Universidad Zaragoza
1542

Trabajo Fin de Máster

Monitorización de temperatura en pacientes de una sala UCI con tecnología Li-Fi.

Temperature monitoring in ICU-room patients with Li-Fi technology.

Autor

Maria Jesús Pérez Saiz

Director

Arturo Mediano Heredia

Escuela de Ingeniería y Arquitectura

2021

Resumen

En la actualidad las capacidades de las tecnologías de acceso a Internet están al limite de su capacidad por la gran saturación que presenta el espectro radioeléctrico. Para resolver este problema, este trabajo pretende analizar las ventajas que supone utilizar la comunicación por luz visible en un entorno sanitario, más concretamente en un área de altos riesgos como es una sala de cuidados intensivos (UCI) mediante el diseño de un prototipo que monitoriza la temperatura de varios pacientes. El objetivo principal de este trabajo es poner en funcionamiento un enlace Li-Fi para la transmisión de información mostrando que no es afectado por las interferencias electromagnéticas y es seguro su uso en un área crítica de un hospital. Además, se realizará el mismo prototipo con un sistema Bluetooth para así observar claramente como si le afectan las interferencias frente al sistema estudiado.

ABSTRACT

At present, the capabilities of Internet access technologies are at the limit of their capacity due to the great saturation of the radioelectric spectrum. To solve this problem, this work aims to analyze the advantages of using visible light communication in a healthcare environment, more specifically in a high-risk area such as an intensive care room (ICU) by designing a prototype that monitors the temperature of various patients. The main objective of this work is to put into operation a Li-Fi link for the transmission of information showing that it is not affected by electromagnetic interference and its use is safe in a critical area of a hospital. In addition, the same prototype will be made with a Bluetooth system in order to clearly observe how interferences affect the system under study.

ÍNDICE

CAPÍTULO 1. INTRODUCCIÓN Y OBJETIVOS.

1.1 INTRODUCCIÓN Y MOTIVACIÓN.

La continua evolución de las tecnologías de la información y la comunicación (TIC) ha supuesto una revolución en los últimos años transformando aspectos sociales y económicos en nuestra sociedad que se han traducido en un desarrollo y expansión de las telecomunicaciones. Internet ha sido el desencadenante de esta nueva era tecnológica, eliminando cualquier tipo de barrera fronteriza. Actualmente, el flujo de datos aumenta de forma exponencial gracias al creciente número de personas que tienen acceso a las nuevas tecnologías.

Los avances tecnológicos aumentan la eficiencia en los diferentes sectores económicos conllevando consigo una digitalización del mundo empresarial. El Índice de Economía y Sociedad Digital (DESI) situaba a España en el undécimo puesto de los países miembros de la Unión Europea (UE) en el año 2019. Esto supone que nos encontramos por encima de la media europea en materia de competitividad digital teniendo en cuenta indicadores tales como el uso de Internet, la conectividad y/o la integración tecnológica digital.

El sector sanitario no ha sido una excepción. El uso de las nuevas tecnologías, con la incorporación de un elevado número de dispositivos IoT (Internet Of Things), se ha visto condicionada por la red corporativa. Muchos de estos dispositivos ya llevan incorporados tecnología Wi-Fi; es por ello, que los departamentos de tecnología de la información (TI) de este sector tienen que dar servicio a una gran cantidad de equipos médicos dentro de la red WLAN (Wireless Local Área Network).

La movilidad constante y la tecnología ubicua en el entorno laboral supone un reto en la búsqueda de nuevas alternativas a la arquitectura tradicional constituida por un componente más estático empleado para dar servicio a dispositivos dentro de la red LAN (Local Área Network), entorno que poco tiene que ver con el actual. La transformación digital tiene que converger hacia una red más flexible a la vez que más segura.

La Sociedad Española de Informática de la Salud (SEIS) en su informe Índice SEIS 2019 nos muestra como el Sistema Nacional de Salud (SNS) ha sido uno de los sectores que mayores recortes ha sufrido en nuestro país y de su presupuesto sanitario, un promedio de sólo un 1,16% fue destinado a TIC en las diferentes Comunidades Autónomas (CCAA) [1].

La búsqueda de la infraestructura idónea que soporte los nuevos equipamientos debe disponer de la capacidad de habilitar de una mayor movilidad al personal en el entorno hospitalario garantizando una conectividad segura y que no sufra interrupciones en ninguna de las áreas que abarca. La red, por tanto, se ha convertido en un elemento clave en esta era de la transformación digital.

El mayor inconveniente en las redes inalámbricas y concretamente en la red Wi-Fi es la superposición de frecuencia y las interferencias que se derivan de las mismas. El creciente número de dispositivos móviles acrecienta este problema porque la mayoría de éstos utilizan una banda de frecuencia de 2,4 GHz determinada en el estándar IEEE 802.11. En el caso de los dispositivos utilizados en el ámbito hospitalario, aunque a priori no utilicen la misma banda de frecuencia, su uso simultáneo hace que se produzcan emisiones electromagnéticas. Cuantos más dispositivos estén funcionando, mayor es la utilización del espectro de radiofrecuencia (abreviado RF) que genera de este modo una interferencia electromagnética (EMI) de manera simultánea que resulta en determinadas situaciones altamente peligrosas.

En el entorno hospitalario hay que prestar especialmente atención a las áreas de cuidados intensivos (UCI), donde la conexión Wi-Fi está limitada debido al alto número de interferencias por radiofrecuencia. Otro factor clave es la seguridad asociada a la red pues ésta es fácilmente vulnerable además del tráfico de datos que se almacenan y se transmiten, los cuales requieren de un mayor ancho de banda.

Consecuentemente, surge la necesidad de desarrollar nuevas infraestructuras de transmisión de la información en las que tiene que estar presente la optimización de recursos y la conservación del medio ambiente. Teniendo en cuenta que el sistema debe de ser robusto, libre de fallos de transmisión y con gran capacidad de almacenamiento y transmisión.

Una solución sería la implantación de un sistema Li-Fi de comunicación que transmite datos vía luz que propone el físico alemán Harald Haas, siendo de este modo una alternativa que se da para la conexión Wi-Fi, es decir, se pretende conseguir una dualidad mediante una lámpara, esto es, iluminación e información. Esta tecnología es novedosa en cuanto a la cobertura inalámbrica de datos de alta densidad y se prevé que en un corto plazo sea un serio competidor.

El problema de trabajar en paralelo con varios sistemas EMI que se veía cuando se utilizaba un sistema Wi-Fi es factible con Li-Fi y resulta ventajoso para cirugías robóticas y procedimientos automatizados. En una cirugía que se utilice un sistema Li-Fi con varios sensores se obtiene la orientación inmediata de los expertos compartiendo datos, videos o detalles en tiempo real sobre

el paciente obteniendo así los mejores resultados. Por tanto, la tecnología Li-Fi mejora el campo médico y tiene una gran cantidad de méritos cuando se instala y se utiliza de manera beneficiosa.

Por último, cabe destacar que una de las motivaciones para desempeñar este trabajo ha sido la actual crisis sanitaría que se está viviendo. Hablo de primera mano ya que yo fui uno de los primeros casos Covid-19 y pude comprobar lo caótico que fueron esos días, así como lo importante que es estar controlado en una situación tan crítica, ya no sólo dentro del Hospital sino todo el seguimiento que hay después que hoy en día continúa dificultando la vida diaria. Por lo tanto, tener una tecnología que descongestione el espectro y mejore el campo médico es de especial interés.

1.2 OBJETIVOS.

El objetivo principal de este proyecto se basa en el estudio de la tecnología Li-Fi en un entorno hospitalario destacando las ventajas de su utilización en entornos críticos como son las áreas de cuidados intensivos (UCI) de un Hospital debido a que es una zona altamente sensible a las interferencias por radiofrecuencia. Para ello, se diseñará un prototipo experimental que permita la transmisión de datos de diversas temperaturas, que corresponden a las temperaturas de los pacientes que se encuentran en una UCI. Se utilizarán varios emisores de luz, tantos como sensores de temperatura y un fotorreceptor para la transmisión de dicha información bajo la tecnología Li-Fi utilizando un hardware libre.

Del mismo modo, se realizará el mismo prototipo, pero con módulos Bluetooth para realizar la comunicación y de esta forma poder hacer una comparativa y ver las características de cada tecnología.

Para llevar a cabo este objetivo es fundamental realizar un estudio previo de la tecnología Li-Fi. De esta forma se podrá realizar un análisis comparativo con el Wi-Fi y el Bluetooth, para ver en mayor detalle los beneficios que tiene la tecnología propuesta. Por otro lado, se debe elaborar las especificaciones y el diagrama de bloques que va a regir el prototipo para posteriormente diseñarlo con el fin de demostrar la funcionabilidad de la tecnología alternativa que se propone en este proyecto. Por último, se realizará la puesta a punto y se harán diversas pruebas de funcionamiento del prototipo.

CAPÍTULO 2. LA TECNOLOGÍA LI-FI.

En este capítulo se sustentarán los puntos clave para llevar a cabo el proyecto de la tecnología Li-Fi. Para ello, se va a introducir la tecnología inalámbrica actual y lo que son las comunicaciones por luz visible (*Visual Light Communications* o sus siglas VLC), explicando sus orígenes y sus características. A su vez, se profundizará en la tecnología Bluetooth para realizar una comparativa con la tecnología Li-Fi.

Por último, la descripción se centrará en los sistemas Li-Fi (*Light Fidelity*, o del inglés Fidelidad de la Luz) y las características que esta tecnología posee y su línea futura. Además, se explicará en detalle la red inalámbrica actual de los hospitales y los beneficios que supone implementar esta tecnología en zonas de riesgo como es el sector sanitario.

2.1. INTRODUCCIÓN A LA TECNOLOGÍA INALÁMBRICA. LA TECNOLOGÍA WI-FI Y BLUETOOTH.

Con los años se ha recurrido a muchas tecnologías inalámbricas para la transmisión de datos, las cuales son elegidas de acuerdo con la aplicación que se necesite en cada momento y el ancho de banda que pueden ofrecer. Las más utilizadas actualmente son el Bluetooth y el Wi-Fi.

El término Bluetooth se refiere a un protocolo de comunicaciones que sirve para la transmisión inalámbrica de datos y voz a través de ondas de radio que operan en la banda ISM de los 2,4 GHz. Se trata de bandas no comerciales que tienen un uso industrial, médico y científico (ISM), por lo tanto, su uso en los entornos sanitarios debe de estar controlado. A pesar de que el Bluetooth está diseñado para distancias de corto alcance, con el Bluetooth 5.0 consigue alcanzar distancias de unos 200 metros a velocidades de hasta 2 Mbps.

El Li-Fi, a diferencia del Bluetooth, es capaz de transportar todo tipo de información debido a las grandes velocidades que puede alcanzar (hasta de 10 Gbps) y también está diseñado para distancias de corto alcance.

En la actualidad, la tecnología Bluetooth está presente en prácticamente cualquier dispositivo siendo uno de los sistemas de comunicaciones más cómodo y práctico para transferir datos. Eso se manifiesta actualmente en el desarrollo de sensores electrónicos Bluetooth a raíz de la crisis sanitaria, que permiten regular la temperatura y la monitorización de las vacunas de Covid-19

[2]. Del mismo modo, para hacer un seguimiento de los contagios se creó un sistema basado en Bluetooth que permitiendo así hacer un rastreo, pero no tuvo mucho éxito la aplicación [3].

Por otro lado, el Wi-Fi cuyo nombre procede de la organización comercial Wi-Fi Alliance, la cual adopta, prueba y certifica que los equipos cumplen los estándares relacionados con las redes inalámbricas de área local (*Local Area Network*, LAN) [4]. Esta tecnología utiliza la banda de 2.4 GHz y la de 5 GHz del espectro electromagnético, todo ello está regulado por el IEEE *(Institute of Electrical and Electronics Engineers* o en español Instituto de Ingeniería Eléctrica y Electrónica*)*, en concreto en los estándares 802.11. Siguiendo dichos estándares, se alcanzan velocidades de 433 Mbps, aunque puede llegar alcanzar velocidades de hasta 10 Gbps [5]. A diferencia de la tecnología Bluetooth, la tecnología Wi-Fi puede ofrecer una cobertura de 100 m de radio en campo abierto sin obstáculos, pero como cualquier sistema de radiofrecuencia se ve afectada por las interferencias electromagnéticas (EMI) y la seguridad entre otros muchos problemas.

A pesar de que estas tecnologías están ampliamente extendidas, ha surgido la necesidad de explorar nuevas alternativas para transmitir información de forma inalámbrica y más eficiente. De esta forma, nace una nueva tecnología en redes inalámbricas llamada Li-Fi.

2.2. LA TECNOLOGÍA VLC: PRECURSORA DEL LI-FI. MARCO HISTÓRICO.

El Li-Fi es una tecnología basada en VLC que utiliza dispositivos LED (*Light-Emitting Diode*) para la transmisión de datos de forma inalámbrica siendo su principal propósito la iluminación. Tanto la tecnología VLC como el Li-Fi pertenecen a un subconjunto de tecnologías llamadas OWC (*Optical Wireless Communications* o "Comunicaciones Ópticas Inalámbricas"), las cuales son reguladas por el estándar IEEE 802.15.7.

La línea de investigación sobre las comunicaciones por medio de luz se remonta años atrás, aunque basada en LEDs solo desde el siglo XXI. El primer equipo de comunicación inalámbrica sofisticado fue inventado por Alexander Graham Bell en 1880, el fotófono, que en vez de utilizar ondas de radio empleaba la luz para la transmisión de la voz. Este invento no se tomó como ejemplo a la hora de crear el teléfono móvil, pero si gesto las bases de la fibra óptica [6].

Desarrollar componentes y tecnologías para la utilización de un VLC de alta velocidad de datos llevo más de un siglo. Las primeras versiones utilizaban lámparas fluorescentes con las que

alcanzaban pequeñas velocidades (de unos pocos Kbps), pero con el tiempo se sustituyeron por LED, que permitió multiplicar las velocidades finales gracias a que la intensidad de luz que emana de ellos puede ser modulada a velocidades extremamente altas donde el ojo humano no las percibe y solo pueden ser percibidas con ayuda de fotodetectores [7].

Son muchas las instituciones que estudian el tema de las comunicaciones por luz visible, pero la presentación pública más conocida de VLC tuvo lugar el 15 de septiembre del 2011 cuando el profesor Harald Hass demostró el trabajo realizado con ayuda de su equipo: *"Wireless data from every light bulb"* en la conferencia de la Tecnología, Entretenimiento y Diseño (TED). Mediante la cual mostro el primer dispositivo que denominaba Li-Fi, enviando una señal de video a 10 Mbps con una sola bombilla LED [8]. Hass comentó que en poco tiempo podría incrementar esa velocidad hasta los 500 Mbps.

En España también se están dando avances sobre este tipo de comunicaciones través del IDeTIC (Instituto para el Desarrollo Tecnológico y la Innovación en Comunicaciones) en su proyecto BALDUR, basado en conectar un monitor táctil a una red de datos convencional a través de un enlace VLC en un entorno de operación como puede ser el caso de los hospitales, donde existen zonas donde hay serias limitaciones al uso de enlaces convencionales de radiofrecuencia [9].

2.3. LI-FI.

Li-Fi es una tecnología de comunicación inalámbrica de alta velocidad, y al igual que la tecnología VLC, se basa principalmente en un LED y un fotodiodo con una buena respuesta a la región de longitud de onda visible como receptor. Aunque a diferencia de VLC, el Li-Fi ofrece conectividad bidireccional y multiusuario, es decir, comunicaciones punto a multipunto y multipunto a punto. En nuestro sistema se llevará a cabo una comunicación multipunto a punto.

Al ser una comunicación por luz visible comparte muchas ventajas con VLC. Entre las multitudes de ventajas que posee esta tecnología se pueden destacar la velocidad, las interferencias y la seguridad.

La disponibilidad que posee el Li-Fi es mucho mayor que otras tecnologías debido a que gracias a las bombillas LED se desempeña una doble funcionalidad, mientras que estas obteniendo iluminación se transmiten datos sin ningún coste adicional como sucede con las

comunicaciones por radiofrecuencia. Esto ayuda a reducir la arquitectura de coste para un punto de acceso y del mismo modo a la naturaleza, ya que las bombillas LED tienden a ser más ecológicas que las bombillas convencionales porque no contienen mercurio ni otros elementos perjudiciales para el medio ambiente.

Entre las ventajas más destacables se da que las EMI pasan a un segundo plano ya que los sistemas Li-Fi tienen una interferencia nula de radiofrecuencia con otros dispositivos. Y esto es debido a que esta tecnología puede usar todo el espectro de luz visible, garantizando así un ancho de banda 10.000 veces más grande que el utilizado en las tecnologías de radiofrecuencia. De esta forma, se pueden alcanzar velocidades de transferencia de datos mucho más altas. Por esta razón, es una gran solución que se debe de tener en cuenta para las zonas críticas de los hospitales como sucede en Aviano (Italia), donde se está llevando a cabo el proyecto AAL X AAL en el Centro de Referencia Oncológica (Oncological Refernce Center con siglas CRO) para llevar la conexión Li-Fi al sector sanitario ya que no produce contaminación electromagnética que puede causar implicaciones altamente problemáticas con maquinaria que salva vidas [10]. El CRO National Cancer Institute es un instituto público sin fines de lucro que opera bajo la autoridad del Ministerio de Salud italiano para mejorar la salud publica mediante el avance del conocimiento médico. Además de ofrecer una mejor eficiencia operativa, Li-Fi proporcionó una supervisión avanzada dónde los pacientes pueden acceder a los datos al tiempo que se mejora la experiencia de los pacientes.

Un aspecto que a simple vista podría parecer un inconveniente resulta una ventaja y se trata de que la luz no traspasa paredes y esto hace que se vuelva una tecnología más segura que el Wi-Fi. Con el Wi-Fi pueden interceptar la red y vulnerarla mientras con el Li-Fi el atacante necesita tener acceso a la luz, es decir, tiene que estar iluminado con la misma luz de aquellos que quiera vulnerar.

El principal inconveniente que presenta esta tecnología es el corto alcance, entre los cinco y diez metros, lo que implica que la sala debe de estar muy bien iluminada. A esto, hay que sumarle que para se produzca la comunicación los LED siempre deben de estar encendidos. Si no hay luz, no hay conexión a pesar de que la luz solar directa actúa como una gran fuente de interferencia pues la señal emitida no puede ser captada por los receptores.

2.3.1. ESTÁNDAR DE FUNCIONAMIENTO DEL LI-FI. IEEE 802.15.7

El grupo de trabajo 802.15 está especializado en WPAN (*Wireless Personal Area Network*), es decir, en redes inalámbricas de área personal [11]. Esta norma se centra en redes de cortas distancias, como por ejemplo Bluetooth.

El estándar IEEE 802.15.7 engloba a los protocolos que se utilizan para las comunicaciones inalámbricas mediante la luz visible. Esta norma define dos capas fundamentales la capa de acceso al medio (MAC) y la capa física (PHY), con una velocidad de datos que es capaz de soportar audio, vídeo y multimedia.

La capa física se encarga de interactuar entre los dispositivos transductores, es decir, diodos LED y fotorreceptores, y la capa de acceso al medio. Para su implementación la norma define tres modelos diferentes, cada uno con sus características respectivas: PHY I, PHY II y PHY III.

Tanto el modelo PHY I como el PHY II se utilizan para transmisiones que tienen una única entrada y salida (*Single-Input and Single-Output*, siglas SISO), con una sola fuente de luz mediante la utilización de modulación OOK (*On/Off Keying*, en español modulación digital de amplitud) o VPPM (*Variable Pulse Position Modulation*, en español modulación por posición de pulso variable). En nuestro prototipo se utilizará la modulación OOK mediante la codificación Manchester, en el apartado 4.3.1. se explica en detalle.

El modelo PHY III se utiliza en transmisiones de múltiples entradas y salidas (MIMO) soportando de esta forma múltiples fuentes de luz mediante el uso de la técnica de modulación CSK (*Color Shift Keying*, en español modulación por desplazamiento de color). Este modelo puede ofrecer velocidades entre los 12 Mb/s y 96 Mb/s [12].

En el Anexo A se puede ver en más detalle el funcionamiento de los modelos PHY I, PHY II y PHY III.

En cuanto a la capa de acceso al medio (MAC) se encarga de resolver los problemas de gestión de la capa física (PHY), como pueden ser: el direccionamiento físico, la prevención de colisiones, la asignación de canales y los protocolos de acuse de recibo de datos. Esta capa, es compatible con tres topologías de acceso: punto a punto, configuración en estrella y en modo de difusión (Ver Ilustración 1).

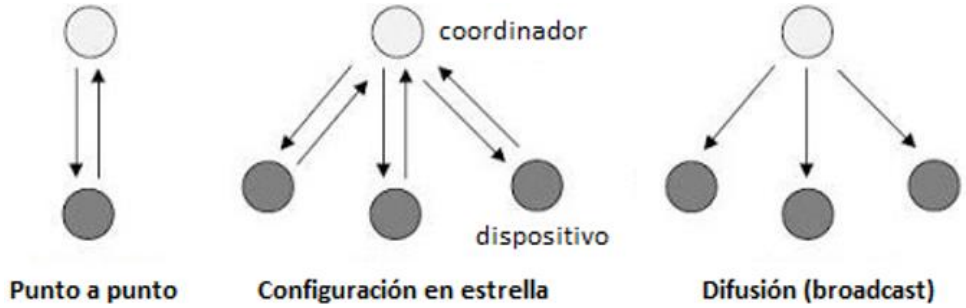

Ilustración 1. Tipos de topologías soportadas [13].

En la topología punto a punto (*peer to peer),* uno de los dos dispositivos que van a comunicarse toma la función de coordinador. Normalmente, el coordinador usará una fuente de energía principal, mientras que los dispositivos asociados a él utilizarán baterías. Por otro lado, la configuración en estrella (*star*) establece la comunicación entre diferentes dispositivos y un coordinador.

Además de estas topologías, los dispositivos que cumplen con este estándar también pueden operar mediante la topología de difusión (*broadcast*) mientras que no estén asociados a ningún dispositivo o no tengan ningún dispositivo asociado a él. Este es el caso de nuestro sistema, hay dos difusores de información y un coordinador.

El estándar 802.15.7 no tiene en cuenta otras técnicas de modulación que pueden implementarse en los modelos PHY que son utilizados en las comunicaciones ópticas inalámbricas y que son muy usadas en Li-Fi. Algunas de ellas son adaptaciones de las técnicas que ya se utilizan para las comunicaciones de radiofrecuencia como es el caso de Multiplexación por División de Frecuencias Ortogonales (OFDM) y en técnicas de modulación como la óptica y la Modulación Espacial de la Luz (SLM). Por lo tanto, este estándar está obsoleto al no considerar los últimos avances tecnológicos sobre las comunicaciones ópticas inalámbricas.

2.4. LA RED INALÁMBRICA EN LOS HOSPITALES.

El desarrollo de la Sociedad de la Información (SI) y la incorporación a ésta de las TIC a la atención sanitaria ha supuesto una transformación única en la historia, con un futuro centrado en el consumidor y orientado a los resultados y a la prevención. Dicha transformación se basa en tres pilares fundamentales que son los pacientes, el personal médico y las infraestructuras.

Las redes de comunicaciones desempeñan un papel muy importante en lo que salud y seguridad de sus pacientes se refiere. Del mismo modo, se facilita la interacción entre médico y paciente, de esta forma, el beneficio es mutuo.

Por otro lado, la utilización de sistemas inalámbricos integrados para aplicaciones clínicas en el recinto hospitalario supone un funcionamiento mas eficiente, efectivo y competitivo del sistema sanitario [14].

