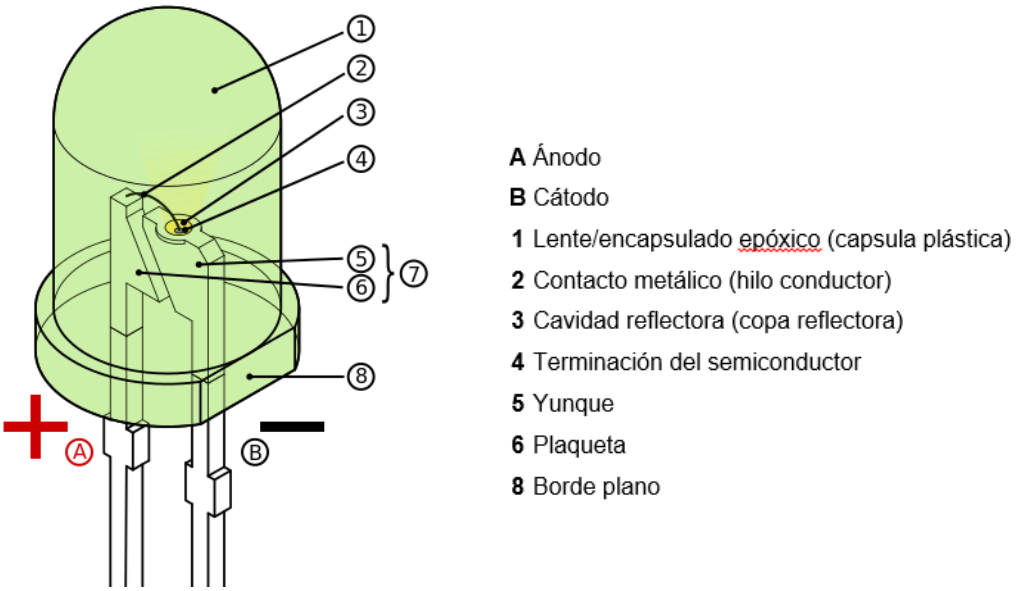


Fig. 2.12 *Partes que componen un LED*

Para obtener buena intensidad luminosa debe escogerse bien la corriente que atraviesa el led; para ello, hay que tener en cuenta que el voltaje de operación va desde 1,8 hasta 3,8 voltios aproximadamente (lo que está relacionado con el material de fabricación y el color de la luz que emite) y la gama de intensidades que debe circular por él varía según su aplicación. Valores típicos de corriente directa de polarización de un led corriente están comprendidos entre los 10 y los 40 mA. En general, los leds suelen tener mejor eficiencia cuanto menor es la corriente que circula por ellos, con lo cual, en su operación de forma optimizada, se suele buscar un compromiso entre la intensidad luminosa que producen (mayor cuanto más grande es la intensidad que circula por ellos) y la eficiencia (mayor cuanto menor es la intensidad que circula por ellos).

CAPÍTULO 3

ARQUITECTURA HARDWARE DE CONTROL

En este capítulo se realiza un análisis profundo del software de control formado por un microcontrolador, que incorpora la plataforma Arduino y el medio inalámbrico usado que da sentido a nuestro proyecto.

3.1 INTRODUCCIÓN

El software que utilizaremos para controlar nuestro sistema domótico de acondicionamiento exterior va a estar centrado en la plataforma Arduino.

Primeramente, se tendrán en cuenta las necesidades del proyecto a realizar para la elección de un microcontrolador adecuado. Se pesaran las alternativas que ofrece el mercado en cuanto al microcontrolador, haciendo un análisis de los distintos controladores existentes y de sus características generales.

Una vez elegido el microcontrolador, nos sumergiremos en el universo Arduino, sopesando, al igual que se hace con el microcontrolador, el modelo de Arduino que más nos conviene. Al seleccionar nuestro modelo, exploraremos sus principales características así como su configuración.

Por último, veremos el control inalámbrico, crucial en el desarrollo de nuestro proyecto, donde seleccionaremos los puntos más importantes del módulo XBee shield y la manera de configurarlo para su correcto funcionamiento.

3.2 MICROCONTROLADOR

Un microcontrolador es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora (Reyes, 2006):

1. Unidad Central de Procesamiento
2. Memoria
3. Periféricos de entrada y salida

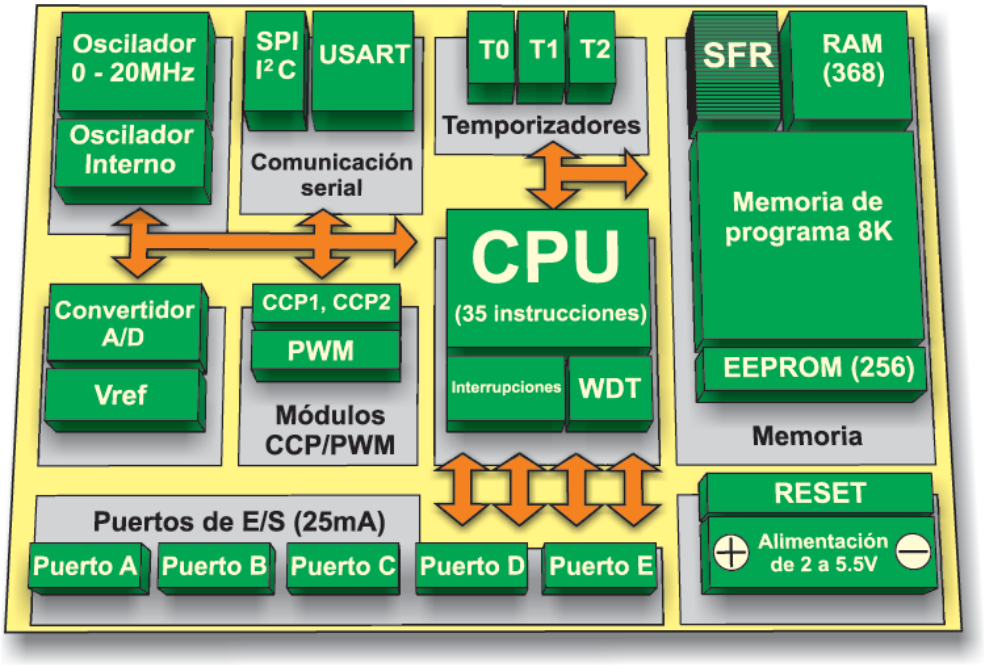


Fig. 3.1 *Arquitectura básica de un microcontrolador*

Algunos microcontroladores pueden utilizar palabras de cuatro bits y funcionan a velocidades de reloj con frecuencias bajas, teniendo un consumo de baja potencia (mW o microvatios). Destaca la capacidad para mantener la funcionalidad a la espera de un evento como o interrupción, con un consumo de energía en modo “sleep” que puede ser se tan solo nanovatios, lo que hace los hace adecuados para aplicaciones con batería de larga duración.

Los microcontroladores están diseñados para reducir el coste económico y el consumo de energía de un sistema en particular. Por eso el tamaño de la unidad central de procesamiento, la cantidad de memoria y los periféricos incluidos dependerán de la aplicación.

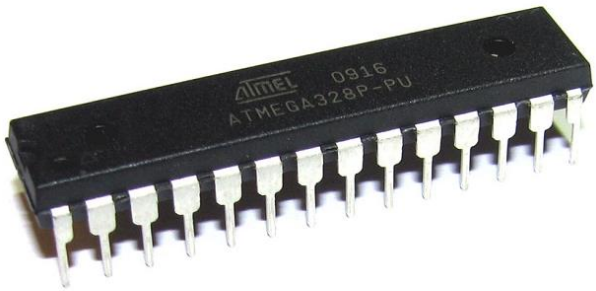


Fig. 3.2 *Microcontrolador ATmega328 encapsulado en un DIP de 28 pines*

De fábrica, la memoria ROM del microcontrolador no posee datos. Para que pueda controlar algún proceso es necesario generar o crear y luego grabar en la memoria EEPROM algún programa, el cual puede ser escrito en lenguaje ensamblador u otro lenguaje para microcontroladores. Para que el programa pueda ser grabado en la memoria requiere de grabador y debe ser codificado en sistema numérico hexadecimal, que es finalmente el sistema que hace trabajar al microcontrolador cuando éste es alimentado con el voltaje adecuado y asociado a dispositivos analógicos y discretos para su funcionamiento.

La principal diferencia frente a una unidad central de procesamiento normal, es que un microcontrolador es más fácil convertirlo en una computadora en funcionamiento, con un mínimo de circuitos integrados externos de apoyo. La idea es que el circuito integrado se coloque en el dispositivo, alimentado por una fuente de energía y de información que necesite.

Un microprocesador tradicional no permite hacer esto, ya que hay que agregarle los módulos de entrada/salida (puertos) y la memoria para almacenamiento de información.

3.2.1 Características Comunes

Al estar todos los microcontroladores integrados en un chip, su estructura fundamental y sus características básicas son similares. Todos deben disponer de los bloques esenciales: procesador (CPU), memoria de datos (RAM) y de instrucciones (ROM), líneas de entrada/salida, oscilador de reloj y diversos módulos para de periféricos. Sin embargo, cada fabricante intenta enfatizar los recursos más idóneos para las aplicaciones a las que se destinan preferentemente.

En esta sección, se hace un recorrido sobre los recursos presentes en todos los microcontroladores, describiendo las diversas alternativas y opciones que pueden encontrarse:

3.2.1.1 Arquitectura básica

Básicamente existen dos arquitecturas de computadoras, y por supuesto, están presentes en el mundo de los microcontroladores: Von Neumann y Harvard. Ambas se diferencian en la forma de conexión de la memoria al procesador y en los buses que cada una necesita.

Aunque inicialmente todos los microcontroladores adoptaron la arquitectura clásica de Von Neumann, en la actualidad es más utilizada la arquitectura Harvard.

Arquitectura Von Neumann

Tradicionalmente los sistemas con microprocesadores se basan en esta arquitectura, en la cual la unidad central de proceso (CPU), está conectada a una memoria principal única (casi siempre sólo RAM) donde se guardan las instrucciones del programa y los datos. A dicha memoria se accede a través de un sistema de buses único (control, direcciones y datos)

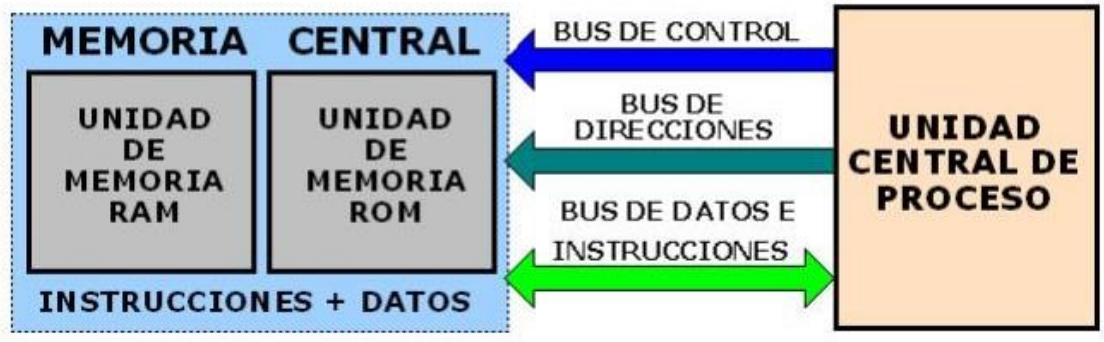


Fig. 3.3 *Estructura de Arquitectura Von Neumann*

En un sistema con **arquitectura Von Neumann** el tamaño de la unidad de datos o instrucciones está fijado por el ancho del bus que comunica la memoria con la CPU. Así un microprocesador de 8 bits con un bus de 8 bits, tendrá que manejar datos e instrucciones de una o más unidades de 8 bits (bytes) de longitud. Si tiene que acceder a una instrucción o dato de más de un byte de longitud, tendrá que realizar más de un acceso a la memoria.

El tener un único bus hace que el microprocesador sea más lento en su respuesta, ya que no puede buscar en memoria una nueva instrucción mientras no finalicen las transferencias de datos de la instrucción anterior.

Las principales limitaciones que nos encontramos con la **arquitectura Von Neumann** son:

- La limitación de la longitud de las instrucciones por el bus de datos, que hace que el microprocesador tenga que realizar varios accesos a memoria para buscar instrucciones complejas.
- La limitación de la velocidad de operación a causa del bus único para datos e instrucciones que no deja acceder simultáneamente a unos y otras, lo cual impide superponer ambos tiempos de acceso

Arquitectura Harvard

Este modelo, que utilizan los Microcontroladores PIC, tiene la unidad central de proceso (CPU) conectada a dos memorias (una con las instrucciones y otra con los datos) por medio de dos buses diferentes.

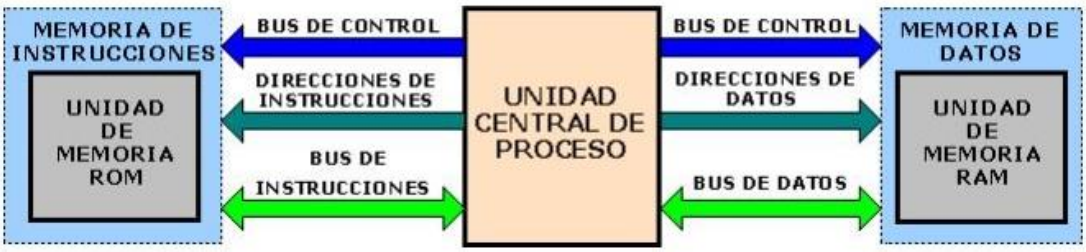


Fig. 3.4 *Estructura de Arquitectura Harvard*

Una de las memorias contiene solamente las instrucciones del programa (Memoria de Programa), y la otra sólo almacena datos (Memoria de Datos).

Ambos buses son totalmente independientes lo que permite que la CPU pueda acceder de forma independiente y simultánea a la memoria de datos y a la de instrucciones. Como los buses son independientes estos pueden tener distintos contenidos en la misma dirección y también distinta longitud.

También la longitud de los datos y las instrucciones puede ser distinta, lo que optimiza el uso de la memoria en general. Para un procesador de **Set de Instrucciones Reducido**, o **RISC (Reduced Instruction Set Computer)**, el set de instrucciones y el bus de memoria de programa pueden diseñarse de tal manera que todas las instrucciones tengan una sola posición de memoria de programa de longitud.

Además, al ser los buses independientes, la CPU puede acceder a los datos para completar la ejecución de una instrucción, y al mismo tiempo leer la siguiente instrucción a ejecutar.

Aunque inicialmente todos los microcontroladores adoptaron la arquitectura clásica de Von Neumann, en la actualidad es más utilizada la arquitectura Harvard.

3.2.1.2 Procesador o CPU

Es el elemento más importante del microcontrolador y determina sus principales características, tanto a nivel hardware como software. Es el encargado de realizar diversas funciones, como direccionar la memoria de instrucciones, recibir el código OP de la instrucción en curso, su decodificación y la ejecución de la operación que implica la instrucción, así como la búsqueda de los operandos y el almacenamiento del resultado.

Existen tres tipos básicos de repertorios de instrucciones que determinan la arquitectura del procesador:

- ***CISC:*** denominado "Complex instruction set computing" o Computadores de Juego de Instrucciones Complejo. Disponen de un conjunto de instrucciones que se caracteriza por ser muy amplio y permiten realizar operaciones complejas entre operando situados en la memoria o en los registros internos. La principal ventaja de los procesadores CISC es que ofrecen al programador instrucciones complejas que actúan como macros.

- ***RISC:*** "Reduced Instruction Set Computer" o Computadores de Juego de Instrucciones Reducido. En estos procesadores el repertorio de instrucciones máquina es muy reducido y las instrucciones son simples y generalmente se ejecutan en un ciclo. Esto permite la optimización del hardware y el software del procesador.

- ***SISC:*** nombrado "Specific Instruction Set Computer" o Computadores de Juego de Instrucciones Específico. En los microcontroladores destinados a aplicaciones muy concretas, el juego de instrucciones, además de ser reducido, es específico, es decir, las instrucciones se adaptan a las necesidades de la aplicación prevista.

3.2.1.3 Memoria

En los microcontroladores la memoria de instrucciones y datos está integrada en el propio chip. Una parte debe ser no volátil, tipo ROM, y se destina a contener el programa de instrucciones que gobierna la aplicación. Otra parte de memoria será tipo RAM, volátil, y se destina a guardar las variables y los datos.

Hay dos propiedades que diferencian a los microcontroladores de los computadores personales:

- No disponen de sistemas de almacenamiento masivo, como disco duros.

- Como el microcontrolador sólo se destina a una tarea en la memoria ROM, sólo hay que almacenar un único programa de trabajo.

La RAM en estos dispositivos es de poca capacidad pues sólo debe contener las variables y los cambios de información que se produzcan en el transcurso del programa. Por otra parte, como sólo existe un programa activo, no se requiere guardar una copia del mismo en la RAM pues se ejecuta directamente desde la ROM.

Los usuarios de computadores personales están habituados a manejar Megabytes de memoria, pero, los diseñadores con microcontroladores trabajan con capacidades de ROM comprendidas entre 512 bytes y 8 k bytes y de RAM comprendidas entre 20 y 512 bytes.

En el caso de la memoria de ROM se utilizan diferentes tecnologías, y el uso de una u otra depende de las características de la aplicación a desarrollar, a continuación se describen las cinco tecnologías existentes, que son más utilizadas:

- ***Máscara ROM.***
  Es una memoria no volátil de sólo lectura cuyo contenido se graba durante la fabricación del chip. El elevado coste del diseño de la máscara sólo hace aconsejable el empleo de los microcontroladores con este tipo de memoria cuando se precisan cantidades superiores a varios miles de unidades.

- ***Memoria PROM.***
  "Programmable Read-Only Memory", también conocida como OTP ("One Time Programmable"). Los microcontroladores con memoria OTP se pueden programar una sola vez, con algún tipo de programador. Se utilizan en sistemas donde el programa no requiera futuras actualizaciones y para series pequeñas, donde la variante de máscara sea muy costosa.

  Tanto en este tipo de memoria como en la EPROM, se suele usar la encriptación mediante fusibles para proteger el código contenido.

- ***Memoria EPROM.***
  Los microcontroladores que disponen de memoria EPROM ("Erasable Programmable Read Only Memory") pueden borrarse y grabarse muchas veces. La grabación se realiza, como en el caso de los OTP, con un grabador gobernado desde un ordenador.

  Si posteriormente se desea borrar el contenido almacenado, disponen de una ventana de cristal en su superficie por la que se somete a la memoria a rayos ultravioleta durante varios minutos. Las cápsulas son de material cerámico y son más caros que los microcontroladores con memoria OTP que están hechos con material plástico.

- ***Memoria EEPROM.***
  Su denominación proviene del término en inglés "Electrical Erasable Programmable Read Only Memory". Este tipo de memorias fueron el sustituto natural de las memorias EPROM, la diferencia fundamental es que pueden ser borradas eléctricamente, por lo que la ventanilla de cristal de cuarzo y los encapsulados cerámicos no son necesarios. Al disminuir los costos de los encapsulados, los microcontroladores con este tipo de memoria son más económicos y cómodos para trabajar que sus equivalentes con memoria EPROM.
  Los microcontroladores dotados de memoria EEPROM una vez instalados en el circuito, pueden grabarse y borrarse cuantas veces se quiera sin ser retirados de dicho circuito. Para ello se usan "grabadores en circuito" que confieren una gran flexibilidad y rapidez a la hora de realizar modificaciones en el programa de trabajo. El número de veces que puede grabarse y borrarse una memoria EEPROM es finito, por lo que no es recomendable una reprogramación continúa. Son muy idóneos para la enseñanza y la ingeniería de diseño.

- ***Memoria Flash.***
  Se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar. Funciona como una ROM y una RAM pero consume menos y tiene un tamaño menor.
  A diferencia de la ROM, la memoria Flash es programable en el circuito. También es más rápida, se recomienda cuando se precisa gran cantidad de memoria de programa no volátil y tolera más ciclos de escritura que la EEPROM.
  Las memorias Flash son muy útiles al permitir que los microcontroladores que las incorporan puedan ser reprogramados "en circuito", es decir, sin tener que sacar el circuito integrado de la tarjeta. Así, un dispositivo con este tipo de memoria incorporado al control del motor de un automóvil permite que pueda modificarse el programa durante la rutina de mantenimiento periódico, compensando los desgastes y otros factores tales como la compresión, la instalación de nuevas piezas, etc. La reprogramación del microcontrolador puede convertirse en una labor rutinaria dentro de la puesta a punto.

3.2.1.4 Puertos de entrada y salida

Los puertos de entrada y salida o puertos E/S, se agrupan generalmente en conjuntos de 8 bits de longitud, que permiten leer datos del exterior o escribir en ellos desde el interior del microcontrolador, el destino habitual es el trabajo con dispositivos simples como relés, leds o motores.

Algunos puertos de también E/S tienen características especiales que le permiten manejar salidas con determinados requerimientos de corriente o incorporar mecanismos especiales de interrupción para el procesador.

Normalmente, cualquier pin de E/S puede ser considerado como E/S de propósito general, compartiendo los pines con otros periféricos.

Para usar un pin con cualquiera de las características a él asignadas se debe configurar mediante los registros destinados a ello.

3.2.1.5 Interrupciones

Las interrupciones son esencialmente llamadas a subrutina generadas por los dispositivos físicos, al contrario de las subrutinas normales de un programa en ejecución. Como el salto de subrutina no es parte del hilo o secuencia de ejecución programada, el controlador guarda el estado del procesador en la pila de memoria y entra a ejecutar un código especial denominado controlador de interrupciones que atiende al periférico específico que generó la interrupción.

Al terminar la rutina, una instrucción especial le indica al procesador el fin de la atención de la interrupción. En ese momento el controlador restablece el estado anterior, y el programa que se estaba ejecutando antes de la interrupción sigue como si nada hubiese pasado. Las rutinas de atención de interrupciones deben ser lo más breves posibles para que el rendimiento del sistema sea satisfactorio, porque normalmente cuando una interrupción es atendida, todas las demás interrupciones están en espera.

3.2.1.6 Oscilador

Todos los microcontroladores disponen de un circuito oscilador que se encarga de generar una onda cuadrada de alta frecuencia, que configura los impulsos de reloj usados en la sincronización de todas las operaciones del sistema.

Generalmente, el circuito de reloj está incorporado en el microcontrolador y sólo se necesitan unos pocos componentes exteriores para seleccionar y estabilizar la frecuencia de trabajo. Estos componentes suelen consistir en un cristal de cuarzo junto a elementos pasivos. También se puede utilizar un resonador cerámico o una red RC.

Aumentar la frecuencia de reloj supone disminuir el tiempo en que se ejecutan las instrucciones pero lleva aparejado un incremento del consumo de energía.

3.2.2 Recursos especiales

Cada fabricante oferta numerosas versiones de la arquitectura básica del microcontrolador. En algunas amplía las capacidades de las memorias, en otras incorpora nuevos recursos, en otras reduce las prestaciones al mínimo para aplicaciones muy simples, etc. La labor del diseñador es encontrar el modelo mínimo que satisfaga todos los requerimientos de su aplicación. De esta forma, minimizará el coste, el hardware y el software.

Los principales recursos específicos que incorporan los microcontroladores son:

- **Temporizadores:** también denominados “timers”, son circuitos síncronos para el conteo de los pulsos y se empleados para tareas como medición de frecuencia e implementación de relojes.
- **Perro guardián:** se denomina también “watchdog”, es un mecanismo de seguridad que provoca un reset del sistema en el caso de que éste se bloquee.
- **Brownout:** protección ante fallo de alimentación.
- **Estado de reposo:** o modo “sleep”, que optimiza el rendimiento a bajo consumo.
- **Conversor analógico – digital:** realiza la conversión de una señal de naturaleza analógica a digital. Las resoluciones más frecuentes son 8 y 10 bits.
- **Comparador analógico:** es un circuito analógico basado en amplificadores operacionales que tiene la característica de comparar dos señales analógicas y dar como salida los niveles lógicos ‘0’ o ‘1’ según el resultado de la comparación.
- **Modulador de ancho de pulsos o PWM:** realiza una técnica de gran utilidad en diferentes periféricos, como en el control de motores.
- **Puertos de comunicación:** existen numerosos tipos de comunicaciones, entre los que destacan el puerto serie, SPI, USB, I2C, Ethernet y Can.

3.2.3 Familias de microcontroladores

Existen numerosas familias de microcontroladores, cada una de las cuales posee un gran número de variantes. En la siguiente tabla, se indicará los más populares y con mayor uso del mercado:

FABRICANTE	8 bits	16 bits	32 bits
Atmel	89SXXXX, ATmega serie 8XX y 4XX		SAM7, SAM3, SAM9
Freescale (Motorola)	68HC05, 68HC08, 68HC11, HCS08	68HC12, 68HCS12, 68HCSX12, 68HC16	ColdFire, PowerPC, 683XX
Intel	Familias 8048 y 8051	MCS96, MXS296	
Microchip	Familias 10f2XX, 12CXX, 12FXX, 16CXX, 16FXX, 18CXX Y 18FXX	PIC24F, PIC24H, PIC30FXX, dsPIC33F	PIC32
NXP Semiconductor (Philips)	80C51	XA	Cortex-M3, Cortex-M0, ARM7, ARM9
Renesas (Hitachi, Mitchubisi y NEC)	78K, H8	H8S, 78K0R, R8C, R32C/M32C/M16C	RX, V850, SuperH, SH-Mobile, H8SX
STMicroelectronics	ST 62, ST7		
Texas Instruments	TMS370, MSP430		C2000, TMS570

Tabla 3.1 *Familias de microcontroladores más utilizadas.*

A nivel individual, destacan los siguientes microcontroladores:

- ✓ **8048 (Intel):** es el padre de los microcontroladores actuales, el primero de todos. Su precio, disponibilidad y herramientas de desarrollo hacen que todavía sea muy popular.
- ✓ **ATmega328 (Atmel):** usado para la plataforma de hardware libre Arduino en sus diferentes versiones. Dispone de buenas prestaciones a precio reducido.
- ✓ **683XX (Freescale):** surgido a partir de la popular familia 68k, a la que se incorporan algunos periféricos. Son microcontroladores de altas prestaciones.
- ✓ **PIC (Microchip):** familia de microcontroladores de gran popularidad. Fueron los primeros microcontroladores RISC.
- ✓ **ATmega2560 (Atmel):** empleado en las versiones más completa de Arduino, el Mega2560. Destaca por su gran cantidad de entradas y salidas, así como de diferentes periféricos.

3.2.4 Aplicaciones y ventajas

Cada vez existen más productos que incorporan un microcontrolador con el fin de aumentar sustancialmente sus prestaciones, reducir el coste y tamaño, mejorar su fiabilidad y disminuir el consumo.

Algunos fabricantes de microcontroladores superan el millón de unidades de un modelo determinado producidas en una semana**.** Este dato puede dar una idea de la masiva utilización de estos componentes.

Los microcontroladores son empleados en multitud de sistemas presentes en la vida cotidiana de las personas, como pueden infinidad de dispositivos electrónicos, como reproductores musicales, juguetes, electrodomésticos, sistema de arranque de vehículos, etc. También se utilizan para otras aplicaciones no tan cotidianas que requieren de gran precisión, como en instrumentación electrónica, el control de sistemas de una nave espacial y tecnología militar, entre otros. Una aplicación típica puede necesitar varios microcontroladores para controlar pequeñas partes del sistema. Estos dispositivos podrían comunicarse entre ellos y con un procesador central, probablemente más potente, para compartir la información y coordinar sus acciones, como ocurre habitualmente en cualquier ordenador personal.

Los productos que para su regulación y funcionamiento incorporan un microcontrolador, disponen de las siguientes ventajas:

- Aumento de prestaciones, debido al mayor control sobre un determinado elemento, esto representa una mejora considerable en el mismo.
- Gran fiabilidad, ya que al remplazar este dispositivo por un elevado número de elementos reduce considerablemente el riesgo de averías y se precisan menos ajustes.
- Disminución del tamaño en el producto acabado, la integración del microcontrolador en un chip disminuye el volumen, la mano de obra y los stocks.
- Mayor flexibilidad, puesto que las características de control están programadas, su modificación sólo necesita cambios en el programa de instrucciones o software.

3.2.5 Justificación de la elección del microcontrolador

Para seleccionar el microcontrolador a utilizar en un diseño concreto, se deben tener en cuenta multitud de factores, como la documentación y herramientas de desarrollo disponibles,

coste económico, fabricantes que lo producen, además de las características propias del dispositivo.

Entre las cualidades intrínsecas de los microcontroladores, es imprescindible tener en cuenta las siguientes:

- ***Procesamiento de datos.***

  En ocasiones, se requiere que el dispositivo pueda realizar numerosos cálculos críticos en el menor tiempo posible. En ese caso, habrá que seleccionar un dispositivo suficientemente rápido para ello y tener en cuenta la precisión de los datos a manejar.
  Si no es suficiente con un microcontrolador de 8 bits, puede ser necesario utilizar uno de 16 o 32 bits. También existe una alternativa más económica, utilizar librerías para manejar los datos de alta precisión.

- ***Entradas y salidas.***

  Para determinar entradas y salidas que necesita el sistema es conveniente dibujar un diagrama de bloques del mismo, de tal forma que sea sencillo identificar la cantidad y tipo de señales a controlar. Una vez realizado este análisis, puede ser necesario añadir periféricos hardware externos o cambiar a otro microcontrolador más adecuado a ese sistema.

- ***Memoria.***

  Las necesidades de memoria de la aplicación a realizar, se debe dividir en memoria volátil (RAM), memoria no volátil (como ROM o EPROM) y memoria no volátil modificable (EEPROM). El último tipo de memoria puede ser útil para incluir información específica de la aplicación, como un número de serie o diferentes parámetros de calibración.