Los sistemas de telemetría suponen una serie de ventajas en el campo sanitario como son:

- Mejoras en la gestión hospitalaria, telecitas en tiempo real, intercambio de información, movilidad de historias clínicas, etc.
- Utilización de las TIC para la formación de profesionales médicos mediante videoconferencias, plataformas de e-learning y soportes digitales diversos.
- Realización de tele-diagnósticos o tele-consultas gracias al uso de equipos especializados para la captación y transmisión de datos e imágenes.
- Realización de intervenciones quirúrgicas a través de estaciones de trabajo virtuales y tele-actuación robotizada.
- Tele-monitorización, seguimiento y control de pacientes y de sus enfermedades a través de dispositivos específicos, así como la atención domiciliaria personal y continua.

A pesar de todas estas ventajas, los sistemas de telemetría son los que mayores problemas pueden causar por las consecuencias que puedan tener en los dispositivos médicos.

En el ámbito médico, los sistemas de comunicaciones inalámbricas utilizan un rango frecuencial reservado denominado ICM, garantizando de este modo la disponibilidad de frecuencias libre de emisiones intencionadas por parte de transmisores de aplicaciones no médicas. Según el Cuadro Nacional de Atribución de Frecuencias (CNAF) se usan 8 bandas de frecuencias destinadas a fines Industriales, Científico y Médico:

- ICM 1: 13,553 – 13,567 MHz
- ICM 2: 26,957 – 27,283 MHz
- ICM 3: 40,660 – 40,700 MHz
- ICM 4: 433,050 – 434,790 MHz
- ICM 5: 2400 – 2500 MHz
- ICM 6: 5725 – 5875 MHz
- ICM 7: 24,00 – 24,25 GHz

- ICM 8: 61,00 – 61,50 GHz

Los equipos ICM deberán cumplir los limites de radiaciones establecidos sobre los requisitos de protección relativos a la compatibilidad electromagnética (EMC) para de esta forma asegurarnos que no existan problemas de este tipo. Pero en el entorno sanitario no solo se utilizan tecnologías presentes en estas bandas, sino que hay una gran versatilidad de tecnologías que conviven y hay que prestar atención para que no causen ningún tipo de problema. En la Ilustración 2 se muestra un esquema gráfico de las distintas tecnologías que dan soporte a las aplicaciones sanitarias.

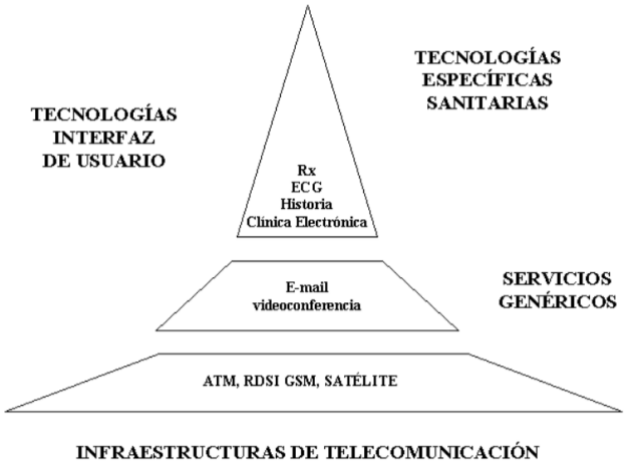

Ilustración 2. Tecnologías que dan soporte a las aplicaciones sanitarias [14].

Asimismo, hay que prestar atención a los posibles entornos electromagnéticos sanitarios y sus características, en el área en el que se centra este proyecto, cuidados intensivos, es conveniente realizar un análisis de todos los elementos que componen la sala para que no pueda surgir ningún contratiempo.

2.4.1. PROBLEMÁTICA DE LAS COMUNICACIONES EN ENTORNOS SANITARIOS.

La revolución digital en el entorno sanitario ha sido muy diferente a la de otros sectores debido a la existencia de una serie de necesidades especificas por parte de los proveedores en lo que respecta a la infraestructura de red y de TI. Por este motivo, siempre ha resultado complicado adoptar estándares tecnológicos vanguardistas en los hospitales, clínicas o en las viviendas asistidas para el cuidado de mayores.

Las TIC nos permiten en el sector de la salud acercarnos más al paciente sin perder el contacto en ningún momento. Consecuentemente se ha producido un incremento del uso de dispositivos eléctricos y electrónicos que conlleva un mayor uso del espectro de RF. Tal y como se observa en la Ilustración 3 los rangos de frecuencia dedicados a las telecomunicaciones y a las aplicaciones médicas no están cercanos a priori, pero aún así generan una interferencia electromagnética. Las EMI pueden ser un problema considerable para cualquier dispositivo electrónico, pero en los dispositivos médicos, las consecuencias pueden ser contraproducentes, pudiendo provocar problemas en el cuerpo humano que se manifestarán en diferentes enfermedades, hipersensibilidad electromagnética, etc.

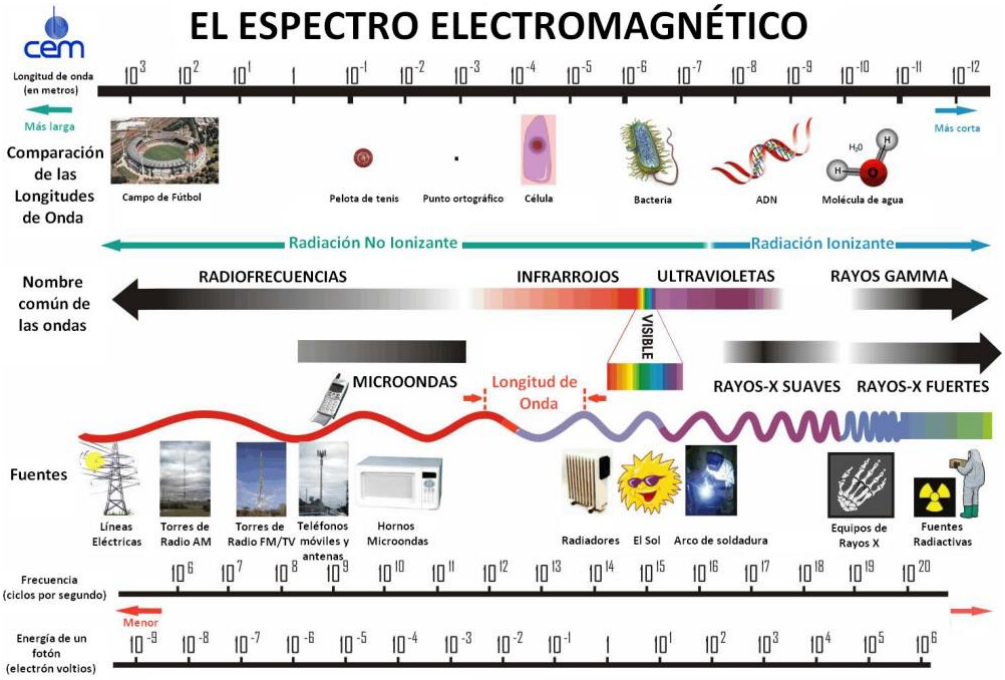

Ilustración 3. Espectro electromagnético y sus usos [15].

Los principales motivos por los que se genera una interferencia electromagnética en los entornos hospitalarios son la incorrecta instalación de equipos radiotransmisores, señales radioeléctricas intensas desde un transmisor cercano y/o el blindaje o filtrado insuficiente en el equipo electrónico para evitar que se capten señales no deseadas, esto es, por la simultaneidad que se da entre los diferentes dispositivos.

Dentro de dichas interferencias se pueden clasificar en constantes o en transitorias. Las señales constantes, como su propio nombre indica, están presentes todo el tiempo siendo de menor energía, más periódicas y de menor contenido frecuencial que las transitorias, que solo se dan en momentos puntuales.

Por otro lado, los efectos que producen las EMI sobre los equipos están divididos en clases según el nivel de daño que producen a los mismos [14]:

- Clase O: No se produce mal funcionamiento del equipo o dispositivo. La perturbación no influye.
- Clase A: La perturbación produce efectos aceptables, pero no altera el funcionamiento del equipo o dispositivo.
- Clase B: La perturbación altera temporalmente el funcionamiento del equipo o dispositivo, pero éste no sufre efectos irreversibles, pudiendo funcionar de nuevo sin intervención técnica.
- Clase C: La perturbación altera el funcionamiento del equipo o dispositivo, haciendo necesaria la intervención técnica para volver a funcionar.
- Clase D: La perturbación produce daños irreversibles en el equipo o dispositivo, quedando irrecuperable.

Hay que puntualizar que el número de fallos producidos por una EMI es mucho menor que el resto de los fallos que se pueden dar, esto se debe a la protección que ofrece la compatibilidad electromagnética. Por ello, es importante generar un ambiente electromagnético sin efectos adversos, sin introducir perturbaciones intolerables en ese ambiente y soportar las producidas por otros equipos. En la Tabla 1 se presentan algunos ejemplos de incidencias médicas registradas y su fuente de origen.

Dispositivo médico	Fuente de origen
Monitores de apnea	Radiodifusión en FM
Monitores de fas de anestesia	Electrobisturíes
Electrocardiograma	Telefonía móvil celular analógica y digital
Bombas de infusión y de jeringa	Telefonía móvil celular analógica y digital y equipos de rayos-X portátiles
Sillas de ruedas electrónicas	Equipos de comunicaciones de policía, bomberos y radioaficionados
Análisis hematológicos	Buscapersonas

Indicación de temperatura y presión sanguínea	Electrobisturíes
Monitores de incubadoras	Radioaficionados, telefonía móvil celular
Marcapasos	Comunicaciones de ambulancias, walky-talkies, detector de metales
Monitor de telemetría cardíaco	Comunicaciones en 160 - 174 MHz
Respirador	Equipos de rayos-X portátiles, walky-talkies y radiodifusión en FM
Equipos de diálisis	Telefonía móvil celular
Desfibriladores	Telefonía móvil celular
Monitor de telemetría	Pagging
Ayudas a la audición	Walky-talkies, telefonía móvil celular
Equipos de laparoscopia	Electrobisturíes

Tabla 1. Incidencias médicas [14].

En la compatibilidad electromagnética hay que tener en cuenta dos factores, el nivel de perturbación de las interferencias del generador y la susceptibilidad (EMS) del receptor. El término EMS y su opuesto inmunidad, se emplean para indicar la mayor o menor tendencia de un dispositivo o equipo a ser afectado por interferencias radioeléctricas, es decir, el nivel de susceptibilidad de un equipo es la capacidad que tiene éste para funcionar correctamente en un ambiente electromagnéticamente complejo. De esta forma, resulta prácticamente imposible hablar de susceptibilidad, inmunidad o medidas de protección en términos generales, sin referirse a equipos o dispositivos en concreto, debido a que cada uno de estos tendrá un comportamiento diferente [14].

Todos los productos europeos deben de cumplir con la Directiva sobre EMC y los fabricantes deben cumplir la norma en la etapa de construcción e instalación de la instrumentación, así como durante la selección de nuevos equipos electrónicos. Los usuarios también deben recibir indicaciones sobre identificación y eliminación de problemas referidos con EMC, y en caso de que se produjeran alertar de ellos.

Al mismo tiempo, la privacidad del paciente también tiene una enorme importancia porque la historia clínica permite llevar un seguimiento de la salud de cada paciente, pero del mismo modo es algo que afecta de lleno a la intimidad de una persona ya que puede verse amenazada, por ese motivo, se debe dar correcta conservación. Estos dos bienes que entran en colisión deben ser adecuadamente compatibilizados ya que son bienes constitucionales.

Centrándonos en el proyecto, mientras que el Wi-Fi interfiere en algunos aparatos o instrumentos médicos, esto no sucede con Li-Fi que puede emplearse en zonas críticas como es la unidad de cuidados intensivos de un hospital.

En la actualidad ya se está poniendo a prueba dicha tecnología en los hospitales como, por ejemplo, en el Hospital Universitario de Motol (Praga) a través de un estudio experimental que están llevando a cabo el equipo de investigación de Fraunhofer HHI en cooperación con la Universidad Técnica de Checa (CTU) de Praga. En dicho estudio se han establecido una red de varios transmisores y receptores LI-FI en el quirófano microquirúrgico del Hospital y se están probando varias técnicas *multiple-input multiple-output* (MIMO). El sistema Li-Fi ha logrado hasta el momento transmitir datos de forma rápida y sin perdidas de señal a velocidades de 600 Mbit/s, superiores a las actuales redes Wi-Fi y móvil [16].

CAPÍTULO 3. ELEMENTOS PARA EL SISTEMA DE COMUNICACIÓN POR LI-FI.

Un aspecto esencial para la aceptación comercial es que esta nueva tecnología, Li-Fi, cuenta con la disponibilidad de transceptores miniaturizados de bajo coste. De esta forma se puede utilizar de forma más cómoda en aplicaciones médicas [17].

Los componentes principales en los que se basan las comunicaciones visibles son un LED blanco que actúa como fuente de transmisión y un fotodiodo que actúa como elemento receptor, de forma que este último debe de poseer una buena respuesta a la luz visible. En este capítulo, se desarrollará con más detalle los elementos que se han utilizado para diseñar el prototipo de un sistema Li-Fi.

3.1. EMISIÓN DE LA SEÑAL.

La tecnología Li-Fi utiliza emisores de luz de alta velocidad que pueden encenderse y apagarse a una velocidad imperceptible por el ojo humano dando una apariencia de estar continuamente encendido, ya que la velocidad de los LED es inferior a 1μs. De esta forma, el LED se enciende y se apaga generando cadenas digitales de diferentes combinaciones de unos y ceros que permite la transmisión de datos.

Dependiendo del material de que está hecho el LED se tendrá un color u otro y, por lo tanto, la longitud de onda variará (ver Tabla 2). Para nuestro diseño se ha optado por el color blanco, ya que para la iluminación habitual bien sea de una sala de hospital o en el hogar es la tonalidad que se utiliza.

Composición del chip	Nombre del compuesto	Color de la luz emitida	Tensión de trabajo en volt (V)	Frecuencia en hertz (Hz)	Longitud de onda en nm
GaAs	Arseniuro de galio	Infrarrojo	< 1,9	< $4,0 \times 10^{14}$	> 760
GaAlAs	Arseniuro de galio y aluminio				
GaP	Fosfuro de galio		± 1,8	$4,8 - 4,0 \times 10^{14}$	610 - 760
GaAlAs	Arseniuro de galio y aluminio				
AlInGaP	Fosfuro de aluminio indio y galio				
GaAsP/GaP	Fosfuro de galio y arsénico / Fosfuro de galio				
AlInGaP	Fosfuro de aluminio indio y galio		± 2,0	$5,1 - 4,8 \times 10^{14}$	590 - 610
GaAsP/GaP	Fosfuro de galio y arsénico / Fosfuro de galio				
AlInGaP	Fosfuro de aluminio indio y galio			$5,3 - 5,1 \times 10^{14}$	570 - 590
GaP	Fosfuro de galio		± 3,0		500 - 570
InGaN	Nitruro de indio y galio			$5,8 - 5,3 \times 10^{14}$	
GaN	Nitruro de galio				
InGaN / Zafiro	Nitruro de indio y galio / Zafiro				
SiC	Carburo de silicio		± 3,3	$6,7 - 6,0 \times 10^{14}$	450 - 500
InGaN / Zafiro	Nitruro de indio y galio / Zafiro				
GaN	Nitruro de galio				
InGaN / Zafiro	Nitruro de indio y galio / Zafiro				
InGaN	Nitruro de indio y galio		± 3,4	$7,9 - 6,7 \times 10^{14}$	
GaN	Nitruro de galio	Ultravioleta	± 3,7	> $7,9 \times 10^{14}$	< 400

Tabla 2. Características de los Leds [18].

Las principales ventajas por las que se opta por este elemento son el precio, la gran implementación y crecimiento en el ámbito de la iluminación gracias a su eficacia, durabilidad. Además, los LEDs en el uso residencial requieren un 75% menos de energía y pueden durar incluso 25 veces más que las bombillas incandescentes [19].

Los datos que se mandan a través del diodo son los distintos valores de temperatura que tienen diversos pacientes en una sala UCI de un hospital, obtenidos a partir de varios sensores LM35. Estos sensores son dispositivos de temperatura que tienen un voltaje de salida linealmente proporcional a la temperatura, de esta forma incrementa el valor a razón de 10 mV por cada grado centígrado. El rango de temperatura que es capaz de medir depende del modelo que se utilice.

El modelo por el que se ha optado es el LM35DZ que cuenta con un rango de temperaturas de -55 º C a un máximo de 150 º C con una precisión a temperatura ambiente de 0,5 º C [20]. Dado que se quiere medir la temperatura de seres humanos el rango de temperaturas es más que de sobra ya que como ya se sabe, los seres humanos somos seres isotérmicos, es decir, seres de sangre caliente y sangre fría, y esto no nos obliga a mantener una temperatura óptima al margen de las condiciones ambientales que nos rodean. La temperatura normal que roda un ser humano está entre los 36 º C y 37 º C, por lo tanto, ya un caso superior sería ya un posible caso de fiebre y debería saltar una alarma en nuestro prototipo y lo mismo cuando la temperatura sea inferior

a 35 [21]. En la Ilustración 4 se puede ver las diferentes consecuencias que se pueden sufrir debido a ciertas temperaturas.

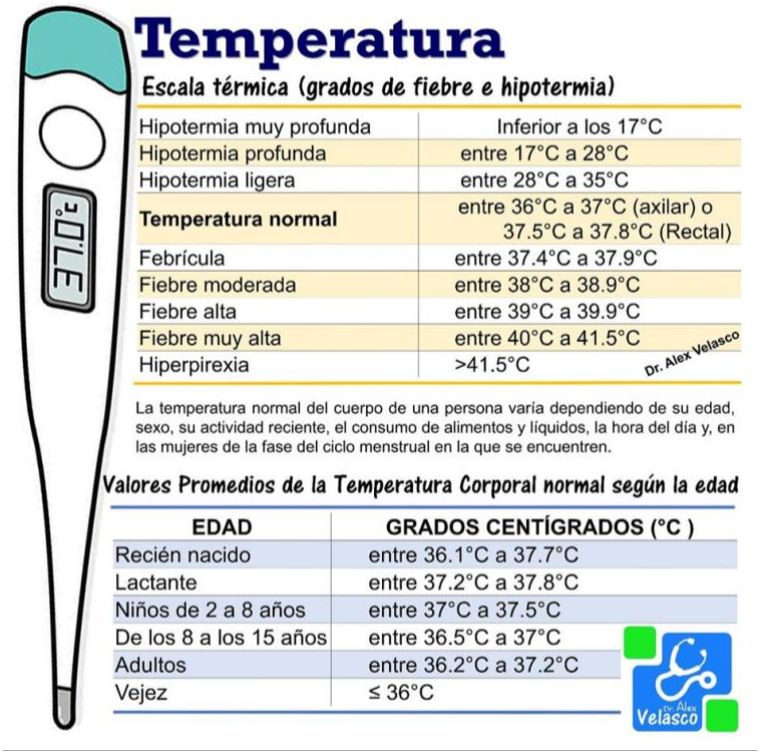

Ilustración 4. Rango de temperaturas de un ser humano [21].

Dicho sensor cuenta con un amplio rango de tensiones de alimentación que va desde los 4V hasta los 30V, pero en nuestro caso está alimentado con 5 voltios que nos proporciona el Arduino y conectado a una de las entradas analógicas de este debido a que es un sensor analógico [20]. En la Ilustración 5 en la parte derecha se puede ver el conexionado del sensor con el Arduino.

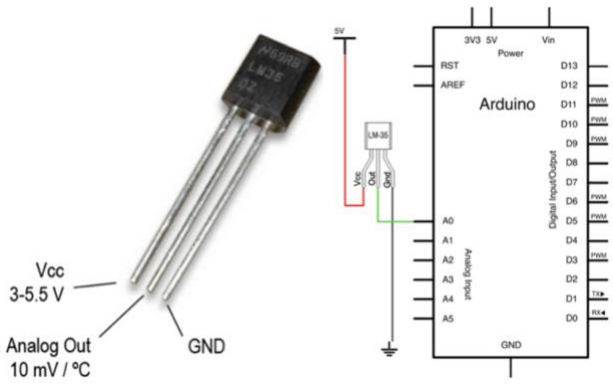

Ilustración 5. Sensor LM35DZ [22].

Por otro lado, el encapsulado que utiliza es el más común que corresponde con el T-92, siendo este muy compacto lo que le permite estar protegido frente a cualquier golpe inesperado ayudando de esta forma a que no se deteriore. Además, este encapsulado viene con tres patas

de conexión (ver Ilustración 5), dos de alimentación y una de salida en una referencia de tensión a razón de 10mV/ºC como ya se comento anteriormente [22]. Para que sea seguro el sensor con el contacto de la piel de una persona se debe de cubrir las patas del sensor con termofit para de esta forma aislar los terminales y los cables de conexión tal y como se muestra en la Ilustración 6.

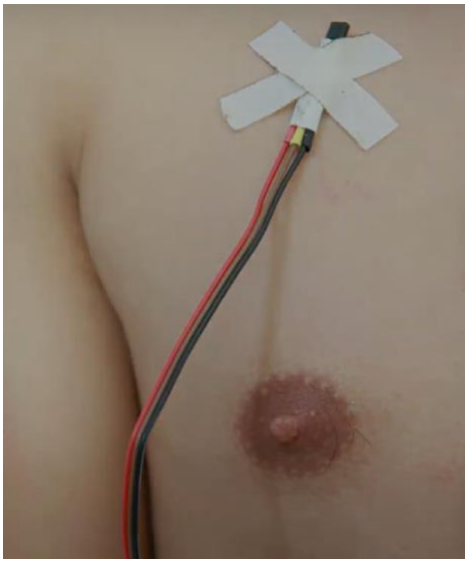

Ilustración 6. Sensor de temperatura en contacto con la piel [23].

La baja impedancia de salida, su salida lineal y su precisa calibración hace posible que este integrado sea instalado fácilmente en un circuito de control, por ese motivo se ha elegido este sensor.

3.2. RECEPCIÓN DE LA SEÑAL.

En un sistema de comunicación Li-Fi la función del receptor es transformar la señal óptica recibida en una señal electrónica. Para ello, existen una gran cantidad de sensores, pero hay que tener en cuenta el más adecuado prestando atención tanto en la velocidad de respuesta como en la sensibilidad.

La velocidad de respuesta hace referencia al tiempo finito desde que se enciende la fuente de luz hasta que hay un valor constante de salida, mientras que la sensibilidad determina la relación entre la salida del sensor ante une entrada determinada [24]. Se ha decido utilizar un fotodiodo ya que en estos dos aspectos tienen una respuesta muy alta y son los más utilizados.

El fotodiodo por el que se ha optado es BPW 34, el cual detecta longitudes de onda comprendidas entre los 400nm y 1100nm, por lo tanto, detecta la luz visible. Este fotodiodo tiene su pico de sensibilidad en el color rojo pero la iluminación con el led rojo no es la apropiada para un entorno hospitalario.

A diferencia del Light-Dependent Resistor (LDR) o fotorresistencia, el fotodiodo responde a los cambios de oscuridad a iluminación y viceversa con mucha más rapidez, y puede utilizarse en circuitos con tiempo de respuesta más pequeño [24].

3.3. EL SISTEMA DE CONTROL.

El papel de sistema de control lo desempeñan diversas placas de Arduino debido a que es una plataforma abierta y versátil para el desarrollo de productos electrónicos. En el caso del receptor se utiliza Arduino UNO ya que resulta indiferente el tamaño porque es donde se recibe y procesa toda la información. Mientras que en los transmisores se emplea Arduino NANO a causa de su pequeño tamaño. Recordemos que el sistema de transmisión que se está diseñado consiste en monitorizar la temperatura de pacientes, por lo tanto, que sea pequeño y fácil de mover es un punto de especial interés. También hay que tener en cuenta que a la hora de alimentarlo también sea así, es decir, que no haya mucho cable. Por esa razón, se utilizarán baterías rectangulares de nueve voltios para dar energía al Arduino NANO, es decir, una pila alcalina. De esta forma se cumple la forma de alimentar los dispositivos asociados al coordinador según la topología de la capa de acceso al medio, es decir, que los dispositivos asociados estén alimentados mediante baterías y el coordinador con la fuente de alimentación de energía principal, es decir, con el ordenador (ver 2.3.1).

A cada una de estas placas se les conectarán los elementos necesarios para que se pueda producir la comunicación del sistema que se detallará más en profundidad en los apartados siguientes.

3.4. SISTEMA DE VISUALIZACIÓN.

Para visualizar nuestro sistema Li-Fi se ha optado por la plataforma de Processing. Processing es un lenguaje de programación y entorno de desarrollo integrado de código abierto basado en

Java. Fue iniciado por Ben Fry y Casey Reas a partir de reflexiones en el Aesthetics and Computation Group del MIT Media Lab dirigido por John Maeda [25].

Processing es muy similar al IDE de Arduino debido a que está basado en este, por lo tanto, es más sencillo a la hora de crear un entorno de visualización de datos ya que no hay mucha diferencia en cuanto a su manejo. Processing cuenta con muchas más posibilidades de visualización que si usáramos el Monitor Serial del IDE de Arduino puesto que es un entorno de desarrollo basado en Java y existen muchas funciones para crear formas y figuras visuales. Para conectar ambos programas se utilizará el puerto serie y se diseñará una ventana donde aparecen diversas camas de hospital cada una de ellas correspondiente a un sensor de temperatura.

CAPÍTULO 4. DISEÑO DEL SISTEMA LI-FI.

En este capítulo se aborda el diseño e implementación del escenario de desarrollo del prototipo de transmisión unidireccional Li-Fi, en el cual se describe el montaje de las tarjetas Arduino UNO (para el receptor) y Arduino Nano (para los transmisores). Para la transmisión de datos se utilizan diodos led blancos en cada uno los transmisores, mientras que para la recepción de información se utiliza un fotodiodo.

4.1. DIAGRAMA DE BLOQUES.

El diseño que se utiliza en este proyecto es una comunicación basada en la difusión, es decir, se tienen varios emisores que mandan información hacia un único receptor. En el esquema de la Ilustración 7 está representado el sistema que se va a llevar a cabo. Consta de dos Arduinos Nano, cada uno de ellos alimentado mediante una batería que se encarga de medir la temperatura con ayuda del sensor LM35 y mandarlo al Arduino principal, Arduino UNO, mediante un LED. El Arduino principal recibe la información de la temperatura mediante un fotodiodo y gracias al ordenador que alimenta dicho Arduino se procesa la información recibida conforme a un umbral. En el caso de que supere dicho umbral se encenderá una alarma para alertar de que la temperatura no es la adecuada.

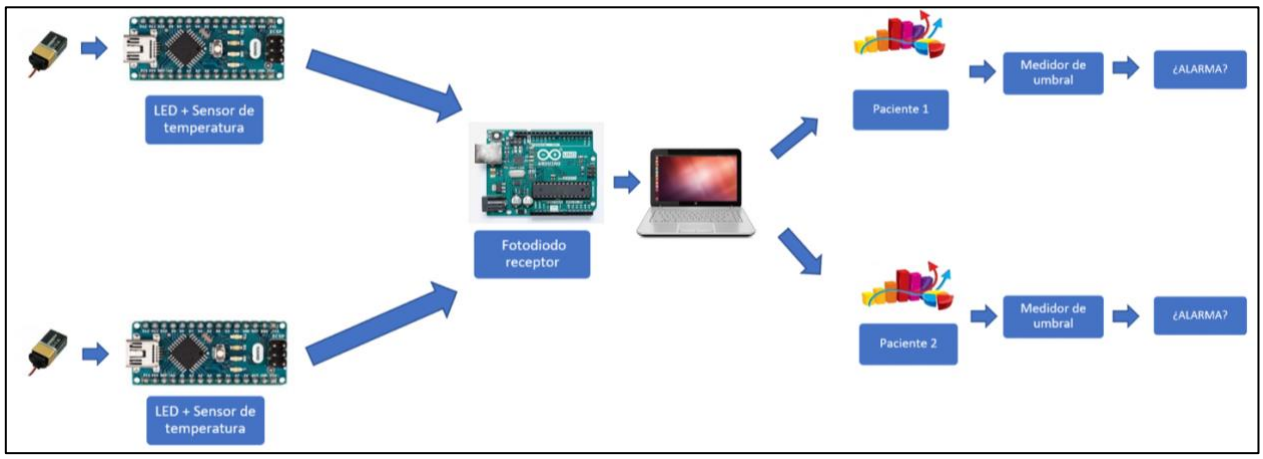

Ilustración 7. Diagrama del escenario.

4.1.1. DIAGRAMA DEL ESCENARIO EN PROTEUS.

En la plataforma de Proteus se ha realizado el diagrama del proyecto técnico. Proteus es una aplicación para la ejecución de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño del esquema electrónico, programación del software, construcción de la placa de circuito impreso, simulación de todo el conjunto, depuración de errores, documentación y construcción [26]. Gracias a Proteus las fases de prueba no suponen la necesidad de volver a construir nuevos prototipos, suponiendo así un ahorro en costes y en tiempo.

En la Ilustración 8 se puede ver que a diferencia del diseño original se han utilizado tanto para los transmisores como para el receptor un Arduino UNO. Por otro lado, se utiliza un optoacoplador que en el diseño físico se utilizará como ya se ha dicho un led y un fotodiodo en vez de un fototransistor ya que tiene mejores prestaciones a la hora de detectar la luz visible.

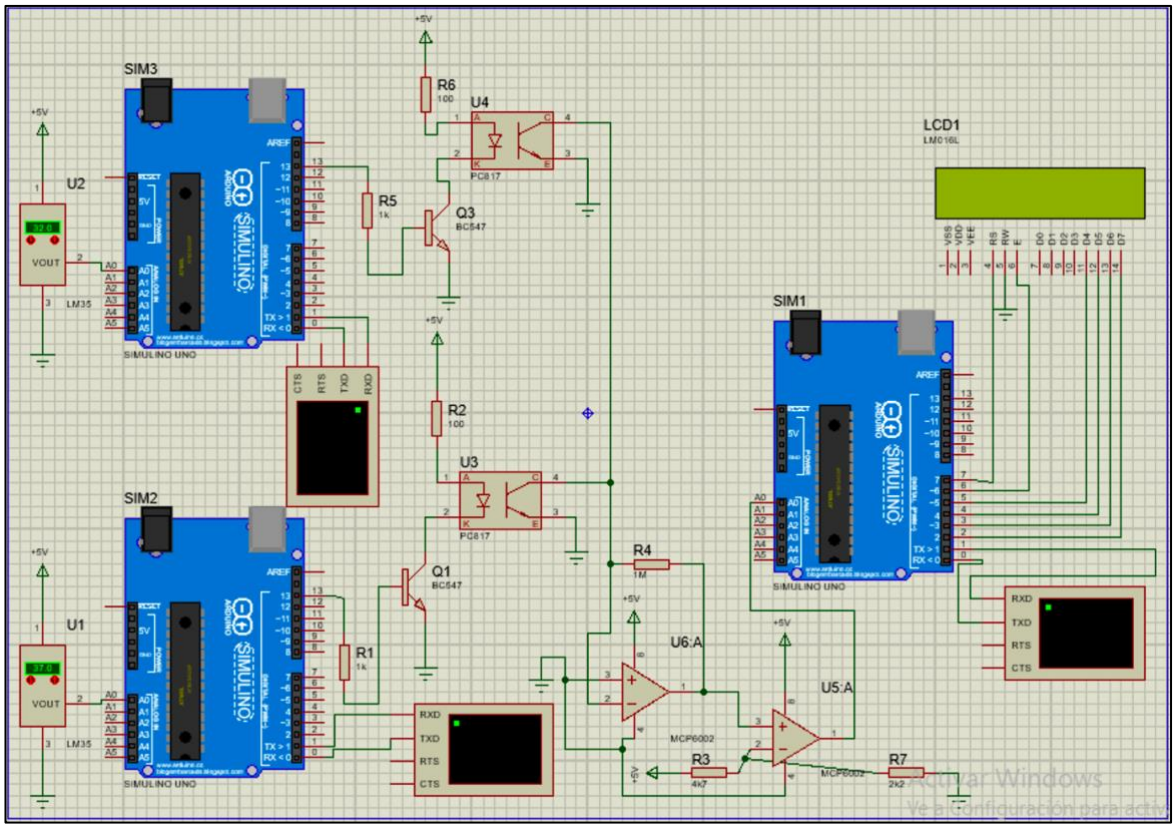

Ilustración 8. Diagrama del escenario en Proteus. Prueba 1.

En el montaje también se pueden observar diferentes amplificadores operacionales (AO) para amplificar la señal débil que se recibe del optoacoplador. En la Ilustración 9 se puede comprobar como el sistema funciona correctamente y que detecta los cambios de temperatura perfectamente.

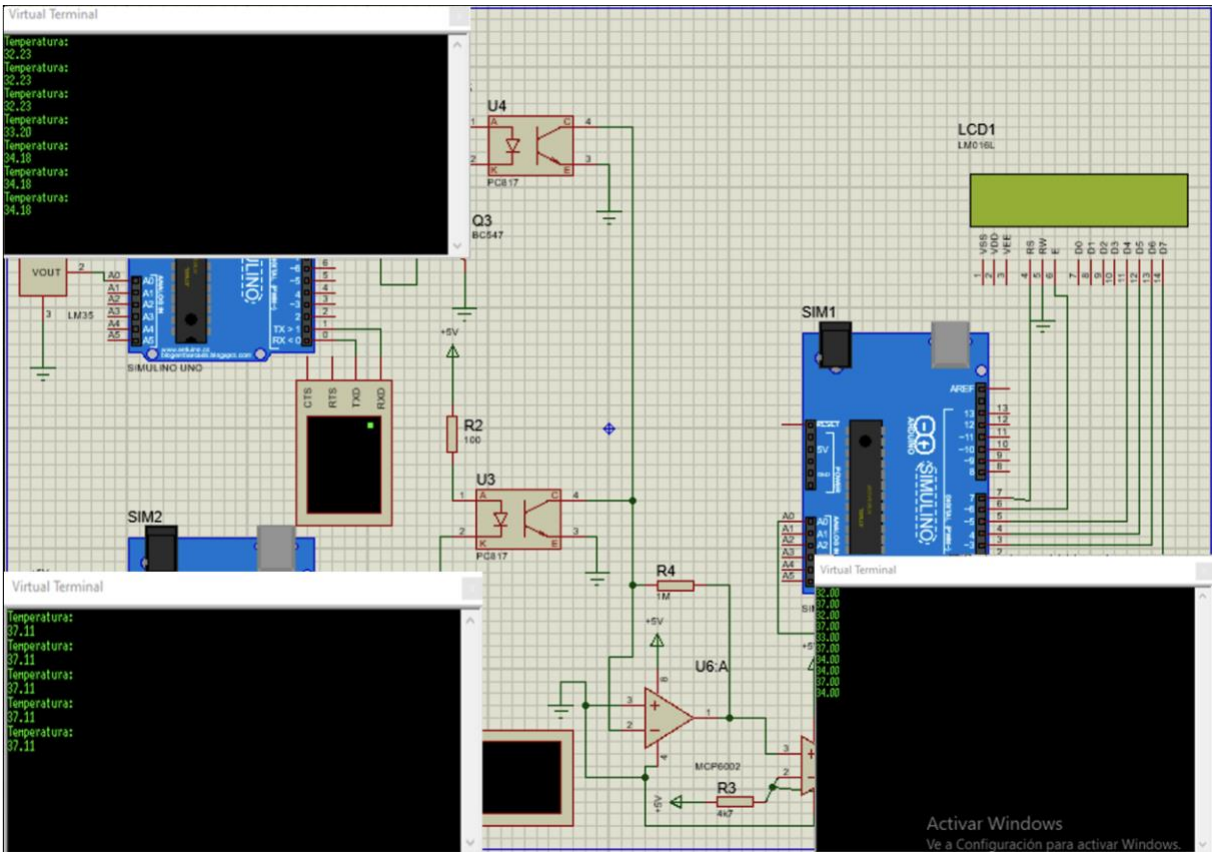

Ilustración 9. Simulación en Proteus. Prueba 1.

En la Ilustración 10 se puede comprobar las señales que se han obtenido del osciloscopio de la simulación pertinente. La señal de color amarillo hace referencia a la señal que emite el Arduino de la parte inferior y la señal azul la señal que emite el Arduino superior. La señal de color rosa es la señal que se obtiene después del optoacoplador y vemos que ha perdido amplitud. En cambio, la señal de color verde corresponde con la señal que se introduce al Arduino, la cual es de una amplitud adecuada para que se pueda producir el proceso de demodulación. Todas las señales están en la misma escala, 5 Voltios por división, para que así sea más visual su comprensión.

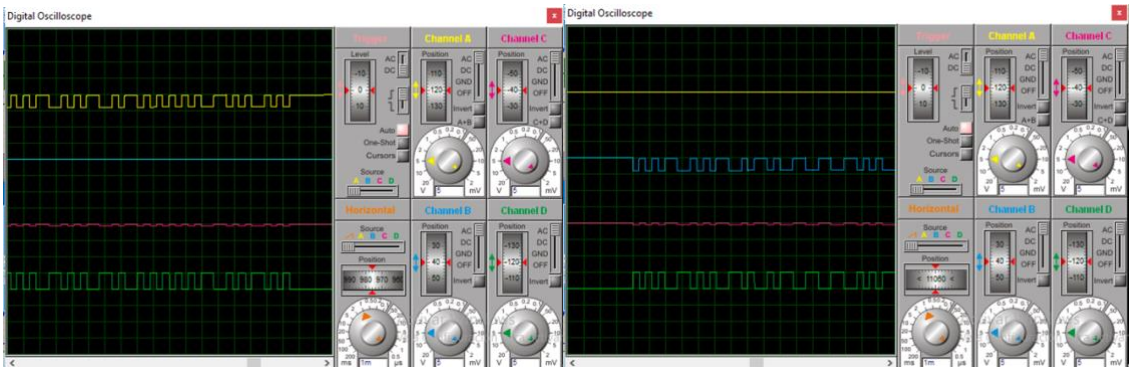

Ilustración 10. Señal del osciloscopio 5V/división. Prueba 1.

4.2. GENERACIÓN DE ESPECIFICACIONES.

En este apartado se explicará más en detalle la elección de cada componente. En el Anexo B se encuentra de forma detallada las características esenciales de cada componente. Para los transmisores se utiliza el mismo montaje que consiste en dos resistencias, un transistor para realizar la modulación directa y un LED, además de la conexión a masa y dos puertos de alimentación, uno en el puerto serie del Arduino y otro a una fuente de alimentación como se muestra en la Ilustración 11. Por otro lado, estará el sensor de temperatura conectado al Arduino.

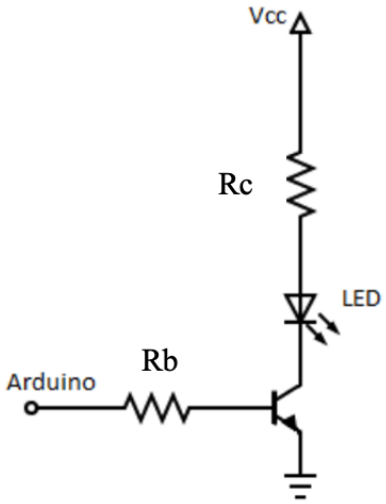

Ilustración 11. Circuito transmisor.

Como se aprecia en la figura la rama que conecta a la base del transistor contiene una resistencia que se ha llamado Rb y está conectada a la salida del puerto serie de la placa Arduino. El emisor esta conectado a masa y el colector se conecta al LED y a la alimentación tras una resistencia de protección que se llamará Rc.

Con las características de los componentes ya seleccionados se calculan los valores de las resistencias. La rama de la base del transistor tiene una entrada de 5V en los valores altos de la transmisión y 0V en los valores bajos. En los valores bajos no hay corriente y por lo tanto el transistor se encuentra en corte impidiendo el paso de corriente y por lo tanto apagando el LED. En el caso de tener tensión debemos elegir la resistencia Rb de forma en que el transistor se sature. Por lo tanto, se selecciona una resistencia base que produce una ganancia de corriente de 10, es decir, (βDC = 10).

La resistencia base que se ha seleccionado es de 1 KΩ y tomando el voltaje de saturación del transistor se obtiene:

$$R_B = 1\ K\Omega$$

$$V_{BB} = I_{BRB} + V_{BE} = \frac{V_{BB} - V_{BE}}{R_B} = \frac{5 - 0.9}{1K} = 4.1\ mA \quad (1)$$

De esta forma usando cálculos básicos se puede obtener la corriente del colector de la siguiente manera:

$$I_C = \beta_{DC} * I_B = 10 * 4.1\ mA = 41\ mA \quad (2)$$

De forma similar se obtiene el valor de Rc con la ecuación 3 utilizando la ley de Ohm:

$$V_{CE} = V_{CC} - V_{LED} - I_{CRC} \quad (3)$$

$$R_C = \frac{V_{CC} - V_{LED} - V_{CE}}{I_C} = \frac{9 - 3.2 - 0.25}{41\ mA} = 135.36\ \Omega \approx 100\ \Omega \quad (4)$$

Una vez realizados los cálculos se puede calcular la potencia del LED, que en los casos de alimentación a 9V la corriente que pasa por él es de 41 mA y en el resto de los casos la alimentación es de 5 V y la intensidad se calcula a partir de la siguiente ecuación.

$$P_{LED-9V_{CC}} = V_{LED} * I_C = 3.5V * 41\ mA = 143.5\ mW \quad (5)$$

$$I_C = \frac{V_{CC} - V_{LED} - V_{CE}}{R_C} \rightarrow P_{LED-5V_{CC}} = V_{LED} * \frac{V_{CC} - V_{LED} - V_{CE}}{R_C} = 3.5 * \frac{5 - 3.5 - 0.25}{135.36} = 32.3\ mW \quad (6)$$

Después del diseño del circuito se implementa en una placa como se muestra en la Ilustración 12.

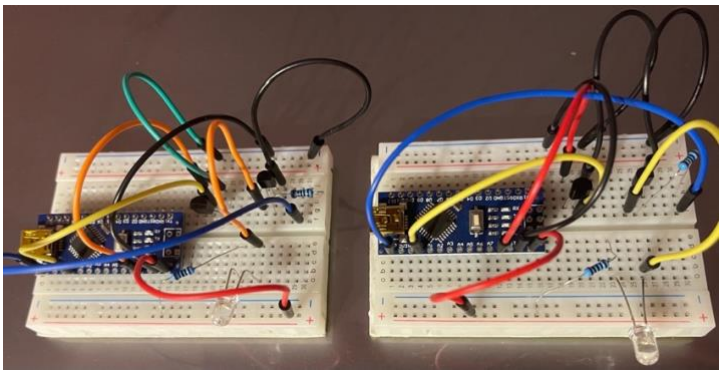

Ilustración 12. Circuitos transmisores.

Para diseñar el circuito receptor se utiliza el montaje que se muestra en la Ilustración 13, el cual consiste en dos etapas una primera amplificadora y la otra comparadora. Inicialmente el fotorreceptor se conecta a la entrada negativa del amplificador de transimpedancia negativa porque este genera corriente inversa. Adicionalmente, se añaden una resistencia para crear retroalimentación y de esta forma se amplifica la señal débil del fotodiodo. Esta primera etapa se puede describir su salida con la siguiente expresión:

$$V_{OUT1} = A_V * V_{in} \tag{7}$$

Según la expresión obtenida llegamos a la conclusión de que la tensión de salida V_{OUT1} corresponde con el valor de entrada multiplicado por una constante (R_1). A esta constante se llama ganancia del circuito, es decir, este circuito tiene una ganancia de R_1.

Como la señal que detecta el fotodiodo es una señal débil se ha optado por elegir un valor alto para esa resistencia, en nuestro caso de 2 MΩ, para de esta forma asegurar que la señal de entrada no se pierde.

La salida de esta primera etapa es conectada a la entrada positiva del comparador y la entrada negativa a un divisor resistivo donde una de las dos resistencias es variable. El resultado del divisor resistivo se calcula mediante la siguiente ecuación:

$$V_{O6} = V_i * \frac{R_3}{(R_2 + R_3)} \tag{8}$$

Dependiendo de la salida del primer AO con el LED apagado, es decir, con el fotodiodo recibiendo la luz ambiente, se fija la salida del divisor resistivo. De esta forma, el comparador compara la salida del primer amplificador con la salida del divisor resistivo, obteniendo a la salida un valor lógico para su posterior descodificación.

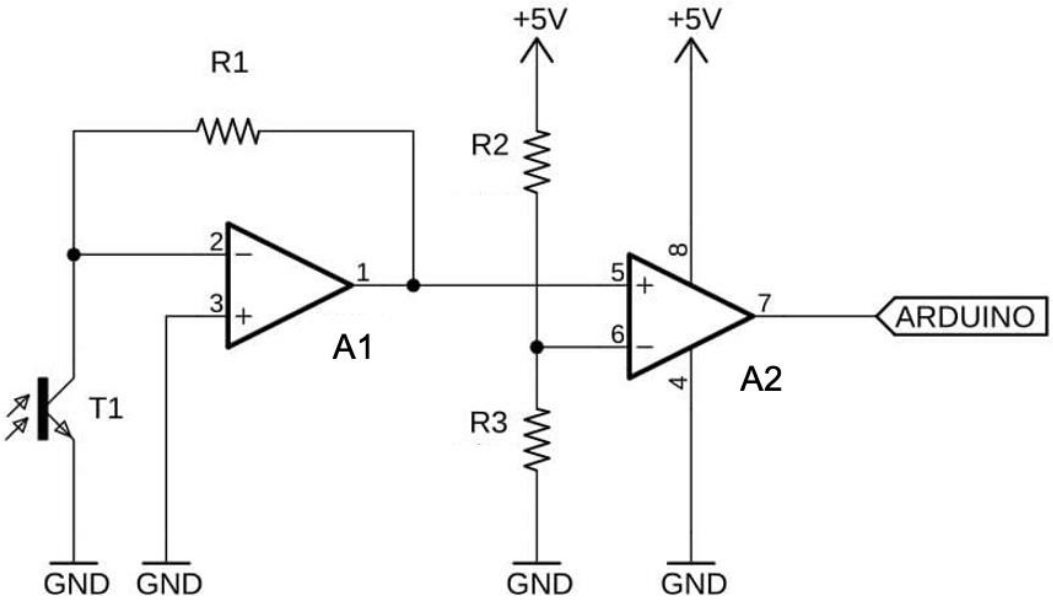

Ilustración 13. Circuito receptor.

Por último, solo queda elegir las resistencias que componen el divisor resistivo. Como se ha comentado anteriormente está compuesto por un potenciómetro y una resistencia de valor 2K2Ω. Por lo tanto, a partir de la ecuación siete se obtiene que el valor de salida mínimo que en el divisor resistivo será de 1.52V.

$$V_{O6} = V_i * \frac{R_3}{(R_2 + R_3)} = 5 * \frac{2K2}{50K + 2K2} = 5 * \frac{11}{36} = \frac{55}{36} = 1.52\,V \quad (9)$$

Una vez diseñado el circuito receptor se implementa en una placa como se muestra en la Ilustración 14.