  El tipo de memoria a emplear vendrá determinado por el volumen de ventas previsto del producto: de menor a mayor volumen será conveniente emplear EPROM, OTP y ROM.

  En cuanto a la cantidad de memoria necesaria puede ser imprescindible realizar una versión preliminar, aunque sea en pseudo-código, de la aplicación y a partir de ella hacer una estimación de cuánta memoria volátil y no volátil es necesaria y si es conveniente disponer de memoria no volátil modificable.

- ***Ancho de palabra.***

  El criterio de diseño debe ser seleccionar el microcontrolador de menor ancho de palabra que satisfaga los requerimientos de la aplicación. Usar un microcontrolador de 4 bits supondrá una reducción en los costes importante, mientras que uno de 8 bits puede ser el más adecuado si el ancho de los datos es de un byte. Los dispositivos de 16 y 32 bits, debido a su elevado coste, deben reservarse para aplicaciones que requieran sus altas prestaciones.

- ***Consumo.***

  Algunos productos que incorporan microcontroladores están alimentados con baterías y su funcionamiento puede ser tan vital como activar una alarma antirrobo. Lo más conveniente en un caso como éste puede ser que el microcontrolador esté en estado de bajo consumo o “sleep” pero que despierte ante la activación de una señal (una interrupción) y ejecute el programa adecuado para procesarla.

- ***Costes.***

Como es lógico, los fabricantes de microcontroladores compiten para vender sus productos. Sus ventas no son precisamente bajas, incluso se venden 10 veces más microcontroladores que microprocesadores.

A modo de ejemplo, para el fabricante que usa el microcontrolador en su producto una diferencia de precio en el microcontrolador de algunos céntimos de euro es importante (el consumidor deberá pagar además el coste del empaquetado, el de los otros componentes, el diseño del hardware y el desarrollo del software). Si el fabricante desea reducir costes debe tener en cuenta las herramientas de apoyo con que va a contar: emuladores, simuladores, ensambladores, compiladores, etc. Es habitual que muchos de ellos siempre se decanten por microcontroladores pertenecientes a una única familia.

- ***Diseño de la placa.***

La selección de un microcontrolador concreto condicionará el diseño de la placa de circuitos. Debe tenerse en cuenta que quizá usar un microcontrolador económico, en ocasiones puede encarecer el precio, debido al requerimiento de mayor cantidad componentes del diseño adicionales.

3.3 PLATAFORMA ARDUINO

3.3.1 Arduino como elección

Después de analizar los diferentes microcontroladores disponibles en el mercado basándose en las características de importancia para el prototipo a diseñar, se tomó la decisión final de emplear la plataforma de hardware Arduino, equipada con un microcontrolador de Atmel, para desarrollar el proyecto.

Arduino es una plataforma de hardware de código abierto (“open source”), implementada en una placa de circuito impreso con un microcontrolador y un entorno de desarrollo integrado (IDE), diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares.

El hardware consiste en una placa con un microcontrolador Atmel, de la familia ATmega y numerosos puertos de entrada/salida. El software consiste en un entorno de desarrollo que implementa el lenguaje de programación “Processing/Wiring” derivado de C++ y el gestor de arranque o “bootloader”.

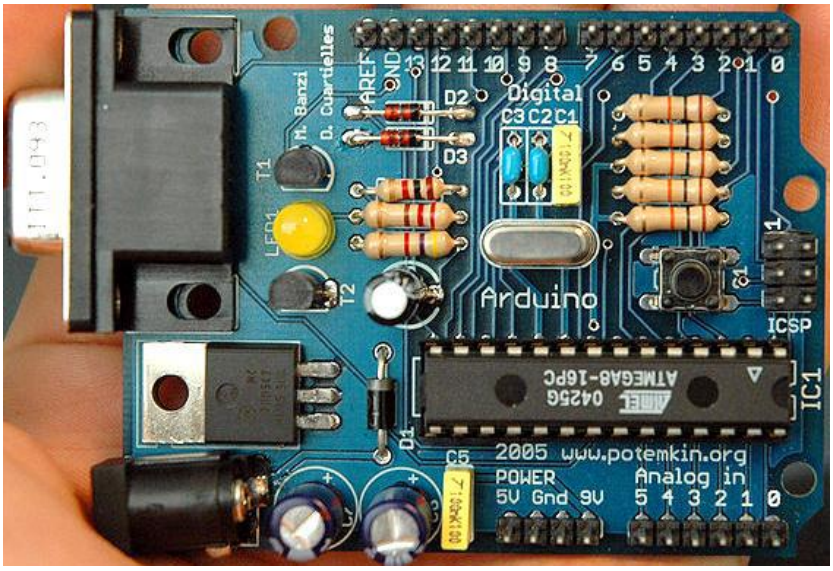


Fig. 3.5 *Primera placa Arduino, llamada Serial*

Arduino se puede utilizar para desarrollar objetos interactivos autónomos o puede ser conectado a diferente software de ordenador, como Macromedia Flash, Max/MSP y Pure Data.

Al ser open-hardware, tanto su diseño como su distribución es libre, es decir, puede utilizarse libremente para el desarrollo de cualquier tipo de proyecto sin haber adquirido ninguna licencia. Las placas se pueden construir de forma propia o adquirirse. El entorno de desarrollo integrado, también libre se puede descargar gratuitamente.

Esta plataforma fue creada por los ingenieros Banzi y Cuartielles, entre otros.

Los microcontroladores disponen de numerosas ventajas y aplicaciones, aunque que no pueden competir con esta completa plataforma, que reúne las características propias de microcontrolador más las que añaden los componentes adicionales que forman la placa.

3.3.2 Modelos de Arduino

Desde el año 2005, cuando apareció el primer prototipo de Arduino como tal, muchos han sido los modelos que se han implementado, aportando cada uno determinadas características que lo diferenciaban del resto. En ocasiones, las nuevas placas servían de versiones mejoradas de las anteriores, por lo que a nivel de fabricación algunos modelos han desaparecido.

En este apartado, se tratarán exclusivamente las plataformas que en la actualidad se tienen más utilidad, debido a que son las versiones más modernas y potentes:

Arduino Nano.

La plataforma Nano es una pequeña y completa placa que implementa el microcontroladores de Atmel, ATmega168, en su revisión 2.X o ATmega328 en su revisión 3.0. Para su correcto uso se recomienda conectar la plataforma a una placa de prototipos. Dispone similares características funcionales que Arduino Uno, pero con una presentación diferente. No tiene conector para alimentación externa y funciona mediante un cable USB Mini-B en vez del cable estándar, tipo B. Nano es diseñado y producido por la empresa estadounidense *Gravitech*.

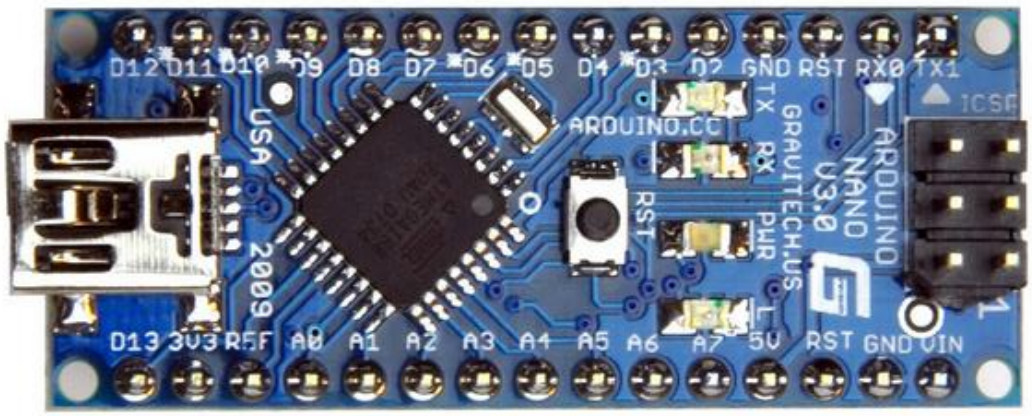


Fig. 3.6 *Vista en planta de Arduino Nano revisión 3.0*

Características de Arduino Nano	
Microcontrolador	Atmel ATmega168 o ATmega328
Tensión de Operación (nivel lógico)	5 V
Tensión de Entrada	7 – 12 V (6 – 20 V límite)
Pines E/S Digitales	14 (6 con salida PWM)
Entradas Analógicas	8
Corriente máx. por cada pin de E/S	40 mA
Memoria Flash	16KB (ATmega168) o 32 KB (ATmega328) de los cuales 2KB son usados por el bootloader
SRAM	1 KB (ATmega168) o 2 BK (ATmega328)
EEPROM	512 bytes (ATmega168) o 1 KB(ATmega328)
Frecuencia de reloj	16 MHz
Dimensiones	18.5mm x 43.2mm

Tabla 3.2 *Especificaciones técnicas de Arduino Nano*

Arduino Leonardo.

La placa Arduino Leonardo es una plataforma que dispone del microcontrolador ATmega32u4, fabricado por la empresa Atmel. Dispone 20 entradas/salidas digitales (7 de las cuales se pueden emplear como salidas PWM), además de 12 entradas analógicas. También implementa un cristal oscilador de 16MHz, una conexión de tipo micro USB, una toma de corriente, un cabezal ICSP y un botón de reset.

Contiene todo lo necesario para utilizar el microcontrolador, sólo es necesario conectarlo al ordenador mediante USB o alimentarlo con un transformador o batería para empezar a trabajar con él.

Leonardo, se diferencia del resto de placas Arduino en el microcontrolador que utiliza, el ATmega32u4 que incorpora comunicación USB, por lo que no requiere de un segundo procesador para esta comunicación. Esto permite que aparezca conectado como un periférico más, junto con un puerto serie virtual (COM).

Fig. 3.7 *Vista frontal (izquierda) y trasera (derecha) de Arduino Leonardo*

Características de Arduino Leonardo	
Microcontrolador	Atmel ATmega32u4
Tensión de Operación (nivel lógico)	5 V
Tensión de Entrada	7 – 12 V (6 – 20 V límite)
Pines E/S Digitales	20 (7 con salida PWM)
Entradas Analógicas	12
Corriente máx. por cada pin de E/S	40 mA
Corriente máx. para pin 3.3 V	50 mA
Memoria Flash	32 KB (ATmega32u4) de los cuales 4 KB son usados por el bootloader
SRAM	2.5 KB (ATmega32u4)
EEPROM	1 KB (ATmega32u4)
Frecuencia de reloj	16 MHz
Dimensiones	68.6mm x 53.3mm

Tabla 3.3 *Especificaciones técnicas de Arduino Leonardo.*

Arduino Mega 2560.

Arduino Mega 2560 es una placa equipada con un microcontrolador fabricado por Atmel, el ATmeg2560. Está formada por 54 entradas/salidas digitales (de las cuales 14 proporcionan salida PWM), 16 entradas digitales, 4 UARTS (puertos serie por hardware), un cristal oscilador de 16MHz, conexión USB, entrada de corriente mediante jack, conector ICSP y botón de reset.

Contiene todo lo necesario para utilizar el microcontrolador, sólo es necesario conectarlo al ordenador mediante USB o alimentarlo con un transformador o batería para empezar a trabajar con él.

Mega 2560, es una versión actualizada que remplaza a Arduino Mega, compatible con la mayoría de “shields” diseñados para Arduino Uno y versiones anteriores de iguales dimensiones.

Fig. 3.8 *Vista frontal (izquierda) y trasera (derecha) de Arduino Mega 2560.*

Características de Arduino Leonardo	
Microcontrolador	Atmel ATmega2560
Tensión de Operación (nivel lógico)	5 V
Tensión de Entrada	7 – 12 V (6 – 20 V límite)
Pines E/S Digitales	54 (15 con salida PWM)
Entradas Analógicas	16
Corriente máx. por cada pin de E/S	40 mA
Corriente máx. para pin 3.3 V	50 mA
Memoria Flash	256 KB (ATmega2560) de los cuales 8 KB son usados por el bootloader
SRAM	8 KB (ATmega2560)
EEPROM	4 KB (ATmega2560)
Frecuencia de reloj	16 MHz
Dimensiones	101.6mm x 53.3mm

Tabla 3.4 *Especificaciones técnicas de Arduino Mega 2560*.

Arduino Uno.

Arduino Uno es una plataforma equipada con el microcontrolador ATmega 328 de Atmel. Está compuesta por 14 entradas/digitales (6 de las cuales se pueden emplear como salidas PWM), además de 6 entradas analógicas. También implementa un cristal oscilador de 16MHz, una conexión de USB tipo B, una toma de corriente, conector ICSP y un botón de reset.

Contiene todo lo necesario para utilizar el microcontrolador, sólo es necesario conectarlo al ordenador mediante USB o alimentarlo con un transformador o batería para empezar a trabajar con él.

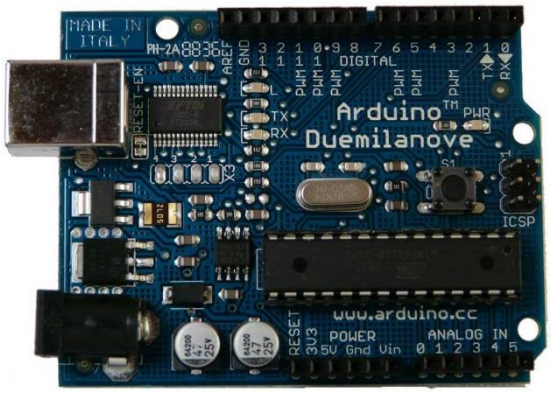
Uno, se diferencia del resto de versiones de Arduino en que difiere de todas las placas anteriores puesto que no utiliza el chip de controlador de FTDI USB a puerto serie. En cambio, cuenta con el Atmega8U2 programado como un convertidor de USB a puerto serie.

Fig. 3.9 *Vista frontal (izquierda) y trasera (derecha) de Arduino UNO (revisión 3.0).*

Características de Arduino UNO	
Microcontrolador	Atmel ATmega328
Tensión de Operación (nivel lógico)	5 V
Tensión de Entrada	7 – 12 V (6 – 20 V límite)
Pines E/S Digitales	14 (6 con salida PWM)
Entradas Analógicas	6
Corriente máx. por cada pin de E/S	40 mA
Corriente máx. para pin 3.3 V	50 mA
Memoria Flash	32 KB (ATmega328) de los cuales 0.5 KB son usados por el bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Frecuencia de reloj	16 MHz
Dimensiones	68.6mm x 53.3mm

Tabla 3.5 *Especificaciones técnicas de Arduino UNO.*

Otros modelos de Arduino

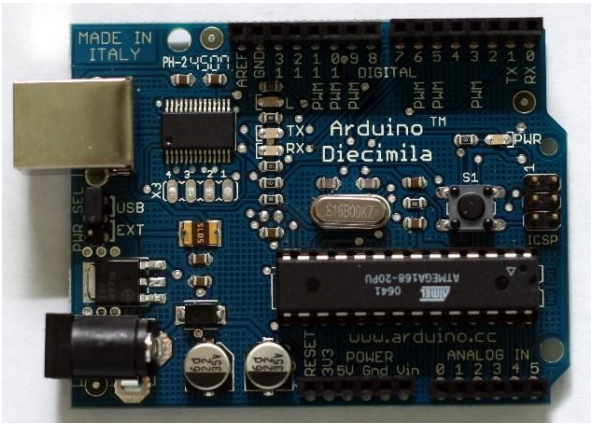


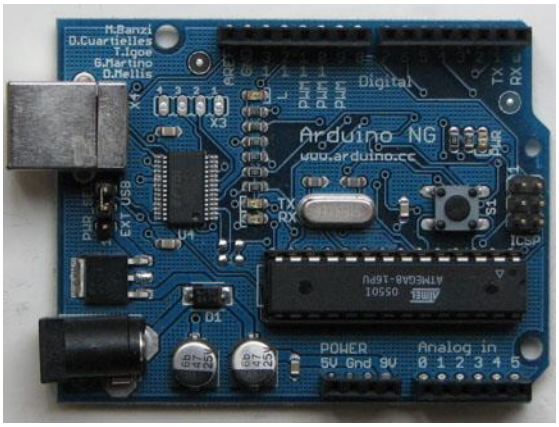
Arduino Duemilanove.

El Duemilanove automáticamente selecciona la fuente de alimentación adecuada (USB o externa), eliminando la necesidad de usar un jumper (especie de conmutador) de selección de fuente como ocurría en placas anteriores. Para que resulte cómodo se puede cortar la pista para deshabilitar el auto-reset y soldar un jumper en el corte para habilitarlo cuando sea necesario. Diseñado originalmente en base al ATmega168, pero a partir de marzo del 2009 empezó a comercializarse con el ATmega328p.

Arduino Diecimila.

La principal diferencia en el Arduino Diecimila es que puede ser reseteado desde el PC, sin necesidad de ser reseteado físicamente usando el botón de reset de la placa. El Diecimila usa un regulador de baja caída de tensión lo cual reduce el consumo de la placa cuando se alimenta con una fuente externa (Adaptador de pared o batería). La placa posee un fusible reseteable que protege el puerto USB de tu PC contra cortocircuitos y sobre tensiones. También posee pines hembra para la línea de reset y de 3.3V.





Arduino NG.

Usa el conversor serie a USB FTDI FT232RL, el cual necesita menos componentes externos que el FT232BM. Este también posee un LED en el pin 13, aunque en su última revisión se eliminó este LED porque podía interferir con la comunicación SPI. Originalmente comercializado con el ATmega8, paso a producirse en base al ATmega168.

3.3.3 Elección de Arduino UNO

Tras el análisis de los diferentes modelos disponibles actualmente de la plataforma Arduino, se realizó una tabulación comparativa de sus características y se analizó que versión podría ser la más conveniente para su utilización en nuestro sistema de control domótico para acondicionamiento exterior.

A continuación, se comparan los parámetros de mayor importancia a la hora de obtener un prototipo eficiente y con viabilidad:

Arduino	*NANO*	*LEONARDO*	*MEGA 2560*	*UNO*
E/S Digitales	14	20	54	14
Entradas Analógicas	8	12	16	6
Microcontrolador	ATmega168 (SMD) o ATmega328 (SMD)	ATmega32u4 (SMD)	ATmega2560 (SMD)	ATmega328
Dimensiones	18.5mm x43.2mm	68.6mm x 53.3mm	101.6mm x 53.3mm	68.6mm x 53.3mm
Acepta Shields	No	Si	Si	Si
Precio (Sin IVA)	33 €	18 €	39 €	20 €

Tabla 3.6 *Comparativa de los diferentes módulos Arduino*

Atendiendo al número de entradas y salidas digitales que ofrecen las alternativas propuestas, no se excluye ningún modelo, ya que disponen del número mínimo o superan las necesarias para la realización del control domótico. En cuanto al número de entradas analógicas, tampoco supone una limitación que elimine a algún candidato.

Al analizar los diferentes microcontroladores implementados, destacan los que disponen de mayores recursos, como el ATmega328 y el ATmega32u4. Puesto que el proyecto será sometido a múltiples experimentos de comportamiento y a su continua depuración para la corrección de errores, las plataformas Arduino equipadas con microcontroladores de montaje superficial o SMD, tienen un punto en contra. Esto se considera un inconveniente, ya que en el caso de sustitución del microcontrolador debido a una avería, en los de tecnología de montaje

superficial habría que sustituir toda la plataforma Arduino, lo que aumentaría los costes. La versión Uno es la única con microcontrolador intercambiable.

Otro factor importante es el precio del material seleccionado, por lo que debe adaptar su coste en función de las características necesarias para el prototipo, adecuándose a sus requisitos.

Por los argumentos antes expuestos, el modelo que más se ajusta a las necesidades es el Uno, la versión más conocida dentro de toda la familia de Arduino.

3.3.4 Características de Arduino UNO

A continuación se exponen las principales características de Arduino UNO, modelo con el que desarrollamos nuestro proyecto.

Arduino UNO es una placa con microcontrolador basada en el ATmega328, Tiene 14 pines con entradas/salidas digitales (6 de las cuales pueden ser usadas como salidas PWM), 6 entradas analógicas, un cristal oscilador a 16Mhz, conexión USB, entrada de alimentación, una cabecera ISCP, y un botón de reset. Contiene todo lo necesario para utilizar el microcontrolador; simplemente conectarse al ordenador a través del cable USB o alimentarlo con un transformador o una batería para empezar a trabajar con él. (Arduino, 2012)

"Uno" significa uno en italiano, y se nombró así para remarcar el lanzamiento próximo de Arduino 1.0, dado que el Arduino UNO y la versión 1.0 están llamados a ser la versión de referencia de Arduino.

El Arduino UNO difiere de todos sus precedentes en que no usa el chip driver FTDI USB-to-serial. En vez de esto viene con un Atmega16U2 programado como convertidor USB-to-serial.

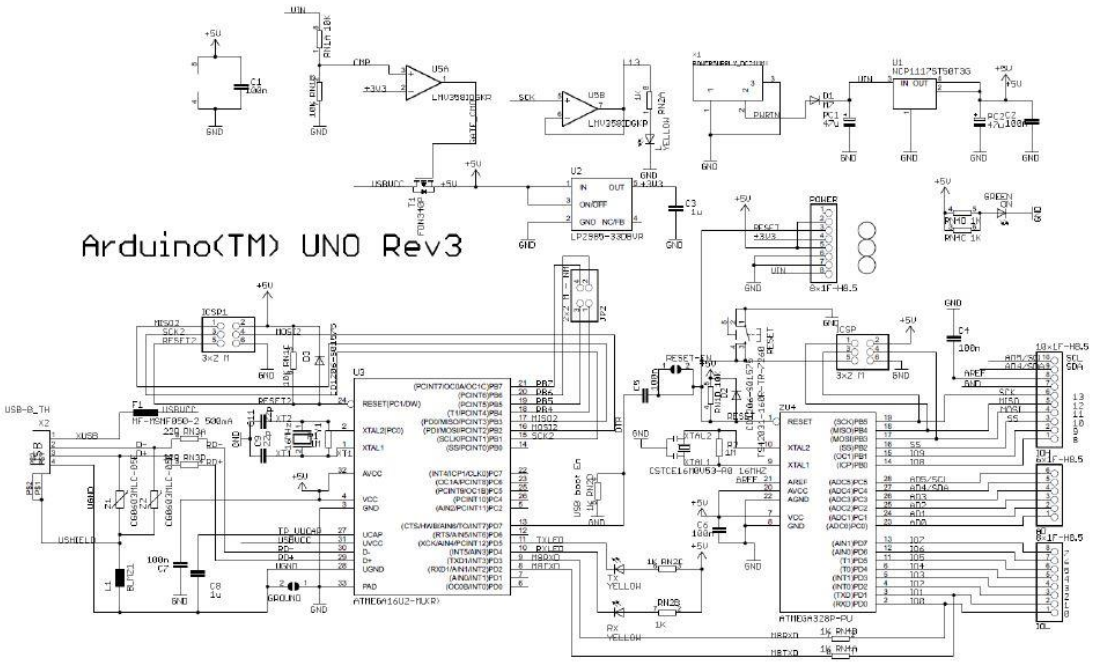


Fig. 3.10 *Esquemático de Arduino UNO (Revisión 3)*

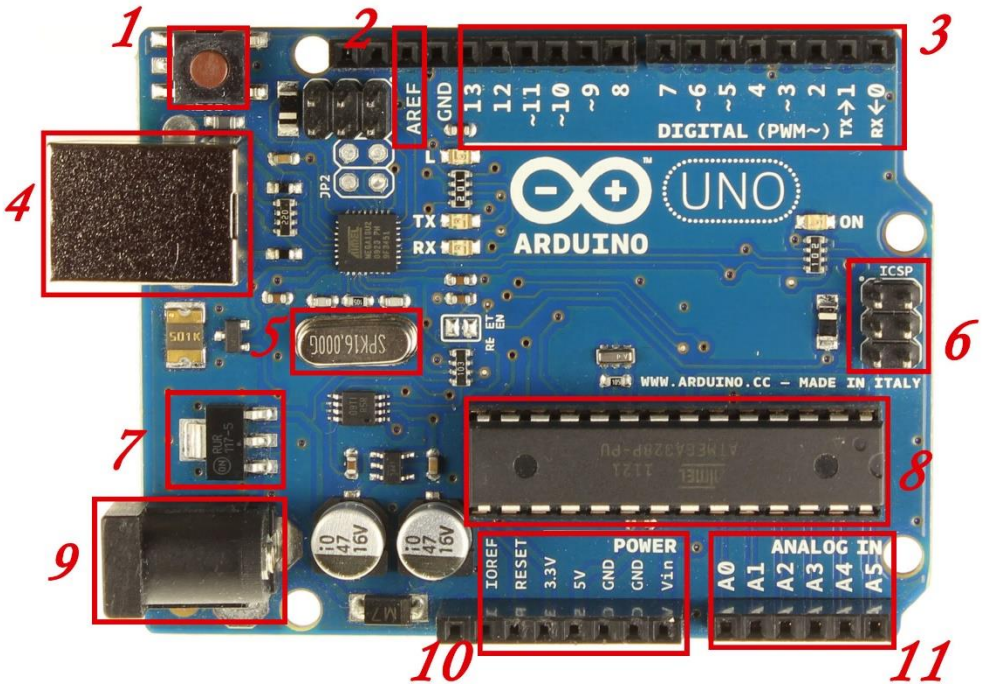


Fig. 3.11 *Componentes de Arduino UNO*

A continuación, señalamos las partes más importantes de la placa Arduino UNO:

1. Botón de Reset
2. AREF
3. Zócalo de entradas/salidas digitales
4. Conector USB
5. Cristal oscilador
6. Conector ICSP
7. Regulador de voltaje
8. Microcontrolador ATMega 328
9. Conector de alimentación
10. Zócalo de alimentación
11. Zócalo de entradas analógicas

3.3.4.1 Alimentación

El Arduino UNO puede ser alimentado vía la conexión USB o con una fuente de alimentación externa. El origen de la alimentación se selecciona automáticamente.

Las fuentes de alimentación externas (no-USB) pueden ser tanto un transformador o una batería. El transformador se puede conectar usando un conector macho de 2.1mm con centro positivo en el conector hembra de la placa. Los cables de la batería pueden conectarse a los pines Gnd y Vin en los conectores de alimentación (POWER).

La placa puede trabajar con una alimentación externa de entre 6 a 20 voltios. Si el voltaje suministrado es inferior a 7V el pin de 5V puede proporcionar menos de 5 Voltios y la placa puede volverse inestable, si se usan más de 12V los reguladores de voltaje se pueden sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios.

Los pines de alimentación, que encontramos en el punto 10 de la figura 3.11, son los siguientes:

- **VIN.** La entrada de voltaje a la placa Arduino cando se está usando una fuente externa de alimentación (en opuesto a los 5 voltios de la conexión USB). Se puede proporcionar voltaje a través de este pin, o, si se está alimentado a través de la conexión de 2.1mm, acceder a ella a través de este pin.

- **5V.** este pin ofrece una salida regulada a 5V por el regulador de la placa independientemente de la fuente y voltaje utilizado para alimentar la placa (7-12v vía conector externo o 5V vía USB). No se recomienda suministrar voltaje a la placa a través de este pin pues puede dañarla.

- **3V3**. Similar al pin 5V, ofrece una salida de 3.3V por el regulador de la placa.

- **GND.** Pines de toma de tierra.

- **Reset.** Suministrar un valor LOW (0V) para reiniciar el microcontrolador. Típicamente usado para añadir un botón de reset a los shields que no dejan acceso a este botón en la placa.

3.3.4.2 Memoria

El ATmega328 tiene 32KB de memoria flash para almacenar código (2KB son usados para el arranque del sistema (bootloader)), 2 KB de memoria SRAM y 1KB de EEPROM (al cual puede ser leída y escrita usando la librería EEPROM). El ATMega 328 está desarrolladlo por la compañía Atmel.

En la siguiente figura se muestra la asignación del patillaje o "pinout" del microcontrolador utilizado:

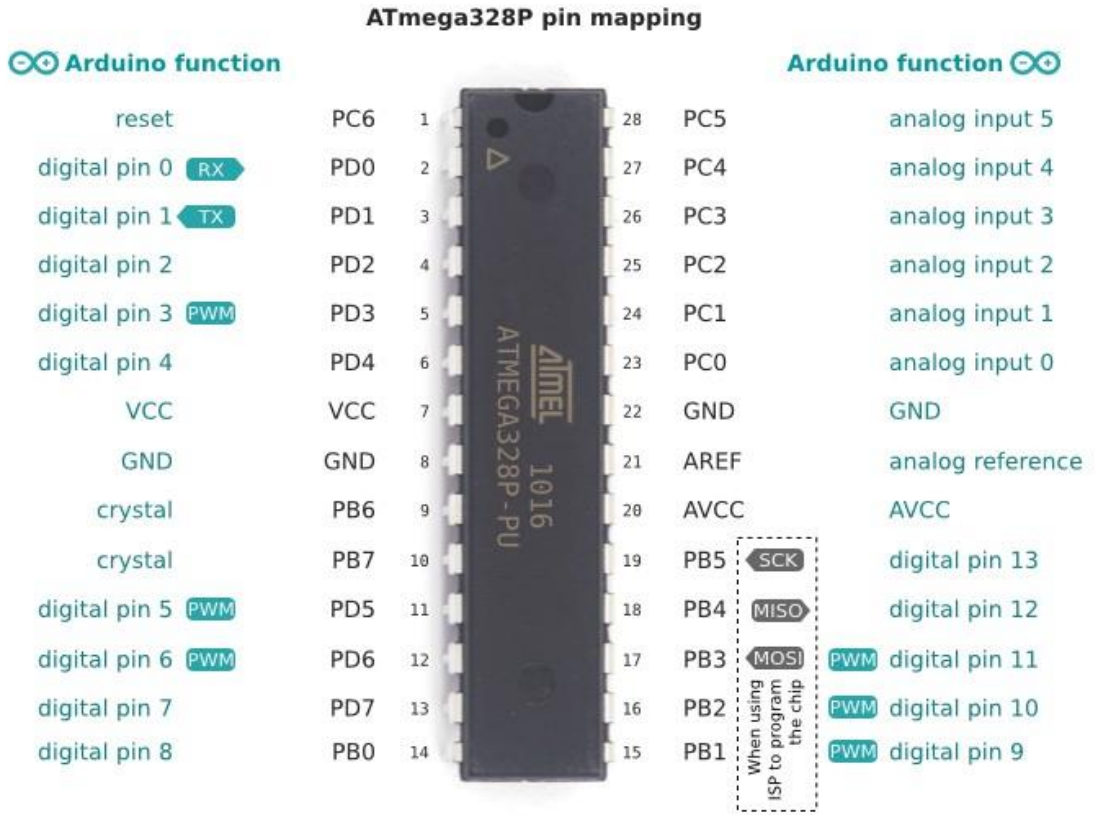


Fig. 3.12 *Pineado del microcontrolador ATMega328*

3.3.4.3 Entradas y salidas

Cada uno de los 14 pines digitales en el UNO pueden utilizarse como entradas o como salidas usando las funciones pinMode(), digitalWrite(), y digitalRead(). Las E/S operan a 5 voltios. Cada pin puede proporcionar o recibir una intensidad máxima de 40mA y tiene una resistencia interna (desconectada por defecto) de 20-50kOhms. El zócalo de pines de entradas/ salidas digitales lo encontramos en el punto 3 de la *figura 4.11*. Además, algunos pines tienen funciones especializadas:

- **Serie**: 0 (RX) y 1 (TX). Usado para recibir (RX) transmitir (TX) datos a través de puerto serie TTL. Estos pines están conectados a los pines correspondientes del chip FTDI USB-to-TTL.

- **Interrupciones Externas:** 2 y 3. Estos pines se pueden configurar para lanzar una interrupción en un valor LOW (0V), en flancos de subida o bajada (cambio de LOW a HIGH (5V), o viceversa), o en cambios de valor.

- **PWM:** 3, 5, 6, 9, 10, y 11. Proporciona una salida PWM de 8 bits de resolución (valores de 0 a 255) a través de la función analogWrite().

- **SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Estos pines proporcionan comunicación SPI usando la librería SPI.

- **LED:** 13. Hay un LED integrado en la placa conectado al pin digital 13, cuando este pin tiene un valor HIGH(5V) el LED se enciende y cuando este tiene un valor LOW(0V) este se apaga.

El UNO tiene 6 entradas analógicas, ubicadas en el punto 11 de la *figura 4.11*, y cada una de ellas proporciona una resolución de 10bits (1024 valores). Por defecto se mide de tierra a 5 voltios, aunque es posible cambiar la cota superior de este rango usando el pin AREF y la función analogReference(). Además algunos pines tienen funciones especializadas:

- **I2C: 4 (SDA) y 5 (SCL).** Soporte del protocolo de comunicaciones I2C (TWI) usando la librería Wire.

Otros pines en la placa:

- **AREF.** Voltaje de referencia para las entradas analógicas. Usado por analogReference(). Podemos ver su ubicación en el punto 2 de la figura 4.11.

3.3.4.4 Comunicaciones

El Arduino UNO facilita en varios aspectos la comunicación con el ordenador, otro Arduino u otros microcontroladores. El ATmega328 proporciona comunicación vía serie UART TTL (5V), disponible a través de los pines digitales 0(RX) y 1(TX). El chip ATmega16U2 en la palca canalice esta comunicación serial a través del Puerto USB y aparece como un Puerto virtual en la computadora. El firmware del 16U2 utiliza drivers estándar para USB, por lo que no es necesario suministrarle los drivers al ordenador, aun así, para Windows se necesita suministrar un archive de extensión inf. El software incluye un monitor de puerto serie que permite enviar y recibir información textual de la placa Arduino. Los LEDS RX y TX de la placa parpadean cuando transmite información vía el chip USB-to-serial y vía USB a la computadora (no parpadearan si se usa la comunicación serie a través de los pines 0 y 1).

La librería SoftwareSerial permite comunicación serie por cualquier par de pines digitales del UNO.

El ATmega328 también soporta la comunicación I2C (TWI) y SPI. El software de Arduino incluye una librería Wire para simplificar el uso del bus I2C. Para el uso de la comunicación SPI se usa la librería SPI.

3.3.4.5 Programación

El Arduino UNO se puede programar a través del software Arduino. El ATmega328 viene pre cargado con un gestor de arranque (bootloader) que permite cargar nuevo código sin necesidad de un programador por hardware externo. Se comunica utilizando el protocolo STK500 original. También es posible sobrepasar el gestor de descarga y programar el microcontrolador a través del cabezal ICSP. (Arduino, 2012)

3.3.4.6 Reinicio Automático (Software)

En vez de necesitar reiniciar presionando físicamente el botón de reset antes de cargar, el Arduino UNO está diseñado de manera que es posible reiniciar por software desde el ordenador donde esté conectado. Una de las líneas de control de flujo (DTR) del ATmega16U2 está conectada a la línea de reinicio del ATmega328 a través de un condensador de 100 nanofaradios. Cuando la línea se pone a LOW (0V), la línea de reinicio también se pone a LOW el tiempo suficiente para reiniciar el chip. El software de Arduino utiliza esta característica para permitir cargar los sketches con solo apretar un botón del entorno. Dado que el gestor de arranque tiene un lapso de tiempo para ello, la activación del DTR y la carga del sketch se coordinan perfectamente.

Esta configuración tiene otras implicaciones. Cuando el UNO se conecta a un ordenador con Mac OS X o Linux, esto reinicia la placa cada vez que se realiza una conexión desde el software (vía USB). El medio segundo aproximadamente posterior, el gestor de arranque se está ejecutando. A pesar de estar programado para ignorar datos mal formateados (ej. cualquier cosa que la carga de un programa nuevo) intercepta los primeros bytes que se envían a la placa justo después de que se abra la conexión. Si un sketch ejecutándose en la placa recibe algún tipo de configuración inicial u otro tipo de información al inicio del programa, es necesario asegurarse que el software con el cual se comunica espera un segundo después de abrir la conexión antes de enviar los datos.

El UNO contiene una pista que puede ser cortada para deshabilitar el auto-reset. Las terminaciones a cada lado pueden ser soldadas entre ellas para rehabilitarlo. Están etiquetadas con "RESET-EN". También podéis deshabilitar el auto-reset conectando una resistencia de 110 ohms desde el pin 5V al pin de reset.

3.3.4.7 Protección contra sobretensiones en USB

El Arduino UNO tiene un multifusible reinicializable que protege la conexión USB de tu ordenador de cortocircuitos y sobretensiones. Aparte que la mayoría de ordenadores proporcionan su propia protección interna, el fusible proporciona una capa extra de protección. Si más de 500mA son detectados en el puerto USB, el fusible automáticamente corta la conexión hasta que el cortocircuito o la sobretensión desaparecen.

3.3.4.8 Características físicas

La longitud y amplitud máxima de la placa Arduino UNO es de 74.8 y 53.3 cm, respectivamente, con el conector USB y la conexión de alimentación sobresaliendo de estas dimensiones. Cuatro agujeros para fijación con tornillos permiten colocar la placa en superficies y cajas.

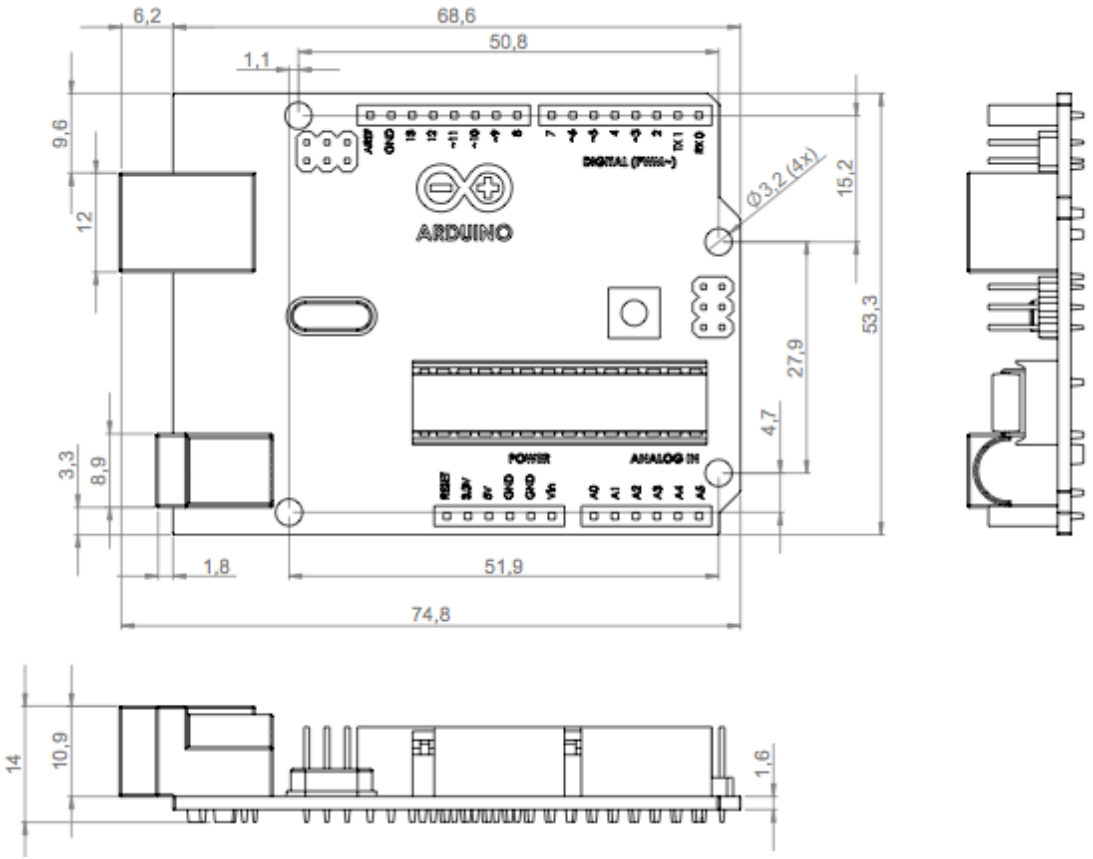


Fig. 3.13 *Vistas con dimensiones en alzado, planta y perfil de Arduino UNO*

3.4 COMUNICACIÓN INALAMBRICA

3.4.1 IEEE 802.15.4

IEEE 802.15.4 es un estándar que define el nivel físico y el control de acceso al medio de redes inalámbricas de área personal con tasas bajas de transmisión de datos (low-rate wireless personal area network, LR-WPAN), es la base sobre la que se define la especificación de ZigBee, cuyo propósito es ofrecer una solución completa para este tipo de redes construyendo los niveles superiores de la pila de protocolos que el estándar no cubre.

El propósito del estándar es definir los niveles de red básicos para dar servicio a un tipo específico de red inalámbrica de área personal (WPAN) centrada en la habilitación de comunicación entre dispositivos ubicuos con bajo coste y velocidad (en contraste con esfuerzos más orientados directamente a los usuarios medios, como WiFi). Se enfatiza el bajo coste de

comunicación con nodos cercanos y sin infraestructura o con muy poca, para favorecer aún más el bajo consumo.

En su forma básica se concibe un área de comunicación de 10 metros con una tasa de transferencia de 250 kbps. Se pueden realizar compromisos que favorezcan aproximaciones más radicales a los sistemas embebidos con requerimientos de consumo aún menores. Para ello se definen no uno, sino varios niveles físicos. Se definieron inicialmente tasas alternativas de 20 y 40 kbps; la versión actual añade una tasa adicional de 100 kbps. Se pueden lograr tasas aún menores con la consiguiente reducción de consumo de energía. Como se ha indicado, la característica fundamental de 802.15.4 entre las WPAN's es la obtención de costes de fabricación excepcionalmente bajos por medio de la sencillez tecnológica, sin perjuicio de la generalidad o la adaptabilidad.

Entre los aspectos más importantes se encuentra la adecuación de su uso para tiempo real por medio de slots de tiempo garantizados, evasión de colisiones por CSMA/CA y soporte integrado a las comunicaciones seguras. También se incluyen funciones de control del consumo de energía como calidad del enlace y detección de energía.

3.4.1.1 Arquitectura de la red

Los dispositivos se relacionan entre sí a través de una red inalámbrica sencilla. La definición de los niveles se basa en el modelo OSI. Aunque los niveles inferiores se definen en el estándar, se prevé la interacción con el resto de niveles, posiblemente por medio de un subnivel de control de enlace lógico basado en IEEE 802.2, que acceda a MAC a través de un subnivel de convergencia. La implementación puede basarse en dispositivos externos o integrarlo todo en dispositivos autónomos. (IEEE-SA Standards Board, 2003)

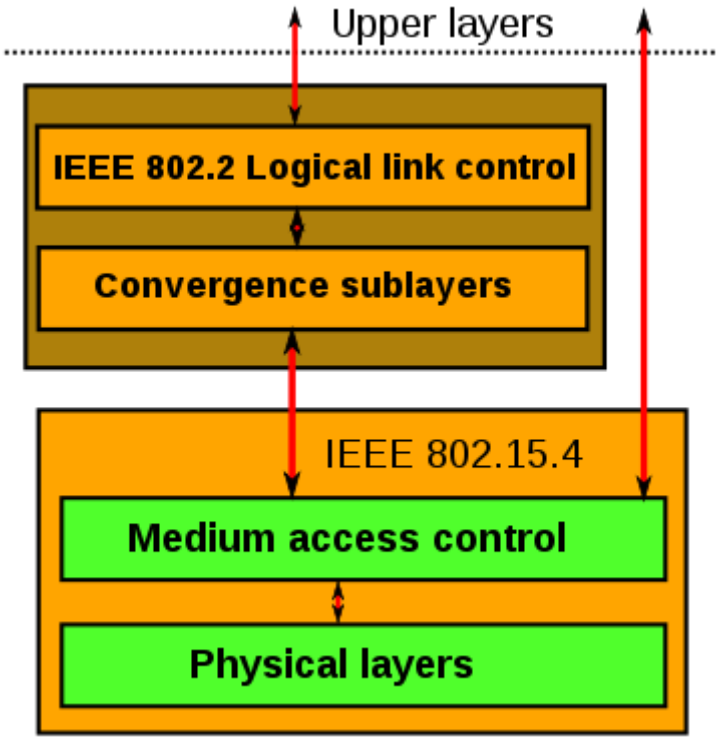


Fig. 3.19 *Arquitectura de red*

El nivel físico (PHY) provee el servicio de transmisión de datos sobre el medio físico propiamente dicho, así como la interfaz con la entidad de gestión del nivel físico, por medio de la cual se puede acceder a todos los servicios de gestión del nivel y que mantiene una base de datos con información de redes de área personal relacionadas. De esta forma, PHY controla el transceptor de radiofrecuencia y realiza la selección de canales junto con el control de consumo y de la señal. Opera en una de tres posibles bandas de frecuencia de uso no regulado:

- *868-868,8 MHz*: Europa, permite un canal de comunicación (versión de 2003), extendido a tres en la revisión de 2006.
- *902-928 MHz:* Norte América, hasta diez canales (2003) extendidos a treinta (2006).
- *2400-2483,5 MHz:* uso en todo el mundo, hasta dieciséis canales (2003, 2006).

La versión original del estándar especifica dos niveles físicos basados en espectro ensanchado por secuencia directa (direct sequence spread spectrum, DSSS): uno en las bandas de 868/915 MHz con tasas de 20 y 40 kbps; y otra en la banda de 2450 MHz con hasta 250 kbps.

La revisión de 2006 incrementa las tasas de datos máximas de las bandas de 868/915 MHz, que permiten hasta 100 y 250 kbps. Aún más, define cuatro niveles físicos en base al método de modulación usado. Tres de ellas preservan el mecanismo por DSSS: las bandas de 868/915 MHz, que usan modulación en fase binaria o por cuadratura en offset (offset quadrature phase shift keying, ésta segunda opcional). En la banda de 2450 MHz se usa esta segunda técnica. Adicionalmente, se define una combinación opcional de modulación binaria y en amplitud para las bandas de menor frecuencia, basadas por lo tanto en una difusión de espectro paralela, no secuencial (PSSS). Si se usan éstas bandas de menor frecuencia, se puede cambiar dinámicamente el nivel físico usado de entre los soportados.

El estándar no define niveles superiores ni subcapas de interoperabilidad. Existen extensiones, como la especificación de ZigBee, que complementan al estándar en la propuesta de soluciones completas.

3.4.1.2 Modelo de red

El estándar define dos tipos de nodo en la red. El primero es el dispositivo de funcionalidad completa (full-function device, FFD). Puede funcionar como coordinador de una red de área personal (PAN) o como un nodo normal. Implementa un modelo general de comunicación que le permite establecer un intercambio con cualquier otro dispositivo. Puede, además, encaminar mensajes, en cuyo caso se le denomina coordinador (coordinador de la PAN si es el responsable de toda la red y no sólo de su entorno). (IEEE-SA Standards Board, 2003)

Contrapuestos a éstos están los dispositivos de funcionalidad reducida (reduced-function device, RFD). Se plantean como dispositivos muy sencillos con recursos y necesidades de comunicación muy limitadas. Por ello, sólo pueden comunicarse con FFD's y nunca pueden ser coordinadores.

Las redes de nodos pueden construirse como redes punto a punto, en estrella, en árbol o en malla. En cualquier caso, toda red necesita al menos un FFD que actúe como su coordinador. Las redes están compuestas por grupos de dispositivos separados por distancias suficientemente reducidas; cada dispositivo posee un identificador único de 64 bits, aunque si se dan ciertas condiciones de entorno en éste pueden utilizarse identificadores cortos de 16 bits. Probablemente éstos se utilizarán dentro del dominio de cada PAN separada.

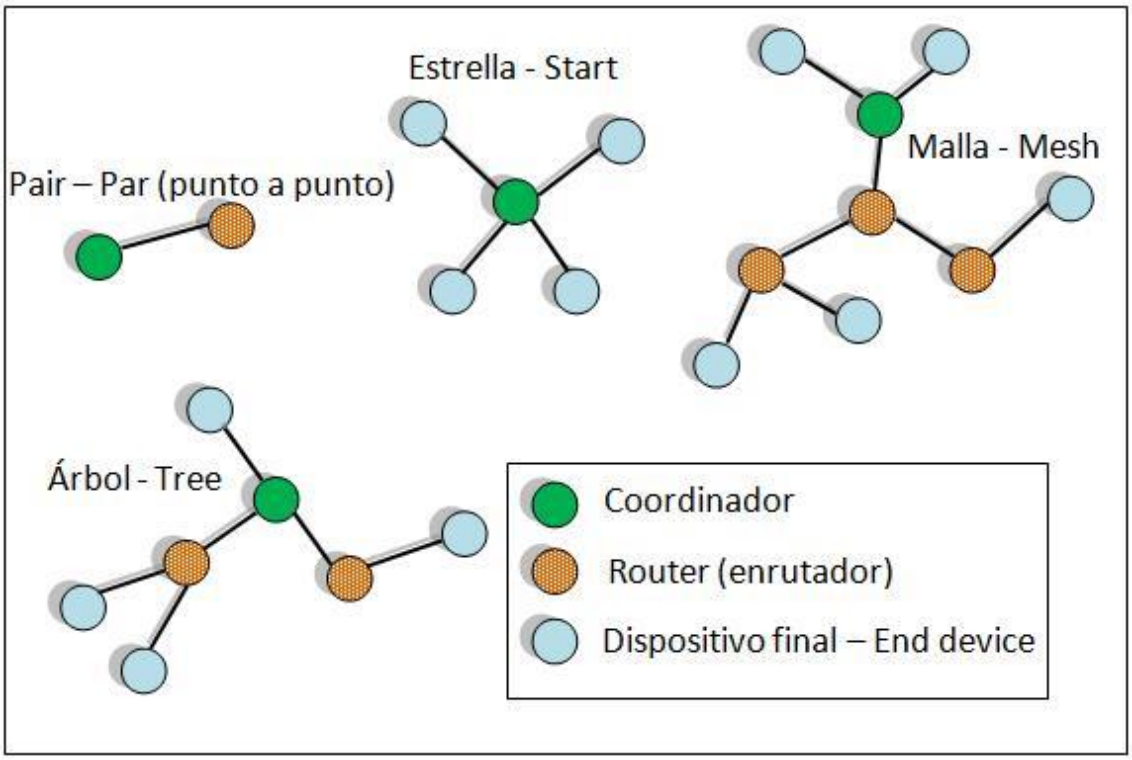


Fig. 3.20 *Estructuras de red en árbol, punto a punto, estrella y malla.*

3.4.1.3 Fiabilidad y seguridad

El medio físico es un recurso al que se accede utilizando CSMA/CA. Las redes que no utilizan métodos balizado hacen uso de una variación del mismo basada en la escucha del medio, balanceada por un algoritmo de backoff exponencial aleatorio, salvo en el caso de las confirmaciones. Las transmisiones de datos típicas utilizan slots no reservados cuando se utilizan balizas; de nuevo, la excepción son las confirmaciones. (IEEE-SA Standards Board, 2003)

El entorno de funcionamiento previsto para este tipo de redes exige que se maximice la vida de la fuente de energía (baterías, posiblemente), por lo que se favorecen los protocolos que conducen a estos fines. Para ello, se programan comprobaciones periódicas de mensajes pendientes, más o menos frecuentes según la aplicación concreta.

En lo que respecta a seguridad en las comunicaciones, el subnivel MAC ofrece funcionalidades que los niveles superiores pueden utilizar para lograr alcanzar el nivel de seguridad deseado. Estos niveles pueden especificar claves simétricas para proteger los datos y restringir éstos a un grupo de dispositivos o a un enlace punto a punto. Estos grupos se especifican en listas de control de acceso. Además, MAC realiza comprobaciones de frescura (freshness check) entre recepciones sucesivas para asegurar que las tramas viejas, cuyo contenido no se considera útil o válido ya, no trascienden a los niveles superiores.

3.4.2 Protocolo ZigBee

ZigBee es el nombre de la especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica para su utilización con radiodifusión digital de bajo consumo. Está basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (wireless personal área network, WPAN). (ZigBee Alliance, 2014)

Su objetivo son las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de sus baterías. ZigBee forma un ecosistema

global de organizaciones que crean soluciones inalámbricas para uso en aplicaciones de gestión energética, residenciales, comerciales y de consumo.

ZigBee utiliza la banda ISM para usos industriales, científicos y médicos; en concreto, 868 MHz en Europa, 915 en Estados Unidos y 2,4 GHz en todo el mundo. Sin embargo, a la hora de diseñar dispositivos, las empresas optarán prácticamente siempre por la banda de 2,4 GHz, por ser libre en todo el mundo. El desarrollo de la tecnología se centra en la sencillez y el bajo costo más que otras redes inalámbricas semejantes de la familia WPAN. (ZigBee Alliance, 2014)

Los protocolos ZigBee están definidos para su uso en aplicaciones encastradas con requerimientos muy bajos de transmisión de datos y consumo energético. Se pretende su uso en aplicaciones de propósito general con características auto organizativas y bajo costo (redes en malla, en concreto). Puede utilizarse para realizar control industrial, albergar sensores empotrados, recolectar datos médicos, ejercer labores de detección de humo o intrusos o domótica. La red en su conjunto utilizará una cantidad muy pequeña de energía de forma que cada dispositivo individual pueda tener una autonomía de hasta 5 años antes de necesitar un recambio en su sistema de alimentación. (Daintree Networks, 2009)

Un nodo ZigBee reduce su consumo gracias a que puede permanecer dormido la mayor parte del tiempo (incluso muchos días seguidos). Cuando se requiere su uso, el nodo ZigBee es capaz de despertar en un tiempo ínfimo, para volverse a dormir cuando deje de ser requerido. Un nodo cualquiera despierta en aproximadamente 15 ms.

3.4.2.1 ZigBee vs. Bluetooth

ZigBee es muy similar al Bluetooth pero con algunas diferencias:

- ✓ Una red ZigBee puede constar de un máximo de 65535 nodos distribuidos en subredes de 255 nodos, frente a los 8 máximos de una subred (Piconet) Bluetooth.

- ✓ Menor consumo eléctrico que el de Bluetooth. En términos exactos, ZigBee tiene un consumo de 30 mA transmitiendo y de 3 uA en reposo, frente a los 40 mA transmitiendo y 0,2 mA en reposo que tiene el Bluetooth. Este menor consumo se debe a que el sistema ZigBee se queda la mayor parte del tiempo dormido, mientras que en una comunicación Bluetooth esto no se puede dar, y siempre se está transmitiendo y/o recibiendo.

- ✓ Tiene una velocidad de hasta 250 kbps, mientras que en Bluetooth es de hasta 3 Mbps.

- ✓ Debido a las velocidades de cada uno, uno es más apropiado que el otro para ciertas cosas. Por ejemplo, mientras que el Bluetooth se usa para aplicaciones como los teléfonos móviles y la informática casera, la velocidad del ZigBee se hace insuficiente para estas tareas, desviándolo a usos tales como la Domótica, los productos dependientes de la batería, los sensores médicos, y en artículos de juguetería, en los cuales la transferencia de datos es menor.

- ✓ Existe una versión que integra el sistema de radiofrecuencias característico de Bluetooth junto a una interfaz de transmisión de datos vía infrarrojos desarrollado por IBM mediante un protocolo ADSI y MDSI.

3.4.2.2 Tipos de dispositivos

Se definen tres tipos distintos de dispositivo ZigBee según su papel en la red (ZigBee Alliance, 2014):

1. **Coordinador ZigBee (ZigBee Coordinator, ZC).** El tipo de dispositivo más completo. Debe existir uno por red. Sus funciones son las de encargarse de controlar la red y los caminos que deben seguir los dispositivos para conectarse entre ellos.

2. **Router ZigBee (ZigBee Router, ZR).** Interconecta dispositivos separados en la topología de la red, además de ofrecer un nivel de aplicación para la ejecución de código de usuario.

3. **Dispositivo final (ZigBee End Device, ZED).** Posee la funcionalidad necesaria para comunicarse con su nodo padre (el coordinador o un router), pero no puede transmitir información destinada a otros dispositivos. De esta forma, este tipo de nodo puede estar dormido la mayor parte del tiempo, aumentando la vida media de sus baterías. Un ZED tiene requerimientos mínimos de memoria y es por tanto significativamente más barato.

Mientras que basándose en su funcionalidad, puede plantearse una segunda clasificación (Daintree Networks, 2009):

1. **Dispositivo de funcionalidad completa** (FFD): También conocidos como nodo activo. Es capaz capacidad de computar, puede funcionar como Coordinador o Router ZigBee, o puede ser usado en dispositivos de red que actúen de interfaz con los usuarios.

2. **Dispositivo de funcionalidad reducida** (RFD): También conocido como nodo pasivo. Tiene capacidad y funcionalidad limitadas (especificada en el estándar) con el objetivo de conseguir un bajo coste y una gran simplicidad. Básicamente, son los sensores/actuadores de la red.

3.4.2.3 Protocolos

Los protocolos de los ZigBee se basan en investigaciones recientes sobre algoritmos de para la construcción de redes ad-hoc de baja velocidad. La mayoría de redes grandes están pensadas para formar un cluster de clusters. También puede estructurarse en forma de malla o como un solo cluster. Los perfiles actuales de los protocolos soportan redes que utilicen o no facilidades de balizado. (Daintree Networks, 2009)

Las redes sin balizas (aquéllas cuyo grado de balizado es 15) acceden al canal por medio de CSMA/CA *(acceso múltiple por detección de portadora con evasión de colisiones)*. Los routers suelen estar activos todo el tiempo, por lo que requieren una alimentación estable en general. Esto, a cambio, permite redes heterogéneas en las que algunos dispositivos pueden estar transmitiendo todo el tiempo, mientras que otros sólo transmiten ante la presencia de estímulos externos. El ejemplo típico es un interruptor inalámbrico: un nodo en la lámpara puede estar recibiendo continuamente ya que está conectado a la red; por el contrario, un interruptor a pilas estaría dormido hasta que el mecanismo se activa. En una red así la lámpara sería un router o coordinador, y el interruptor un dispositivo final.

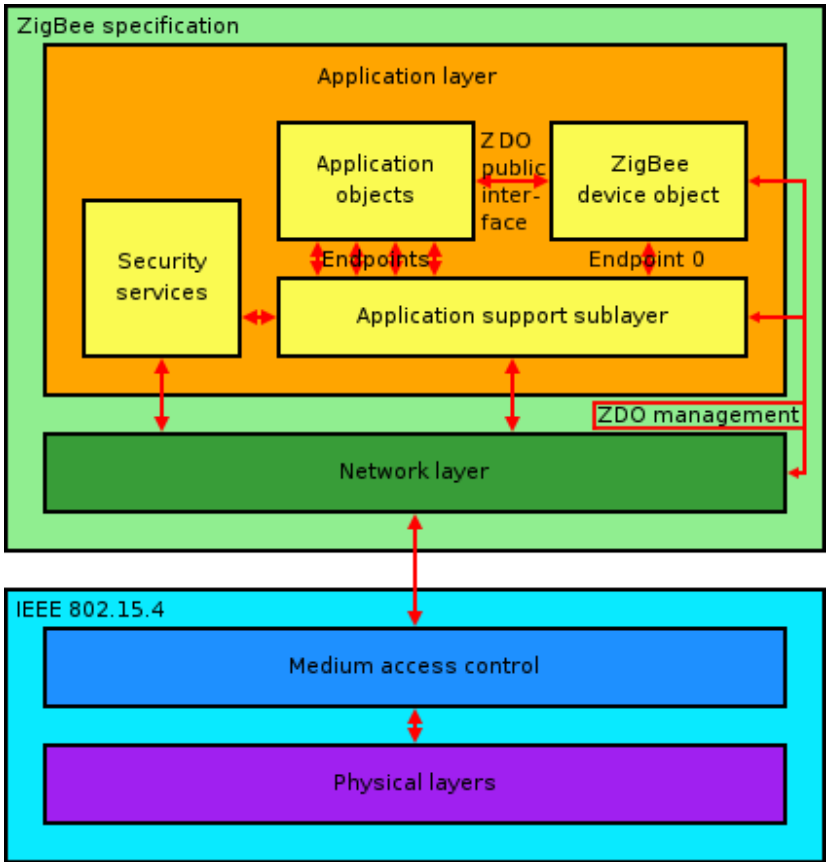


Fig. 3.21 *Estructura del protocolo ZigBee*

Si la red utiliza balizas, los routers las generan periódicamente para confirmar su presencia a otros nodos. Los nodos pueden desactivarse entre las recepciones de balizas reduciendo su ciclo de servicio (duty cycle). Los intervalos de balizado pueden ir desde 15,36 ms a 15,36 ms * 214 = 251,65824 segundos a 250 kbps; de 24 ms a 24 ms * 214 = 393,216 segundos a 40 kbps; y de 48 ms a 48 ms * 214 = 786,432 segundos a 20 kbps. Sin embargo, los periodos largos con ciclos de servicio cortos necesitan que una temporización precisa, lo que puede ir en contra del principio de bajo coste.