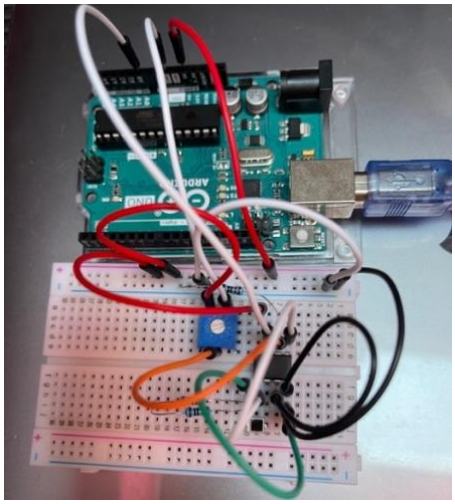

Ilustración 14. Circuito receptor.

4.3. PROGRAMACIÓN DE LA PLACA DE ARDUINO.

En este trabajo la placa Arduino es una parte fundamental del prototipo ya que funciona como interfaz entre el ordenador y el prototipo. Para no hacer que el sistema sea menos redundante, la placa Arduino se utiliza como microcontrolador y como unidad de fuente de alimentación (PSU) para los circuitos, suministrando voltaje y actuando también como tierra virtual [27].

En cuanto a la programación se han creado dos programas, uno para los transmisores y otro para el receptor. En los apartados siguientes se comenta más detalladamente cada uno de ellos.

4.3.1. CODIFICACIÓN DEL EMISOR.

Para enviar datos a través del LED es necesario modular la información en una señal portadora, y en este caso se realiza mediante la codificación Manchester con la ayuda de la librería Manchester.h. En el Anexo C se muestra detalladamente el código de esta librería.

Como se ha dicho anteriormente la modulación OOK es el esquema de modulación más simple para VLC ya que los LEDs se encienden o se apagan dependiendo de si los bits de datos son '1' o '0', es decir un uno digital representa el estado de luz encendida y un cero digital representa el estado de luz apagada. Hay varios tipos de codificación Manchester, en nuestro caso se utiliza la que se muestra en la Ilustración 15 en la parte inferior. Dicha codificación incorpora el reloj en los datos al representar un cero como un símbolo OOK '10' y uno lógico como un símbolo OOK '01'. De esta manera, el periodo de los pulsos positivos es el mismo que el de los negativos, pero se duplica el ancho de banda requerido para la transmisión OOK. En este caso no es problema ya que solo se quieren mandar las temperaturas y un identificador.

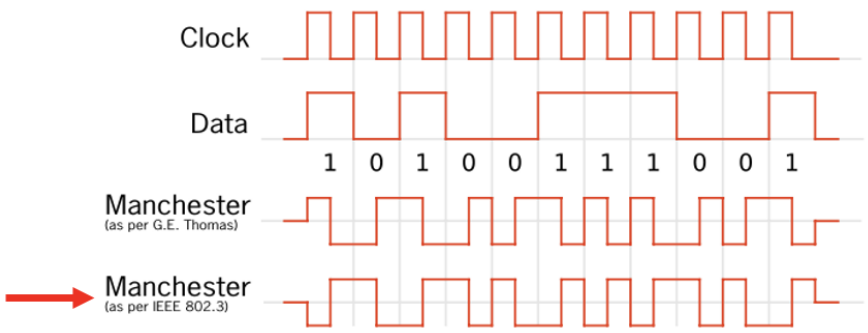

Ilustración 15. Codificación Manchester [28].

En la Ilustración 16 se muestra el código que se ha utilizado para los transmisores, la única diferencia es que la variable *"direc"* será diferente en cada uno de los transmisores, permitiendo saber de donde procede cada temperatura, en nuestro caso a que paciente corresponde.

Transmisor_array

```
// Transmitter code:

#include <Manchester.h>
#define TxPin 13 //el pin digital que se utilizará para transmitir datos

uint8_t datos[3]={0,0,0};

void setup()
{
  pinMode(A0,INPUT);
  pinMode(13, OUTPUT);
  man.setupTransmit(TxPin, MAN_2400); // establece el pin digital como salida
  Serial.begin (9600);
}

void loop () {
    float temperatura = analogRead(A0);
    float milivoltios = ( temperatura / 1024.0 ) * 5000;
    float Tdata = milivoltios / 10;
    float direc = 1;

    //Comprobar que es la misma temperatura la del sensor que la que se envía
    Serial.println("temperatura sensor:");
    Serial.println(Tdata);

    datos[0]= 3;
    datos[1]= Tdata;
    datos[2]= direc;

    // comprobación por el serial de la temperatura que manda
    Serial.println("Temperatura: ");
    Serial.println(datos[1]);
    Serial.println("Cama: ");
    Serial.println(datos[2]);

    //Enviar el array con la temperatura y la dirección
    man.transmitArray(3,datos);

     delay(1000);
}
```

Ilustración 16. Código del transmisor.

El código es muy sencillo de entender, el pin A0 es configurado como una entrada a través del cual se obtienen los datos de la temperatura que serán enviados por el pin 13 gracias al comando *man.transmitArray()* en el bucle principal. Además, en el *setup()* se configura la velocidad de transmisión con el comando *man.setupTransmit()*. Por último, he de comentar que la información se manda cada segundo.

4.3.2. CODIFICACIÓN DEL RECEPTOR.

En cuanto a la recepción de datos también se emplea la librería Manchester para la descodificación del mensaje tal y como se puede ver en la Ilustración 17. En el *setup()* se configura igual que en el transmisor la velocidad, la cual tiene que ser la misma para que se produzca correctamente la transmisión, y además, se indica la dimensión del array que se espera. Por otro lado, en el bucle principal se obtienen los datos del array.

Receptor_array

```
#include <Manchester.h>
#define RxPin A0

uint8_t datos[3];
String enviar;

void setup()
{
 man.setupReceive(RxPin,MAN_2400);
 man.beginReceiveArray(3,datos);
 Serial.begin (9600);
}

void loop()
{
 if (man.receiveComplete())
 {
    uint8_t mensaje = 0;
    mensaje = datos[0];

    for (uint8_t i = 1; i < mensaje; i++)
    {
      datos[i];
    }
    Serial.println();
    Serial.println("Temperatura: ");
    Serial.println(datos[1]);
    Serial.println("Cama: ");
    Serial.println(datos[2]);

    enviar = "";
    enviar += datos[0];
    enviar += ",";
    enviar += datos[1];
    enviar += ",";
    enviar += datos[2];

    //Serial.write(enviar);
    Serial.println(enviar);

    man.beginReceiveArray(3,datos);

  }
}
```

Ilustración 17. Código del receptor.

4.4. IMPLEMENTACIÓN DEL PROTOTIPO.

Una vez que se ha realizado el diseño, verificado los materiales y los componentes, se procede a la implementación del prototipo Li-Fi. Para ver de una forma más visual el funcionamiento del sistema se ha creado en Processing un programa que se puede ver en el Anexo D.

La aplicación nos muestra una ventana que simula una sala de un Hospital en este caso con dos camas ya que se utilizan dos sensores de temperatura, pero podría ampliarse de acorde a cuantos sensores haya (ver Ilustración 18). Debajo de cada cama, se encuentra la temperatura que está teniendo el paciente, que es recibida directamente por el puerto serie del Arduino junto con un circulo rojo o azul en caso de si está ocupada o no la cama respectivamente y, por tanto, si se están recibiendo temperaturas procedentes de ese paciente o no. Junto con la temperatura hay

un cuadrado que se pondrá de color rojo en caso de que la temperatura del paciente supere los 37ºC, es decir, cuando comience a tener fiebre.

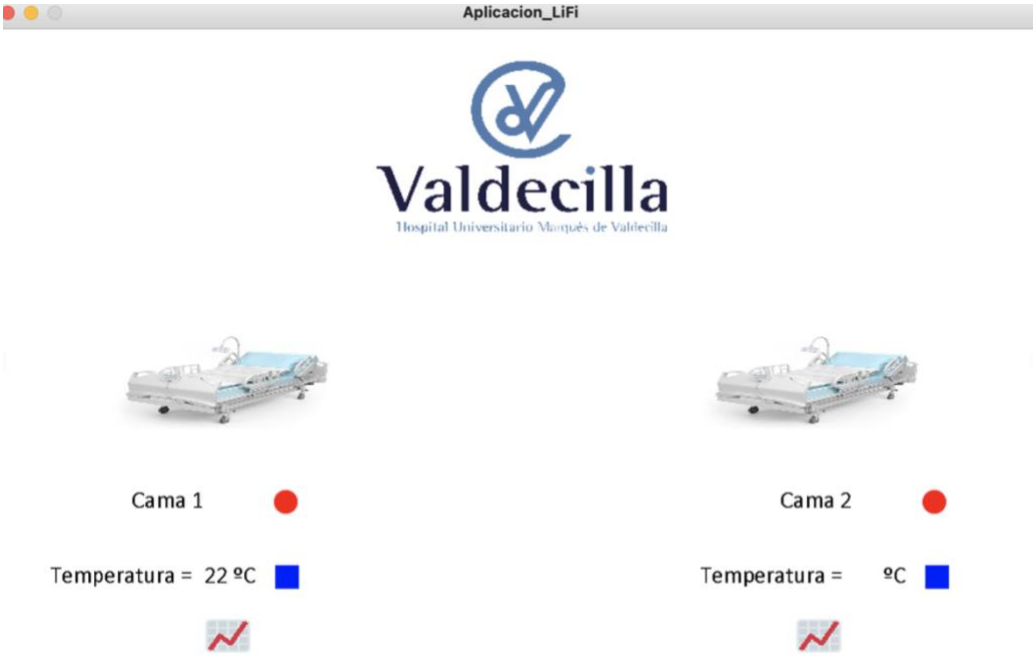

Ilustración 18. Aplicación Li-Fi en Processing.

Además, la aplicación consta de un botón interactivo que nos mostraría la información detallada de la persona como se observa en la Ilustración 19.

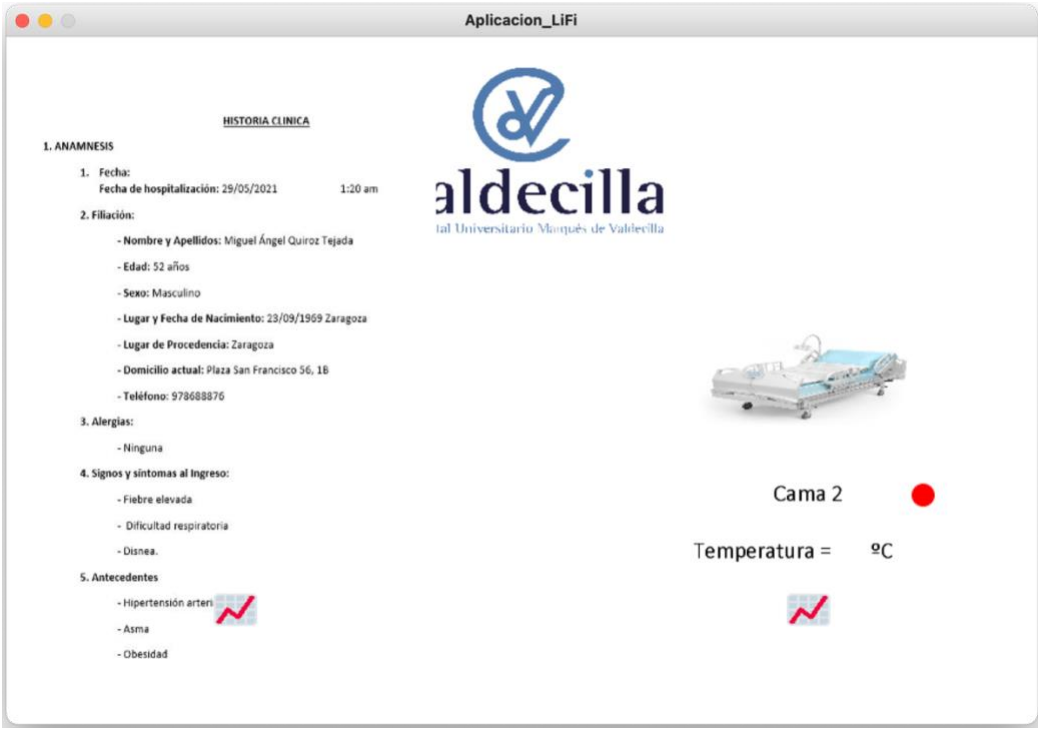

Ilustración 19. Aplicación Li-Fi con Processing usando el botón interactivo.

Al mismo tiempo que se muestra por pantalla a través de la aplicación los datos de temperatura se guardan en un documento denominado *temperatura_datos.txt* que recoge la hora exacta de la temperatura junto con la cama correspondiente para así llevar a cabo mejor un estudio por los profesionales de la materia.

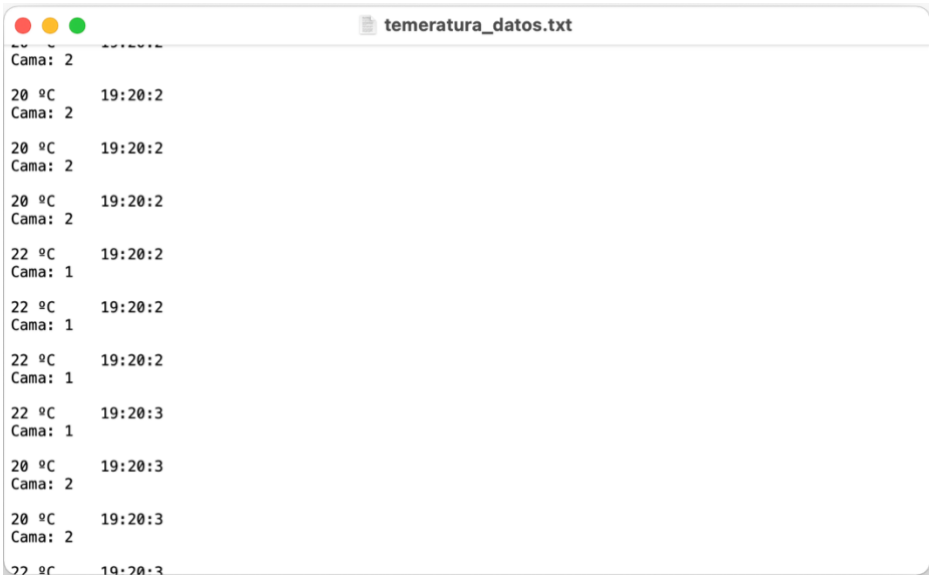

Ilustración 20. Documento temperatura_datos.txt.

Para que la visualización de la temperatura sea mejor, se ha creado de forma adicional una aplicación con el entorno de programación de Matlab ya que a la hora de representar valores lo hace de una forma más precisa y clara ya que con Processing se puede representar en tiempo real, pero con ausencia de ejes, por lo tanto, la comprensión no es clara.

En la aplicación de Matlab se cogen los valores de un fichero añadido que genera Processing en el cuál en cada fila solo hay los datos de la temperatura y la cama. De esta forma se puede obtener el resultado que se muestra en la Ilustración 21. De igual manera que en la aplicación de Processing hay dos botones interactivos que corresponden a cada uno de los sensores y una lamparita que se iluminará de color rojo en el caso de que la temperatura sea mayor a 37 º C. El código de esta aplicación también se encuentra en el Anexo D. ANEXO D. CÓDIGO DEL ENTORNO DE VISUALIZACIÓN.

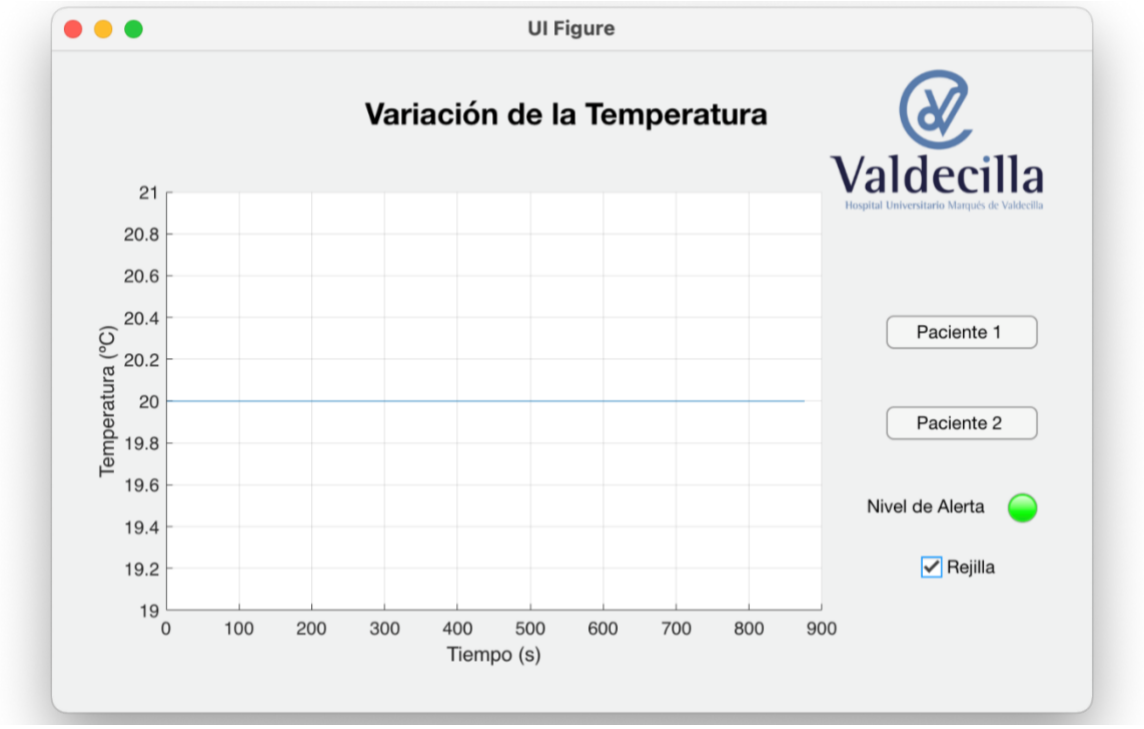

Ilustración 21. Aplicación en Matlab.

Con estas dos aplicaciones lo que se pretende conseguir es monitorizar la temperatura del paciente, mejorando el seguimiento de este a la vez que se disminuyen los desplazamientos innecesarios del personal sanitario. Ya que o bien se puede ver en el momento o bien se puede estudiar más en detalle los cambios de temperatura a posteriori.

4.4.1. MONTAJE DEL ESCENARIO DE EXPERIMENTACIÓN.

Las medidas que se van a realizar son en dos ejes de desplazamiento: distancia y altura. Al variar la altura, como tenemos dos transmisores se varía asimismo el ángulo en el que incide en el fotodetector. Para asegurar que las medidas sean fiables y la alineación del entre el transmisor y el receptor se pueda conseguir con relativa facilidad se ha diseñado una estructura para que esto se consiga.

El montaje consiste en dos elementos estructurales tal y como se muestra en la Ilustración 22, que permite realizar las medidas con estabilidad y comodidad. La estructura del receptor es fija, la única que se puede desplazar verticalmente es la correspondiente al transmisor.

Como se puede apreciar la estabilidad nos la proporciona la base y las diferentes alturas la barra vertical. Al mismo tiempo, se ha decido sustituir las placas de protoboard por dos placas de

pistas (mirar Ilustración 23) para soldar los componentes haciendo las medidas más reales ya que se elimina el ruido que las protoboard proporcionan.

Para realizar las medidas se ha utilizado un osciloscopio PicoScope 2203 y con ayuda software se permite obtener un mejor estudio.

Ilustración 22. Montaje.

Ilustración 23. Montaje del receptor (izquierda) y de los transmisores (derecha).

CAPÍTULO 5. PRUEBAS EXPERIMENTALES Y RESULTADOS

5.1. PRUEBAS DE COMUNICACIÓN DEL SISTEMA LI-FI.

Las comunicaciones por luz visible no se ven afectadas por las interferencias electromagnéticas, pero si por la luz. Por esta razón, se va a realizar una prueba basada en la ausencia de luz, es decir, completamente oscuras, solo la iluminación del sistema y otra con luz ambiente para observar la calidad de la información recibida. Otro impedimento que tiene el Li-Fi es el alcance que suele ser muy escaso. En nuestro caso el fotodiodo que se ha seleccionado tiene un ángulo de 60º y el LED de 25º.

Para las pruebas que se han realizado se ha modificado el ángulo variando la disposición vertical, la distancia de ambos componentes y la velocidad de transmisión.

En la Ilustración 24 se puede observar la señal que se emite con el LED variando las velocidades en 300, 2400 y 9600 baudios respectivamente. Al lado izquierdo está la señal a un segundo por división (en todos los casos se va a usar siempre la misma división para que sea más fácil comparar las señales) y al lado derecho el zoom de dicha señal. Como se puede observar cuanto mayor es la velocidad mayor es la distorsión que se produce en la señal, por eso con la velocidad de 300 baudios se ve una señal cuadrada y a medida que se aumenta la velocidad no es exactamente cuadrada, sino que aparece algún pico. Pero a pesar de que la señal no sea perfectamente cuadrada el valor de la temperatura, se obtiene correctamente.

En todas las situaciones que se han planteado la señal siempre toma un valor constante de 4,25 voltios mientras el LED está apagado. Cuando el LED se enciende, cada un segundo, toma valores entre 2,90V y 4,25V para formar el tren de pulsos que tiene que enviar.

A esta señal no le influye que haya más o menos luz, por lo tanto, se obtiene la misma señal con el escenario apagado que con luz ambiente. Del mismo modo, si se conecta algún aparato que produzca ondas de radiofrecuencia como en nuestro caso un taladro, no se produce ninguna perturbación en la señal, obteniendo así el mismo resultado.

Ilustración 24. Salida de la señal de transmisión con distintas velocidades.

En cuanto al receptor la cosa cambia ya que, si que hay diferencia entre un escenario luminoso y un escenario oscuro iluminado únicamente con los LEDs y las pantallas del ordenador. Se va a comenzar estudiando el caso de ausencia de luz, es decir el escenario oscuro (ver Ilustración 25).

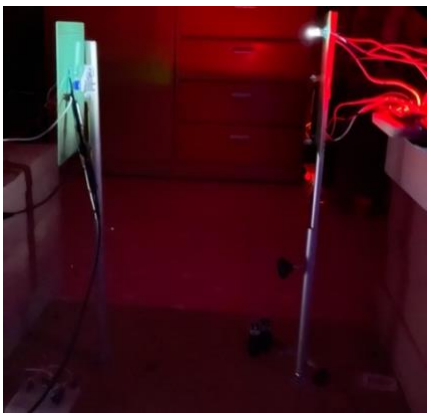

Ilustración 25. Escenario oscuro.

En la Ilustración 26 se plasma la señales recibidas por el fotodetector una vez que se ha amplificado con una distancia de 31 centímetros variando las velocidades en 300, 2400 y 9600 baudios respectivamente. Al igual que la señal que envían los transmisores a la derecha se encuentra el zoom de la señal. Del mismo modo a medida que se aumenta la velocidad se distorsiona la señal. En este caso la señal que se recibe en ausencia de señal es de 0,80 voltios y cuando se recibe las señales de cada transmisor aumenta a valores de 1,56 voltios o de 1,90 voltios aproximadamente. El valor más pequeño corresponde con el transmisor que está más arriba en la barra del transmisor ya que está a mayor distancia.

Si la señal recibida se compara con la señal transmitida (ver Ilustración 24) se puede observar que es la misma (los colores ayudan a ver que señales hay que fijarse), pero con algo más de ruido una vez que se ha pasado por la etapa amplificadora pero no dificulta su compresión a la hora de obtener los datos de temperatura ya que se obtienen de forma correcta.

Ilustración 26. Señal recibida del fotodetector amplificada a distintas velocidades con distancia de 31cm.

Si se toma como referencia la velocidad de 2400 baudios y se varia la distancia horizontal a 45 centímetros y 62 centímetros respectivamente, se obtienen los resultados que se muestran en la Ilustración 27. Al aumentar la distancia, la señal que se recibe es más pequeña y hay mayor distorsión de la señal. Y, por lo tanto, se parece menos a la señal que se ha transmitido desde los transmisores.

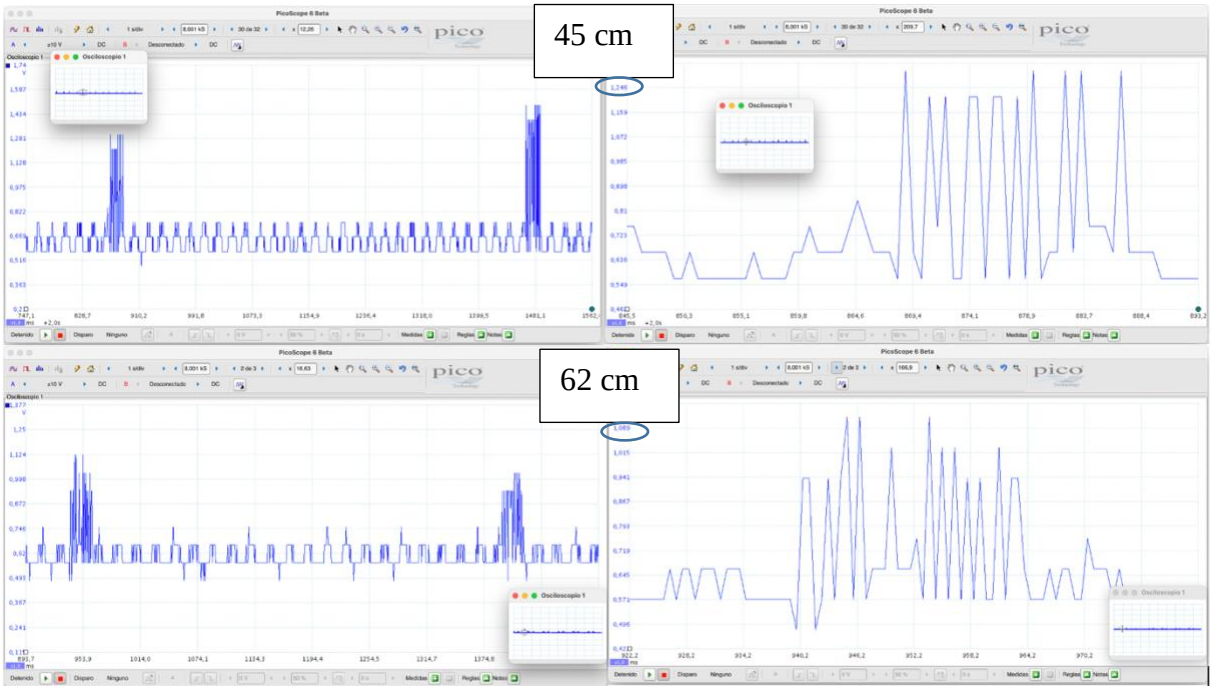

Ilustración 27. Señal recibida del fotodetector amplificada a distintas distancias con una velocidad de 2400 baudios.

A diferencia de la señal de los transmisores si se apunta con una luz al fotodetector se producen interferencias tal y como se ve en la Ilustración 28. Pero al igual que en las señales transmisoras si se conecta algún aparato que produzca ondas de radiofrecuencia no se produce ninguna perturbación en la señal, obteniendo así el mismo resultado en todas las pruebas que se han realizado. Cabe destacar que si se interpone un obstáculo entre la comunicación establecida se corta la comunicación y no recibe nada.

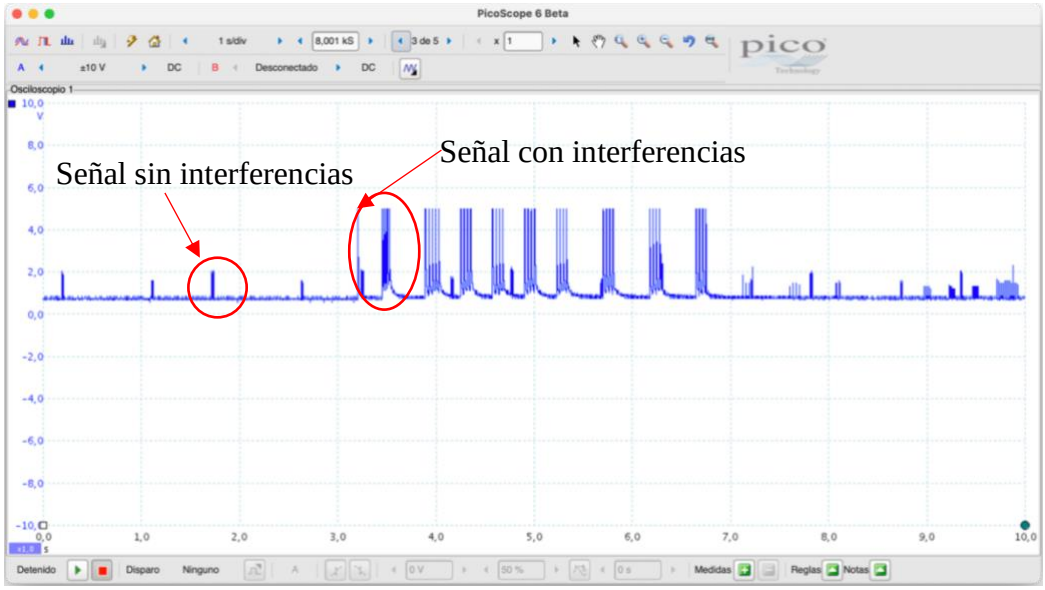

Ilustración 28. Señal recibida del fotodetector amplificada a una distancia de 31 cm con una velocidad de 2400 baudios con interferencias.

Hay que tener en cuenta que ninguno de los transmisores en los escenarios propuestos está alineado con el fotodiodo, por lo tanto, si se alinea el fotodetector inferior y el superior formando un ángulo de 22º con 31 centímetros de distancia se obtienen los resultados de la Ilustración 29.

En el caso de mayor ángulo formado con el transmisor se ve que el voltaje que se obtiene es 864mV que comparado con los 2,5V que se reciben con la menor distancia es una gran diferencia. Por lo tanto, se puede concluir que el ángulo si afecta y si se aumenta la distancia y el ángulo la señal que se recibe es muy escasa.

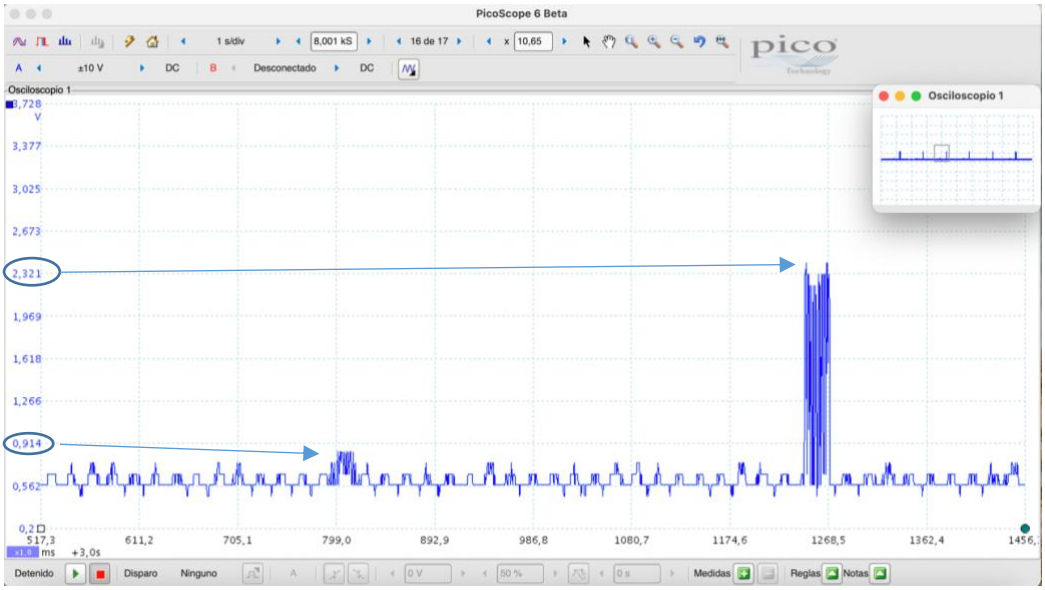

Ilustración 29. Señal recibida del fotodetector amplificada con un transmisor perfectamente alineado con una velocidad de 2400 baudios.

La máxima distancia que se ha conseguido con el escenario anterior y recibiendo de forma correcta el valor del sensor es de 102 cm (ver Ilustración 30). Se observa como el LED alineado con el fotodetector se recibe mejor que el que tiene ángulo de 22º.

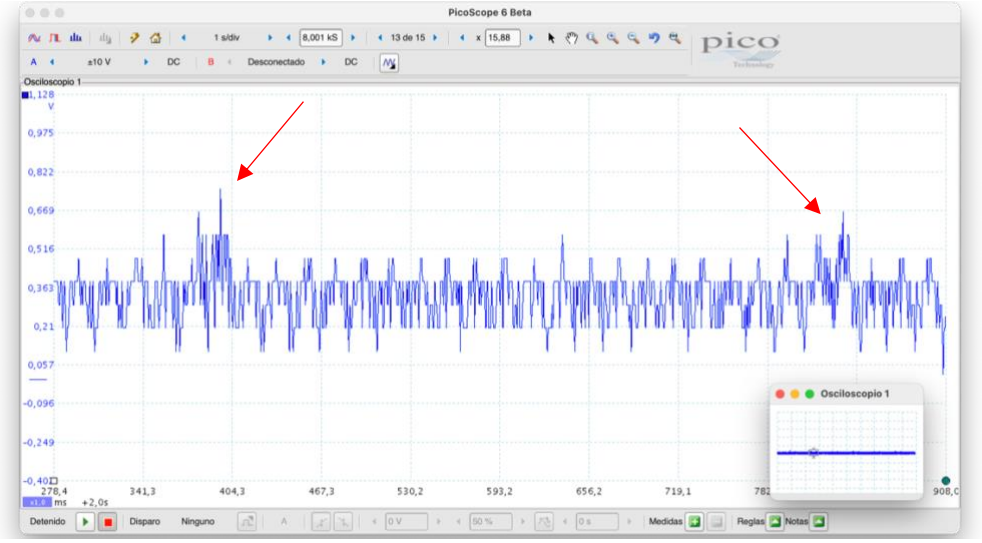

Ilustración 30. Señal recibida del fotodetector amplificada a 102 cm de distancia con una velocidad de 2400 baudios.

La diferencia principal entre el escenario oscuro y el escenario con luz ambiente (ver Ilustración 31) es que la señal que recibe el fotodector con los leds apagados es de 2,63 voltios mientras que cuando recibe la secuencia de pulsos del led alcanza los 4,5 voltios.

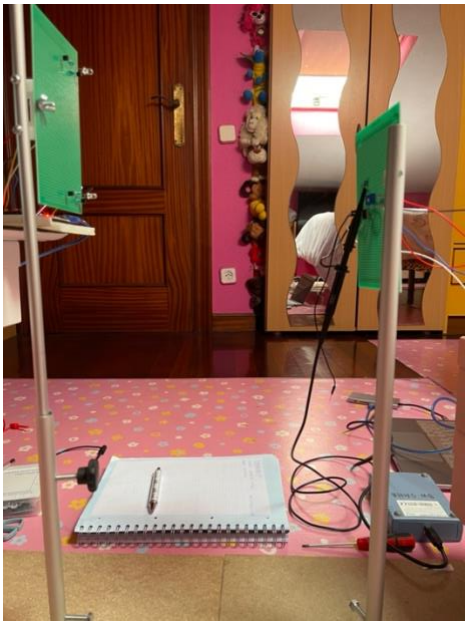

Ilustración 31. Escenario con luz ambiente.

Si se observa la Ilustración 32, la cual equivale al mismo escenario que la Ilustración 29 pero con el escenario iluminado con luz ambiente, se puede apreciar que se obtiene la misma señal, pero con un voltaje de inicio con el led apagado mayor.

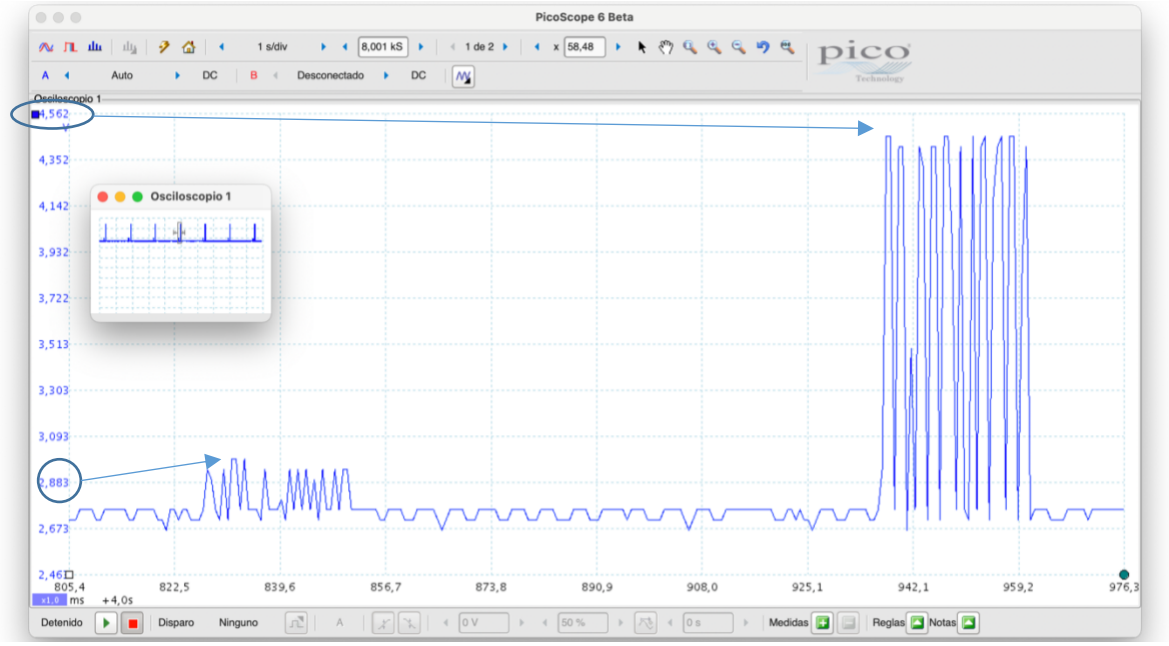

Ilustración 32. Señal recibida del fotodetector amplificada con un transmisor perfectamente alineado con una velocidad de 2400 baudio en un entorno iluminado con luz ambiente.

En este escenario si se alumbra al fotodiodo produce tal interferencia que ya ni se distinguirían los pulsos que mandan los transmisores produciendo una señal continua de unos 5 voltios aproximadamente (ver Ilustración 33). Por lo tanto, podemos concluir que cuanto mayor este iluminado el escenario más influyen las interferencias impidiendo la recepción de información.

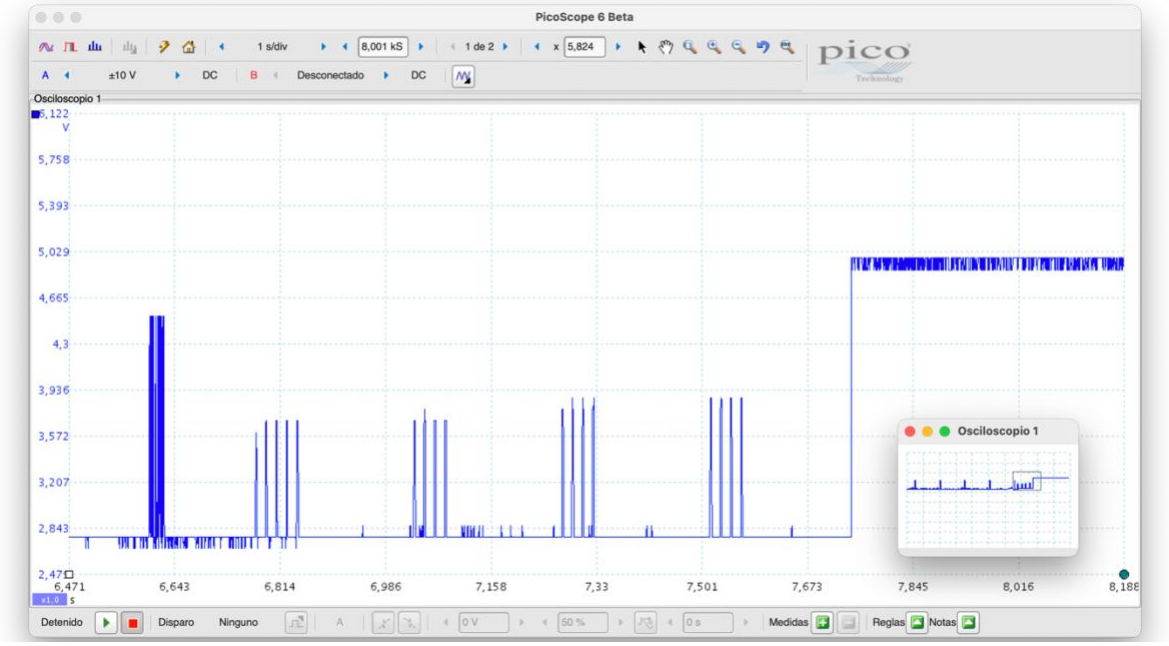

Ilustración 33. Señal recibida del fotodetector amplificada a una distancia de 31 cm con una velocidad de 2400 baudios con interferencias en un escenario con luz ambiente.

5.2. DESARROLLO DEL SISTEMA CON BLUETOOTH.

Para poder analizar más en profundidad la tecnología Li-Fi se ha decidido realizar una comparativa con otra tecnología que está en auge como es el Bluetooth. Según un informe de ABI Research, en 2024 habrá en el mercado más de 800 millones de dispositivos que lo utilizarán [29]. Este es uno de los principales motivos por los que se ha optado por está tecnología para hacer una comparación con la tecnología Li-Fi. Para ello, se ha optado por una comunicación punto a punto a través de dos módulos Bluetooth a una distancia cercana. Uno de estos módulos actúa como esclavo, que manda la información de la temperatura, y otro como maestro, el cual recibe dicha información.

A la hora de la elección de los módulos Bluetooth podemos observar que existe una gran diversidad pero que los más comunes para su uso con Arduino son los HC-05 y HC-06. El HC-06 sólo puede ser esclavo mientras que el HC-05 puede ser esclavo y maestro, en nuestro caso se ha elegido para ambos módulos el modelo HC-05 para que no haya ningún tipo de problema [30]. La diferencia entre esclavo y maestro es que modo esclavo es el dispositivo quien se conecta al módulo, mientras que en modo maestro es el módulo quien se conecta con un dispositivo.

Para la utilización de dicho módulo se requiere un uso del puerto serie de nuestra placa Arduino. Para establecer la comunicación con el puerto serie se puede hacer o bien desde los pines que posee la propia placa, o se puede emplear la librería SoftSerial para establecer una comunicación puerto serie por cualquier pareja de pines digitales [31]. En nuestro caso, como se observa en la Ilustración 34 se hace utilizando dicha librería mediante los pines digitales 10 (pin de recepción) y 11 (pin de transmisión).

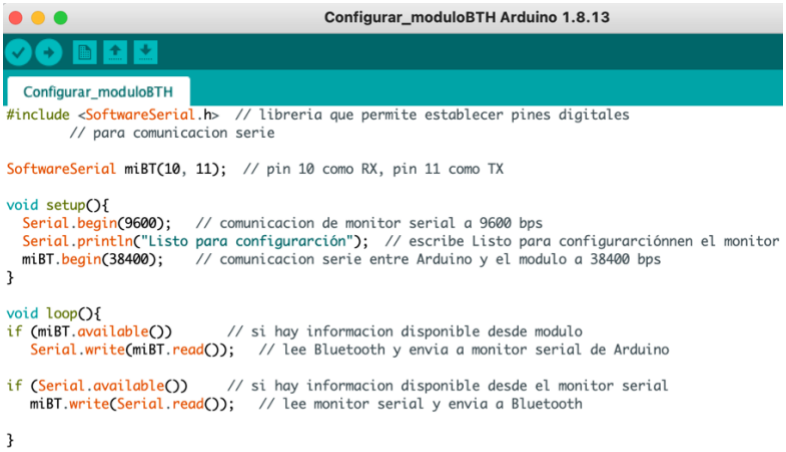

Ilustración 34. Código de la Configuración del módulo Bluetooth.

En la Ilustración 35 se muestra como la configuración se ha realizado correctamente y como se ha enlazado el módulo maestro al módulo esclavo a través de la MAC del esclavo.

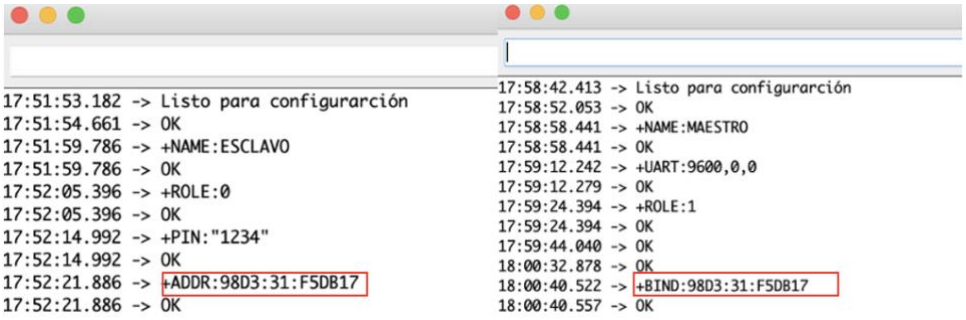

Ilustración 35. Comprobación de la configuración del módulo HC-05.

La conexión que hay que realizar en el módulo es sencilla, primero se alimenta a través de VCC y GND, y posteriormente se conecta el TXD (pin de transmisión) y el RXD (pin de recepción) a los opuestos de la placa Arduino. En la Ilustración 36 se muestra el conexionado del Arduino a la izquierda y a la derecha el del módulo Bluetooth.

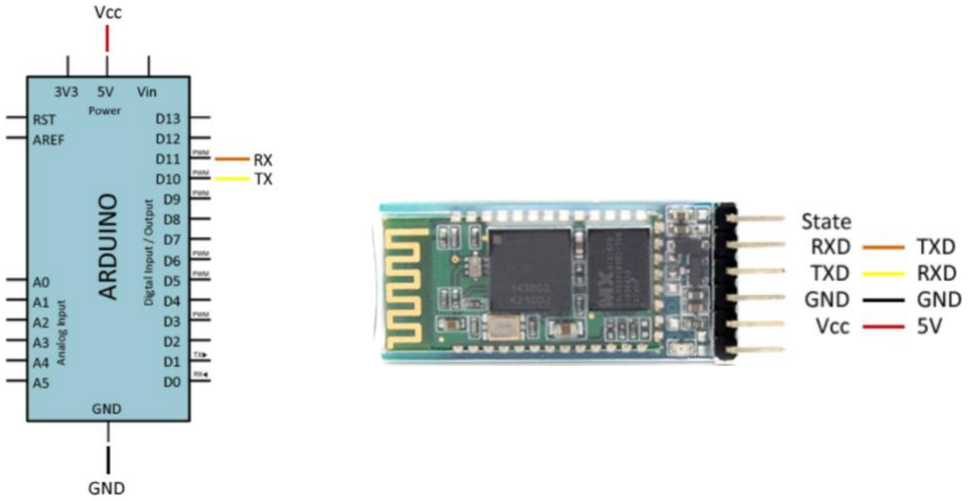

Ilustración 36. Conexionado de la placa Arduino y el módulo Bluetooth [31].

Además, como el Arduino trabaja con una lógica de 5V no es aconsejable conectar el módulo Bluetooth directamente con el Arduino ya que el pin RXD soporta un voltaje de 3.3V. Por eso, hay que hacer una adaptación intermedia mediante un divisor de tensión con tres resistencias iguales tal y como se muestra en la Ilustración 37. En el punto A se obtiene un tercio del valor total, es decir, aproximadamente 3.3V.