En general, los protocolos ZigBee minimizan el tiempo de actividad de la radio para evitar el uso de energía. En las redes con balizas los nodos sólo necesitan estar despiertos mientras se transmiten las balizas (además de cuando se les asigna tiempo para transmitir). Si no hay balizas, el consumo es asimétrico repartido en dispositivos permanentemente activos y otros que sólo no están esporádicamente.

Los dispositivos ZigBee deben respetar el estándar de WPAN de baja tasa de transmisión IEEE 802.15.4-2003. Éste define los niveles más bajos: el nivel físico (PHY) y el control de acceso al medio (MAC, parte del nivel de enlace de datos, DLL). El estándar trabaja sobre las bandas ISM de uso no regulado detalladas más arriba. Se definen hasta 16 canales en el rango de 2,4 GHz, cada uno de ellos con un ancho de banda de 5 MHz. La frecuencia central de cada canal puede calcularse como: FC = (2405 + 5*(k-11)) MHz, con k = 11, 12,...,26.

Las radios utilizan un espectro de dispersión de secuencia directa. Se utiliza BPSK en los dos rangos menores de frecuencia, así como un QPSK (Quadrature Phase-Shift Keying) ortogonal que transmite dos bits por símbolo en la banda de 2,4 GHz. Ésta permite tasas de transmisión en el aire de hasta 250 kbps, mientras que las bandas inferiores se han ampliado con la última revisión a esta tasa desde los 40 kbps de la primera versión. Los rangos de transmisión oscilan entre los 10 y 75 metros, aunque depende bastante del entorno. La potencia de salida de las radios suele ser de 0 dBm (1 mW).

Si bien en general se utiliza CSMA/CA para evitar colisiones en la transmisión, hay algunas excepciones a su uso: por una parte, las tramas siguen una temporización fija que debe ser respetada; por otra, las confirmaciones de envíos tampoco siguen esta disciplina; por último, si se asignan slots de tiempo garantizados para una transmisión tampoco es posible que exista contención.

3.4.2.4 Hardware y software

El software se ha diseñado para ejecutarse en procesadores y microcontroladores de bajo coste, con un diseño de radio muy optimizado para lograr bajos costes con altos volúmenes de producción. Utiliza circuitos digitales siempre que es posible y evita los componentes analógicos. (ZigBee Alliance, 2014)

Si bien el hardware es sencillo, el proceso de certificación de un dispositivo conlleva una validación completa de los requerimientos del nivel físico. Esta revisión intensiva tiene múltiples ventajas, ya que todas las radios fabricadas a partir de una misma máscara de semiconductor gozarán de las mismas características de radiofrecuencia. Por otro lado, un nivel físico mal controlado podría perjudicar no sólo al propio dispositivo, sino al consumo de energía de otros dispositivos en la red. Otros estándares pueden compensar ciertos problemas, mientras que ZigBee trabaja en márgenes muy estrechos de consumo y ancho de banda. Por ello, según el 802.15.4, las radios pasan validaciones ISO 17025(establecen los requisitos que deben cumplir los laboratorios de ensayo y calibración). La mayoría de fabricantes integran la radio y el microcontrolador en un único chip, lo cual permite crear dispositivos más compactos.

3.4.2.5 Conexión de los dispositivos en una red ZigBee

ZigBee permite tres topologías de red (ZigBee Alliance, 2014):

- *Topología en estrella:* el coordinador se sitúa en el centro.
- *Topología en árbol:* el coordinador será la raíz del árbol.
- *Topología de malla*: al menos uno de los nodos tendrá más de dos conexiones.

La topología más interesante (y una de las causas por las que parece que puede triunfar ZigBee) es la topología de malla. Ésta permite que si, en un momento dado, un nodo del camino falla y se cae, pueda seguir la comunicación entre todos los demás nodos debido a que se rehacen todos los caminos. La gestión de los caminos es tarea del coordinador.

Por otro lado, las redes ZigBee han sido diseñadas para conservar la potencia en los nodos esclavos. De esta forma se consigue el bajo consumo de potencia. La estrategia consiste en que, durante mucho tiempo, un dispositivo "esclavo" está en modo "dormido", de tal forma que solo se "despierta" por una fracción de segundo para confirmar que está "vivo" en la red de dispositivos de la que forma parte. Esta transición del modo "dormido" al modo "despierto" (modo en el que realmente transmite), dura unos 15ms, y la enumeración de "esclavos" dura alrededor de 30ms, como ya se ha comentado anteriormente.

En las redes Zigbee, se pueden usar dos tipos de entornos o sistemas (Daintree Networks, 2009):

- **Con balizas.** Es un mecanismo de control del consumo de potencia en la red. Permite a todos los dispositivos saber cuándo pueden transmitir. En este modelo, los dos caminos de la red tienen un distribuidor que se encarga de controlar el canal y dirigir las transmisiones. Las balizas que dan nombre a este tipo de entorno, se usan para poder sincronizar todos los dispositivos que conforman la red, identificando la red domótica, y describiendo la estructura de la "supertrama". Los intervalos de las balizas son asignados por el coordinador de red y pueden variar desde los 15ms hasta los 4 minutos.

Este modo es más recomendable cuando el coordinador de red trabaja con una batería. Los dispositivos que conforman la red, escuchan a dicho coordinador durante el "balizamiento" (envío de mensajes a todos los dispositivos -broadcast-, entre 0,015 y 252 segundos). Un dispositivo que quiera intervenir, lo primero que tendrá que hacer es registrarse para el coordinador, y es entonces cuando mira si hay mensajes para él. En el caso de que no haya mensajes, este dispositivo vuelve a "dormir", y se despierta de acuerdo a un horario que ha establecido previamente el coordinador. En cuanto el coordinador termina el "balizamiento", vuelve a "dormirse".

- **Sin balizas.** Se usa el acceso múltiple al sistema Zigbee en una red punto a punto cercano. En este tipo, cada dispositivo es autónomo, pudiendo iniciar una conversación, en la cual los otros pueden interferir. A veces, puede ocurrir que el dispositivo destino puede no oír la petición, o que el canal esté ocupado.

  Este sistema se usa típicamente en los sistemas de seguridad, en los cuales sus dispositivos (sensores, detectores de movimiento o de rotura de cristales), duermen prácticamente todo el tiempo (el 99,999%). Para que se les tenga en cuenta, estos elementos se "despiertan" de forma regular para anunciar que siguen en la red. Cuando se produce un evento (en nuestro sistema será cuando se detecta algo), el sensor "despierta" instantáneamente y transmite la alarma correspondiente. Es en ese momento cuando el coordinador de red, recibe el mensaje enviado por el sensor, y activa la alarma correspondiente. En este caso, el coordinador de red se alimenta de la red principal durante todo el tiempo.

3.4.3 XBee

XBee es el nombre comercial de Digi Internacional para una familia de módulos de radio con formato de forma compatibles. Los primeros módulos XBee se introdujeron al mercado bajo la marca MaxStream en 2005, estaban basados en el estándar 802.15.4-2003 diseñado para comunicación inalámbrica punto a punto y punto a multipunto y con una velocidad de transmisión de 250 kbit/s. Dos modelos fueron presentados inicialmente; el primero ofrecía un menor costo y potencia de 1 mW, y el segundo ofrecía una mayor potencia (100 mW), el XBee-PRO. Desde entonces, diversos diferentes modelos de estos módulos se ha producidos y se comercializan actualmente bajo la marca “Digi”. (Digi International Inc., 2013)

A continuación los diversos radios XBee en el mercado (Digi International Inc., 2013):

- XBee 802.15.4 (también conocido como Serie 1) – funciona punto-a-punto (PTP), punto-a-multipunto (PTM) y corre bajo el protocolo IEEE 802.15.4. Este es el modelo que utilizaremos en el presente proyecto, y por tanto sobre el cual abundaremos.
- XBee-PRO 802.15.4 (Series 1) – versión más potente de la anterior
- XBee ZB (conocido como Serie 2) – es un módulo XBee que incorpora el protocolo de red en malla de ZigBee PRO.
- XBee-PRO ZB (Serie 2) – versión más potente de la anterior
- XBee ZB SMT – Un montaje en superficie XBee que ejecute el protocolo ZigBee
- XBee-PRO ZB SMT – versión más potente de la anterior
- XBee SE – An XBee ZB módulo que incorpora el grupo de seguridad para el perfil de “ZigBee Smart Energy public”.
- XBee PRO SE – versión más potente de la anterior
- XBee PRO 900 – una versión a 900 MHz que funciona PTP y PTM
- XBee PRO 868 – una versión a 868 MHz que funciona PTP y PTM para Europa
- XBee-PRO DigiMesh 900 – un radio a 900 MHz que incorpora un protocolo de dormida en red malla.
- XBee DigiMesh 2.4 – igual al anterior, pero funcionando a 2.4 GHz
- XBee-PRO DigiMesh 2.4 – versión más potente de la anterior

La mayoría de los XBees vienen con varias opciones de antena, aunque no todas las variantes tienen exactamente los mismos conectores. El conector U.F.L (un conector coaxial

miniatura de RF para señales de alta frecuencia de hasta 6 GHz y manufacturadas por Hirose Electric Group en Japón) es común en toda la familia XBee, y todas las variantes vienen o con un chip antena, o con una antena empotrada en el PCB. Otras antenas incluyen una tira ¼ de onda integrada y un conector RP-SMA para antenas de cabezas grandes. (Digi International Inc., 2013)

3.4.3.1 Características principales de XBee Serie 1

Los módulos XBee, en este caso el Serie 1, han sido diseñados para cumplir con los requerimientos de bajo costo y bajo consumo de las redes de sensores. Por lo tanto, estos módulos requieren un consumo muy bajo y proveen una fiable transmisión de datos a una velocidad de hasta 250 kbps.

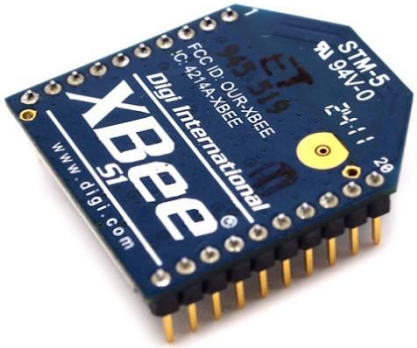


Fig. 3.22 *XBee series 1*

A continuación las principales características de estos módulos:

Características Generales XBee series 1	
Rango en interior o entorno urbano	Hasta 30 m
Exterior/ vista directa entre módulos	Hasta 90 m
Potencia de transmisión	1 mW
Velocidad por Radio-frecuencia	250 kbps
Velocidad por puerto serie	1200 bps – 250 kbps
Sensibilidad de recepción	-92 dBm
Tensión de alimentación	2,8 – 3,4 VDC
Corriente de transmisión	45 mA
Corriente en recepción	50 mA
Corriente en inactividad	< 10 uA
Dimensiones	2,438 cm x 2,761 cm
Temperatura de operación	-40 to 85 º C

Tabla 3.7 *Características principales de XBee series 1*

Contrario a las especificaciones anteriores, el módulo XBee PRO ofrece un alcance de hasta 90 metros en interiores y hasta 1.5 kilómetros (pueden ser hasta 10 kilómetros con una

antena expandida) en exterior, pero a cambio de un mayor consumo de potencia (63 mW). Sin embargo el rango de alcance que brinda el XBee serie 1 es suficiente para aplicaciones demóticas que es el enfoque de nuestro proyecto.

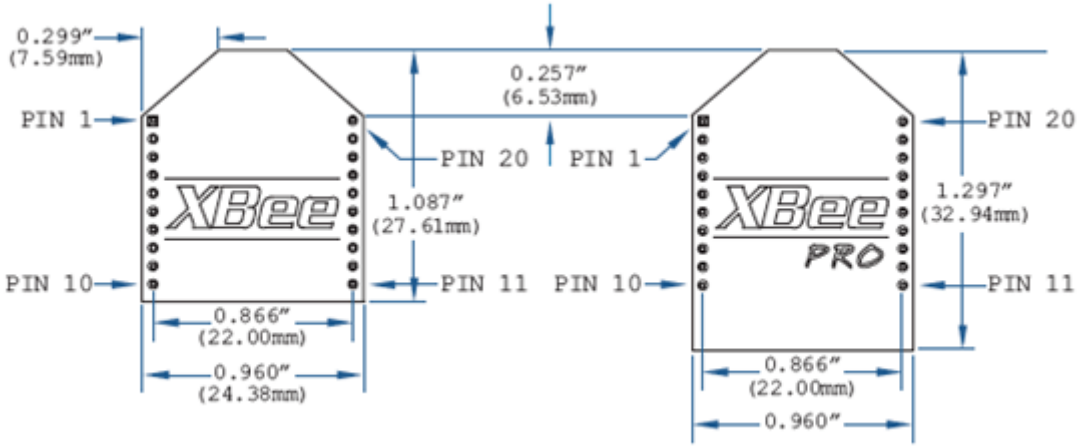


Fig. 3.22 *Dimensiones físicas*

Características de Red	
Topologías de red soportadas	Malla, punto a punto, punto a multipunto
Número de canales	16 canales en secuencia directa
Encriptación	128 bits AES
Opciones de direccionamiento	PAN ID, canales y direcciones de 64 bits

Tabla 3.8 *Características de red*

3.4.3.2 XBee Shield

La Xbee shield permite a una placa Arduino comunicarse de forma inalámbrica usando Zigbee. Está basada en el módulo Xbee de MaxStream. El módulo puede comunicarse hasta 100ft (30 metros) en interior o 300ft (90 metros) al aire libre (en visión directa). Puede ser usado como reemplazo del puerto serie/USB o puedes ponerlo en modo de comandos y configurarlo para una variedad de opciones de redes broadcast o malladas. La shield tiene pistas desde cada pin del Xbee hasta un orificio de soldar. También provee conectores hembra para usar los pines digitales desde 2 hasta 7 y las entradas analógicas, las cuales están cubiertas por la shield (los pines digitales de 8 a 13 no están cubiertos por la placa, así que puedes usar los conectores de la placa directamente).

La Xbee shield fue creada en colaboración con Libelium, quienes la desarrollaron para usarlo en sus SquidBee motes(usados para crear redes de sensores).

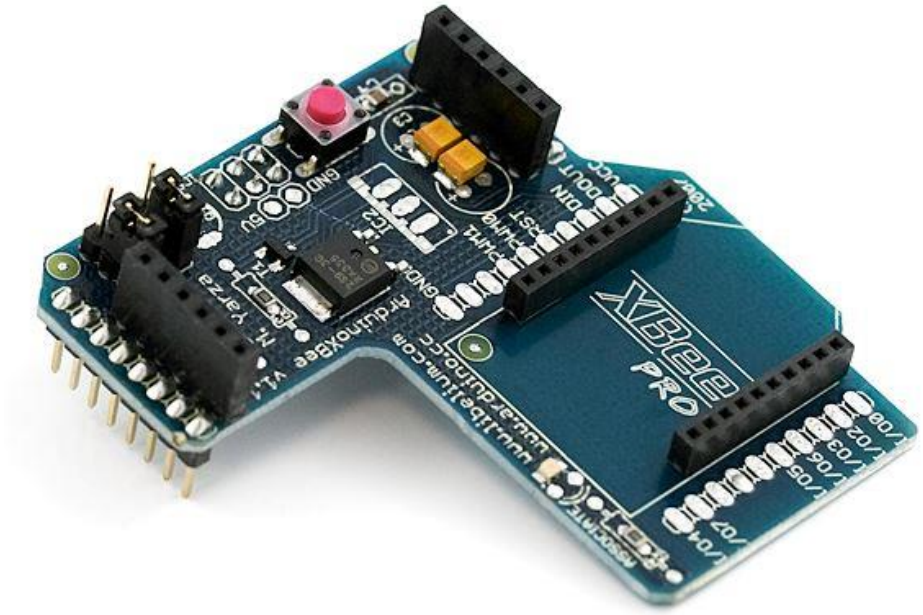


Fig. 3.23 *XBee Shield*

Configuración de los jumpers

La Xbee shield tiene dos jumpers (las pequeñas fundas de plásticos que están sobre los tres pines etiquetados como XBEE/USB). Estos determinan como se conecta la comunicación serie del Xbee entre el microcontrolador (ATmega168) y el chip serie FTDI de la placa Arduino.

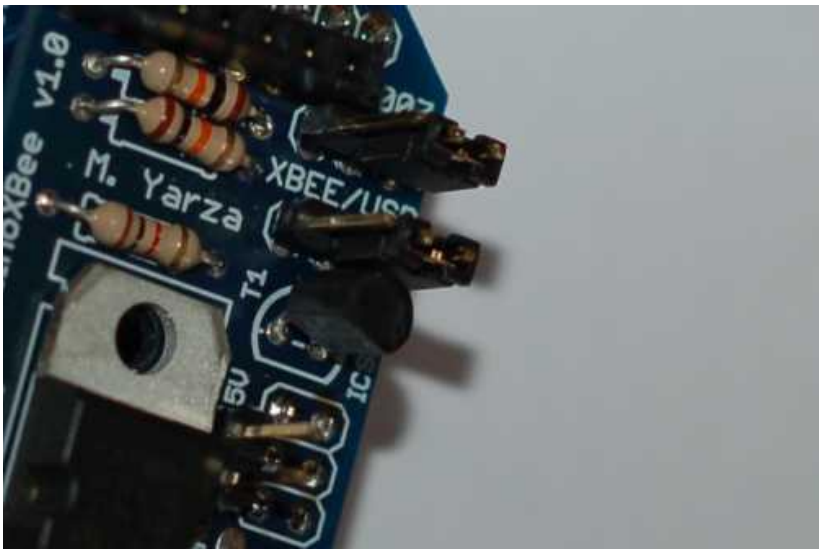


Fig. 3.24 *Jumpers XBEE shield*

Con los jumpers en la posición Xbee (en los dos pines más cercanos al interior de la placa, como se ilustra en la *figura 3.24*), el pin DOUT del módulo Xbee está conectado al pin RX del microcontrolador; y el pin DIN está conectado a TX. Notar que los pines RX y TX del microcontrolador están todavía conectados a los pines TX y RX (respectivamente) del chip FTD. Los datos enviados desde el microcontrolador serán transmitidos al ordenador vía USB y a la vez enviados de forma inalámbrica por el módulo Xbee. El microcontrolador, sin embargo, solo será capaz de recibir datos desde el módulo Xbee, no desde el USB del ordenador.

Con los jumpers en la posición USB (e.g. en los dos pines más cercanos al borde de la placa), el pin DOUT del módulo Xbee está conectado al pin RX del pin del chip FTDI, y el DIN del módulo Xbee está conectado al pin TX del el chip FTDI. Esto significa que el módulo Xbee puede comunicarse directamente con el ordenador. Sin embargo, esto solo funciona si el microcontrolador ha sido quitado de la placa Arduino. Si el microcontrolador se deja en la placa Arduino, solo será capaz de comunicarse con el ordenador vía USB, pero ni el ordenador ni el microcontrolador podrán comunicarse con el módulo Xbee.

3.4.3.3 Información sobre los pines

La descripción de los pines del módulo XBee Serie 1 se muestra en la tabla 4.9 y la ubicación de los mismos en la figura 4.25. De estos 20 pines, los únicos requeridos para llevar a cabo una comunicación elemental son VCC (alimentación), GND (tierra), DIN (entrada de datos) y DOUT (salida de datos). Para actualizar el firmware aparte de los pines citados anteriormente se habrán de usar el RTS y el DTR. (Digi International Inc., 2013)

Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / **CONFIG**	Input	UART Data In
4	DO8*	Output	Digital Output 8
5	**RESET**	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI	Output	PWM Output 0 / RX Signal Strength Indicator
7	PWM1	Output	PWM Output 1
8	[reserved]	-	Do not connect
9	$\overline{\text{DTR}}$ / SLEEP_RQ / DI8	Input	Pin Sleep Control Line or Digital Input 8
10	GND	-	Ground
11	AD4 / DIO4	Either	Analog Input 4 or Digital I/O 4
12	**CTS** / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7
13	ON / **SLEEP**	Output	Module Status Indicator
14	VREF	Input	Voltage Reference for A/D Inputs
15	Associate / AD5 / DIO5	Either	Associated Indicator, Analog Input 5 or Digital I/O 5
16	**RTS** / AD6 / DIO6	Either	Request-to-Send Flow Control, Analog Input 6 or Digital I/O 6
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0	Either	Analog Input 0 or Digital I/O 0

Tabla 3.9 *Descripción de pines de Xbee Series 1.*

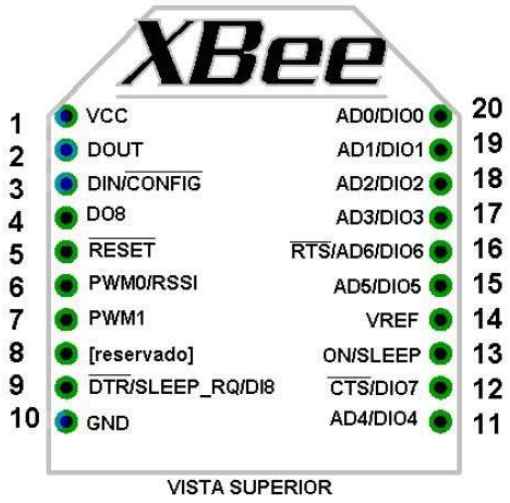


Fig. 3.25 *Localización de pines en Xbee Series 1*

Todos los pines no utilizados deben dejarse desconectado, pero no hay un tratamiento específico o requerido para las salidas no utilizadas. Otros pines pueden estar conectados a circuitos externos por conveniencia de operación, como lo son el pin 15 asociado a LED y el pin de botón de puesta en marcha (pin 20). El pin 15 parpadeará de forma diferente dependiendo del estado del módulo, y un pulsador conectado a la patilla 20 puede permitir el despliegue y diversas soluciones de problemas de funciones sin tener que enviar comandos UART. Si se desea muestreo analógico, el pin VREF (pin 14) debe ser conectado a una tensión de referencia.

3.4.3.4 Comunicación Serial

El módulo XBee se puede comunicar con un dispositivo anfitrión a través un puerto serie a nivel asíncrono. A través de este puerto serie, el módulo se puede comunicar con cualquier lógica y a cualquier voltaje compatible con UART (Universal Asynchronous Receiver Transmitter); o si es adaptado a cualquier dispositivo serial, ejemplo de este último es el XBee Shield utilizado en este proyecto y descrito en el acápite 3.7. (Digi International Inc., 2013)

Los dispositivos que tienen una interfaz UART pueden conectarse directamente a los terminales del módulo de RF, como se muestra en la figura 4.26.

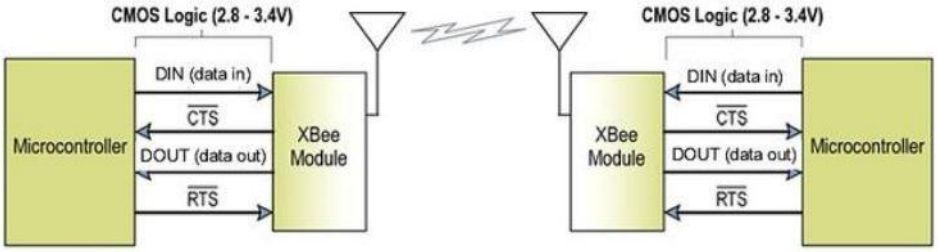


Fig. 3.26 *Comunicación entre módulos Xbee*

Los datos entran al módulo UART a través del DIN (pin 3) como una señal serie asíncrona. La señal debe permanecer en high cuando no se está transmitiendo información. Cada byte de dato consiste en un bit de inicio (low), 8 bit de datos (el menos significativo se envía primero) y un bit de parada (high). La figura 4.27 ilustra el patrón de transmisión de datos serial del módulo.

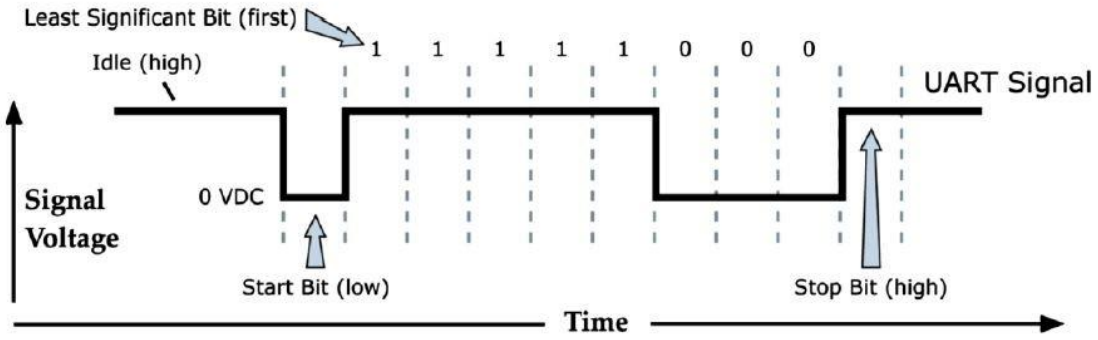


Fig. 3.27 *Patrón de transmisión de datos serial en los módulos XBee*

3.4.3.5 Configuración

Hay múltiples parámetros que necesitan ser configurados correctamente para que dos módulos puedan comunicarse entre ellos, en la tabla 4.10 se muestra un listado de todos los parámetros configurables en el módulo XBee serie 1. Para realizar la configuración se ha utilizado el software Xctu.

El módulo XBee se puede programar por puerto serie utilizando comandos AT, aunque lo más sencillo es utilizar la herramienta X-CTU.

X-CTU: Digi pone a disposición una herramienta de configuración para Windows: X-CTU. Esta permite configurar los parámetros del módulo y actualizar el firmware. Además, también permite:

- Descubrir todos los dispositivos XBee de la red
- Actualizar el firmware de un módulo local (por USB o puerto serie)
- Leer y escribir los parámetros de configuración del módulo en un dispositivo local o remoto
- Grabar y cargar perfiles de configuración conteniendo configuración personalizada

Para que dos módulos se puedan comunicar necesitan estar en la misma red, definida por el parámetro ID, los módulos necesitan estar en el mismo canal, definido por el parámetro CH, y finalmente, la dirección de destino de un módulo (parámetros DH y DL) determina que módulo en esa red y canal recibirá los datos transmitidos. Esto puede suceder de las siguientes formas:

- Si el DH de un módulo es 0 y su DL es menor de 0xFFF, los datos transmitidos por ese módulo serán recibidos por cualquier módulo cuyos 16 bits de dirección del parámetro MY sea igual al DL.
- Si el DH es 0 y el DL es igual a 0xFFFF, las transmisiones del módulo serán recibidas por todos los módulos.
- Si el DH no es cero o el DL es mayor de 0xFFFF, la transmisión solo será recibida por el módulo cuyo número de serie sea igual a la dirección de destino del módulo transmisor (cuyos SH es igual al DH del módulo transmisor y cuyo SL sea igual a su DL).

De nuevo, esta correspondencia de direcciones solo sucederá entre módulos en la misma red y canal. Si dos módulos están en diferentes redes o canales, no podrán comunicarse sea cual sea sus direcciones.

Comando	*Descripción*	*Valores Válidos*	*Valores por Defecto*
ID	El ID de la red del módulo Xbee.	0 – 0xFFFF	3332
CH	El canal del módulo Xbee.	0x0B – 0x1A	0X0C
SH y SL	El número serie del módulo Xbee (SH devuelve los 32 bits superiores, SL los 32 inferiores). De solo-lectura.	0 - 0xFFFFFFFF (para ambos SH y SL)	Diferente para cada módulo
MY	La dirección de 16 bit del módulo.	0 – 0xFFFF	0
DH y DL	La dirección de destino para las comunicaciones inalámbricas (DH son los 32 bits superiores, DL son los 32 inferiores).	0 – 0xFFFFFFFF (para ambos DH y DL)	0 (Para ambos DH y DL)
BD	La velocidad de transmisión usada para las comunicaciones con el Arduino o el ordenador.	0 (1200 bps) 1 (2400 bps) 2 (4800 bps) 3 (9600 bps) 4 (19200 bps) 5 (38400 bps) 6 (57600 bps) 7 (115200 bps)	3 (9600 bps)

Tabla 3.10 *Parámetros de configuración del módulo XBee series 1*

CAPÍTULO 4

IMPLEMENTACIÓN HARDWARE

En este capítulo, se abordará el montaje e implementación de los diferentes bloques funcionales que forman el control domótico de acondicionamiento exterior, detallando todo lo relacionado a sus componentes, esquemas de montaje y proceso de fabricación. Se mostrará la evolución de los diseños circuitales empleados, su depuración y optimización, además del proceso de ensamblado y su acabado final.

4.1 INTRODUCCIÓN

El proceso de implementación de la arquitectura hardware de control está dividido en dos partes bien diferenciadas. En primer lugar se abarca toda la información fundamental sobre los componentes de cada módulo, así como sus principales características, ventajas e inconvenientes.

Por otro lado, veremos el desarrollo y montaje más adecuado paso a paso de dichos componentes en sus respectivos módulos con todo detalle, su diseño electrónico y la versión final de los mismos en el que observaremos su correcto funcionamiento

Durante toda la etapa de implementación, se aplicó el máximo afán de mejora e innovación, para realizar los diseños tanto a nivel hardware electrónico, como a nivel software con la interfaz de control desarrollada.

4.2 MATERIALES NECESARIOS PARA SU DESARROLLO

4.2.1 Sensor de Movimiento

Un detector de movimiento es un dispositivo electrónico equipado de sensores que responden un movimiento físico.

Los detectores de presencia son equipos eléctricos que encienden la luz con total fiabilidad cuando detectan movimiento en la zona que cubren. De esta forma, la luz solo se enciende cuando es necesario

La aplicación más extendida de los detectores de presencia es la iluminación, aunque también pueden encontrarse en instalaciones de climatización, de control o de seguridad. Su uso permite el ahorro de energía en espacios de uso esporádico.

Por lo general, su funcionamiento se basa en combinar un sensor de movimiento u ocupación junto con un temporizador y un interruptor electrónico para encender o apagar las luces cuando no son necesarias. Sin embargo, existen diferentes tecnologías que es conveniente conocer para saber cuál es la más adecuada en cada caso particular.

4.2.1.1 Tipos de detección

Detección por infrarrojos

Los detectores de presencia más sencillos y habituales son los denominados Passive Infrared (PIR) y se basa principalmente en un sistema que detecta variaciones de temperatura. La luz se enciende automáticamente cuando el sensor detecta la radiación térmica o energía que emite el cuerpo humano. Por otro lado, cuando se colocan en el exterior, han de ser resistentes a factores climatológicos adversos. Sólo así garantizarán un buen funcionamiento.

Detección por ultrasonidos

Está basada en la emisión de ondas de ultrasonidos fuera del rango de audición humana. En este caso, la diferencia entre la frecuencia de la onda emitida y recibida es interpretada como la existencia de personas. Estos sensores, de tipo activo, son capaces de "ver" a través de esquinas y objetos, por lo que son aconsejables para la detección de movimientos pequeños y suelen cubrir superficies mayores.

4.2.1.2 Características principales del sensor de movimiento Evology

Tras una ardua búsqueda, nos decantamos por el sensor de presencia Evology, cumpliendo todas las características necesarias para nuestro proyecto y el correcto funcionamiento con el software de arduino.

Fig. 4.1 *Sensor de movimiento Evology*

Principales características

- ***Cobertura***

  Según el lugar en el que se instale el detector, éste tiene un radio de acción que suele oscilar entre 180º (horizontal) o 60º (vertical).

Las zonas más habituales son la pared -desde donde puede detectar el movimiento en un radio de 180º- o el techo -desde donde abarca una zona de 180º-, pero también es posible colocar los detectores en una esquina.
Respecto a la distancia de detección, también depende del lugar en el que se instale el detector y de la potencia que tenga. La carga media que soportan es de 110 voltios, para cubrir áreas máximo de 12 metros, si las condiciones son óptimas.

- ***Sensor***

  180º horizontal o 60º vertical

- ***Voltaje***

  110-120V @ 50-60Hz

- ***Consumo estático***

  0.5W

- ***Distancia de transmisión***

  12 metros

- ***Material***

  Plástico

4.2.2 Iluminación LED

El uso de los Diodos LEDs en el ámbito de la iluminación es moderado pero con un previsible crecimiento en el futuro cercano, ya que sus prestaciones son superiores a las de la lámpara incandescente y la lámpara fluorescente, desde diversos puntos de vista. La iluminación con LEDs presenta indudables ventajas: fiabilidad, mayor eficiencia energética, mayor resistencia a las vibraciones, mejor visión ante diversas circunstancias de iluminación, menor disipación de energía, menor riesgo para el medio ambiente, capacidad para operar de forma intermitente de modo continuo, respuesta rápida, etc. Asimismo, con LEDs se pueden producir luces de diferentes colores con un rendimiento luminoso elevado, a diferencia de muchas de las lámparas utilizadas hasta ahora, que tienen filtros para lograr un efecto similar (lo que supone una reducción de su eficiencia energética). Cabe destacar también que diversas pruebas realizadas por importantes empresas y organismos han concluido que el ahorro energético varía entre el 70 y el 80% respecto a la iluminación tradicional que se utiliza hasta ahora. Todo ello pone de manifiesto las numerosas ventajas que los LEDs ofrecen en relación al alumbrado público.

4.2.2.1 Tira de LEDs

El primero de los dos dispositivos de iluminación que utilizaremos en nuestro control domótico es una tira de LED adhesiva.

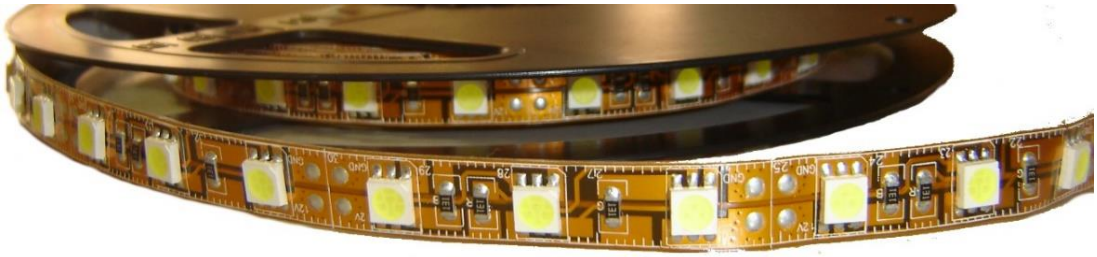


Fig. 4.2 *Tira de LEDs*

Las tiras de LED flexibles de alta tensión se conectan directamente a la red eléctrica de 230V mediante un alimentador. Están fabricadas con componentes de alta calidad, permiten una alta disipación del calor y son impermeables gracias a la cubierta transparente de PVC.

Con la tiras LED de 230V se pueden hacer instalaciones de hasta 50 metros por alimentador, sin riesgo de caídas de tensión. No requieren transformadores, amplificadores ni adaptadores adicionales, consiguiendo grandes ahorros en materiales y mano de obra en comparación con las tiras LED de 12/24V. Se pueden cortar cada metro, por lo que las tiras LED a 220V reducen significativamente la probabilidad de fallos. Como usan circuitos serie/paralelo, si se daña una zona (cada zona 1 metro) no afecta a las demás.

Gracias a su flexibilidad y alta luminosidad, son ideales para crear una iluminación de calidad en todo tipo de ambientes, tanto en interiores como en exteriores.

A continuación, se reúnen las principales especificaciones técnicas de la tira de LED a tener en cuenta en la siguiente tabla:

Especificaciones Técnicas - Tira de LED	
Voltaje	230V/ 12VDC
Tipo de LED	Coolwhite 5500k
Potencia	3,6 W
Tiempo de duración del chip	12000h
Ángulo de dispersión lumínica	120º
Temperatura de trabajo	< 30ºC
Dimensiones	1000mm x 2,5mm x 8mm

Tabla 4.1 *Especificaciones Técnicas*

Destacamos en la siguiente tabla, las características más relevantes que caben señalar en nuestra tira de LEDs:

Características Principales - Tira de LED
Carrete de 1 metro
3 LED/m (30 LED por carrete)
Zona de corte cada 100mm – cada grupo de 3 chips
Autoadhesiva (en la parte posterior)
Muy flexible para poder realizar una instalación domótica
Alto brillo y luminosidad

Tabla 4.2 *Principales características de la tira de LEDs*

4.2.2.2 Foco Exterior

El segundo dispositivo LED que disponemos en el desarrollo de nuestro proyecto, se trata de un foco LED para exterior.

El foco LED, es uno de los mercados donde más claramente se ven las mejoras de la iluminación LED, pudiendo no solamente encontrar importantes ahorros, sino también mejoras lumínicas de mayor calidad, que generen un ambiente cómodo, agradable y una iluminación más adaptada al entorno, pudiendo direccionar la luz allí donde nos interesa.

La tecnología LED (Light Emitting Diode) consume el 92% menos que las lámparas incandescentes de uso doméstico común y el 30% menos que la mayoría de las lámparas fluorescentes. Además, pueden durar hasta 20 años y suponer el 200% menos de coste total si se comparan con las lámparas o tubos fluorescentes convencionales.

Fig. 4.3 *Foco LED*

Las especificaciones técnicas de nuestro modelo de foco LED, mostrado en la figura 3. Las describimos a continuación en la siguiente tabla:

Especificaciones Técnicas – Foco LED	
Tipo	De fijar
Material específico	Aluminio
Color	Negro
Forma	Rectangular
Voltaje	230V
Potencia Máxima	0,06 W
Índice de protección	IP44 (contra agentes externos y/o agua)
Número de LEDs	28 bombillas LED
Dimensiones	10cm x 14cm x 19,2cm

Tabla 4.3 *Especificaciones técnicas del foco LED*

4.2.2.3 Ventajas e inconvenientes de la iluminación LED

Las principales ***ventajas*** de usar la iluminación LED en nuestro control domótico las enumeramos en la siguiente lista:

- **Consumo**. Una lámpara LED consume alrededor de una décima parte que una bombilla incandescente equivalente, y la mitad de su equivalente fluorescente. El LED no genera pérdidas por radiación infrarroja o ultravioleta, la pérdida por calor es inferior a las incandescentes y halógenas. Por todo ello, la iluminación conseguida por vatio consumido es mayor.

- **Tiempo de Vida.** El tiempo de vida de un LED puede sobrepasar las 50,000 horas, esto es 50 veces más que la de una bombilla incandescente y más de dos veces el de una lámpara fluorescente.

- **Emisión de Calor.** Los LEDs proporcionan producen mucho menos calor que sus competidoras, lo que contribuye a un menor encapsulamiento pues necesita disipar menos calor, disminuye los costos de acondicionamiento de temperatura en espacios cerrados, y en general beneficia al medio ambiente al reducir el efecto invernadero en el planeta.

- **Tiempo de respuesta.** El tiempo de respuesta de las lámparas LEDs se encuentra en el orden de los microsegundos, frente a los milisegundos de las lámparas incandescentes, y frente al tiempo de respuesta de las lámparas fluorescentes que es mucho mayor que ambas. La bombilla incandescente está basada en la utilización de filamentos incandescentes, muy largos en algunos casos, lo que genera una gran persistencia a la hora de enfriarse y volver a encenderse. Es decir, su velocidad de encendido y apagado es baja frente a la utilización de componentes semiconductores.

- **Luminosidad.** Los LEDs son más brillantes que una bombilla tradicional, brillan por igual desde el principio y con menor degradación de flujo durante su vida útil. Uno de los problemas principales de los sistemas de iluminación actuales es el mantenimiento del flujo luminoso a lo largo de la vida útil de las lámparas. Por diferentes circunstancias, la mayoría de los sistemas degradan rápidamente su flujo luminoso en función del tiempo.

- **Resistencia mecánica**. Como los LEDs no tienen ampolla de cristal ni un frágil filamento, son muy resistentes y más duraderos que cualquier otra fuente de luz. Dada la temperatura que suelen alcanzar los filamentos de las lámparas actuales, pueden llegar a los 500ºC, los materiales que soportan bien estas temperaturas y son transparentes, son el vidrio y el cristal.

- **Libre de Mercurio**. Las lámparas fluorescentes, las más utilizadas aun el día de hoy debido a su bajo consumo en relación a las bombillas incandescentes, funcionan al pasar electricidad a través de vapor de mercurio. Si una lámpara fluorescente se rompe puede provocar exposición al mercurio, el cual es un elemento altamente contaminante y venenoso.

Por otra parte, el proceso de producción de LEDs blancos es complejo y tiene muchos aspectos por mejorar.

Esto significa que el costo de producción a alto volumen es aun relativamente alta en comparación con las fuentes de luz tradicionales. El proceso utilizado para depositar las capas de semiconductores activos de la LED está constantemente mejorando para aumentar el rendimiento en la producción. También se ha informado sobre problemas relativos a la dificultad de manejo del fosforo para obtener resultados estandarizados en cuanto ancho de banda de la luz obtenida. (Soltic & Chalmers, 2012)

Los LEDs tienen una tolerancia limitada al aumento de temperatura, lo cual puede provocar degradación de la eficiencia de los LEDs o en el peor de los casos el colapso del mismo. Este inconveniente limita el total de LEDs que pueden ser encapsulados de forma práctica en lámparas comerciales y requiere de una correcta aplicación de disipadores de calor en el encapsulado.

4.3 DESARROLLO DE LOS COMPONENTES

En este apartado se desarrolla todo el procedimiento de montaje e implementación del diseño de los dos módulos que componen nuestro sistema domótico.

4.3.1 Diseño módulo emisor

El primero de los módulos, el módulo emisor, está compuesto por el sensor de movimiento con su correspondiente placa de acondicionamiento, una placa que realizará el control manual y el módulo Arduino con el xbee shield y su tarjeta inalámbrica xbee que mandará la información al módulo receptor.

A continuación se expone el diseño e implementación de las placas de acondicionamiento que se han desarrollado en el primer módulo.

4.3.1.1 Alimentación del sensor de movimiento

La alimentación del circuito se realizará con un transformador de 220V /50Hz de corriente alterna a 9V de continua. A su vez, alimentará la etapa previa del circuito que reducirá estos 9V a 5V requeridos por los componentes activos del circuito.

No es necesaria la modelización del transformador y del puente de diodos ya que van integrados en el circuito externo del transformador.

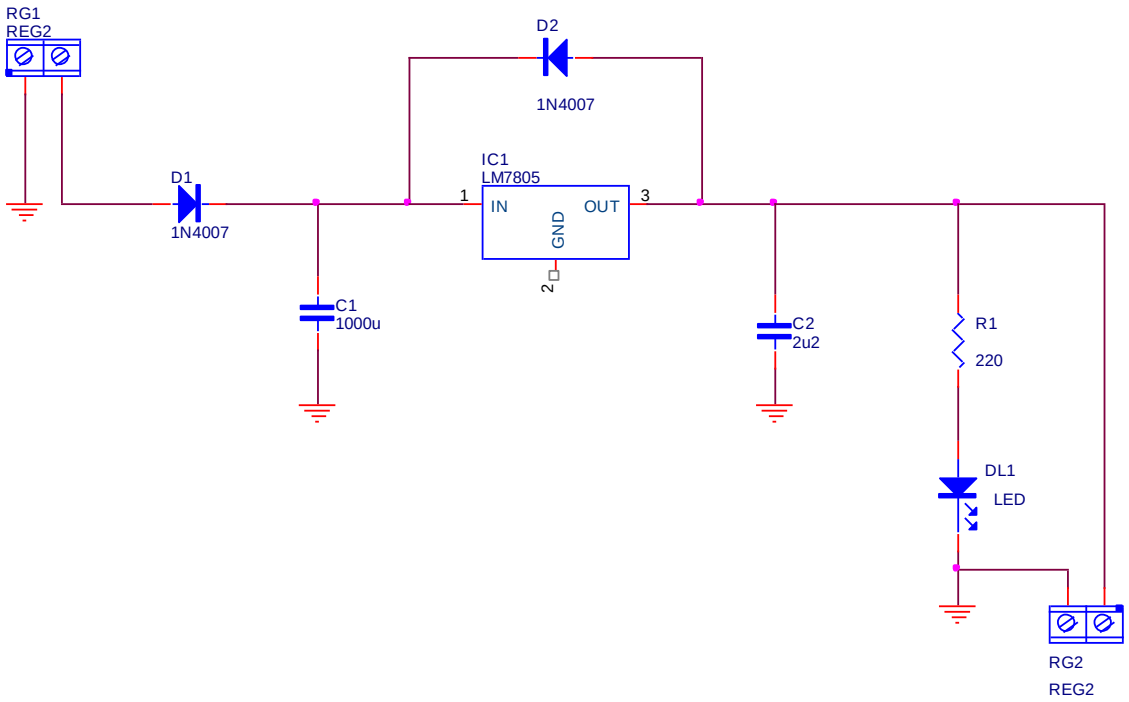


Fig. 4.4 *Esquemático de alimentación de 9V a 5V*

Al no requerir modelización el transformador y el puente de diodos, ya que este circuito tan solo reduce de 9V a 5V, damos por hecho que la tensión no requiere filtrado ya que procede de un transformador comercial (Aunque de todas formas añadimos los condensadores para evitar un posible rizado). Utilizamos una fuente lineal estándar aprendida de la asignatura Circuitos Integrados Analógicos no Lineales.

El rendimiento de la fuente será:

$$\eta_{\%} = \frac{V_{out}}{V_{in}} \cdot 100$$

Con los datos aproximados obtenemos un rendimiento de:

$$\frac{5V}{9V} \cdot 100 = 55{,}6\ \%$$

El rendimiento queda afectado, ya que no podemos ajustar todos los valores que necesitamos para obtener una tensión de salida válida para nuestro circuito.

Una vez diseñado el circuito de acondicionamiento procedimos a realizar el trabajo manual en el laboratorio, donde se dio forma a la primera de las placas que darían funcionamiento a nuestro sistema domótico.

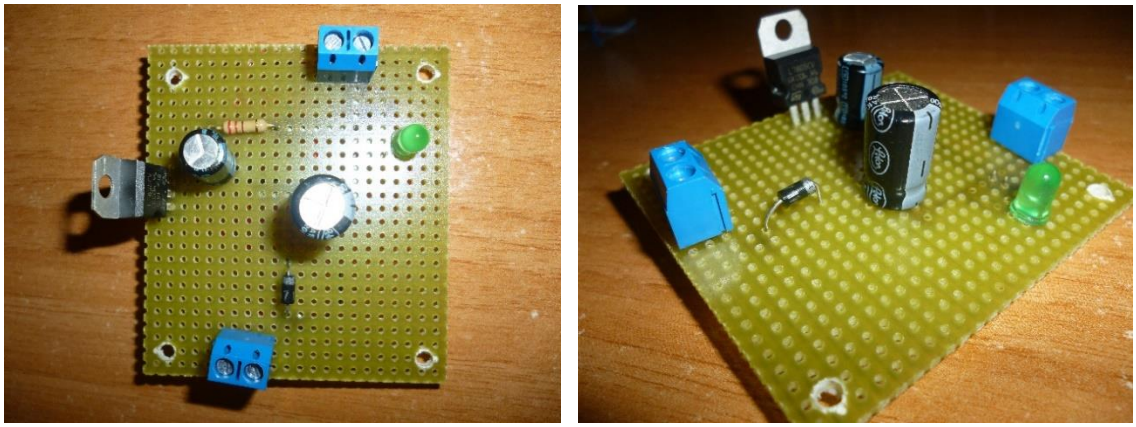


Fig. 4.5 *Imagen en planta (derecha) y perspectiva isométrica (izquierda) del circuito de alimentación del sensor de movimiento.*

Con este circuito conseguimos reducir la tensión proveniente de la fuente de alimentación del sensor y, a su vez, transmitir la señal de nivel alto al arduino de manera que sepa que ha detectado movimiento.

El diodo LED contenido en el circuito nos sirve también como indicativo de que el sistema funciona correctamente ya que, al detectar el sensor movimiento, el LED debe pasar a estado encendido durante el tiempo de detección y una vez el sensor deje de detectar, volver al estado de apagado.

Por otro lado, con ayuda de dos enchufes, uno macho y otro hembra, ensamblamos el cableado del sensor del movimiento para poder conectarlo a la red doméstica y, a su vez, a la placa de acondicionamiento.

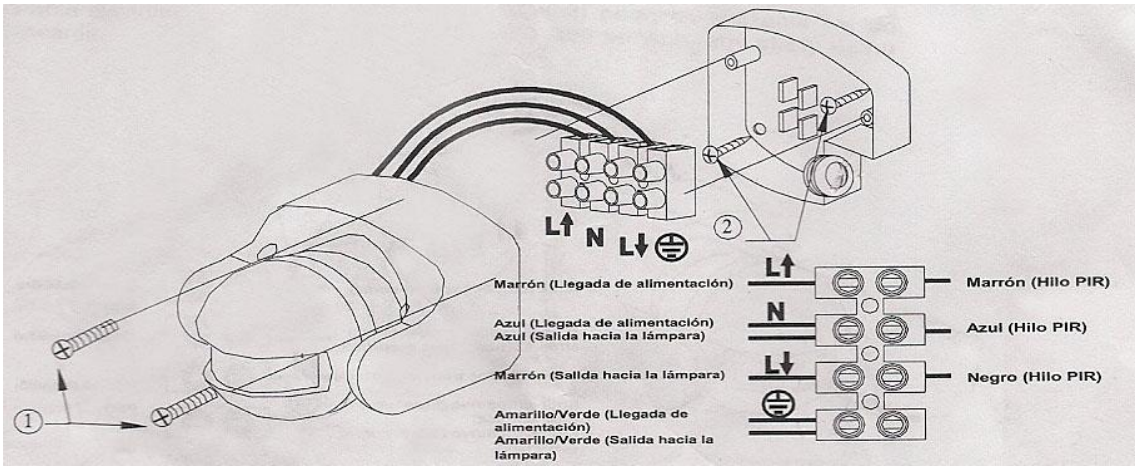


Fig. 4.6 *Instrucciones de instalación del detector*

Como podemos observar en las instrucciones, nos ofrecen la información necesaria para conectar el cableado a las dos cabezas de enchufe, macho y hembra.

Una vez realizadas las uniones, el resultado podemos observarlo en la figura 3.

Fig. 4.7 *Sensor de presencia con enchufes macho y hembra*

4.3.1.2 Placa de control manual

El segundo circuito que forma parte el módulo emisor es una placa de control manual que usaremos para el encendido y apagado de las luminarias cuando el usuario desee.

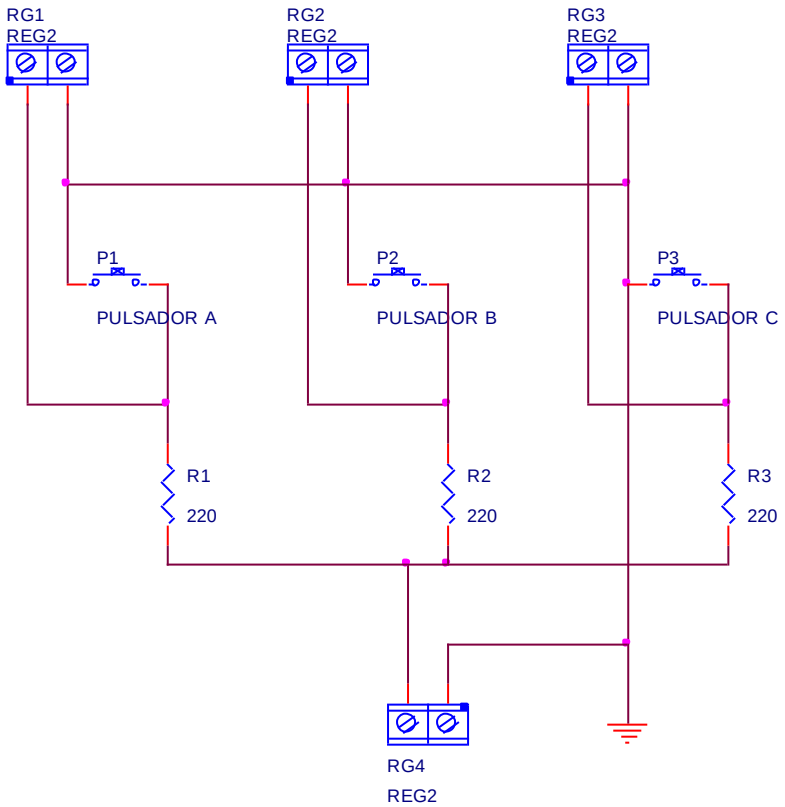


Fig. 4.8 *Esquemático de control manual del sistema domótico*

Como podemos observar, la placa de circuito está compuesta por tres pulsadores. Cada pulsador tiene una función distinta mediante la cual se controla el encendido y apagado, así:

- ✓ ***Pulsador A:*** Enciende la tira de LEDs.
- ✓ ***Pulsador B:*** Enciende el foco exterior.
- ✓ ***Pulsador C:*** Apaga la tira de LEDs, el foco exterior, o ambos, si están encendidos.

Las placas una vez realizadas en el laboratorio nos quedan de la siguiente manera:

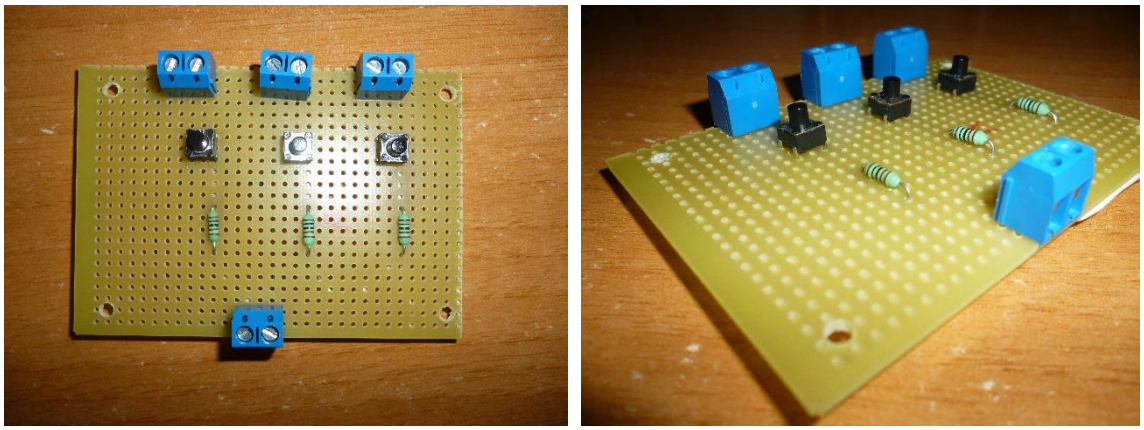


Fig. 4.9 *Vista en planta (izquierda) y perspectiva isométrica (derecha) del control manual*

El uso de resistencias lo realizamos para evitar daños en los pulsadores.

4.3.2 Diseño módulo receptor

El segundo módulo, el módulo receptor, estará formado por las dos luminarias, el foco exterior y la tira de LEDs. Cada una de las luminarias contará con su circuito de alimentación para el acondicionamiento de la red doméstica a la circuitería de arduino, consiguiendo así su correcto funcionamiento.

Así pues, a continuación se expone en diseño e implementación de las placas de acondicionamiento que se han desarrollado en el segundo módulo.

4.3.2.1 Alimentación del foco exterior

El foco exterior es alimentado por la red doméstica a 230V@50Hz, por tanto, este desmesurado voltaje no es compatible con la placa de arduino que trabaja con una tensión de 5V.

Por este motivo desarrollamos a continuación el primer circuito de acondicionamiento de las dos luminarias para poder trabajar correctamente con la plataforma arduino.

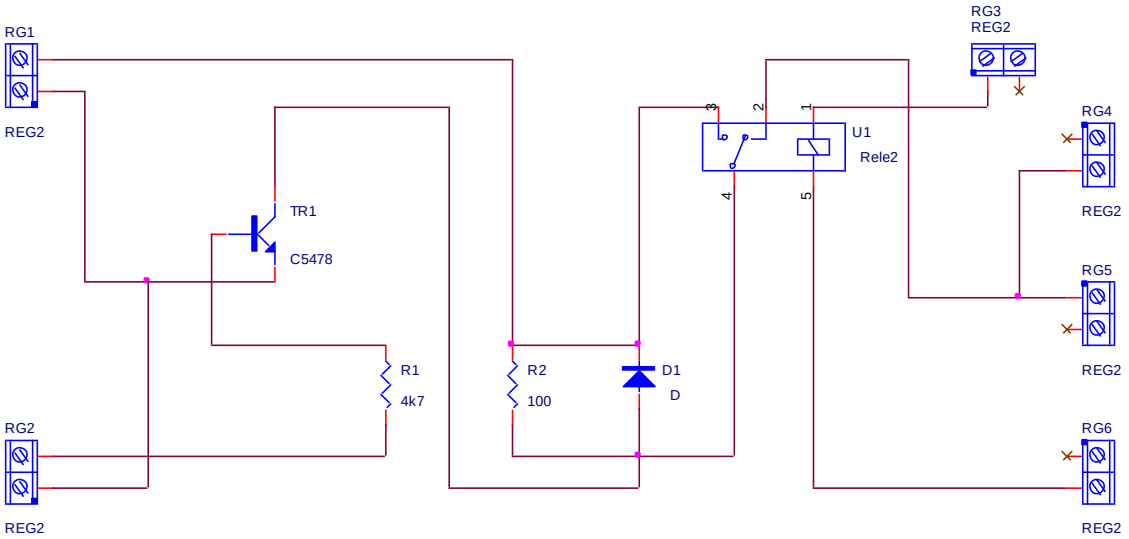


Fig. 4.10 *Esquemático de alimentación del foco exterior*

Este es el circuito más peligroso puesto que estamos tratando con alta tensión directamente sin etapa de filtrado previo, como es el caso de la alimentación del sensor de movimiento, que estudiamos anteriormente, y de la tira de LEDs, como veremos en el siguiente apartado. Por esta razón, las soldaduras realizadas en la placa en el cableado procedente del foco están selladas con una capa de silicona para mayor seguridad y evitando de esta manera la posible formación de arcos eléctricos. Así mismo, los cables utilizados en esta etapa de alta tensión son más gruesos que los cables de laboratorio que usamos en el resto del proyecto.