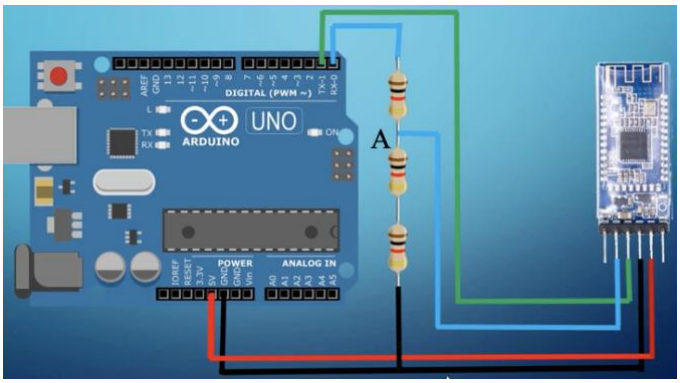

Ilustración 37. Conexionado de programación [32].

Una vez que se ha realizado el conexionado se procede a la configuración correspondiente de cada módulo mediante los comandos AT. Los comandos AT son un tipo de comandos que sirven para configurar el módulo Bluetooth a través de un microcontrolador, un ordenador o con cualquier dispositivo que posea una comunicación serie (TX, RX). Son unas instrucciones que nos permiten cambiar los baudios del módulo, el PIN, el nombre, el role, etc. Para hacer la lectura de un dispositivo es tan simple como añadir después del comando '?', mientras que para asignar un valor hay que poner '=' al final del comando y el valor que se quiere asignar. En la Tabla 3 se muestran algunos de estos comandos.

Comando AT	**Descripción**	**Respuesta**
AT	Test de comunicación	OK
AT+VERSION	Versión del módulo	Retorna la versión del módulo
AT+BAUDx	Configura la velocidad de transmisión del módulo según el valor de "x"	9600 (valor por defecto)
AT+NAME	Configura el nombre con el que se visualizará el módulo, soporta 20 caracteres como máximo.	Nombre
AT+PSWD	Contraseña (PIN).	1234 (pin por defecto)
AT+ROLE	Rol del dispositivo: '0' esclavo o '1' maestro.	'0' o '1'
AT+RESET	Vuelve al modo usuario	OK
AT+CMODE	Averiguar el modo actual de conexión	Retorna el modo '0' o '1'
AT+BIND	Dirección actual	Retorna la dirección
AT+ADDR	Dirección MAC	Retorna la dirección MAC

Tabla 3. Comandos AT [32].

Una vez realizada la configuración se ha creado un código dependiendo de cual sea la función que desempeña cada módulo. En el caso de actuar como esclavo se utilizará el Arduino Nano por su tamaño reducido y teniendo en cuenta que estaría en contacto con el paciente y cuanto menos voluminoso sea mucho mejor. En cambio, para el módulo maestro que se encarga de procesar toda la información se utilizará el Arduino UNO ya que es indiferente el tamaño, es decir, siguiendo la misma lógica que para el prototipo Li-Fi.

Tal y como se puede ver en la Ilustración 38 los códigos que se emplean en este caso son mucho más sencillos ya que a diferencia del sistema Li-Fi no hay que codificar la información que se quiere enviar, solo hay que vincular los dos módulos. En el caso del esclavo (código de la izquierda) se lee la información obtenida por el sensor y la manda a través del enlace establecido y en el caso del esclavo (código de la derecha) recibe y proyecta la información que se obtiene.

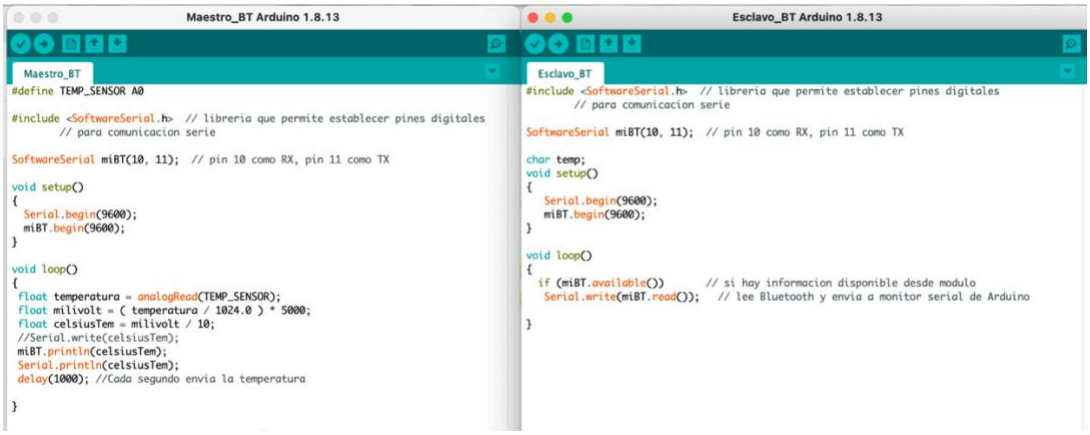

Ilustración 38. Códigos para el prototipo Bluetooth (izquierda maestro y derecha esclavo).

La Ilustración 39 muestra los dos circuitos utilizados tanto para el transmisor (circuito de abajo) como para el receptor (circuito de arriba) y los datos de temperatura que se reciben.

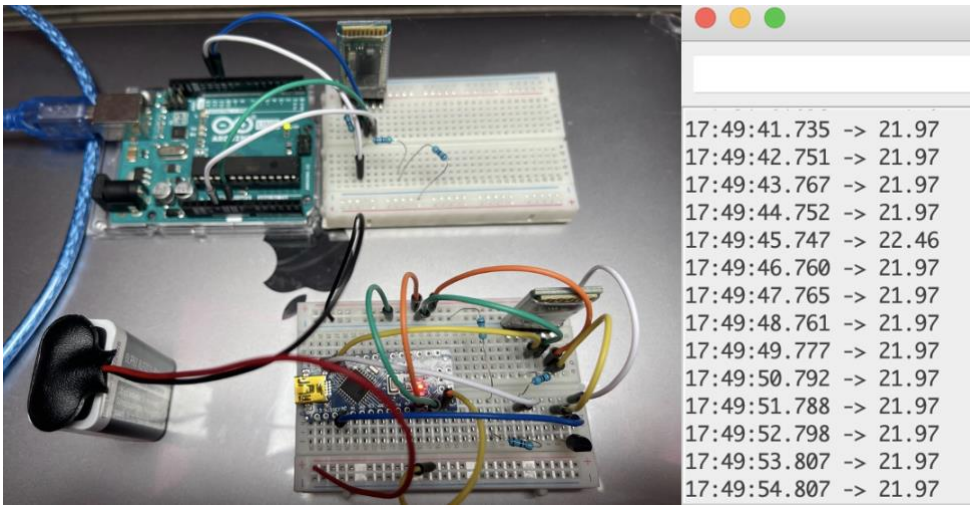

Ilustración 39. Montaje del esclavo y el maestro con los datos recibidos.

5.2.1. PRUEBAS DE COMUNICACIÓN DEL SISTEMA BLUETOOTH.

En las comunicaciones por Bluetooth se han simulado los mismos escenarios que en el apartado anterior para así realizar una mejor comparativa. En la Ilustración 40 se puede observar la señal que se emite con el módulo Bluetooth variando las velocidades en 300, 2400 y 9600 baudios respectivamente. Se sigue con el mismo formato, al lado izquierdo está la señal a un segundo por división y al lado derecho el zoom de dicha señal. Cuanto mayor es la velocidad, mayor es la distorsión que se produce en la señal, por eso con la velocidad de 300 baudios se ve una señal más cuadrada.

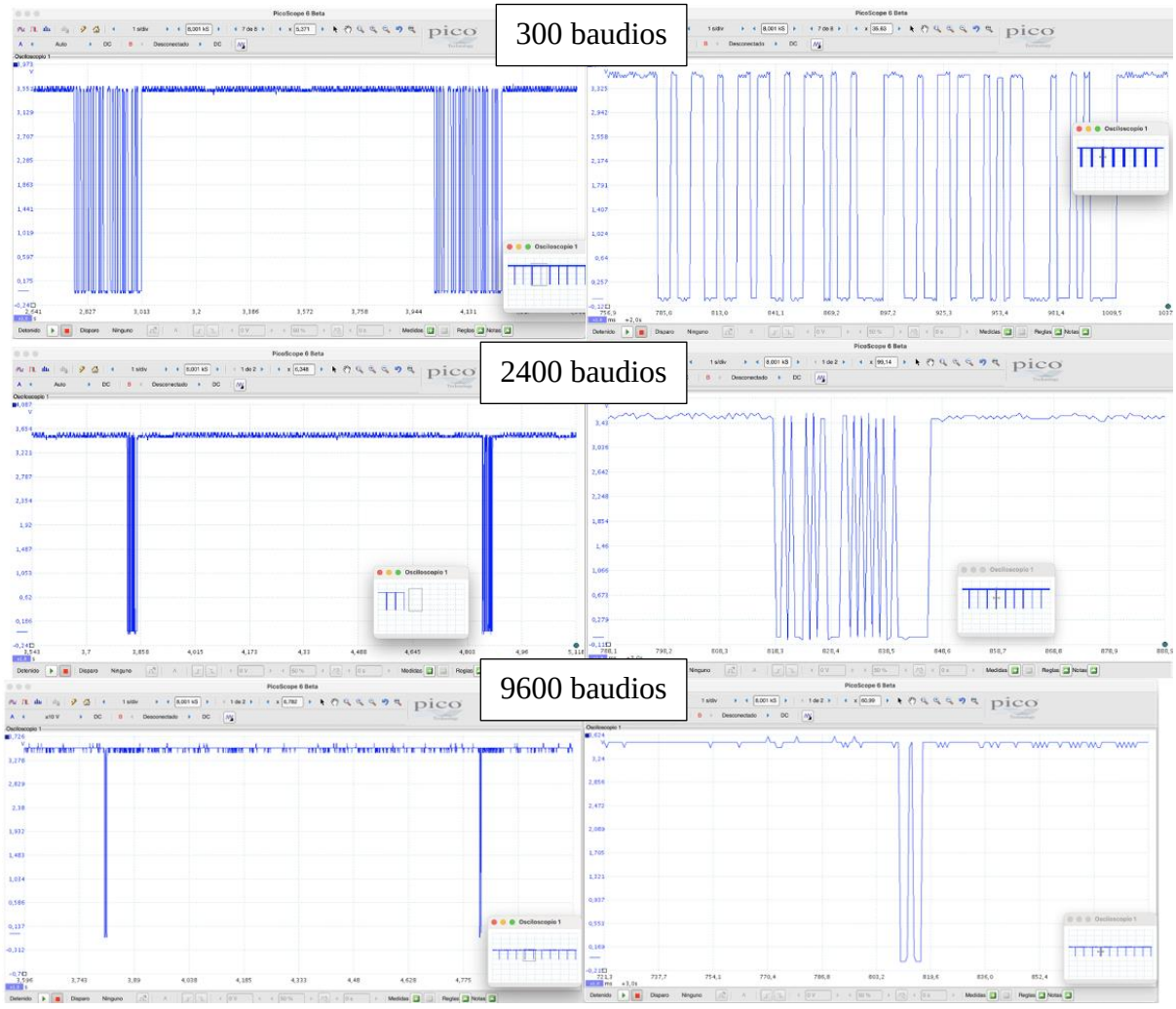

Ilustración 40. Salida de la señal de transmisión con distintas velocidades.

En el receptor, la señal recibida a 31 cm (Ilustración 41) y la recibida a 5 metros (Ilustración 42) son iguales a pesar de tener paredes de por medio en esta última distancia.

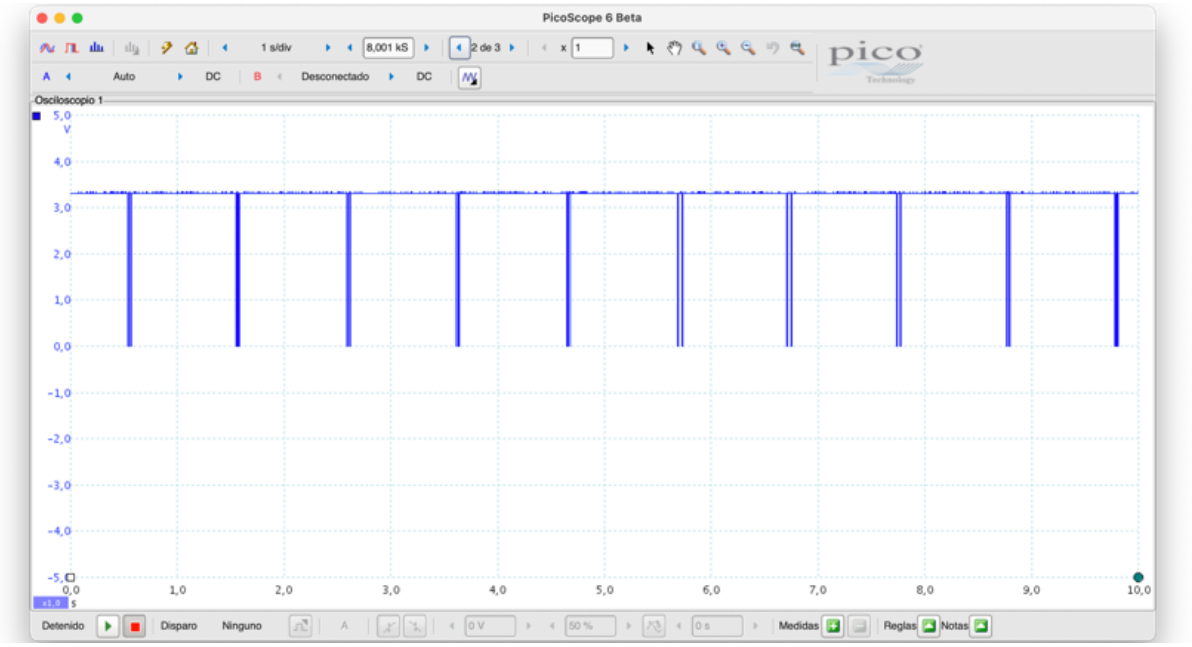

Ilustración 41. Señal recibida a 31 cm de distancia con velocidad de 2400 baudios.

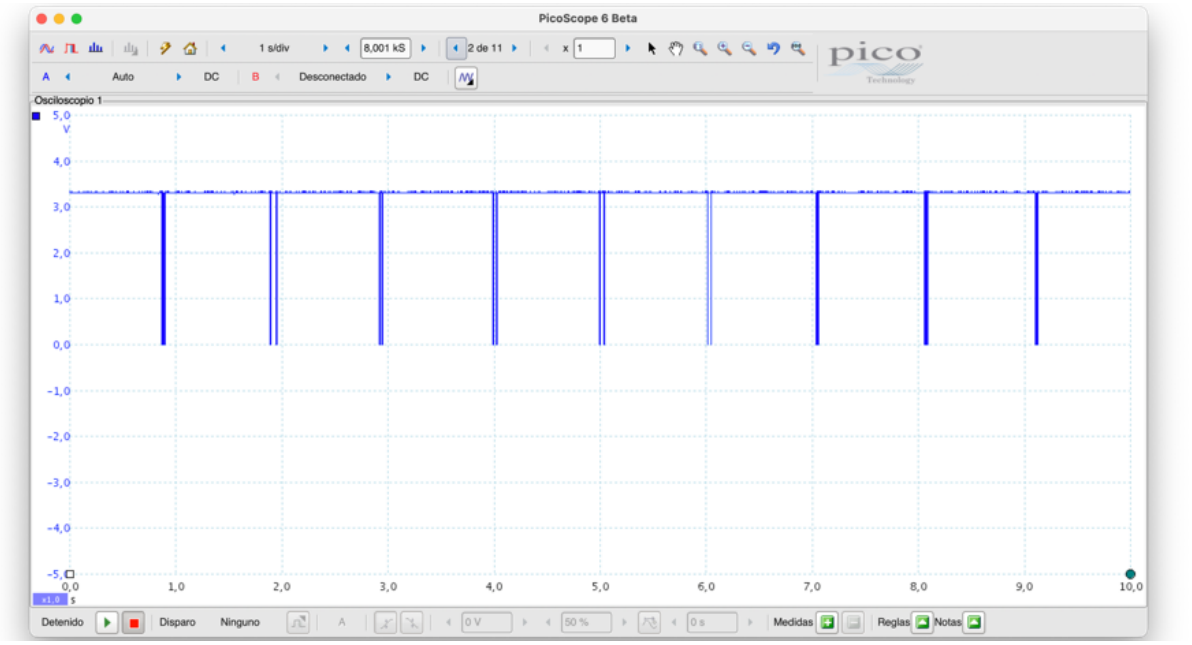

Ilustración 42. Señal recibida a 5 metros de distancia con velocidad de 2400 baudios.

En este sistema las interferencias causadas por la luz no influyen, ya que da igual lo iluminado que se encuentre el escenario que las señales enviadas y recibidas no sufren variación, pero si influyen las interferencias electromagnéticas. En la Ilustración 43 se puede observar que, al provocar una interferencia electromagnética puntual, con ayuda de un aparato que transmita en ondas radio, dificulta la lectura de la señal transmitida. Para producir la interferencia, se utilizó un taladro y se aprovechó un momento en el que había obras en la planta inferior de donde se realizaban las pruebas para que se notará más el efecto.

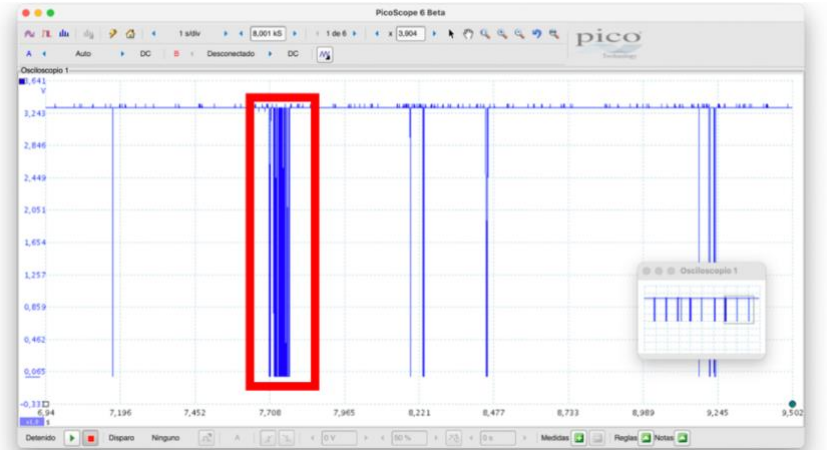

Ilustración 43. Interferencia electromagnética.

5.3. COMPARACIÓN CON AMBOS SISTEMAS.

Como hemos comentado y observado en los apartados anteriores, las comunicaciones por luz visible no se ven afectadas por las interferencias electromagnéticas, pero si por la luz a diferencia del sistema Bluetooth que le sucede lo contrario. Es decir, en el sistema Bluetooth da igual cuanto de iluminado este el escenario que no va a sufrir modificación alguna que si estuviera totalmente a oscuras como si que sucede con el Li-Fi. Por lo tanto, gracias al uso del sistema Li-Fi con sus correspondientes aplicaciones hace que todas las tecnologías que dan soporte a las aplicaciones sanitarias estén libres de cualquier interferencia electromagnética, a diferencia de si se emplea la tecnología Bluetooth.

En cuanto a la señal de los transmisores no se ve apenas diferencia entre ambos sistemas. Ya sea un sistema u otro actúan de la misma forma, a mayores velocidades mayor es la distorsión de la señal cuadrada.

En el receptor si se puede ver alguna diferencia en cuanto a los dos sistemas. La primera diferencia es que la distancia y el ángulo en el sistema Li-Fi afectan más para la recepción de la señal mientras que en el sistema de Bluetooth no, se puede alejar más recibiendo la misma cantidad de señal indiferentemente de si hay o no obstáculos de por medio, cosa que en el sistema Li-Fi es imposible que se reciba esta información ya que no atraviesa paredes.

Para que esto no sucediera y se pudiera superar la distancia máxima alcanzada, una solución sería colocar una matriz de LEDs o una bombilla LED en cada transmisor y con esto nos

aseguraríamos de iluminar perfectamente el escenario y que se recibiera la misma cantidad de señal.

Por otro lado, al variar las velocidades con las que se manda la información la señal recibida en ambos sistemas es la misma.

CAPÍTULO 6. CONCLUSIONES.

El presente trabajo ha examinado la comunicación entre dos transmisores y un receptor mediante la comunicación por luz visible creando para ello un prototipo que ha sido construido a través de diversos componentes electrónicos. Para poder realizar y desarrollar este prototipo se ha realizado una investigación de los antecedentes de la tecnología inalámbrica, su diferente tipología, así como el análisis de la red inalámbrica en un entorno hospitalario, entre otros, todo ello con el fin de obtener los conocimientos necesarios para conseguir implementar dicho sistema.

Para una mayor comprensión de la información se ha diseñado una aplicación interactiva con Processing y otra con Matlab. Processing se ha comprobado que es una aplicación más visual, en la cual se produce una simulación de una sala de hospital en la que se puede visualizar (debajo de cada cama) la temperatura correspondiente de cada sensor. En el caso de que la temperatura supere un umbral determinado (estado febril: temperatura superior a los 37 grados) se enciende una alerta. Del mismo modo, la cama al estar monitorizada avisa si ésta se encuentra libre u ocupada, esto es, si está recibiendo temperatura. Además, permite la opción de mostrar la información clínica de cada paciente a la vez que recoge en un documento .txt la temperatura, la cama y la hora a la que se ha obtenido, volcándose estos datos posteriormente en la aplicación Matlab con la que se ha trabajado también. Esta otra aplicación es más precisa y concisa en el sentido de que la información se vuelca en una gráfica temporal en referencia a la variable temperatura a partir de los datos recopilados por Processing.

En cuanto a las pruebas experimentales para que fueran más realistas se han diseñado dos estructuras, una para los transmisores y otra para el receptor. La estructura consiste en una base sólida con una barra vertical, la cual en el caso de los transmisores se puede variar la altura para así de esta forma realizar las medidas en dos ejes de desplazamiento: distancia y altura, que a su vez variará asimismo el ángulo en el que incide en el fotodetector.

Como se ha podido comprobar el mayor problema que tiene el sistema Li-Fi que se ha implementado son las interferencias causadas por la luz ambiente. Con la implementación de los dos escenarios, uno luminoso (con luz ambiente) y otro oscuro (ausencia de luz), se pudo observar como la señal recibida era menor cuanto más iluminado estaba el sistema por factores externos, es decir, por la luz solar. Además, si se incidía directamente en el fotodiodo con una fuente de luz externa o bien se ponía un obstáculo cortando la comunicación directa no se

recibía nada de información. Al contrario, nada de esto sucede con la tecnología Bluetooth, ya que es indiferente el escenario que se utilice y si hay obstáculos cortando la comunicación directa pues se recibe la misma cantidad de información.

Por otro lado, la distancia que alcanza el sistema Li-Fi se ha observado que supone un inconveniente a la hora de la recepción de la señal, pues a medida que se aumentaba la distancia la señal recibida era más débil. La mayor distancia que se obtuvo con ausencia de luz fue de 102 centímetros. En cuanto al ancho del haz se obtuvo un valor cercano a los 25º especificados por el fabricante del LED, manteniéndose dentro de los parámetros esperados. Al comparar los resultados con la tecnología Bluetooth se vio que no sucedía ya que recibía la misma señal a pesar de la distancia o al ancho del haz.

Tanto en el sistema Bluetooth como en el Li-Fi, al variar las velocidades de transmisión, las señales que se recibían en ambos sistemas era la misma.

En cuanto a las interferencias electromagnéticas, en las que se centra este proyecto, se verificó como el sistema Li-Fi no sufría variación en la recepción de la señal al provocar una interferencia puntual de radiofrecuencia, cosa que no sucedía con el sistema Bluetooth. Por ello, resulta de gran interés implementar el prototipo en zonas críticas donde haya un gran uso del espectro como es en el caso de los hospitales. En definitiva, en la implantación de las TIC en las instituciones sanitarias juegan un papel muy importante, pero no hay que olvidar la problemática que pueden causar ya que en ocasiones pueden provocar grandes daños no solo a los equipos médicos sino también a las personas.

De cara a una posible mejora en el rendimiento del sistema se podría sustituir los diodos por una bombilla LED para aumentar de esta forma el alcance máximo. Además, se podría hacer más eficiente el sistema añadiéndole más sensores para controlar además de la temperatura el oxigeno en sangre, la tensión y el ritmo cardiaco entre muchos otros.

Para finalizar las conclusiones a las que se ha llegado en la realización de este proyecto, cabe mencionar en último lugar que el objetivo planteado se ha cumplido pues se ha logrado satisfactoriamente la implementación del diseño de un sistema de transmisión por luz visible mediante el cual se envía a través de dos transmisores diferentes la temperatura de los sensores.

Bibliografía

[1] M. Navarro, «revista byte,» 6 Mayo 2020. [En línea]. Available: https://revistabyte.es/covid-19/hospital-12-de-octubre/.

[2] M. Garduño, «Así funcionan los sensores Bluetooth para la ultracongelación de vacunas,» *Forbes*, 5 Marzo 2021.

[3] J. Jiménez, «La utilidad real de que tu móvil controle de quién has estado cerca: esto es lo que puede hacer el bluetooth frente al coronavirus,» *Xataka,* 14 Abril 2020.

[4] O. M. N. Meléndez, «ANÁLISIS COMPARATIVO DE LA TECNOLOGÍA WIFI Y LIFI PARA LA SELECCIÓN ADECUADA EN LA FACULTAD DE CIENCIAS ADMINISTRATIVAS, GESTIÓN EMPRESARIAL E INFORMÁTICA,» GUARANDA, 2017.

[5] Y. Fernández, «Cómo saber la velocidad máxima teórica que puede dar tu router WiFi,» *Xataka*, 29 Enero 2021.

[6] J. M. López, «¿Llamadas de voz a través de la luz?,» *Hipertextual*, 10 Marzo 2021.

[7] S. J. Dinesh Khandal, de *Li-Fi (Light Fidelity): The Future Technology in Wireless Communication*, vol. 4, Yaipur, International Research Publications House, 2014.

[8] P. Rodriguez, «Comunicaciones por luz visible: Cuando los bits nos lleguen de las bombillas,» *Xataka smart home,* 5 Febrero 2013.

[9] D. R. Pérez Jiménez, D. J. A. Rabadán Borges, D. F. Delgado Rajó, J. F. Rufo Torres y A. Perera Casiano, «BALDUR: Comunicaciones ópticas en espectro visible».

[10] «pureLIFI,» [En línea]. Available: https://purelifi.com/case-study/lifi-in-the-healthcare-sector/.

[11] «Wikipedia,» 11 Diciembre 2020. [En línea]. Available: https://en.wikipedia.org/wiki/IEEE_802.15.

[12] S. Rajagopal, R. D. Roberts y S.-K. Lim, «IEEE 802.15.7 Visible Light Communication: Modulation Schemes and Dimming Support,» *IEEE,* Marzo 2012.

[13] Unknown, «Visible Light Communications - Technology and Standards,» 19 Junio 2014. [En línea]. Available: http://vlc-fiatlux.blogspot.com/2014/06/ieee-standard-802157-short-range.html.

[14] G. NAP, «Elementos Técnicos para la Gestión de Frecuencias en Espacios Complejos: Entornos Sanitarios,» Colegio Oficial de Ingenieros de Telecomunicación, Madrid, 2010.

[15] Anonymous, «EL ESPECTRO ELECTROMAGNÉTICO,» 30 Septiembre 2017. [En línea]. Available: https://microondas2017.blogspot.com/2017/09/el-espectro-electromagnetico.html.

[16] J. E. Álvarez, «Poniendo a prueba la tecnología LiFi en hospitales,» *Smartlighting A Journal on Lighting Technologies,* 12 Mayo 2020.

[17] H. Haas, L. Yin, Y. Wang y C. Chen, «What is LiFi?,» *Lightwave Technology,* vol. 34, 15 Marzo 2016.

[18] Asifunciona, «Características de los diodos LEDs,» Septiembre 2015. [En línea]. Available: http://www.asifunciona.com/tablas/leds/leds.htm.

[19] A. B. Beltran, «Implementación experimental de un enlace óptico inalámbrico para comunicaciones en la región visible (VLC),» Universitat Politècnica de València, Valencia, 2019.

[20] T. Instruments, «LM35 Precision Centigrade Temperature Sensors,» Diciembre 2107. [En línea]. Available: https://docs.rs-online.com/abdd/A700000006857169.pdf.

[21] D. M. F. Calderón, «Cuál es la temperatura normal de una persona,» 10 Marzo 2020. [En línea]. Available: https://www.miguelfernandezcalderon.com/temperatura-normal-de-una-persona/.

[22] L. LLamas, «Medir temperatura con arduino y sensor LM35,» 15 Julio 2015. [En línea]. Available: https://www.luisllamas.es/medir-temperatura-con-arduino-y-sensor-lm35/.

[23] B. Industries, «YouTube: LM35 Arduino - TEMPERATURA CORPORAL | PCBway,» 11 Septiembre 2019. [En línea]. Available: https://www.youtube.com/watch?v=zsqXbjzLkqE.

[24] B. L. Grandes, «Estudio del Estado del Arte de los sistemas de comunicaciones por luz visible (VLC),» Dep. Teoría de la Señal y Comunicaciones Escuela Técnica Superior de Ingeniería Universidad de Sevilla, Sevilla , 2016.

[25] DIYMakers, «ARDUINO + PROCESSING: PRIMEROS PASOS,» 5 Enero 2013. [En línea]. Available: http://diymakers.es/arduino-processing-primeros-pasos/.

[26] Hubor, «¿Qué es proteus?,» 2015. [En línea]. Available: https://www.hubor-proteus.com/proteus-pcb/proteus-pcb/2-proteus.html.

[27] Y. Abdullahi Badamasi, «The Working Principle Of An Arduino.,» *IEEE Xplore,* p. 4, 2014.

[28] Wikipedia, «Manchester code,» 12 Junio 2021. [En línea]. Available: https://en.wikipedia.org/wiki/Manchester_code.

[29] A. Research, «El mercado de IoT Bluetooth casi se cuadruplica para 2024, ya que el hogar inteligente supera los 800 millones de envíos de dispositivos,» 14 Enero 2020. [En línea]. Available: https://www.abiresearch.com/press/bluetooth-iot-market-set-nearly-quadruple-2024-smart-home-exceeds-800-million-device-shipments/.

[30] DIYMakers, 3 Febrero 2014. [En línea]. Available: http://diymakers.es/arduino-bluetooth/.

[31] L. LLamas, «Conectar arduino por Bluetooth con los módulos HC-05 ó Hc-06,» 27 Noviembre 2015. [En línea]. Available: https://www.luisllamas.es/conectar-arduino-por-bluetooth-con-los-modulos-hc-05-o-hc-06/.

[32] C. Online, «YouTube: Comunicación Maestro/ esclavo con dos módulos bluetooth y arduino.,» 23 Noviembre 2020. [En línea]. Available: https://www.youtube.com/watch?v=crxukoldLOs.

[33] E. Sarbazi y M. Uysal, «PHY Layer Performance Evaluation of the IEEE 802.15.7 Visible Light Communication Standard,» 2013. [En línea]. Available: https://ieeexplore-ieee-org.cuarzo.unizar.es:9443/stamp/stamp.jsp?tp=&arnumber=6777772.

[34] R. W. World, «RF Wireless World,» 2012. [En línea]. Available: https://www.rfwireless-world.com/Tutorials/LiFi-PHY-MAC-layer-frame-structures.html.

[35] «Cree, Icn.,» 2014. [En línea]. Available: https://docs.rs-online.com/3d78/0900766b815247e8.pdf.

[36] «Unisonic Technologies,» 2009. [En línea]. Available: https://pdf1.alldatasheet.com/datasheet-pdf/view/413060/UTC/2N3904.html.

[37] «OSRAM,» 16 Julio 2019. [En línea]. Available: https://docs.rs-online.com/4359/0900766b817157b6.pdf.

[38] «Microchip Technology Inc.,» 14 Mayo 2019. [En línea]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/MCP6001-1R-1U-2-4-1-MHz-Low-Power-Op-Amp-DS20001733L.pdf.

[39] Bourns, «Datasheet,» Abril 2014. [En línea]. Available: https://docs.rs-online.com/698e/0900766b8069ccf9.pdf.

[40] «IEEE Standard 802.15.7: Short – Range Wireless Optical Communication Using Visible Light – A Brief Overview,» *IEEE,* 19 Junio 2014.

[41] «RF Wireless World,» 2012. [En línea]. Available: https://www.rfwireless-world.com/Tutorials/LiFi-PHY-MAC-layer-frame-structures.html.

[42] Anónimo, «El Espectro Electromagnético,» 30 Septiembre 2017. [En línea]. Available: https://microondas2017.blogspot.com/2017/09/el-espectro-electromagnetico.html.

[43] Jecrespom, «Aprendiendo Arduino,» 27 Junio 2016. [En línea]. Available: https://aprendiendoarduino.wordpress.com/2016/06/27/arduino-uno-a-fondo-mapa-de-pines-2/.

ANEXO A. DETALLES DEL ESTÁNDAR IEEE 802.15.7

Detalles del funcionamiento de la capa física (PHY)

Modelo PHY I

El modelo PHY I se utiliza en exteriores con velocidades bajas de transmisión de decenas a cientos de Kb/s. En el diagrama de la Ilustración 44 se observa el funcionamiento del transmisor y receptor para este modelo.

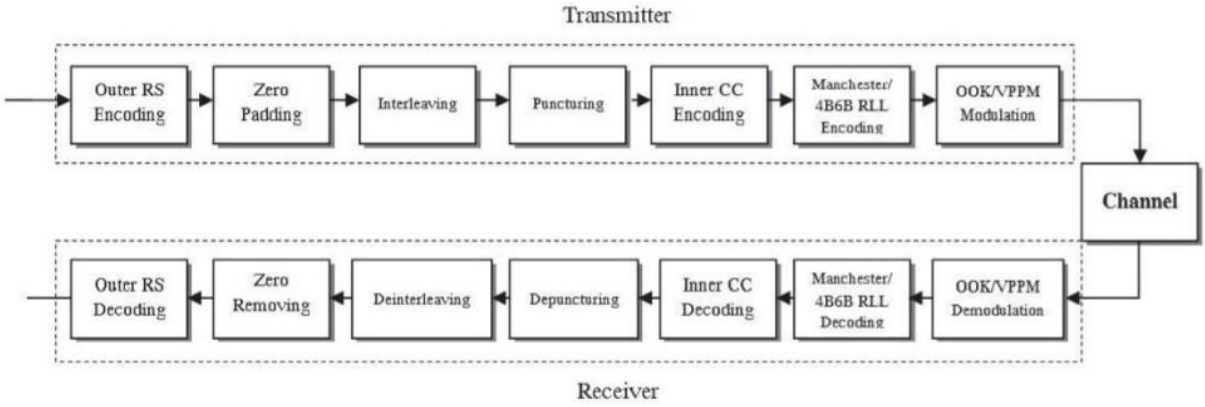

Ilustración 44. Diagrama de bloques del modelo PHY I [33].

Los bits de entrada son codificados utilizando la codificación denominada Reed Salomon (RS) y posteriormente rellenados con "ceros" a modo de padding, obteniendo de esta forma un intercalador entre símbolos. Una vez hecho esto se pasa por un codificador convolucional. El objetivo que se pretende alcanzar utilizando estas técnicas de tipo FEC es corregir los posibles errores de canal durante la transmisión además de optimizar el flujo de datos. [33].

La salida de este proceso se hace pasar por un codificador RLL (*Run-length limited)* aplicando la técnica de Manchester o 4B6B para transformar los datos en un conjunto de símbolos donde cada símbolo está compuesto por 2, 4 o 6 bits. En este proceso se pretende equipar de un mecanismo simple de sincronización de la transmisión, y del mismo modo, detectar retardos en la señal y controlar el ancho de banda de la transmisión. [33].

Finalmente, una vez que la señal está codificada, se procede a la modulación. Para ello, se utiliza la técnica OOK (*On-Off Keying*) o la técnica VPPM (*Variable Pulse Position Modulation*) y se envía por un canal con una sola fuente emisora de luz.

En el receptor, se procede a la demodulación y decodificación de la señal, aplicando el proceso descrito, pero en orden inverso.

En la Tabla 4 se indica el proceso descrito anteriormente con diferentes velocidades y modulaciones, observando como se comporta el modelo PHY I.

Modulation	RLL code	Optical clock rate	FEC		Data rate
			Outer code (RS)	Inner code (CC)	
OOK	Manchester	200 kHz	(15,7)	1/4	11.67 kb/s
			(15,11)	1/3	24.44 kb/s
			(15,11)	2/3	48.89 kb/s
			(15,11)	None	73.3 kb/s
			None	None	100 kb/s
VPPM	4B6B	400 kHz	(15,2)	None	35.56 kb/s
			(15,4)	None	71.11 kb/s
			(15,7)	None	124.4 kb/s
			None	None	266.6 kb/s

Tabla 4. PHY I modos de operación [12].

Modelo PHY II

El modelo PHY II se utiliza en interiores con velocidades medias que va en las decenas de Mb/s. De la misma manera se muestra en la Ilustración 45 el funcionamiento del transmisor y receptor para el modelo PHY II.

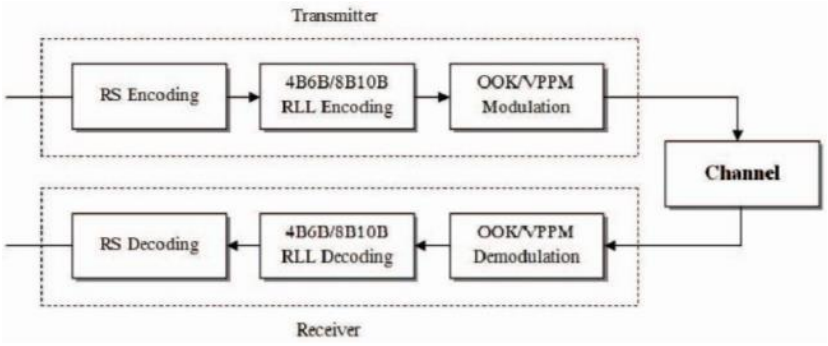

Ilustración 45. Diagrama de bloques del modelo PHY II [33].

Este diagrama de bloques es mucho mas sencillo y eficiente que el que se utiliza en el modelo PHY I, ya que en este esquema se suprime el relleno de caracteres y la codificación

convolucional. Las técnicas de modulación que utiliza este modelo son las mismas que en el caso del modelo PHY I.

En la Tabla 5 se pueden ver los distintos modos de operación para el modelo PHY II conforme a los parámetros utilizados para su implementación.

Modulation	RLL code	Optical clock rate	FEC	Data rate
VPPM	4B6B	3.75 MHz	RS(64,32)	1.25 Mb/s
			RS(160,128)	2 Mb/s
		7.5 MHz	RS(64,32)	2.5 Mb/s
			RS(160,128)	4 Mb/s
			None	5 Mb/s
OOK	8B10B	15 MHz	RS(64,32)	6 Mb/s
			RS(160,128)	9.6 Mb/s
		30 MHz	RS(64,32)	12 Mb/s
			RS(160,128)	19.2 Mb/s
		60 MHz	RS(64,32)	24 Mb/s
			RS(160,128)	38.4 Mb/s
		120 MHz	RS(64,32)	48 Mb/s
			RS(160,128)	76.8 Mb/s
			None	96 Mb/s

Tabla 5. PHY II modos de operación [12].

La diferencia que existe entre estos modelos es que el modelo PHY I está previsto para que se utilice en ambientes exteriores con velocidades entre 11,67 kb/s y 266,6 kb/s. Mientras que el modelo PHY II se prevé que sea utilizado en interior con velocidades entre 1,25 Mb/s y 96 Mb/s [12].

Modelo PHY III

El modelo PHY III a diferencia de los dos anteriores trabaja bajo un sistema que tiene múltiples entradas y salidas, es decir, una transmisión MIMO (*Multiple-input Multiple-output*). De esta forma se puede utilizar en aplicaciones con muchas más fuentes de luz. En la Ilustración 46 se muestra el diagrama de bloques que rige este modelo.

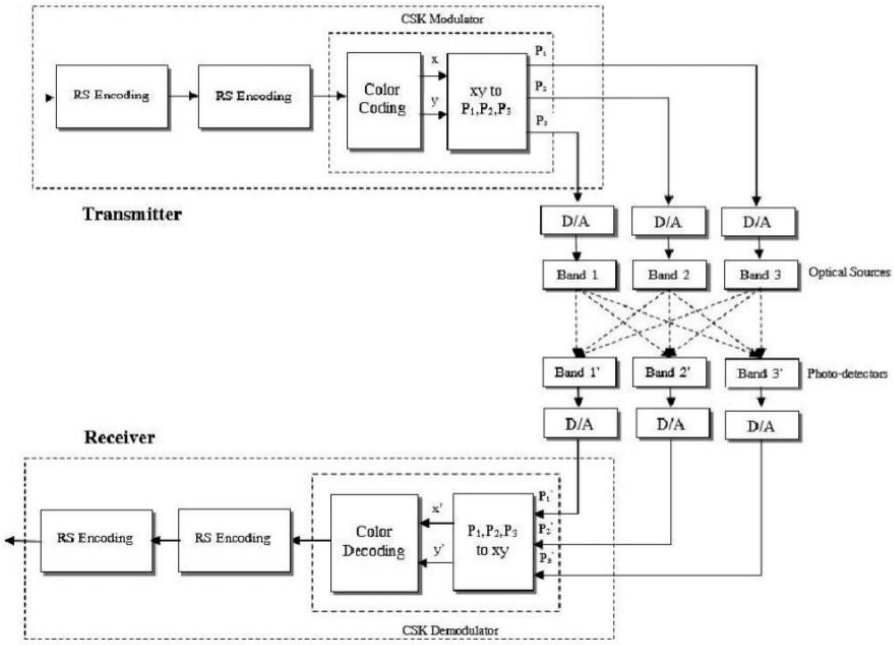

Ilustración 46. Diagrama de bloques del modelo PHY III [33].

De igual manera que los dos modelos anteriores, los datos de entrada ingresan en un codificador Reed-Solomon que transforma la cadena de bits en tramas cortas que luego serán nuevamente codificadas con Reed-Solomon. La diferencia está en la modulación, que en este caso utiliza la modulación CSK (*Color Shift Keying*) que modula mediante incrustación de color.

El espectro de luz visible puede dividirse en 7 grupos (rojo, naranja, amarillo, verde, azul, índigo y violeta) y a cada uno de estos grupos se le puede asignar un código específico de la forma [x - y]. El modulador CSK trabaja con 3 de esos 7 grupos, es decir, con los puntos respectivos en [x - y], y forma los vértices de un triángulo formando así las constelaciones que son necesarias [33].

Los datos que ingresan al modulador son analizados con la función “Log(M)”, donde “M” representa el tamaño de la modulación. Los datos de entrada son tomados de a tres y mapeados cada uno a un valor de [x - y] y posteriormente pasados a valores RGB. Por último, se normaliza la intensidad de cada color y se procede a la transmisión.

El receptor cuenta con tres fotorreceptores donde cada uno detecta las longitudes de onda correspondientes a cada color. Entonces se realiza un trabajo inverso del que se realizó en el transmisor, transformando las longitudes de onda en las ternas [x – y] correspondientes para cada color y a la recuperación de cada símbolo correspondiente dentro de la constelación. Por último, la información se decodifica dos veces con el método Reed-Solomon y se recupera la información enviada. [33].

En la Tabla 6 se ilustran los distintos modos de operación para el modelo PHY III.

Modulation	Optical clock rate	FEC	Data rate
4-CSK	12 MHz	RS(64,32)	12 Mb/s
8-CSK		RS(64,32)	18 Mb/s
4-CSK	24 MHz	RS(64,32)	24 Mb/s
8-CSK		RS(64,32)	36 Mb/s
16-CSK		RS(64,32)	48 Mb/s
8-CSK		None	72 Mb/s
16-CSK		None	96 Mb/s

Tabla 6. PHY III modos de operación [12].

Una trama PHY de Li-Fi está compuesta por un encabezado de sincronización, un encabezado PHY y la carga útil o payload. El encabezado de sincronización hace posible que el receptor sincronice el reloj óptico y se posicione en la ráfaga de bits recibida, mientras que el encabezado PHY contiene información sobre la transmisión. Por último, en el payload se transmiten los datos útiles de la trama PHY.

En la Ilustración 47 se puede apreciar como los diferentes tipos de capa física que conviven, pero no operan entre sí. PHY I y PHY II ocupan diferentes regiones en el espectro de la frecuencia (multiplexación en frecuencia), mientras que PHY II y PHY III ocupan la misma parte del espectro, pero usando diferencia tasa de datos y óptica.

No todos los dispositivos ópticos soportan las múltiples bandas de frecuencia que se necesitan para PHY III. Por ello, los dispositivos PHY III son compatibles con PHY II, para de este modo, detectar otros dispositivos.

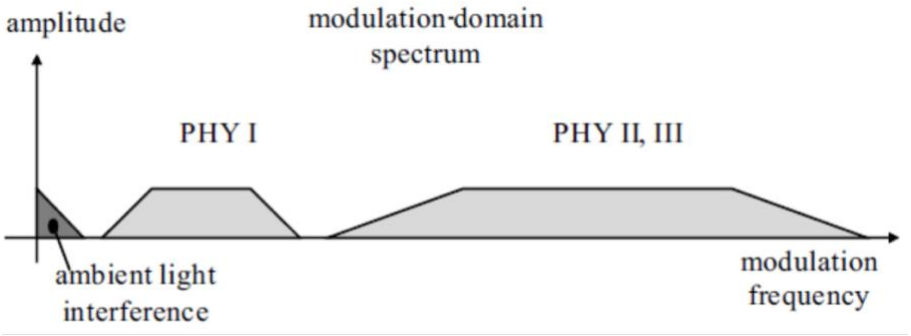

Ilustración 47. Separación de los tipos de PHY en la modulación por dominio [13].

Detalles del funcionamiento de la capa de acceso al medio (MAC)

Descripción de los distintos modos de la capa de acceso al medio (MAC)

Para lograr los objetivos de la capa de acceso al medio se definen cuatro modos de trabajo para la transmisión de datos y el manejo de las tramas que son: el modo sencillo o individual, el modo empaquetado, el modo "ráfaga" y el modo OOK regulado.

- **Modo sencillo o individual (*single mode*).**
  En este modo de funcionamiento se transporta solamente una unidad de datos (*Protocol Data Unit* o PDU) por trama, y para ello se utiliza la comunicación de datos de pequeño tamaño como puede ser un acuse de recibo (ACK) o una señal de baliza (*beacon*).
- **Modo de empaquetado (*packed mode*).**
  A diferencia del modo sencillo, en esta modalidad se transportan varios PDU por trama, los cuales van dirigidos hacia una misma dirección de destino. De este modo, se incrementa la eficiencia de la capa de acceso al medio (MAC) debido a que los encabezados PHY y MAC repetitivos se eliminan para el mismo destino.
- **Modo de ráfaga (*burst mode*).**
  En esta modalidad, se incrementa la eficiencia y se mejora el rendimiento ya que la trama utiliza un preámbulo reducido después de enviar la primera trama; y, además, se utiliza un separador RIFS (*Reduced Inter-Frame Space*) en lugar de un separador SIFS (*Short Inter-Frame Space*) para separar las tramas.
- **Modo On-Off Keying regulado (*dimmed OOK mode*).**
  Este ultimo modo se utiliza para la transmisión de datos en aplicaciones de calibración de intensidad luminosa.