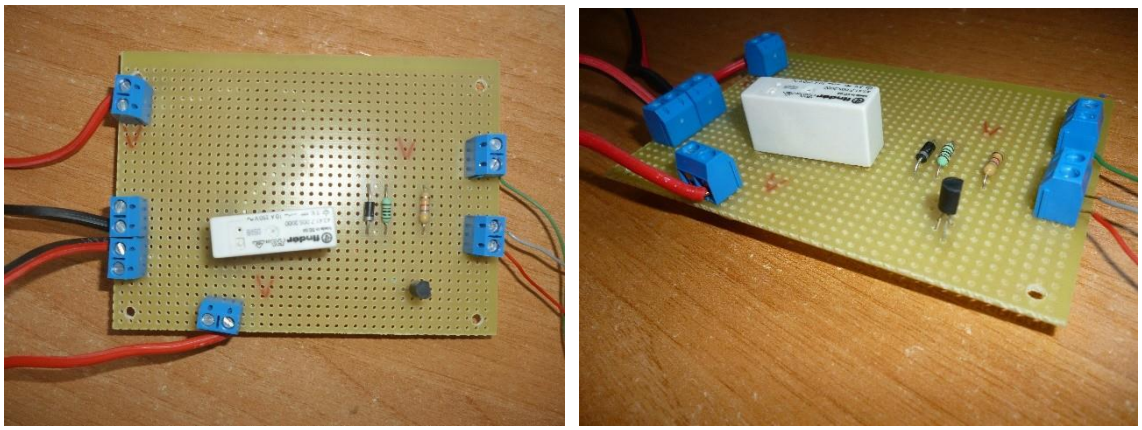


Fig. 4.11 *Vista en planta (izquierda) y perspectiva isométrica (derecha) de la placa de alimentación del foco exterior.*

El circuito cuenta también con un componente característico, un relé. La gran ventaja de los relés electromagnéticos es la completa separación eléctrica entre la corriente de accionamiento, la que circula por la bobina del electroimán, y los circuitos controlados por los contactos, lo que hace que se puedan manejar altos voltajes o elevadas potencias con pequeñas tensiones de control, lo que lo hace idóneo para el uso en nuestro circuito de acondicionamiento para el foco exterior.

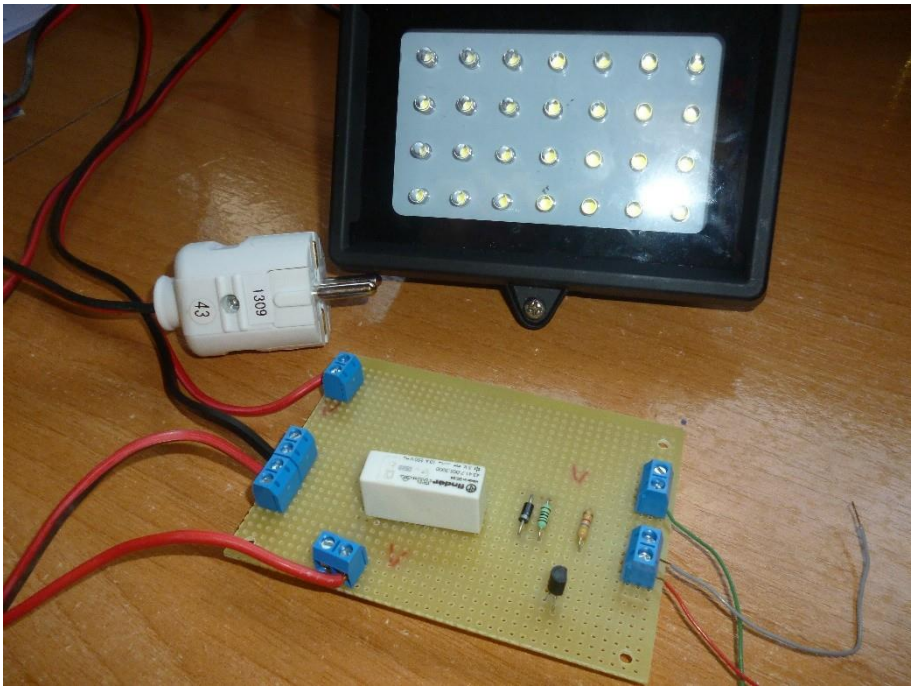


Fig. 4.12 *Montaje completo de la placa de alimentación, con el foco exterior y el enchufe*

Finalmente, se ha realizado la unión del cableado de tensión del foco a un enchufe macho para poder conectarlo a la red doméstica.

4.3.2.2 Alimentación de la tira de LEDs

Por último, mostramos el desarrollo del circuito de alimentación de la tira de LED. En este caso necesitaremos amplificar la señal proveniente del arduino para poder encender la tira de LED.

Cuando el transistor se encuentra en este estado de funcionamiento, permite amplificar la potencia de la señal recibida del arduino receptor y, por tanto, encender la luminaria.

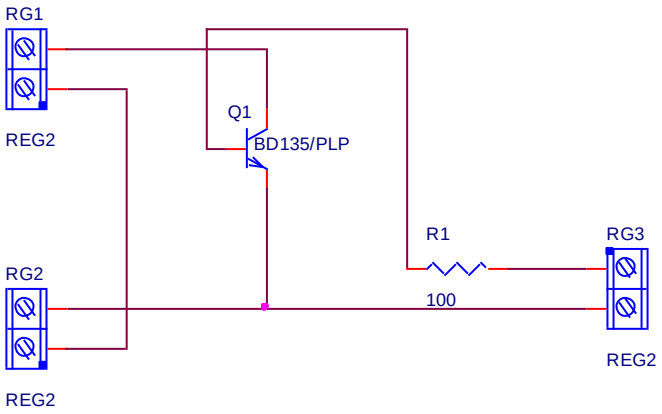


Fig. 4.13 *Esquemático de alimentación de la tira de LEDs*

Aquí mostramos el resultado una vez realizada la placa en el laboratorio.

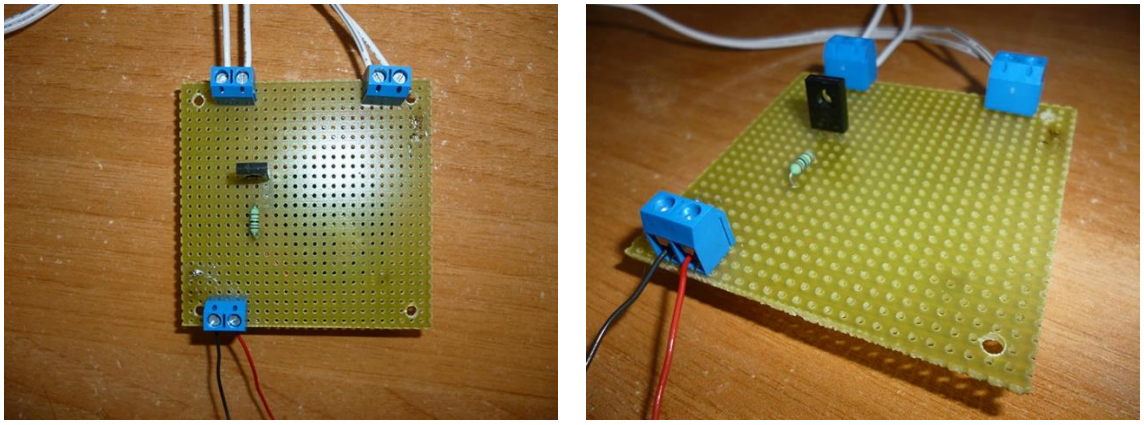


Fig. 4.14 *Vista en planta (izquierda) y perspectiva isométrica (derecha) de la placa de alimentación de la tira de LEDs*

Finalmente, tenemos la visual de la tarjeta conectada al transformador comercial, que realiza la etapa previa de reducción de la tensión de alimentación, y a la tira de LEDs.

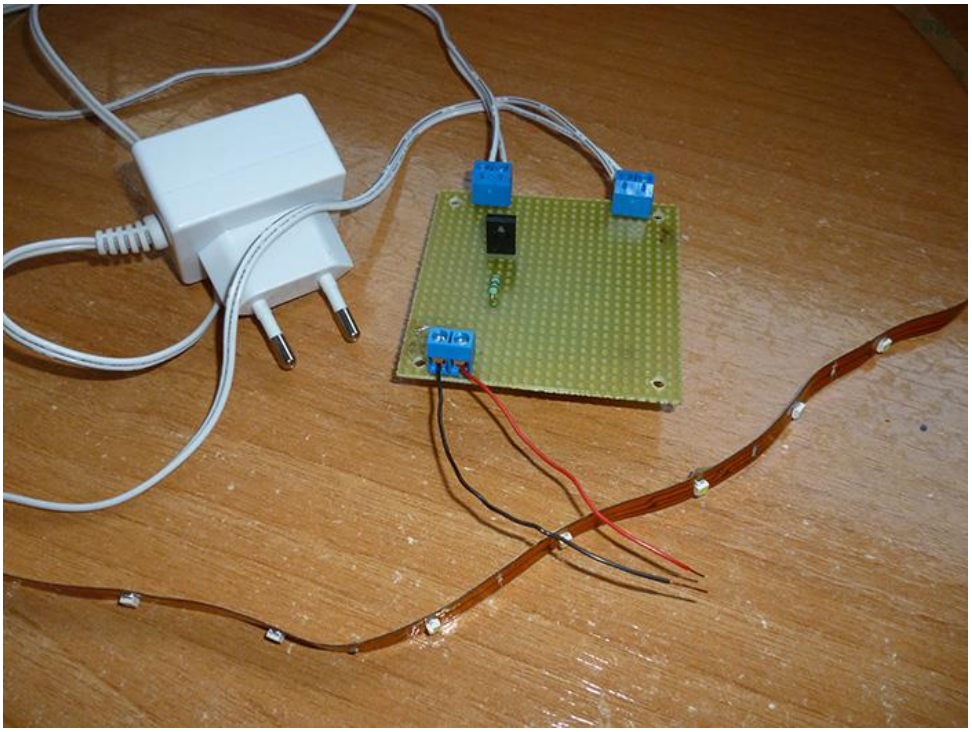


Fig. 4.15 *Montaje completo de la placa de alimentación con la tira de LED y el enchufe*

4.4 COMPROBACIÓN Y RESULTADO FINAL

Una vez que hemos realizado todas las tarjetas y uniones necesarias para nuestro control domótico, situamos los componentes en dos soportes individuales, quedando físicamente diferenciado también el módulo emisor y el módulo receptor y, posteriormente, procedemos a la comprobación de su correcto funcionamiento.

En esta primera imagen tenemos el soporte del módulo emisor, donde nos encontramos las placas de control manual y alimentación del sensor de movimiento, así como la placa Arduino emisora.

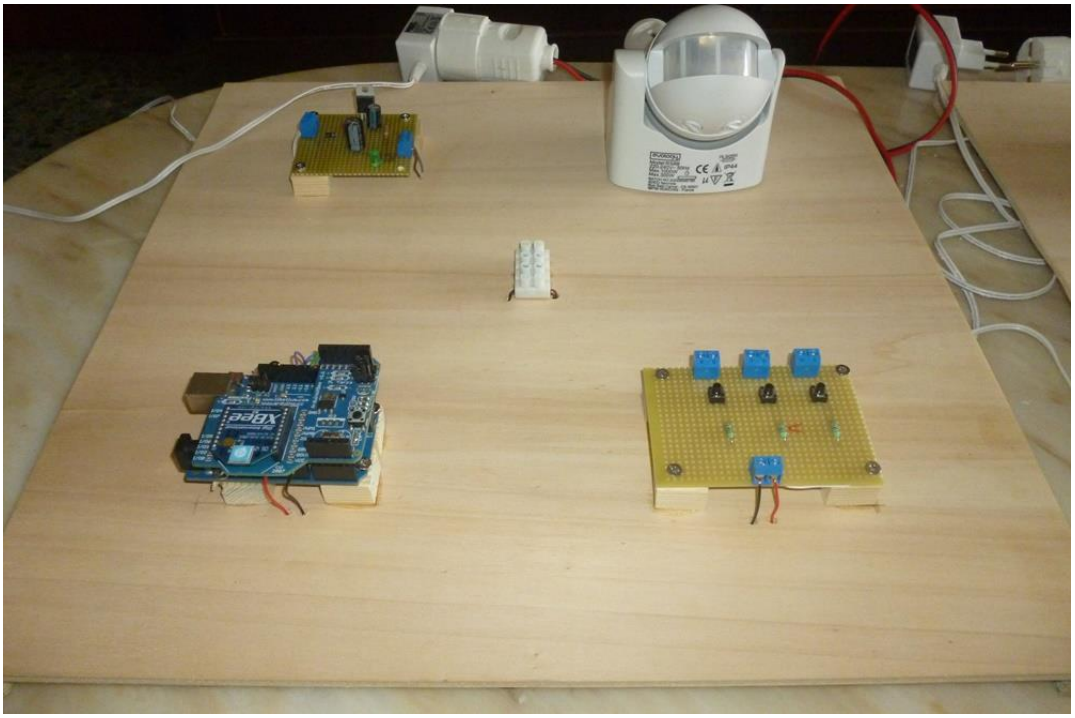


Fig. 4.16 *Módulo emisor*

Por otro lado, como muestra la imagen siguiente, están situadas en este tablero las tarjetas de alimentación del foco y la tira LED que conforman el módulo receptor junto con la placa Arduino receptora.

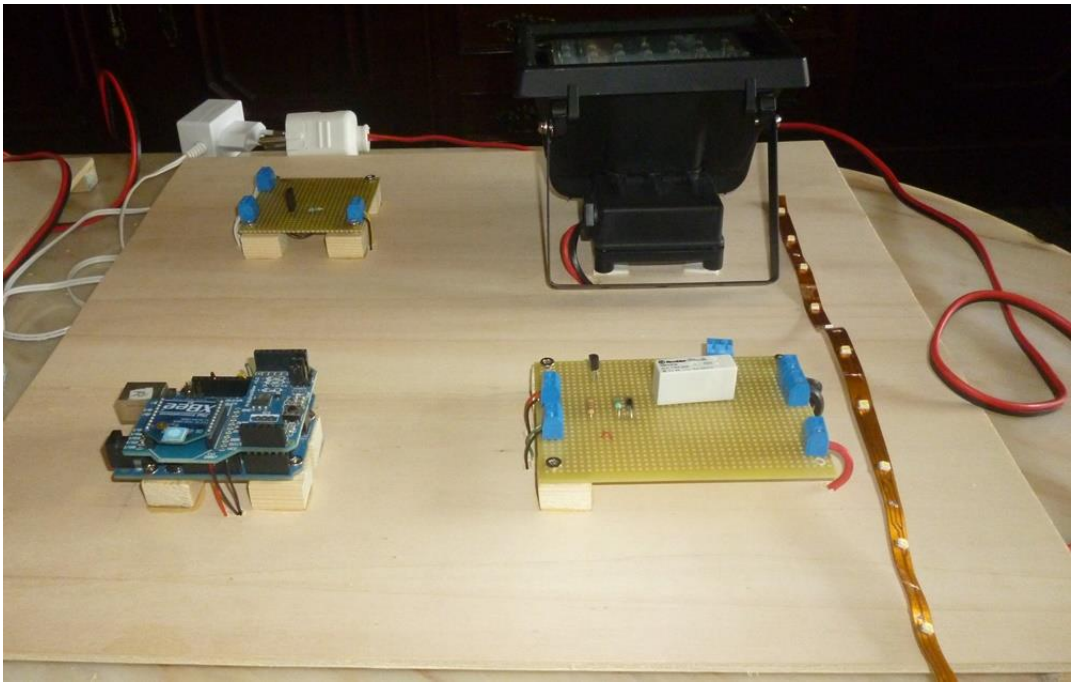


Fig. 4.17 *Módulo receptor*

Como podemos observar, las luminarias se accionarán en el momento en el que el sensor ha detectado una presencia.

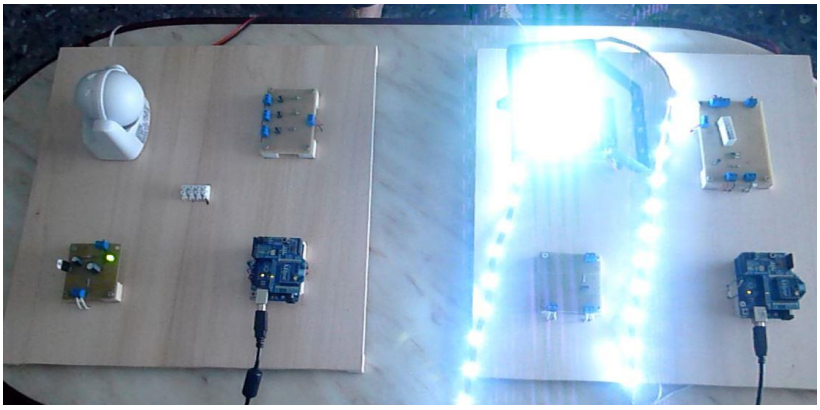


Fig. 4.18 *Luminarias encendidas por detección del sensor*

De la misma manera, con el control manual tendremos:

1. Si se acciona el botón A, se encenderá la tira LED:

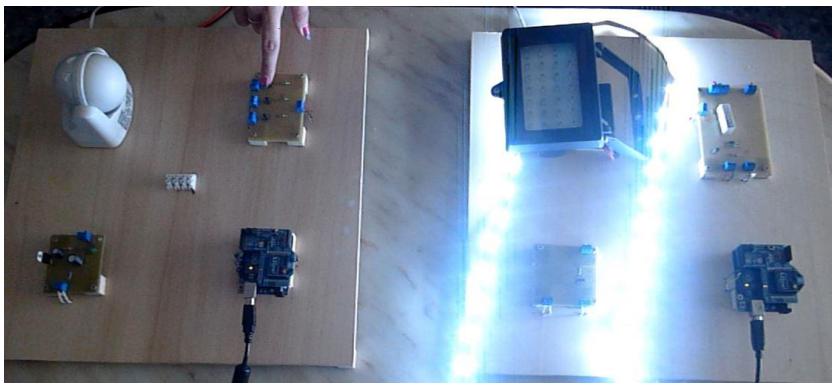


Fig. 4.19 *Tira LED encendida al accionar el botón A*

2. Si se acciona el botón B, se encenderá el foco:

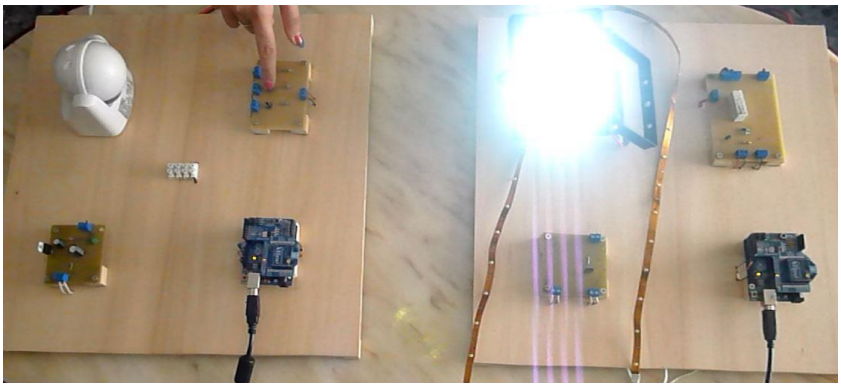


Fig. 4.20 *Foco exterior encendido al accionar el botón B*

3. Si se acciona el botón C, se apagará cualquier luminaria que se encuentre encendida, ya sea el foco, la tira LED, o ambas.

4.5 PLATAFORMA SOFTWARE DE DESARROLLO

En el último apartado de este capítulo damos una pequeña referencia al software utilizado a la hora de desarrollar los circuitos que nos ayudan a implementar el hardware de nuestro control domótico.

4.5.1 OrCAD 9.2

OrCAD es un paquete de software formado por las aplicaciones *Capture CIS*, *OrCAD PCB Editor*, *PSpice y Layout Plus*. El paquete OrCAD, entre otras funcionalidades, permite el diseño de placas de circuito impreso multicapa de forma sencilla y rápida. En el proyecto se utilizaron las aplicaciones *Capture CIS*, con el que se realizaron los esquemas del circuito. En las siguientes líneas se realizará una descripción de estas aplicaciones empleadas:

OrCAD Capture CIS permite el diseño de esquemas electrónicos en los cuales, previamente, se hayan incorporado componentes propios mediante la librerías. Aunque posee gran cantidad de librerías de componentes, permite la creación de librerías propias y el diseño de componentes específicos. Simplemente colocando los símbolos de los componentes y conectándolos, se consigue el circuito esquemático, y permite la opción de poder simular dicho circuito, obteniendo tensiones, intensidades, potencias, además de diversas gráficas.

Esta aplicación, Capture CIS, es la que hemos usado en el desarrollo de los circuitos que hemos diseñado y previamente vistos en los apartados anteriores. Aquí vemos un ejemplo de uno de los circuitos realizados (el circuito de control manual, concretamente) dentro de la interfaz del programa, con las barras de herramientas superior y lateral derecha, esenciales para llevar a cabo los diseños.

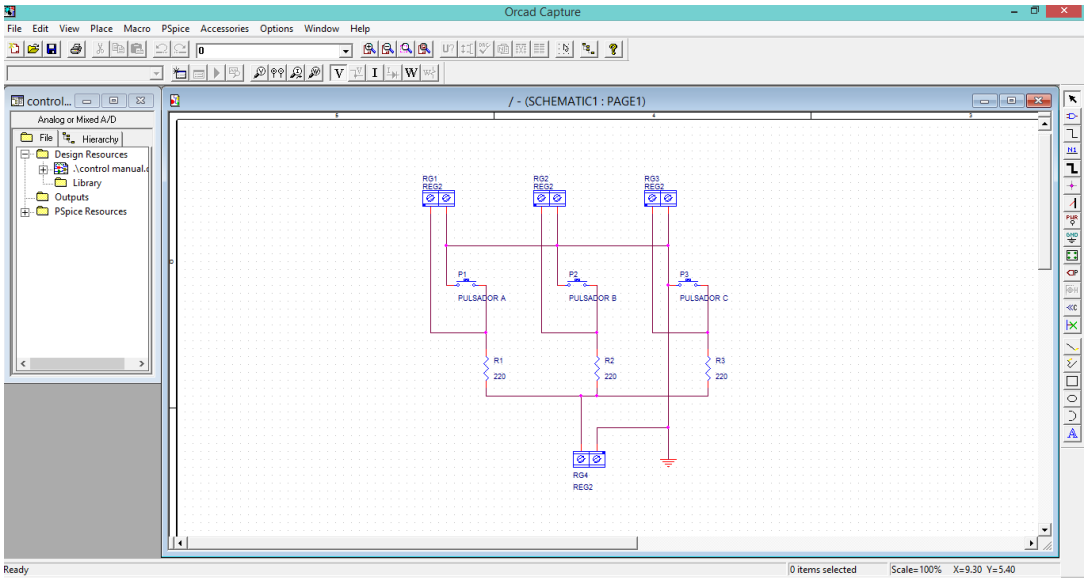


Fig. 4.21 *Interfaz de Orcad Capture CIS*

OrCAD Layout Plus, es el software encargado de transformar el esquemático realizado con la anterior aplicación al diseño de la placa de circuito impreso real. Hay disponibles varias opciones en cuanto a colocación de componentes y ruteo, no obstante lo más aconsejado es desatender la colocación que nos proporciona el programa y disponer los componentes de forma que se optimicen las dimensiones de la placa lo máximo posible. Para el ruteo de placas complejas existe la opción “autorute”, que realiza un ruteado automático de las pistas del diseño así como diferentes opciones de ruteado para minimizar los costes de implementación. También se pueden crear librerías, siendo indispensable para introducir las dimensiones reales de los componentes a utilizar para no encontrar problemas a la hora de su colocación y soldadura.

CAPÍTULO 5

IMPLEMENTACIÓN SOFTWARE

En este penúltimo capítulo se realiza un análisis del software de control del que se ha dotado al microcontrolador que incorpora la plataforma Arduino.

En este análisis se explica el software del módulo emisor y el módulo receptor por separado, comentando sus principales similitudes y diferencias.

5.1 INTRODUCCIÓN

En este capítulo se va a realizar el análisis detallado del software del que se ha dotado al microcontrolador ATMega que incorpora la plataforma Arduino para el correcto funcionamiento del control domótico.

En primer lugar presentaremos el entorno de programación en el que trabajaremos, así como el lenguaje de programación usado y unos ejemplos que nos ayudarán a entender la mecánica de la interfaz de Arduino.

Por último nos meteremos de lleno en el desarrollo del programa que hará que funcione nuestro control domótico, que veremos bien diferenciado en dos bloques, uno para cada módulo; emisor y receptor.

5.2 ENTORNO DE PROGRAMACIÓN

La plataforma Arduino se programa mediante el uso de un lenguaje propio basado en el popular lenguaje de programación de alto nivel Proccessing. Sin embargo, es posible utilizar otros lenguajes de programación y aplicaciones populares en Arduino como son: Java, Flash, proccessing, Python, Visual Basic .NET, C, C++, etc. (Arduino, 2013)

Esto es posible debido a que Arduino se comunica mediante la transmisión de datos en formato serie que es algo que la mayoría de los lenguajes anteriormente citados soportan. Para los que no soportan el formato serie de forma nativa, es posible utilizar software intermediario que traduzca los mensajes enviados por ambas partes para permitir una comunicación fluida. Es bastante interesante tener la posibilidad de interactuar Arduino mediante esta gran variedad de sistemas y lenguajes puesto que dependiendo de cuales sean las necesidades del problema que vamos a resolver podremos aprovecharnos de la gran compatibilidad de comunicación que ofrece. (Evans, 2007)

El entorno de Desarrollo Arduino está constituido por un editor de texto para escribir el código, un área de mensajes, una consola de texto, una barra de herramientas con botones para las funciones comunes, y distintos menús. Permite la conexión con el hardware de Arduino para cargar los programas y comunicarse con ellos.

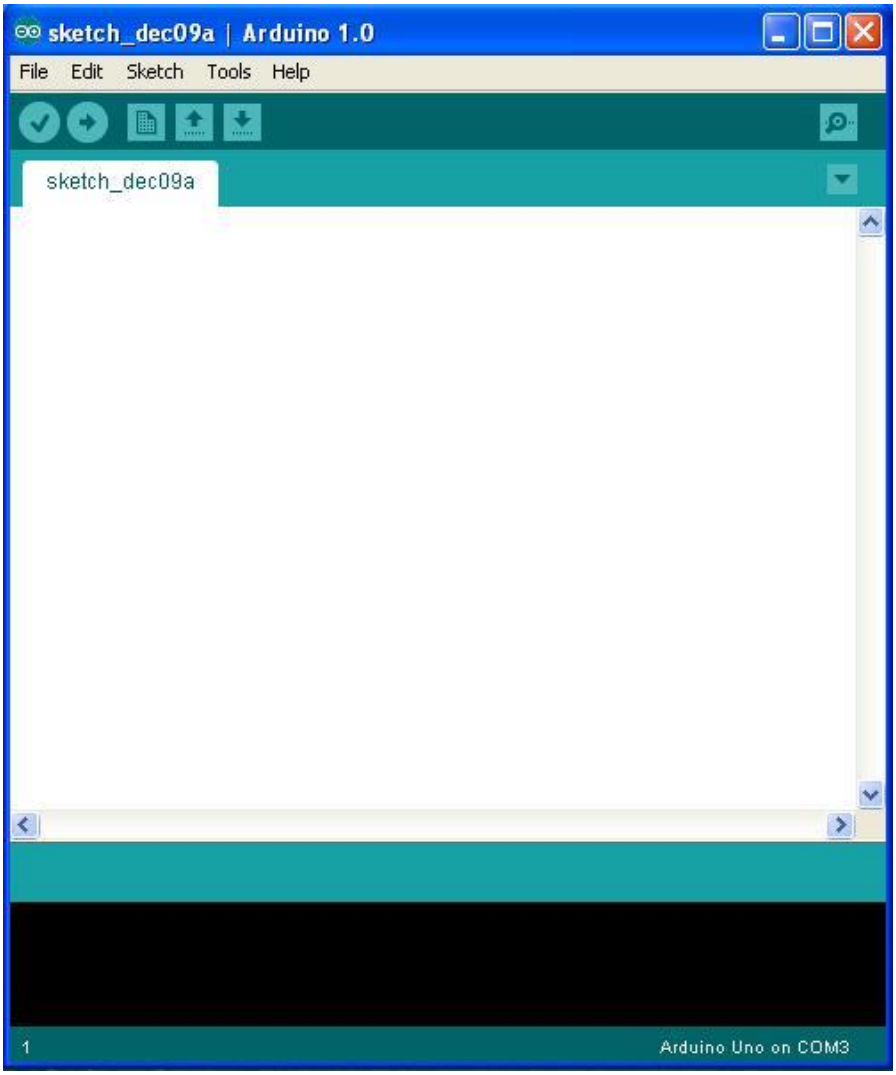


Fig. 5.1 *IDE de Arduino*

Arduino utiliza para escribir el software lo que denomina "sketch" (*programa*). Estos programas son escritos en el editor de texto. En el área de mensajes que será la parte inferior del editor, se muestran mensajes cuando se está compilando o cargando el programa, la finalización de estas tareas y también informan sobre la existencia de errores en el código.

5.2.1 Interfaz de usuario

- ***Barra de herramientas***

La IDE permite realizar determinadas tareas destinadas al desarrollo de proyectos. A continuación vamos a exponer cada una de estas tareas que están asociadas a las opciones de la barra de herramientas:

Fig. 5.2 *Barra de herramientas*

De izquierda a derecha, se explica a continuación la función de cada uno de los botones y accesos directos de la barra de herramientas del IDE de Arduino:

1. ***Verify:*** Chequea el código en busca de errores.
2. ***Upload:*** Compila el código y lo vuelca en la placa E/S de Arduino.
3. ***New:*** Crea un nuevo sketch
4. ***Open:*** Presenta un menú de todos los programas *sketch* de su *librería de sketch*. Un click sobre uno de ellos lo abrirá en la ventana actual.
5. ***Save:*** Guarda el programa.

- ***Menús***

  A continuación el contenido de los Menús más relevantes:

  - ***Menú sketch.***

    Verificar/ compilar: Comprueba tu rutina para errores.

    Mostrar la Carpeta de Sketch: Abre la carpeta de rutinas en tu escritorio.

    Agregar Archivo: Añade otro chero fuente a la rutina. El nuevo archivo aparece en una nueva pestaña en la ventana de la rutina. Esto facilita y agranda proyectos con múltiples archivos fuente. Los archivos pueden ser eliminados de una rutina usando el Tab Menu.

    Importar librería…: Utiliza una librería en tu rutina. Trabaja añadiendo #include en la cima de tu código. Esto añade funcionalidad extra a tu rutina, pero incrementa su tamaño. Para parar de usar una librería, elimina el #include apropiado de la cima de tu rutina.

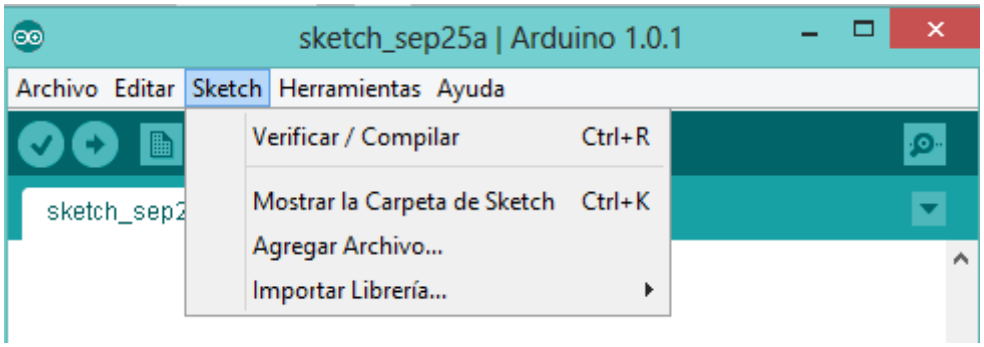


Fig. 5.3 *Opciones de menú sketch*

  - ***Menú herramientas.***

    Formato Automático Esto formatea tu código amigablemente.

    Tarjeta: Selecciona la placa que estas usando. Esto controla la forma en que tu rutina es compilada y cargada así como el comportamiento de los elementos del menú Burn Bootloader.

    Puerto Serial: Este menú contiene todos los dispositivos series (reales o virtuales) de tu máquina. Debería actualizarse automáticamente cada vez que abres el nivel superior del menú Tools. Antes de subir tu rutina, necesitas seleccionar el elemento de este menú que representa a tu placa Arduino. En Windows, es probablemente COM1 o COM2 (para una placa Serie) o COM4, COM5, COM7 o superior (para una placa USB).

    Grabar secuencia de inicio: Los elementos en este menú te permiten grabar un bootloader en tu placa con una variedad de programadores. Esto no es necesario para uso normal de una placa Arduino, pero puede ser útil si encargas

ATmegas adicionales o estás construyendo una placa por tu cuenta. Asegúrate que has seleccionado la placa correcta del menú Boards de antemano. Para grabar un bootloader con el AVR ISP, necesitas seleccionar el elemento que corresponde a tu programador del menú Serial Port.

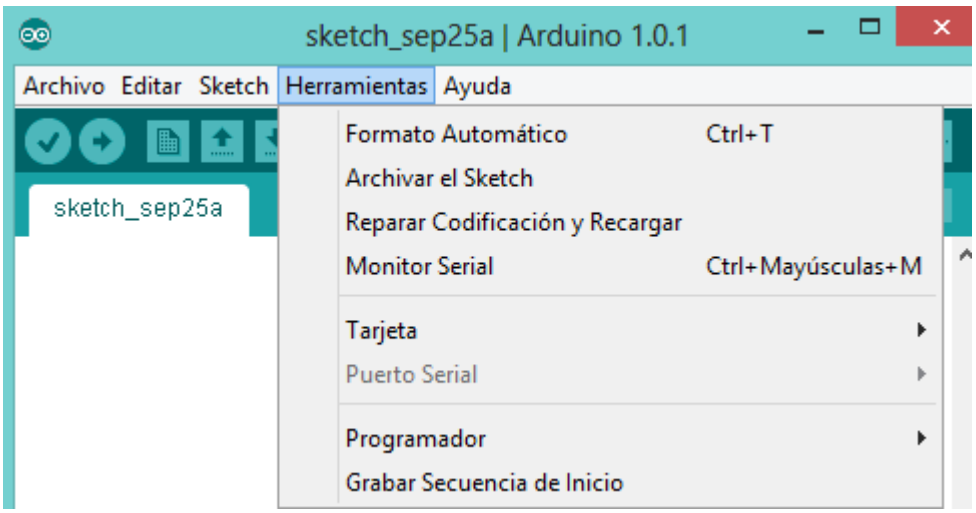


Fig. 5.4 *Opciones del menú herramientas*

5.3 LENGUAJE DE PROGRAMACIÓN

El lenguaje de programación que utiliza la plataforma Arduino es un lenguaje nativo y derivado del C++. Este es un lenguaje muy extendido, que posee múltiples librerías y documentación.

La IDE de Arduino con una librería de C/C++ llamada “Wiring”, la cual hace que las típicas operaciones de entrada/salida resulten más sencillas. Los programas de Arduino están escritos en un lenguaje derivado del C++. La estructura principal de cualquier programa será la siguiente:

```
int buttonPin = 3;

void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}

void loop()
{
  // ...
}
```

Fig. 5.5 *Estructura básica de un programa para Arduino*

La parte superior del programa, antes de la función setup (), es utilizada para la creación e inicialización de variables locales, es decir, variables que serán comunes a todo el programa.

Las principales funciones que contiene todo código en Arduino son:

- **setup ()**: La función setup () se establece cuando se inicia un programa -sketch. Se emplea para iniciar variables, establecer el estado de los pines, inicializar librerías, etc. Esta función se ejecutará una única vez después de que se conecte la placa Arduino a la fuente de alimentación, o cuando se pulse el botón de reinicio de la placa.

- **loop ()**: Después de crear la función setup(), la cual inicializa y prepara los valores iniciales, la función loop () actuará como un bucle de programa. Se ejecuta consecutivamente, permitiéndole al programa variar y responder. Se usa para controlar de forma activa la placa Arduino.

A continuación se comenta las funciones específicas más usuales en la programación con Arduino y que serán de utilidad para la elaboración del software de control:

- **pinMode():** Configura un pin como entrada o salida. Para utilizarla, le pasas el número del pin que vas a configurar y la constante INPUT u OUTPUT. Es decir configura el pin especificado para comportarse como una entrada o una salida. Se usa dentro de la función setup().
  Sintaxis ->pinMode(pin, modo)

- **digitalWrite():** Escribe o envía un valor HIGH o LOW hacia un pin digital. Por ejemplo, la línea:
  digitalWrite(ledPin, HIGH);

- **digitalRead():** Lee el valor de un pin digital especificado, HIGH o LOW. Sintaxis-> digitalRead(pin) donde "pin" será el número de pin digital que quieres leer (*int*). Devolverá HIGH o LOW.

- **delay():** Pausa el programa por un tiempo determinado (en milisegundos) especificado por un parámetro. Hace a Arduino esperar por el número especificado de milisegundos antes de continuar con la siguiente línea.

- **analogRead():** Lee el valor de tensión en el pin analógico especificado. La placa Arduino posee 6 canales conectados a un conversor analógico digital de 10 bits. Esto significa que convertirá tensiones entre 0 y 5 voltios a un número entero entre 0 y 1023. Esto proporciona una resolución en la lectura de: 5 voltios / 1024 unidades, es decir, 0.0049 voltios (4.9 mV) por unidad. El rango de entrada puede ser cambiado usando la función analogReference().

- **analogWrite():** Escribe un valor analógico (PWM) en un pin. Puede ser usado para controlar la luminosidad de un LED o la velocidad de un motor. Después de llamar a la función analogWrite(), el pin generará una onda cuadrada estable con el ciclo de trabajo especificado hasta que se vuelva a llamar a la función analogWrite() (o una llamada a las funciones digitalRead() o digitalWrite() en el mismo pin). La frecuencia de la señal PWM será de aproximadamente 490 Hz. Esta frecuencia podrá ser variada por el usuario.

- **Serial:** Se utiliza para la comunicación entre la placa Arduino con otros dispositivos mediante comunicación serie. Todas las placas Arduino tienen al menos un puerto serie (también conocido como UART). Se comunica a través de los pines digitales 0 (RX) y 1 (TX), así como con el ordenador mediante USB. Por lo tanto, si se utilizan estas funciones, no podrán ser usados los pines 0 y 1 como entrada o salida digital.

Dentro de esta comunicación serie, podemos encontrar distintas funciones que serán muy útiles para la comunicación inalámbrica:

- **Serial.begin:** Establece la velocidad de datos en bits por segundo (baudios) para la transmisión de datos en serie. Para comunicarse con el computador, utilice una de estas velocidades: 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 o 115200. Sin embargo, puedes especificar otras velocidades - por ejemplo, para comunicarte a través de los pines 0 y 1 con un componente que requiere una velocidad de transmisión en particular.

- **Serial.available():** Devuelve el número de bytes (caracteres) disponibles para ser leídos por el puerto serie. Se refiere a datos ya recibidos y disponibles en el buffer de recepción del puerto (que tiene una capacidad de 128 bytes).

- **Serial.read():** Lee los datos entrantes del puerto serie.

- **Serial.write():** Escribe datos binarios en el puerto serie. Estos datos se envían como un byte o una serie de bytes.

5.3.1 Ejemplos de programas en arduino

Antes de meternos de lleno en la programación de los módulos, veremos una serie de ejemplos sencillos y relacionados con lo que será el código que desarrollemos en nuestros programas, contenidos en la propia interfaz de arduino, que nos ayudará a comprender la forma de realizar estos programas.

5.3.1.1 Blink

Este ejemplo muestra lo más simple que se puede hacer con un arduino para ver la salida física: el parpadeo de un LED.

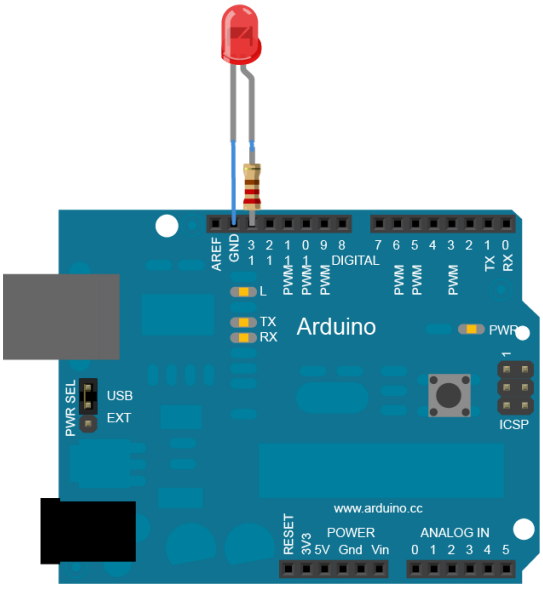


Fig. 5.6 *Montaje Blink*

Para construir el circuito, se debe colocar una resistencia al pin 13. Acto seguido, conectamos la pata larga de un LED (la pata positiva, el ánodo) a la resistencia. Conectamos la pata corta (la pata negativa, el cátodo) a tierra. Después, enchufamos la placa de arduino al ordenador, abrimos el programa de arduino e introducimos el código siguiente:

```
int led = 13;

void setup() {

  pinMode(led, OUTPUT);
}

void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

Fig. 5.7 *Código Blink*

En el programa, lo primero que se hace es inicializar el pin 13 como pin de salida con la línea

```
pinMode(13, OUTPUT);
```

En el loop principal, encendemos el LED con la línea:

```
digitalWrite(13, HIGH);
```

Con ello se proporcionan 5 voltios al pin 13. Eso genera una diferencia de potencial a través de los pines del LED, y lo enciende. A continuación, lo apagamos con la línea:

```
digitalWrite(13, LOW);
```

Esta línea vuelve a poner el pin 13 a 0 voltios, y apaga el LED. Entre el encendido y el apagado, queremos tiempo suficiente para que se vea el cambio, por tanto, el comando **`delay()`** le dice al arduino que no haga nada en 1000 milisegundos, o en un segundo. Cuando se usa el comando **`delay()`** nada pasa en esa cantidad de tiempo.

5.3.1.2 Digital Read Serial

Este ejemplo muestra como monitorizar el estado de un cambio estabilizando la comunicación serial entre el arduino y el ordenador a través de USB.

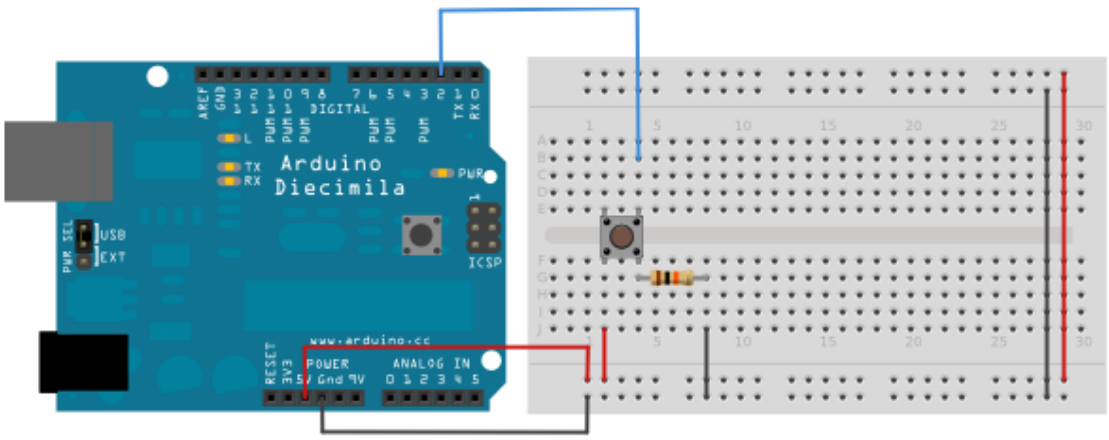


Fig. 5.8 *Montaje Digital Read Serial*

Conectamos tres cables a la placa arduino. Los dos primeros, rojo y negro, se conectan a las dos filas largas verticales en el lado de la protoboard para proporcionar el acceso a los 5 voltios suministrados y a tierra. El tercer cable va del pin 2 digital a una pierna del pulsador. La misma pata del botón se conecta mediante una resistencia a tierra. La otra pata del botón se conecta a los 5 voltios.

Cuando el pulsador está abierto (sin pulsar) no hay conexión entre las dos patas del pulsador, por tanto el pin está conectado a tierra (a través de la resistencia) y se lee como nivel bajo (LOW), o 0. Cuando el pulsador se cierra (se presiona), se produce una conexión entre sus dos patas, conectando el pin a los 5 voltios, por lo que el pin se lee como nivel alto (HIGH), o 1.

Si se desconecta el pin de entrada/salida digital de todo, el LED puede parpadear erróneamente. Esto ocurre porque la entra está "flotante", es decir, no tiene una conexión sólida de voltaje o tierra, y devolverá de manera aleatoria HIGH o LOW. Esta es la razón por la que se necesita una resistencia en el circuito.

```
int pushButton = 2;

void setup() {

  Serial.begin(9600);

  pinMode(pushButton, INPUT);
}

void loop() {

  int buttonState = digitalRead(pushButton);

  Serial.println(buttonState);
  delay(1);
}
```

Fig. 5.9 *Código Read Digital Serial*

En el programa que vemos arriba, la primera cosa que hacemos en la función setup es iniciar la comunicación serial, a 9600 bits de datos por segundo, entre el arduino y el ordenador con la línea:

```
Serial.begin(9600);
```

Lo siguiente es inicializar el pin digital 2, el pin que leerá la salida del botón, como una entrada:

```
pinMode(2,INPUT);
```

Ahora que el setup se ha completado, nos trasladamos al loop principal del código. Cuando el pulsador se presiona, los 5 voltios fluirán libremente a través del circuito, y cuando no está presionado, el pin de entrada estará conectado a tierra mediante la resistencia de 10K. Es una entrada digital, lo que significa que el pulsador solo puede estar entre un estado de encendido (visto en Arduino como un "1", o HIGH) o un estado apagado (visto en el Arduino como un "0", o LOW) sin ningún valor de por medio.

Lo primero que se hace en el loop principal es establecer una variable para almacenar la información que llega del pulsador. Ya que la información que llega del pulsador estará entre los valores "1" y "0", se puede usar un int datatype. Llamamos a esta variable `sensorValue` y la igualamos a lo que se esté leyendo en el pin digital 2. Todo ello lo realizamos con la línea del código:

```
int sensorValue = digitalRead(2);
```

Una vez el Arduino ha leído la entrada, imprime esta información en el ordenador como un valor decimal. Lo hacemos con el comando Serial.println() en la última línea del código:

```
Serial.println(sensorValue);
```

Ahora, cuando abrimos el Monitor Serial en la interfaz de Arduino, veremos un torrente de “0” si el interruptor está abierto, o de “1” si el interruptor está cerrado.

5.3.1.4 Physical Pixel

Por último, veremos este ejemplo en el que veremos una combinación de los dos programas anteriores.

Este ejemplo usa la placa Arduino para recibir datos del ordenador. El Arduino enciende un LED cuando recibe el carácter ‘H’, y lo apaga cuando recibe el carácter ‘L’.

Los datos (caracteres) pueden ser enviados desde el monitor serial del Arduino, o en otros programas de monitorización.

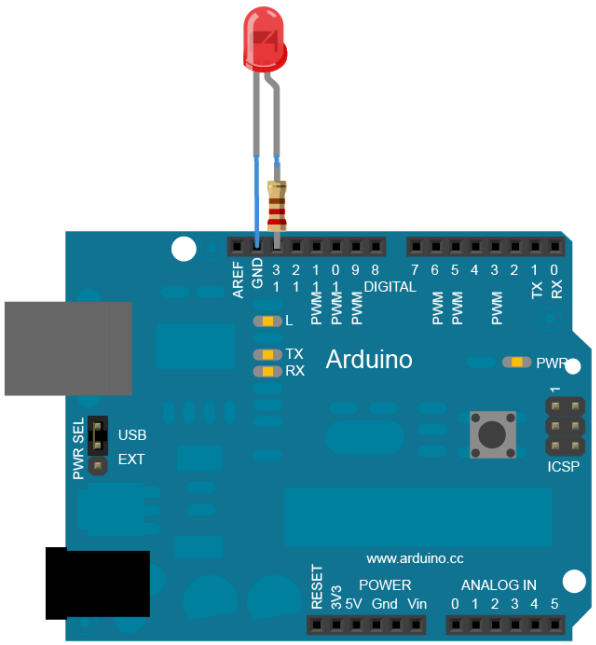


Fig. 5.10 *Montaje Physical Pixel*

Para el montaje de este circuito primeramente conectamos un LED en el pin13. La pata larga, o ánodo, va al pin 13. La pata corta, o cátodo, va conectado a tierra.

```
const int ledPin = 13;
int incomingByte;

void setup() {

  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}
```

```
void loop() {

  if (Serial.available() > 0) {

    incomingByte = Serial.read();
        if (incomingByte == 'H') {
      digitalWrite(ledPin, HIGH);
    }
        if (incomingByte == 'L') {
      digitalWrite(ledPin, LOW);
    }
  }
}
```

Fig. 5.11 *Código Physical Pixel*

En el programa que vemos arriba, la primera cosa que hacemos en la función setup es iniciar la comunicación serial, a 9600 bits de datos por segundo, entre el arduino y el ordenador con la línea:

```
Serial.begin(9600);
```

Lo siguiente es inicializar el pin LED digital 13 como una salida:

```
pinMode(2,INPUT);
```

Ahora que el setup se ha completado, nos trasladamos al loop principal del código. Lo primero que realiza el loop será comprobar si hay algún dato serial entrando en él mediante la línea:

```
if (Serial.available() > 0)
```

En el caso de que efectivamente el programa detecte un dato, éste lee el último byte en el buffer serial con `incomingByte = Serial.read();` con lo que se darán dos casos:

Si hay una “H” mayúscula (ASCII 72), el LED se enciende. Esto lo realiza la línea:

```
if (incomingByte == 'H') {
   digitalWrite(ledPin, HIGH);
```

El otro caso será cuando la siguiente línea lea una “L” mayúscula (ASCII 76), momento en el que el LED se apagará:

```
if (incomingByte == 'L') {
    digitalWrite(ledPin, LOW);
```

La introducción de los caracteres “H” y “L” se realizarán mediante el monitor serial del entorno arduino.

5.4 SOFTWARE MODULO EMISOR

5.4.1 Asignación de pines

Comenzamos nuestro programa emisor. En este primer apartado, explicamos a qué pines de la plataforma Arduino están conectados los distintos pulsadores y Leds.

El código de asignación de pines se realiza mediante la función ***#define***, quedándonos la siguiente lista de comandos:

```
#define Sensor 8
#define BotonA 9
#define BotonB 10
#define BotonC 11
```

Como podemos observar, tanto el sensor como los tres pulsadores tienen adjudicado un número, el cual pertenece a cada uno de los pines digitales de la placa emisor de Arduino, en los que irán conectadas las patas positivas de cada uno de los dispositivos.

5.4.2 Estados

Todo programa debe identificar una serie de estados en los que se encuentran los elementos que conforman el sistema. Con las siguientes líneas realizamos esta función, identificando los posibles escenarios que puedan darse en el módulo emisor de nuestro control domótico.

```
#define INICIAL 0
#define DETECCION 1
#define PULSADOA 2
#define PULSADOB 3
#define PULSADOC 4
#define HOLD 5
```

Se definen cinco estados diferentes:

- ***INICIAL*.** Estado que nos indica que el sistema se encuentra en arranque.
- ***DETECCION.*** Este estado hace referencia al momento en el que el sensor de presencia ha detectado movimiento.
- ***PULSADOA.*** Estado que indica que se ha pulsado el botón A.
- ***PULSADOB.*** Estado que indica que se ha pulsado el botón B.
- ***PULSADOC.*** Estado que indica que se ha pulsado el botón C.
- ***HOLD.*** Es un estado auxiliar en el que el sistema entra cuando no se produce cambio en ninguna variable. El sistema se mantiene en este estado hasta el momento en que alguna de las variables que se están comprobando continuamente cambia de valor.

5.4.3 Inicialización de variables

El sistema debe empezar con unos valores iniciales que debemos de establecer nosotros. En este apartado del código llevamos a cabo esa tarea.

```
int Estado = INICIAL;
int valorSensor = 0;
int valorBA = 1;
int valorBB = 1;
int valorBC = 1;
int sensorAnterior = 0;
int baAnterior = 0;
int bbAnterior = 0;
int bcAnterior = 0;
boolean cambiaEstado = false;
char mensaje;
```

La variable *estado* indica en que condición se encuentra el sistema en cada momento. Como se observa, arranca en el estado *INICIAL*. La variable *valorSensor*, almacena el valor que proporciona el sensor, siendo 0 el estado de no detección y 1 el de detección. Al igual que el anterior, *valorba, valorbb y valorbc* son variables de tipo entero que almacenan el valor correspondiente a cada uno de los botones A, B y C, siendo 1 no pulsado y 0 pulsado.

Por otro lado, tenemos las variables, *sensorAnterior, baAnterior, bbAnterior y bcAnterior,* que se usan para almacenar el valor que indicaban en el estado anterior al estado actual. Todos están inicializados a 0.

La variable *cambiaEstado* es de tipo booleano, y nos servirá para indicar si los valores del estado anterior al compararlos con los del estado actual han cambiado. Se establece en *false*, que corresponde al estado en que no cambia la variable. Si es *true* indica que se ha producido un cambio de variable.

Por último, tenemos la variable de tipo carácter *mensaje,* la cual almacena la información que se le enviará al receptor indicándole en qué estado se encuentra el sistema, que puede ser:

- Si el sensor ha detectado
- Si se ha pulsado alguno de los 3 botones
- Si no se produce ningún cambio

5.4.4 Setup

```
void setup() {

Serial.begin(9600);
pinMode(Sensor, INPUT);
pinMode(BotonA, INPUT);
pinMode(BotonB, INPUT);
pinMode(BotonC, INPUT);
```

Como podemos observar, el setup de nuestro programa emisor contiene los pines que conforman el módulo emisor y que son de tipo entrada (INPUT). A continuación, vemos las características detalladas en la siguiente tabla:

Nombre	Función	Pin de Arduino
Sensor	Entrada sensor movimiento	8
BotonA	Entrada del pulsador A, que encenderá el foco exterior	9
BotonB	Entrada del pulsador B, que encenderá la tira de LEDs	10
BotonC	Entrada del pulsador C, que apagará una o amas luminarias	11

Tabla 5.1 *Correspondencia de entradas y pines en Arduino*

5.4.5 Loop

En este apartado nos metemos en el grueso del programa, donde veremos qué funciones realiza el módulo emisor para su funcionamiento e interacción con el módulo receptor.

Como podemos observar más adelante, la estructura que usaremos para resolver el funcionamiento del programa será la estructura *switch.* Se trata de una estructura de

control empleada en programación que se utiliza para agilizar la toma de decisiones múltiples. Trabaja de la misma manera que lo harían sucesivos if, if else o until anidados, así como combinaciones propias de determinados lenguajes de programación.

```
void loop() {

      switch (Estado) {
      .....
      .....
```

En nuestro caso, disponemos de un sensor de movimiento y tres pulsadores, lo que nos dará lugar a 3 casos, *DETECCION, PULSADOA, PULSADOB* y *PULSADOC,* además de dos casos añadidos, el caso INICIAL y el caso HOLD.

Si nos fijamos en la estructura de cada caso, podemos ver que son prácticamente idénticas, a salvo de pequeñas peculiaridades de cada una, que explicamos detalladamente a continuación:

```
case INICIAL:
      mensaje = 'N';
      Serial.print(mensaje);
      sensorAnterior = valorSensor;
      baAnterior = valorBA;
      bbAnterior = valorBB;
      bcAnterior = valorBC;
      LeeEntrada();
      ComparaEntrada();
      ActualizaEstado();
      break;
```

En este primer *case*, como en todos los demás a excepción del caso *HOLD*, la primera línea que nos encontraremos será la que contiene la variable *mensaje*. En este caso, contiene el carácter 'N' y la estructura ***Serial.****print* se encarga de enviar al receptor el mensaje, en este caso mandará una 'N', lo que significa que el sistema no ha sufrido ningún cambio, ya sea por detección de sensor o por pulsación de alguno de los 3 botones.

Las variables *sensorAnterior, baAnterior, bbAnterior* y *bcAnterior* almacenan los valores que han registrado el sensor y los pulsadores. Una vez enviado el mensaje, éste queda obsoleto, por tanto hay que comprobar si ha cambiado o no. Para ello usamos la función auxiliar *LeeEntrada()* que hemos creado nosotros. Esta función guarda lo que estén diciendo en el momento el sensor y los pulsadores.

La función *ComparaEntrada* anota si alguno de los valores ha cambiado o si, por el contrario, siguen igual con lo que, finalmente, la función *ActualizaEstado* es la encargada de que, si alguno de los valores ha sufrido una variación, pase al estado correspondiente a ese cambio y si, por el contrario, no cambia, pasa al estado *HOLD* manteniéndose en éste hasta que se produzca algún cambio, ya sea por detección del sensor o por la pulsación de alguno de los tres pulsadores.

Veremos el funcionamiento detallado de estas funciones auxiliares en el apartado siguiente, ***5.4.6 Funciones Auxiliares***.

```
case HOLD:
      LeeEntrada();
      ComparaEntrada();
      ActualizaEstado();
      break;
```

Como podemos comprobar el caso *HOLD* no almacena datos anteriores del sensor y los pulsadores ya que sería redundante pues, al no producirse ningún cambio, almacenaría una y otra vez los mismos datos. Al no realizar esta acción nos queda el código más optimizado.

```
                case DETECCION:
                      mensaje = 'D';
                      Serial.print(mensaje);
                      sensorAnterior = valorSensor;
                      baAnterior = valorBA;
                      bbAnterior = valorBB;
                      bcAnterior = valorBC;
                      LeeEntrada();
                      ComparaEntrada();
                      ActualizaEstado();
                      break;

                case PULSADOA:
                      mensaje = 'A';
                      Serial.print(mensaje);
                      sensorAnterior = valorSensor;
                      baAnterior = valorBA;
                      bbAnterior = valorBB;
                      bcAnterior = valorBC;
                      LeeEntrada();
                      ComparaEntrada();
                      ActualizaEstado();
                      break;

                case PULSADOB:
                      mensaje = 'B';
                      Serial.print(mensaje);
                      sensorAnterior = valorSensor;
                      baAnterior = valorBA;
                      bbAnterior = valorBB;
                      bcAnterior = valorBC;
                      LeeEntrada();
                      ComparaEntrada();
                      ActualizaEstado();
                      break;

                case PULSADOC:
                      mensaje = 'C';
                      Serial.print(mensaje);
                      sensorAnterior = valorSensor;
                      baAnterior = valorBA;
                      bbAnterior = valorBB;
                      bcAnterior = valorBC;
                      LeeEntrada();
                      ComparaEntrada();
                      ActualizaEstado();
                      break;
          }
    }
```

Tanto el caso *DETECCION* perteneciente al sensor como los casos *PULSADOA*, *PULSADOB* y *PULSADOC*, que corresponden a los tres pulsadores A, B y C, respectivamente, tienen la misma estructura como ya explicamos anteriormente, con la única diferencia del mensaje que envía cada uno, teniendo asignado un carácter fácilmente reconocible.

5.4.6 Funciones Auxiliares

Para facilitar, optimizar y mejorar visualmente el código de nuestro programa, se han creado tres funciones auxiliares que serán fundamentales para el funcionamiento del módulo emisor. Las funciones las describimos a continuación junto a su código:

- **Función *LeeEntrada***

```
void LeeEntrada() {

      valorSensor = digitalRead(Sensor);
      valorBA = digitalRead(BotonA);
      valorBB = digitalRead(BotonB);
      valorBC = digitalRead(BotonC);
}
```

Como hemos comentado en el apartado anterior, esta función se encarga de guardar los valores tanto del sensor como de los tres pulsadores. Para almacenar estos valores, utilizamos la estructura *digitalRead*, que recoge los datos que recibe de los pines digitales de Arduino en los que están conectados el sensor y los pulsadores. Dichos datos quedan almacenados en las variables *valorSensor*, *valorBA*, *valorBB* y *valorBC*.

- **Función *ComparaEntrada***

```
void ComparaEntrada() {

      if (valorSensor != sensorAnterior || valorBA != baAnterior
      || valorBB != bbAnterior || valorBC != bcAnterior){

          cambiaEstado = true;
      }
      else {
          cambiaEstado = false;
      }
}
```

Esta función como su nombre bien indica, compara los valores que han sido leídos y almacenados con los guardados anteriormente de manera que, si se produce alguna variación, pone la variable *cambiaEstado* a true, indicando que se ha producido un cambio de estado. Si no se produce ninguna variación, es decir, los valores del sensor y los pulsadores son idénticos a los del estado anterior, seguimos en el mismo estado, *HOLD*.

- **Función *ActualizaEstado***

Por último, la función ActualizaEstado, es la encargada de establecer un determinado estado dependiendo de las condiciones que se produzcan.

```
void ActualizaEstado() {

      if (cambiaEstado == false){

                Estado = HOLD;
      }
```

La primera condición establece que, si no se produce ningún cambio, es decir, si la función ComparaEntrada no ha detectado diferencias en cada uno de los valores del sensor y los pulsadores, el estado en el que nos mantenemos es en el estado *HOLD*.

Si no se cumple esta premisa pasamos a la siguiente, de manera que si el sensor está en nivel alto ("1", o HIGH) se establece el estado *DETECCION*.

```
else if (valorSensor == 1) {

      Estado = DETECCION;
}
```

En el caso de no encontrarse tampoco en el estado anterior, el programa sigue su recorrido, por tanto llegamos a la condición de los tres botones. Para evitar casos indeseados, como la pulsación simultánea de dos o más botones, se ha creado un sistema que impedirá el funcionamiento incorrecto del control domótico:

1. Para que se dé el estado *PULSADOA* debe de cumplirse que la variable valorBA se encuentre a "0" y que las otras dos variables sean distintas de 0.

```
else if (valorBA == 0 && valorBB != 0 && valorBC != 0) {

      Estado = PULSADOA;
}
```

2. Para que se dé el estado *PULSADOB* debe de cumplirse que la variable valorBB se encuentre a "0" y que las otras dos variables sean distintas de 0.

```
else if (valorBB == 0 && valorBA != 0 && valorBC != 0) {

      Estado = PULSADOB;
}
```

3. Para que se dé el estado *PULSADOC* debe de cumplirse que la variable valorBC se encuentre a "0" y que las otras dos variables sean distintas de 0.

```
else if (valorBC == 0 && valorBA != 0 && valorBB != 0) {

      Estado = PULSADOC;
}
```

De esta manera, evitamos que, en el caso de que se produzcan pulsados simultáneos, haya desajustes. Además, con ello se establece la preferencia de pulsado, es decir, si se pulsa el botón B o C simultáneamente estando pulsado A, el sistema seguirá tomando como orden el pulsado de A.

```
      else {

            Estado = INICIAL;
      }
}
```

Por último, el estado *INICIAL* se dará en el resto de los casos o cuando se produzca alguna variación anómala en el sistema. El receptor estará preparado para que cuando reciba una 'N' no realice ninguna acción distinta a la que estaba haciendo anteriormente.

5.5 SOFTWARE MÓDULO RECEPTOR

5.5.1 Asignación de pines

Comenzamos nuestro programa receptor. En este primer apartado, explicamos a qué pines digitales de la plataforma Arduino están conectadas las dos luminarias.

Al igual que ocurría con los pines del módulo emisor, el código de asignación de pines se realiza mediante la función ***#define***, quedándonos la siguiente lista de comandos:

```
#define Foco 8
#define Leds 9
```

El foco y la tira de LEDs tienen adjudicado un número, el cual corresponde a cada uno de los pines digitales de la placa receptor de Arduino, en los que irán conectadas las patas positivas de cada uno de los dispositivos.

5.5.2 Estados

En el presente apartado identificamos los estados que se pueden dar en nuestro sistema, como ya hicimos en el apartado 5.4.2. Como podemos observar son prácticamente idénticos.

```
#define INICIAL 0
#define DETECCION 1
#define PULSADOA 2
#define PULSADOB 3
#define PULSADOC 4
```

En este caso, definimos cinco estados distintos:

- ***INICIAL*.** Estado que nos indica que el sistema se encuentra en arranque.
- ***DETECCION.*** Este estado hace referencia al momento en el que el sensor de presencia ha detectado movimiento.
- ***PULSADOA.*** Estado que indica que se ha pulsado el botón A.
- ***PULSADOB.*** Estado que indica que se ha pulsado el botón B.
- ***PULSADOC.*** Estado que indica que se ha pulsado el botón C.

5.5.3 Inicialización de variables

Los valores iniciales que establecemos para el módulo receptor son los siguientes:

```
int Estado = INICIAL;
int valorFoco = 0;
int valorLeds =0;
char mensaje;
unsigned long Retardo = 20;
int Paso = 5;
int Intensidad = 0;
```

La variable estado indica, al igual que en el módulo emisor, en que condición se encuentra el sistema en cada momento. Vuelve a estar establecido en el estado INICIAL. Las variables valorFoco y valorLeds, almacenan el valor tanto del foco como de la tira de LEDs, siendo 0 el estado de apagado y 1 el estado de encendido. Ambas luminarias están inicializadas en 0, o estado apagado.

La variable mensaje de tipo char (carácter) almacenará la información que recibe del emisor la cual le indica en qué estado se encuentra el sistema para el encendido o apagado de las luminarias.

La variable Retardo que nos servirá para poder apreciar el cambio del gradiente que realizará la tira de LEDs, lo establecemos en 20ms.

Por último, las variables paso e intensidad son dos variables que usaremos para el funcionamiento de la tira de LEDs, que veremos más detallado en apartado ***5.5.5 Loop***. Paso se establece a 5, que indica que el valor de intensidad de la tira de LEDs se incrementará y se decrementará luego de 5 puntos en 5 puntos para apreciar bien el gradado de intensidad. Por otra parte, intensidad se inicializa a 0, que será el punto de partida. Por tanto, para el gradado de intensidad, la tira empezará completamente apagada con intensidad= 0, e irá subiendo de 5 puntos en 5 puntos hasta su máximo (intensidad = 255) y, una vez alcanzada la máxima intensidad, volverá a decrecer de nuevo de 5 puntos en 5 puntos hasta volver a estar completamente apagada (intensidad= 0).

5.5.4 Setup

```
void setup() {

Serial.begin(9600);
pinMode(Foco, OUTPUT);  //Salida Foco Exterior
pinMode(Leds, OUTPUT);  //Salida Tira de LEDs
```

Como podemos apreciar, la asignación de pines en el módulo receptor es reducida, lo que nos facilitará el montaje del control domótico.

Nombre	Función	Pin de Arduino
Foco	Salida del Foco que hará que se encienda	8
Leds	Salida de la Tira LED que hará que se encienda	9

5.5.5 Loop

En el loop del código del módulo receptor, veremos las funciones que realiza el mismo para el encendido y apagado de las luminarias en cada caso que se produzca.

En primer lugar vemos que la estructura que usamos es la misma que hemos utilizado en el módulo emisor, la estructura switch. Ésta es usada para agilizar la toma de decisiones múltiples, trabajando de igual manera que sucesivos if, o if else, entre otros.

```
void loop() {

    switch (Estado) {
    ...
    ...
```

En el loop receptor, nos encontramos con los casos *INICIAL*, *DETECCION, PULSADOA, PULSADOB* y *PULSADOC*.

En esta ocasión, las estructuras switch case son distintas entre ellas, solo siendo similares las de los tres pulsadores. A continuación vemos su forma de trabajar:

```
case INICIAL:
      digitalWrite(Foco, valorFoco);
      digitalWrite(Leds, valorLeds);
      LeeEntrada();
      ActualizaEstado();
      break;
```

En este primer *case*, nos encontramos con la estructura que usaremos con frecuencia en el código del módulo receptor. Se trata de la estructura *digitalWrite*, la cual escribe un valor HIGH o LOW hacia un pin digital. En este caso, escribe el valor inicial del foco y de la tira LED.

Lo siguiente que realiza el caso INICIAL mediante la función *LeeEntrada* será guardar la información recibida del módulo emisor sobre lo que han dicho el sensor y los pulsadores.

Por último, la función *ActualizaEstado*, se encargará de que, según si los valores han sufrido cambios o no, pase a otro *case* o se quede en el presente, es decir, en el caso *INICAL*.

Como podemos observar a continuación, el caso DETECCION será el que experimente una mayor dificultad debido a la característica que le hemos asignado a la tira de LED.

```
case DETECCION:
      valorFoco = 1;
      digitalWrite(Foco, valorFoco);
```

Cuando entramos en este caso, es decir cuando el sensor detecta, en primer lugar el valor del foco exterior lo ponemos a '1' o 'HIGH', produciéndose el encendido del mismo mediante la estructura *digitalWrite*.

Acto seguido, se realiza el gradado de intensidad (tanto de subida como de bajada) en la tira de LEDs, mediante el empleo de dos bucles for seguidos.

```
      for(Intensidad = 0; Intensidad <= 255;
          Intensidad += Paso) {
            analogWrite(Leds, Intensidad);
            delay(Retardo);
}
      for(Intensidad = 255; Intensidad >= 0;
          Intensidad -= Paso) {
            analogWrite(Leds, Intensidad);
            delay(Retardo);
}
      valorFoco = 0;
      digitalWrite(Foco, valorFoco);
      LeeEntrada();
      ActualizaEstado();
      break;
```

Finalmente, apagamos el foco nada más acabar con el gradado de la tira de LEDs, y comprobamos de nuevo el estado actual con la función LeeEntrada(), y actualizamos el estado con la función ActualizaEstado(). De esta manera, el efecto visual que se apreciará en caso de que el sensor indique que está detectando presencia durante un tiempo prolongado, será un parpadeo continuo del foco, y unas subidas y bajadas continuas de intensidad de la tira de LEDs.

Seguidamente, establecemos los casos de los tres pulsadores cuyas tareas a realizar se describen a continuación:

```
case PULSADOA:
      valorFoco = 1;
```

```
                digitalWrite(Foco, valorFoco);
                LeeEntrada();
                ActualizaEstado();
                break;
```

En primer lugar tenemos el caso PULSADOA. En él, el valor del foco se establece en '1', o nivel alto, lo que quiere decir que el foco exterior se encenderá a través de digitalWrite.

Posteriormente, se realizan las tareas de actualización de variables mediante las funciones LeeEntrada y ActualizaEstado, descritas anteriormente.

```
            case PULSADOB:
                valorLeds = 1;
                digitalWrite(Leds, valorLeds);
                LeeEntrada();
                ActualizaEstado();
                break;
```

El segundo caso, PULSADOB, vemos que establece al contrario que en el caso anterior, un nivel alto o '1' al valor de la tira LED, por lo que *digitalWrite* realiza la tarea de encender la misma.

Seguidamente tenemos las funciones auxiliares de actualización de variables.

```
            case PULSADOC:
                valorFoco = 0;
                valorLeds = 0;
                digitalWrite(Foco, valorFoco);
                digitalWrite(Leds, valorLeds);
                LeeEntrada();
                ActualizaEstado();
                break;
        }
}
```

Por último, el caso PULSADOC, que se dará cuando el receptor reciba la información de que ha sido pulsado el botón C, será el encargado de apagar el foco y la tira LED. Para ello, se establece el valor de las dos variables del receptor, es decir, el foco y los LEDs, a un nivel bajo o '0', de manera que digitalWrite lo transmitirá a los pines correspondientes a las dos luminarias.

Una vez más, el caso realiza la actualización de variables con las funciones auxiliares explicadas con anterioridad y que veremos más detalladamente en el siguiente apartado.

5.5.6 Funciones Auxiliares

Al igual que en módulo emisor, recopilamos una serie de funciones auxiliares en el programa receptor para obtener un código más limpio y fácil de entender. En el módulo receptor veremos las funciones presentes en el módulo emisor, a excepción de la función ***ComparaEntrada()*** la cual no necesitamos en el desarrollo de dicho código.

- **Función *LeeEntrada***

```
void LeeEntrada() {

        if (Serial.available() > 0) {
                mensaje = Serial.read();
        }
}
```

En esta primera función auxiliar, **Serial**.available() devuelve el número de bytes (caracteres) disponibles para ser leídos por el puerto serie. Se refiere a datos ya recibidos y disponibles en el buffer de recepción del puerto. Con nuestra estructura if, si se cumple la condición de que **Serial.**available() es mayor que 0, **Serial.**read() leerá dichos datos entrantes y los almacenaremos en la variable mensaje.

- **Función *ActualizaEstado***

  Para finalizar, describimos la función auxiliar ActualizaEstado, que se encargará de establecer un determinado estado dependiendo del mensaje recibido del módulo emisor.

```
void ActualizaEstado() {


      if (mensaje == 'N') {

            Estado = INICIAL;
      }

      else if (mensaje == 'D') {

            Estado = DETECCION;
      }

      else if (mensaje == 'A') {

            Estado = PULSADOA;
      }

      else if (mensaje == 'B') {

            Estado = PULSADOB;
      }

      else if (mensaje == 'C') {

            Estado = PULSADOC;
      }

      else {

            Estado = INICIAL;
      }
}
```

Como podemos observar, en cada if se comprueba si el mensaje recibido corresponde al carácter asignado a cada estado para, en el caso de recibir alguno de dichos caracteres, establecer su estado correspondiente.

Así, por orden descendente, nuestra función auxiliar comprobará si coincide cada mensaje con los caracteres '*N*', '*D*', '*A*', '*B*', '*C*', asignados a los estados, *INICIAL*, *DETECCION*, *PULSADOA*, *PULSADOB* Y *PULSADOC*, respectivamente. En caso de no darse ninguno de estos caracteres o producirse cualquier error inesperado, el sistema volverá al estado *INICIO* y a la espera de un nuevo cambio.

CAPÍTULO 6

CONCLUSIONES Y TRABAJOS FUTUROS

En este último capítulo se expondrán las conclusiones que se obtienen tras la elaboración del presente proyecto. Se explicarán los problemas más relevantes que han ido surgiendo en la elaboración del control domótico y la forma de solventarlos. A continuación se realizará una valoración a título personal sobre lo que supone al autor el desarrollo y elaboración del proyecto. Por último se trazarán aquellas posibles líneas en las que se puede orientar la elaboración de futuros trabajos y proyectos relacionados con el mundo de la domótica y en particular con el proyecto expuesto en esta memoria.

6.1 CONCLUSIONES

La primera y más importante conclusión que se puede extraer a la finalización de este proyecto de fin de carrera que culmina los estudios en la titulación de Ingeniero Técnico Industrial especialidad en Electrónica Industrial de la autora, es el éxito y la satisfacción por los resultados obtenidos tras la cantidad de horas de trabajo y el esfuerzo necesarios para desarrollar el proyecto.

A pesar de ser un proyecto con bastante información disponible en la red para llevarlo a cabo, han ido surgiendo una serie de inconvenientes y problemas a lo largo del desarrollo del control domótico.

Recordemos que la finalidad de nuestro proyecto es realizar un control domótico de manera que la comunicación sea inalámbrica, por tanto se tuvo que realizar una comparativa de sistemas inalámbricos para ver cuál de ellos era más conveniente. Así mismo, varios contratiempos en programación y montaje de hardware dificultaron en determinados momentos el desarrollo de nuestro proyecto.

Sin embargo, a base de dedicación los problemas se fueron solventando y a pesar de que se dedicaban muchas horas semanales a la elaboración del proyecto, la ilusión por conseguir un gran objetivo y el sueño de convertirse en ingeniero eran estímulos de grandes dimensiones para obtener ánimos.

A destacar en el desarrollo del proyecto será la satisfacción al comprobar de forma práctica que se está realizando el diseño de un control domótico y al conectarlo a la red eléctrica funciona perfectamente, realizando todas las tareas que se le han programado. Ya que la teoría no siempre se cumple, la comprobación práctica no ofrece duda alguna del correcto funcionamiento del sistema.

A la conclusión del proyecto, valoro el enriquecimiento en conocimientos relacionados con la programación, el control y sobre todo la tarea de hardware ya que siempre han sido los campos que más me han agradado. También he obtenido muchos conocimientos relacionados con el mundo de la domótica, y es un mundo que realmente tiene un gran futuro para la

comodidad, seguridad y bienestar de las personas en sus hogares y/o entornos de trabajo y con grandes expectativas de futuros proyectos a desarrollar y mejorar. Me llevo la sensación de que una idea siempre es posible de realizar, que si te propones algo y le dedicas tiempo y esfuerzo para conseguirlo, el resultado obtenido será satisfactorio.

6.2 TRABAJOS FUTUROS

La elaboración de este proyecto culmina con una sensación de hablar cumplido los objetivos del mismo satisfactoriamente. Se ha desarrollado un sistema de control domótico que utiliza la tecnología ZigBee para su funcionamiento de manera inalámbrica.

Teniendo en cuenta el desarrollo de este sistema de control, se pueden realizar mejoras en el funcionamiento del mismo, así como nuevos proyectos para el control domótico exterior e interior.

Por una parte, si nos centramos en la comunicación, no cabe duda de que la inalámbrica es el gran paso hacia el futuro, pues con ella nos damos cuenta de que se reducen gastos en cableado así como las dificultades que éste conlleva. Como hemos comprobado en la elaboración de nuestro proyecto, la tecnología ZigBee es ampliamente apta para el control domótico por su bajo coste y bajo consumo de potencia que alarga la vida de las baterías.

De cara al futuro nos encontramos con otras tecnologías como es la Ultra-Wideband (UWB). Ésta tecnología, sin embargo, a pesar de no encontrarse aún disponible comercialmente y con la falta de estándares globales más allá del básico IEEE 802.15.3, promete ser la tecnología inalámbrica por excelencia para la transmisión de datos y contenido multimedia a altas velocidades, en el orden de varios cientos de Mbps, con garantías de calidad de servicio.

Además, teniendo en cuenta las peculiaridades de la señal en algunas implementaciones, una aplicación de interés en el hogar puede ser la localización en interiores, aplicable a cuestiones de teleasistencia y seguridad, por ejemplo, detectando en qué estancia se encuentra una persona o si tal vez se ha caído al suelo.

Por otro lado, si nos plantamos de cara al usuario, la idea es que al tiempo de que la domótica facilita las cosas, el propio usuario se vea integrado en el proceso de control de manera que establezca sus preferencias de manera sencilla. En nuestro proyecto, hemos cubierto este frente con el control manual, sin embargo, éste se encontrará en una ubicación física permanente.

Por tanto, una idea en la que podemos ver mejorado lo anterior sería mediante el control manual a través de dispositivos móviles, de manera que el usuario controle a distancia su sistema de control domótico así como establecer preferencias, modos de encendido o apagado, etc.

Como podemos ver se pueden sacar infinidad de mejoras e ideas para desarrollar en el campo de la domótica y en el acondicionamiento interior o exterior, ya sea por motivos de confort o de seguridad y sacar cantidad de nuevas posibilidades, cuyo descubrimiento depende de la creatividad y la imaginación de aquellas personas, como la autora, atraídas por este mundo.

BIBLIOGRAFÍA Y REFERENCIAS

Listado de referencias bibliográficas:

[1] "Estado del Arte de la Domótica en Colombia", Aguirre, J. & Zapata, O., 2006.

[2] "Fundamentos de Robótica", Antonio Barrientos y otros, 2007.

[3] "Electrónica", Hambley, A., Editorial Pearson, 2001.

[4] "Diseño de una casa Inteligente con basado en la Tecnología JINI", Arias. 2004.

[5] "Desarrollo de Sistemas Domóticos utilizando un enfoque dirigido por modelos", Jiménez Buendía, Cartagena, 2009.

[6] "Microcontroladores PIC. Programación en basic", Reyes, C.A., 2006.

[7] "Arduino Notebook: A Beginner´s Reference." Evans, B. W., San Francisco, CA, 2007.

[8] "Electrónica: Teoría de Circuitos y Dispositivos Electrónicos." Boylestad, R., Prentice Hall, 2002.

[9] *"*Getting Started with Arduino*"*, Massimo Banzi, Editorial O'Reilly Media/Make, 2008.

[10] "Arduino Cookbook*"*, Michael Margolis, Editorial O'Reilly Media, 2011.

[11] "Fundamentos de informática. Programación en C", Pedro María Alcover Garau, Universidad Politécnica de Cartagena, 2006.

Listado de referencias a artículos:

[12] Soltic, S., & Chalmers, A. "Differential evolution for the optimisation of multi-band shite LED light sources." 2012 Lighting Res. Rechnol. P.224 – 237.

[13] Huidobro, J. M., "La domótica como solución de futuro", 2007, Madrid. P. 15-17

[15] Kushner, D., "The making of the Arduino", 2011. IEEE Spectrum.

Listado de referencias a direcciones URL:

[16] Página web de tutoriales de Arduino.
http://arduino.cc/en/Tutorial/HomePage

[17] Página web Cooking Hacks con tutoriales de XBee Shield.
http://www.cooking-hacks.com/index.php/documentation/tutorials/arduino-xbee-shield

[18] Página web de Digi con información sobre XBee.
http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rfmodules/point-multipoint-rfmodules/xbee-series1-module#overview

[19] Página web de configuración del módulo Xbee
http://www.jorts.net/images/2/25/Tutorial_XBee_Neira.pdf

[20] Página web de datasheets.
www.alldatasheet.com

[21] Página web de Wikipedia.
http://es.wikipedia.org

[22] Página web de uControl.
http://www.ucontrol.com.ar

[23] Página web de información sobre sensores.
http://www.ladyada.net/learn/sensors/index.html

[24] Página web de la IEEE sobre el estándar 802.15.4.
http://standards.ieee.org/about/get/802/802.15.html

ANEXO I

SOFTWARE DESARROLLADO

A1.1 SOFTWARE MÓDULO EMISOR

```
      //PROGRAMA FINAL PARA MÓDULO EMISOR

//------------------------ASIGNACION DE PINES-------------------------
#define Sensor 8
#define BotonA 9
#define BotonB 10
#define BotonC 11
//-------------------------------ESTADOS-------------------------------
#define INICIAL 0
#define DETECCION 1
#define PULSADOA 2
#define PULSADOB 3
#define PULSADOC 4
#define HOLD 5
//--------------------INICIALIZACION DE VARIABLES--------------------
int Estado = INICIAL;
int valorSensor = 0;
int valorBA = 1;
int valorBB = 1;
int valorBC = 1;
int sensorAnterior = 0;
int baAnterior = 0;
int bbAnterior = 0;
int bcAnterior = 0;
boolean cambiaEstado = false;
```

```
char mensaje;

//--------------------------------SETUP---------------------------------
void setup() {

      Serial.begin(9600);
        pinMode(Sensor, INPUT);
        pinMode(BotonA, INPUT);
        pinMode(BotonB, INPUT);
        pinMode(BotonC, INPUT);
}
//--------------------------------LOOP----------------------------------
void loop() {
      switch (Estado) {
            case INICIAL:
                  mensaje = 'N';
                  Serial.print(mensaje);
                        sensorAnterior = valorSensor;
                        baAnterior = valorBA;
                        bbAnterior = valorBB;
                        bcAnterior = valorBC;
                  LeeEntrada();
                        ComparaEntrada();
                  ActualizaEstado();
                  break;
            case HOLD:
                        baAnterior = valorBA;
                        bbAnterior = valorBB;
                        bcAnterior = valorBC;
                        LeeEntrada();
                        ComparaEntrada();
                        ActualizaEstado();
                        break;
            case DETECCION:
                  mensaje = 'D';
                  Serial.print(mensaje);
                        sensorAnterior = valorSensor;
                        baAnterior = valorBA;
                        bbAnterior = valorBB;
                        bcAnterior = valorBC;
```

```
                LeeEntrada();
                        ComparaEntrada();
                ActualizaEstado();
                break;
            case PULSADOA:
                mensaje = 'A';
                Serial.print(mensaje);
                        sensorAnterior = valorSensor;
                        baAnterior = valorBA;
                        bbAnterior = valorBB;
                        bcAnterior = valorBC;
                LeeEntrada();
                        ComparaEntrada();
                ActualizaEstado();
                break;
            case PULSADOB:
                mensaje = 'B';
                Serial.print(mensaje);
                        sensorAnterior = valorSensor;
                        baAnterior = valorBA;
                        bbAnterior = valorBB;
                        bcAnterior = valorBC;
                LeeEntrada();
                        ComparaEntrada();
                ActualizaEstado();
                break;
            case PULSADOC:
                mensaje = 'C';
                Serial.print(mensaje);
                        sensorAnterior = valorSensor;
                        baAnterior = valorBA;
                        bbAnterior = valorBB;
                        bcAnterior = valorBC;
                LeeEntrada();
                        ComparaEntrada();
                ActualizaEstado();
                break;
        }
}
```

```
void LeeEntrada() {

      valorSensor = digitalRead(Sensor);
      valorBA = digitalRead(BotonA);
      valorBB = digitalRead(BotonB);
      valorBC = digitalRead(BotonC);
}

void ComparaEntrada() {

      if (valorSensor != sensorAnterior || valorBA != baAnterior ||
valorBB != bbAnterior || valorBC != bcAnterior){

                cambiaEstado = true;
        }
        else {
                cambiaEstado = false;
        }
}

void ActualizaEstado() {
        if (cambiaEstado == false){
                Estado = HOLD;
        }
        else if (valorSensor == 1) {
            Estado = DETECCION;
      }
      else if (valorBA == 0 && valorBB != 0 && valorBC != 0) {
            Estado = PULSADOA;
      }
      else if (valorBB == 0 && valorBA != 0 && valorBC != 0) {
            Estado = PULSADOB;
      }
      else if (valorBC == 0 && valorBA != 0 && valorBB != 0) {
            Estado = PULSADOC;
      }
      else {
            Estado = INICIAL;
      }
}
```

A1.2 SOFTWARE MÓDULO RECEPTOR

```
      //PROGRAMA FINAL PARA MÓDULO RECEPTOR


//------------------------ASIGNACION DE PINES------------------------
#define Foco 8
#define Leds 9
//-----------------------------ESTADOS-------------------------------
#define INICIAL 0
#define DETECCION 1
#define PULSADOA 2
#define PULSADOB 3
#define PULSADOC 4
//--------------------INICIALIZACION DE VARIABLES--------------------
int Estado = INICIAL;
int valorFoco = 0;
int valorLeds =0;
char mensaje;
unsigned long Retardo = 20;
int Paso = 5;
int Intensidad = 0;
//--------------------------------SETUP---------------------------------
void setup() {
      Serial.begin(9600);
      pinMode(Foco, OUTPUT);
      pinMode(Leds, OUTPUT);
}
//--------------------------------LOOP---------------------------------
void loop() {

      switch (Estado) {

            case INICIAL:
                  digitalWrite(Foco, valorFoco);
                  digitalWrite(Leds, valorLeds);
                  LeeEntrada();
                  ActualizaEstado();
                  break;
            case DETECCION:
```

```
                valorFoco = 1;
                digitalWrite(Foco, valorFoco);
                for(Intensidad = 0; Intensidad <= 255; Intensidad +=
Paso) {
                      analogWrite(Leds, Intensidad);
                      delay(Retardo);
                }
                for(Intensidad = 255; Intensidad >= 0; Intensidad -=
Paso) {
                      analogWrite(Leds, Intensidad);
                      delay(Retardo);
                }
                valorFoco = 0;
                digitalWrite(Foco, valorFoco);
                LeeEntrada();
                ActualizaEstado();
                break;
          case PULSADOA:
                valorFoco = 1;
                digitalWrite(Foco, valorFoco);
                digitalWrite(Leds, valorLeds);
                LeeEntrada();
                ActualizaEstado();
                break;
          case PULSADOB:
                valorLeds = 1;
                digitalWrite(Foco, valorFoco);
                digitalWrite(Leds, valorLeds);
                LeeEntrada();
                ActualizaEstado();
                break;
          case PULSADOC:
                valorFoco = 0;
                valorLeds = 0;
                digitalWrite(Foco, valorFoco);
                digitalWrite(Leds, valorLeds);
                LeeEntrada();
                ActualizaEstado();
                break;
      }
}
```

```
//-----------------------FUNCIONES AUXILIARES-------------------------
void LeeEntrada() {

      if (Serial.available() > 0) {

            mensaje = Serial.read();
      }
}
//--------------------------------------------------------------------
void ActualizaEstado() {

      if (mensaje == 'N') {

            Estado = INICIAL;
      }
      else if (mensaje == 'D') {

            Estado = DETECCION;
      }
      else if (mensaje == 'A') {

            Estado = PULSADOA;
      }
      else if (mensaje == 'B') {

            Estado = PULSADOB;
      }
      else if (mensaje == 'C') {

            Estado = PULSADOC;
      }
      else {

            Estado = INICIAL;
      }
}
```