En la Ilustración 48 se puede apreciar una comparativa de las distintas tramas MAC dependiendo del modo que se este utilizando. El último modo, OOK regulado, no se incluye debido a que no transporta datos, solo se utiliza para calibrar la señal.

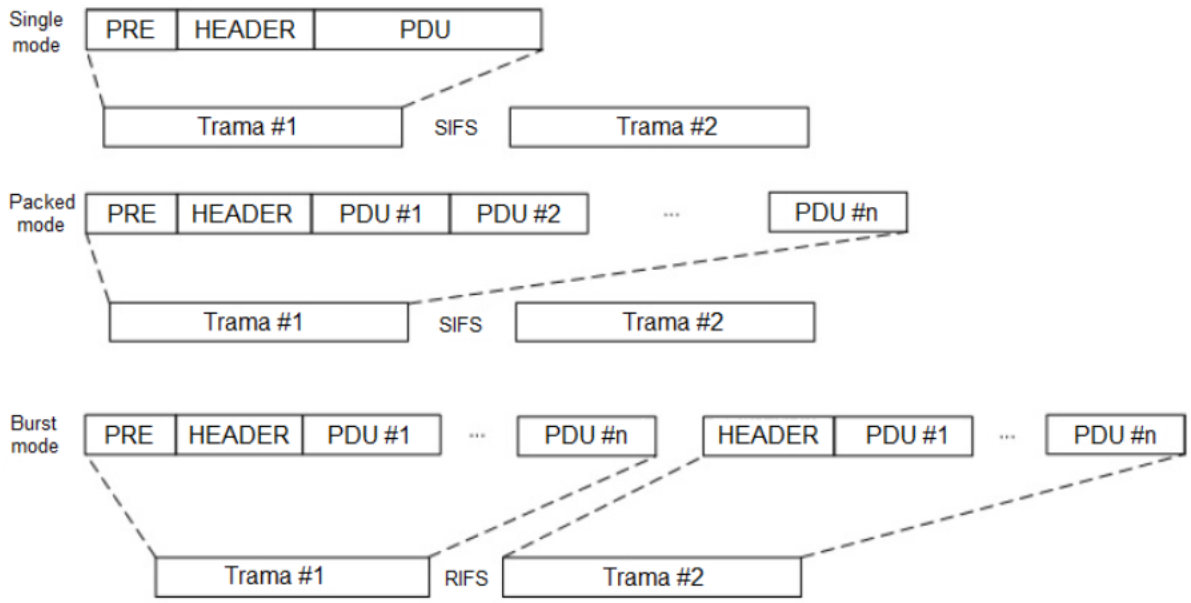

Ilustración 48. Modos de trabajo de la capa MAC en Li-Fi [34].

Una trama MAC Li-Fi está compuesta por un encabezado, una unidad de datos y un pie de página. El encabezado está compuesto por distintos campos que brindan información respecto del tipo de trama, secuenciamiento y direcciones físicas de origen y destino. Por otro lado, la unidad de datos contiene el payload de la trama, es decir, los datos del protocolo subyacente. Y, por último, el footer contiene información para mecanismos destinados al control de la integridad de la trama.

ANEXO B. CARACTERÍSTICAS DE LOS COMPONENTES.

Para la elección del LED se tiene en cuenta principalmente la frecuencia de encendido y apagado que limita la máxima velocidad de transmisión y ésta debe de ser capaz de realizar la transición entre encendido y apagado a velocidades muy altas. Por otra parte, al ser parte de la iluminación de ambiente debe tener luz visible y al mismo tiempo una tonalidad adecuada. Por estos motivos, se ha decidido elegir un LED blanco donde sus características se muestran en la Tabla 7.

Corriente directa	25 mA
Corriente directa máxima	100 mA
Tensión directa	3.2 - 4 V
Temperatura de operación	-40 - 95 ºC
Ángulo de visión	25º
Intensidad de Iluminación	23500 mcd

Tabla 7. Características del LED [35].

Con unos criterios similares a los del LED se ha optado por un transistor bipolar de silicio 2N3904 NPN, cuyas características se pueden ver en la Tabla 8.

Intensidad colector	200 mA
Ganancia de corriente (h_{fe})	80
Voltaje $_{(CE)}$ saturación	0.25 V
Voltaje $_{(BE)}$ saturación	0.9 V
Frecuencia	450 MHz

Tabla 8. Características del transistor referidos a una corriente de 30 mA en el LED [36].

El fotodiodo debe de ser capaz de capturar las señales que se van a transmitir. En cuanto a las características del fotodetector hay que destacar el tiempo de subida, la corriente en cortocircuito, la corriente en oscuridad, la capacidad de la unión, el ángulo de media sensibilidad y el rango de longitudes de onda. Entre estas características los valores que se desean deberían de ser como los que se detallan a continuación:

- **Corriente de oscuridad.**

  Debe de ser un valor bajo ya que es una de las fuentes de ruido más importantes y, por lo tanto, debe de tenerse en cuenta en la elección del detector.
- **La corriente en cortocircuito.**

Tiene que ser lo más alta posible para que la resistencia de feedback sea la menor posible y, por lo tanto, tenga la mejor respuesta posible respecto al ruido y la frecuencia.

- **Tiempo de subida.**

  Debe de ser un valor suficientemente bajo para que el fotodetector sea capaz de captar las señales transmitidas correctamente. Además, se tiene en cuenta que para un sistema óptico el tiempo de subida determina el ancho de banda, f3dB, mediante la relación siguiente:

$$Rise\ time = \frac{0.35}{f_{3dB}} \tag{10}$$

- **Ancho de banda.**

  Se calcula mediante la relación que se muestra a continuación:

$$f_{3dB} = \frac{1}{2\pi * R_L * C} \tag{11}$$

- **Ángulo de media sensibilidad.**

  Cuando es mayor aumenta la sensibilidad del fotodiodo, pero al mismo tiempo aumenta la corriente de oscuridad.

- **Longitud de onda de trabajo.**

  Se debe de situar dentro del rango de emisión de nuestro LED. No solamente deberíamos mirar el rango, sino que se debería buscar la gráfica que indica el valor de respuesta en frecuencia y ajustar el máximo para que esté este dentro de nuestro rango.

Una vez que se han analizado todos los parámetros se ha elegido el fotodiodo BPW 34, cuyas características pueden contemplarse en la Tabla 9.

Tiempo de subida	0.02 µs
Corriente de cortocircuito	47 µA
Corriente de oscuridad	2 nA
Capacidad	72 pF
Ángulo de media sensibilidad	60º
Rango de longitud de onda	400 – 1100 nm

Tabla 9. Características del fotodiodo [37].

Otro componente fundamental del receptor óptico es el amplificador operacional ya que en la primera etapa transforma la corriente generada por el fotodetector en un nivel de tensión al mismo tiempo que amplifica la señal.

A la hora de seleccionar un amplificador para un sistema hay que tener diversos aspectos en cuenta, siendo entre ellos los mas importantes la limitación en la entrada/salida, la estabilidad y los errores V_{IO} e I_B. El V_{IO} representa la diferencia de tensión que hay entre las dos entradas diferenciales y la I_B la corriente que entra el circuito por estas entradas.

Los valores de errores no se calcularán, pero se va a escoger un amplificador con valores pequeños para los mismos para de esta forma asegurar el correcto funcionamiento del circuito. Finalmente, se ha elegido MCP6002, en la Tabla 10 se muestran las principales características.

Parámetros	**MCP6002**
Pendiente de salida	0.6 V/μs
Rango de tensión de entrada	V_{SS} -0.3 V , V_{DD} +0.3 V
Rango de tensión de salida	V_{DD} -25 mV , V_{SS} +25 mV
Máxima corriente de entrada de bias	1 pA
Máximo voltaje de offset a la entrada	7 mV
Corriente inactiva	100 μA
Capacidad de entrada en modo diferencial (C_D) y modo común (C_M)	6 pF || 3 pF
Producto de ganancia de ancho de banda	1 MHz
Rango de voltaje de alimentación	1.8 V – 5.5 V

Tabla 10. Características del amplificador MCP6002 [38].

El chip contiene en su encapsulado ocho pines de entrada/salida y una marca circular en el mismo para indicar el orden de los pines el cual se muestra en la siguiente imagen.

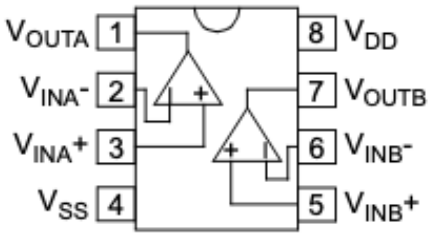

Ilustración 49. Conexionado del MCP6002 [38].

El potenciómetro que se ha seleccionado es de una vuelta y es del fabricante Bourns. En la Tabla 11 se muestran las especificaciones de este componente.

Resistencia máxima	50 kΩ
Tipo de montaje	Orificio pasante
Número de vueltas	1
Orientación	Ajuste superior
Potencia nominal	0.5 W
Serie	3386
Estilo de terminación	Pin
Tolerancia	± 10%
Coeficiente de temperatura	± 100 ppm/º C
Diámetro del Eje	3.15 mm
Longitud	9.53 mm
Mínima temperatura de funcionamiento	-55º C
Máxima temperatura de funcionamiento	+125º C

Tabla 11. Especificaciones del potenciómetro [39].

ANEXO C. CÓDIGO DE LA LIBRERÍA MANCHESTER.

En la siguientes imágenes se muestra la librería utilizada para la modulación.

Manchester.h

Manchester.h

```
#ifndef MANCHESTER_h
#define MANCHESTER_h

//timer scaling factors for different transmission speeds
#define MAN_300 0
#define MAN_600 1
#define MAN_1200 2
#define MAN_2400 3
#define MAN_4800 4
#define MAN_9600 5
#define MAN_19200 6
#define MAN_38400 7

/*
Timer 2 in the ATMega328 and Timer 1 in a ATtiny85 is used to find the time between
each transition coming from the demodulation circuit.
Their setup is for sampling the input in regular intervals.
For practical reasons we use power of 2 timer prescaller for sampling,
for best timing we use pulse lenght as integer multiple of sampling speed.
We chose to sample every 8 ticks, and pulse lenght of 48 ticks
thats 6 samples per pulse, lower sampling rate (3) will not work well for
innacurate clocks (like internal oscilator) higher sampling rate (12) will
cause too much overhead and will not work at higher transmission speeds.
This gives us 16000000Hz/48/256 = 1302 pulses per second (so it's not really 1200)
At different transmission speeds or on different microcontroller frequencies, clock prescaller is
adjusted
to be compatible with those values. We allow about 50% clock speed difference both ways
allowing us to transmit even with up to 100% in clock speed difference
*/

// added by caoxp@github
//
// the sync pulse amount for transmitting and receiving.
// a pulse means : HI,LO   or  LO,HI
// usually SYNC_PULSE_MAX >= SYNC_PULSE_DEF + 2
//         SYNC_PULSE_MIN <= SYNC_PULSE_DEF + 2
//  consider the pulses rising when starting transmitting.
//  SYNC_PULSE_MIN should be much less than SYNC_PULSE_DEF
//  all maximum of 255
#define     SYNC_PULSE_MIN  1
#define     SYNC_PULSE_DEF  3
#define     SYNC_PULSE_MAX  5

//#define       SYNC_PULSE_MIN  10
//#define       SYNC_PULSE_DEF  14
//#define       SYNC_PULSE_MAX  16

//define to use 1 or 0 to sync
// when using 1 to sync, sending SYNC_PULSE_DEF 1's , and send a 0 to start data.
//                       and end the transimitting by three 1's
// when using 0 to sync, sending SYNC_PULSE_DEF 0's , and send a 1 to start data.
//                       and end the transimitting by three 0's
```

```
#define     SYNC_BIT_VALUE      0
//decoding not finished.
//#define     SYNC_BIT_VALUE      0


/*
        Signal timing, we take sample every 8 clock ticks

        ticks:   [0]-[8]--[16]-[24]-[32]-[40]-[48]-[56]-[64]-[72]-[80]-[88]-[96][104][112][120][128]
[136]
        samples:
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
        single:  |                   [--------|----------]
        double:  |                                      [-----------------|--------------------]
        signal:  |_____________________________                            _____________________
                 |                             |__________________________|

*/

//setup timing for receiver
#define MinCount        33  //pulse lower count limit on capture
#define MaxCount        65  //pulse higher count limit on capture
#define MinLongCount    66  //pulse lower count on double pulse
#define MaxLongCount    129 //pulse higher count on double pulse

//setup timing for transmitter
#define HALF_BIT_INTERVAL 3072 //(=48 * 1024 * 1000000 / 16000000Hz) microseconds for speed factor 0
(300baud)

//it's common to zero terminate a string or to transmit small numbers involving a lot of zeroes
//those zeroes may be mistaken for training pattern, confusing the receiver and resulting high packe
lost,
//therefore we xor the data with random decoupling mask
#define DECOUPLING_MASK 0b11001010

#define RX_MODE_PRE 0
#define RX_MODE_SYNC 1
#define RX_MODE_DATA 2
#define RX_MODE_MSG 3
#define RX_MODE_IDLE 4

#define TimeOutDefault -1 //the timeout in msec default blocks

#if defined(ARDUINO) && ARDUINO >= 100
  #include "Arduino.h"
#else
  #include "WProgram.h"
  #include <pins_arduino.h>
#endif

class Manchester
{
  public:
```

```
    Manchester(); //the constructor
    void setTxPin(uint8_t pin); //set the arduino digital pin for transmit.
    void setRxPin(uint8_t pin); //set the arduino digital pin for receive.

    void workAround1MhzTinyCore(uint8_t a = 1); //apply workaround for defect in tiny Core library
for 1Mhz
    void setupTransmit(uint8_t pin, uint8_t SF = MAN_1200); //set up transmission
    void setupReceive(uint8_t pin, uint8_t SF = MAN_1200); //set up receiver
    void setup(uint8_t Tpin, uint8_t Rpin, uint8_t SF = MAN_1200); //set up receiver

    void transmit(uint8_t data); //transmit 16 bits of data
    void transmitArray(uint8_t numBytes, uint8_t *data); // transmit array of bytes

    uint8_t decodeMessage(uint16_t m, uint8_t &id, uint8_t &data); //decode 8 bit payload and 4 bit
ID from the message, return 1 of checksum is correct, otherwise 0
    uint16_t encodeMessage(uint8_t id, uint8_t data); //encode 8 bit payload, 4 bit ID and 4 bit
checksum into 16 bit

    //wrappers for global functions
    void beginReceive(void);
    void beginReceiveArray(uint8_t maxBytes, uint8_t *data);
    uint8_t receiveComplete(void);
    uint8_t getMessage(void);
    void stopReceive(void);
    uint8_t speedFactor;
    uint16_t delay1;
    uint16_t delay2;

  private:
    void sendZero(void);
    void sendOne(void);
    uint8_t TxPin;
    uint8_t applyWorkAround1Mhz;
};//end of class Manchester

// Cant really do this as a real C++ class, since we need to have
// an ISR
extern "C"
{
    //set the arduino digital pin for receive. default 4.
    extern void MANRX_SetRxPin(uint8_t pin);

    //begin the timer used to receive data
    extern void MANRX_SetupReceive(uint8_t speedFactor = MAN_1200);

    // begin receiving 16 bits
    extern void MANRX_BeginReceive(void);

    // begin receiving a byte array
    extern void MANRX_BeginReceiveBytes(uint8_t maxBytes, uint8_t *data);

    // true if a complete message is ready
    extern uint8_t MANRX_ReceiveComplete(void);

    // fetch the received message
    extern uint8_t MANRX_GetMessage(void);

    // stop receiving data
    extern void MANRX_StopReceive(void);
}

extern Manchester man;

#endif
```

Manchester.cpp

Manchester.cpp

```
#include "Manchester.h"

static int8_t RxPin = 255;

static int16_t rx_sample = 0;
static int16_t rx_last_sample = 0;
static uint8_t rx_count = 0;
static uint8_t rx_sync_count = 0;
static uint8_t rx_mode = RX_MODE_IDLE;

static uint16_t rx_manBits = 0; //the received manchester 32 bits
static uint8_t rx_numMB = 0; //the number of received manchester bits
static uint8_t rx_curByte = 0;

static uint8_t rx_maxBytes = 2;
static uint8_t rx_default_data[2];
static uint8_t* rx_data = rx_default_data;

Manchester::Manchester() //constructor
{
  applyWorkAround1Mhz = 0;
}


void Manchester::setTxPin(uint8_t pin)
{
  TxPin = pin; // user sets the digital pin as output
  pinMode(TxPin, OUTPUT);
}


void Manchester::setRxPin(uint8_t pin)
{
  ::RxPin = pin; // user sets the digital pin as output
  pinMode(::RxPin, INPUT);
}

void Manchester::workAround1MhzTinyCore(uint8_t a)
{
  applyWorkAround1Mhz = a;
}

void Manchester::setupTransmit(uint8_t pin, uint8_t SF)
{
  setTxPin(pin);
  speedFactor = SF;
  //we don't use exact calculation of passed time spent outside of transmitter
  //because of high ovehead associated with it, instead we use this
  //emprirically determined values to compensate for the time loss

  #if F_CPU == 1000000UL
    uint16_t compensationFactor = 88; //must be divisible by 8 for workaround
  #elif F_CPU == 8000000UL
```

```
    uint16_t compensationFactor = 12;
  #else //16000000Mhz
    uint16_t compensationFactor = 4;
  #endif

#if (F_CPU == 80000000UL) || (F_CPU == 160000000)   // ESP8266 80MHz or 160 MHz
  delay1 = delay2 = (HALF_BIT_INTERVAL >> speedFactor) - 2;
#else
  delay1 = (HALF_BIT_INTERVAL >> speedFactor) - compensationFactor;
  delay2 = (HALF_BIT_INTERVAL >> speedFactor) - 2;

  #if F_CPU == 1000000UL
    delay2 -= 22; //22+2 = 24 is divisible by 8
    if (applyWorkAround1Mhz) { //definition of micro delay is broken for 1MHz speed in tiny
cores as of now (May 2013)
      //this is a workaround that will allow us to transmit on 1Mhz
      //divide the wait time by 8
      delay1 >>= 3;
      delay2 >>= 3;
    }
  #endif
#endif
}


void Manchester::setupReceive(uint8_t pin, uint8_t SF)
{
  setRxPin(pin);
  ::MANRX_SetupReceive(SF);
}


void Manchester::setup(uint8_t Tpin, uint8_t Rpin, uint8_t SF)
{
  setupTransmit(Tpin, SF);
  setupReceive(Rpin, SF);
}


void Manchester::transmit(uint8_t data)
{
  uint8_t byteData[2] = {2, data};
  transmitArray(2, byteData);
}

/*
The 433.92 Mhz receivers have AGC, if no signal is present the gain will be set
to its highest level.

In this condition it will switch high to low at random intervals due to input noise.
A CRO connected to the data line looks like 433.92 is full of transmissions.

Any ASK transmission method must first sent a capture signal of 101010........
```

```
When the receiver has adjusted its AGC to the required level for the transmisssion
the actual data transmission can occur.

We send 14 0's 1010... It takes 1 to 3 10's for the receiver to adjust to
the transmit level.

The receiver waits until we have at least 10 10's and then a start pulse 01.
The receiver is then operating correctly and we have locked onto the transmission.
*/
void Manchester::transmitArray(uint8_t numBytes, uint8_t *data)
{

#if SYNC_BIT_VALUE
  for( int8_t i = 0; i < SYNC_PULSE_DEF; i++) //send capture pulses
  {
    sendOne(); //end of capture pulses
  }
  sendZero(); //start data pulse
#else
  for( int8_t i = 0; i < SYNC_PULSE_DEF; i++) //send capture pulses
  {
    sendZero(); //end of capture pulses
  }
  sendOne(); //start data pulse
#endif

  // Send the user data
  for (uint8_t i = 0; i < numBytes; i++)
  {
    uint16_t mask = 0x01; //mask to send bits
    uint8_t d = data[i] ^ DECOUPLING_MASK;
    for (uint8_t j = 0; j < 8; j++)
    {
      if ((d & mask) == 0)
        sendZero();
      else
        sendOne();
      mask <<= 1; //get next bit
    }//end of byte
  }//end of data

  // Send 3 terminatings 0's to correctly terminate the previous bit and to turn the
transmitter off
#if SYNC_BIT_VALUE
  sendOne();
  sendOne();
  sendOne();
#else
  sendZero();
  sendZero();
  sendZero();
#endif
}//end of send the data
```

```
void Manchester::sendZero(void)
{
  delayMicroseconds(delay1);
  digitalWrite(TxPin, HIGH);

  delayMicroseconds(delay2);
  digitalWrite(TxPin, LOW);
}//end of send a zero


void Manchester::sendOne(void)
{
  delayMicroseconds(delay1);
  digitalWrite(TxPin, LOW);

  delayMicroseconds(delay2);
  digitalWrite(TxPin, HIGH);
}//end of send one

//TODO use repairing codes perhabs?
//http://en.wikipedia.org/wiki/Hamming_code

/*
    format of the message including checksum and ID

    [0][1][2][3][4][5][6][7][8][9][a][b][c][d][e][f]
    [    ID    ][ checksum ][         data         ]
                  checksum = ID xor data[7:4] xor data[3:0] xor 0b0011

*/

//decode 8 bit payload and 4 bit ID from the message, return true if checksum is correct,
otherwise false
uint8_t Manchester::decodeMessage(uint16_t m, uint8_t &id, uint8_t &data)
{
  //extract components
  data = (m & 0xFF);
  id = (m >> 12);
  uint8_t ch = (m >> 8) & 0b1111; //checksum received
  //calculate checksum
  uint8_t ech = (id ^ data ^ (data >> 4) ^ 0b0011) & 0b1111; //checksum expected
  return ch == ech;
}

//encode 8 bit payload, 4 bit ID and 4 bit checksum into 16 bit
uint16_t Manchester::encodeMessage(uint8_t id, uint8_t data)
{
  uint8_t chsum = (id ^ data ^ (data >> 4) ^ 0b0011) & 0b1111;
  uint16_t m = ((id) << 12) | (chsum << 8) | (data);
  return m;
}

void Manchester::beginReceiveArray(uint8_t maxBytes, uint8_t *data)
```

```
{
  ::MANRX_BeginReceiveBytes(maxBytes, data);
}

void Manchester::beginReceive(void)
{
  ::MANRX_BeginReceive();
}


uint8_t Manchester::receiveComplete(void)
{
  return ::MANRX_ReceiveComplete();
}


uint8_t Manchester::getMessage(void)
{
  return ::MANRX_GetMessage();
}


void Manchester::stopReceive(void)
{
  ::MANRX_StopReceive();
}

//global functions

#if defined( ESP8266 )
   volatile uint16_t ESPtimer = 0;
   void timer0_ISR (void);
#endif

void MANRX_SetupReceive(uint8_t speedFactor)
{
  pinMode(RxPin, INPUT);
  //setup timers depending on the microcontroller used

  #if defined( ESP8266 )
   #if F_CPU == 80000000
      ESPtimer = (512 >> speedFactor) * 80;  // 8MHZ, 300us for MAN_300, 128us for MAN_1200
   #elif F_CPU == 160000000
      ESPtimer = (512 >> speedFactor) * 160;
   #endif

   noInterrupts();
   timer0_isr_init();
   timer0_attachInterrupt(timer0_ISR);
   timer0_write(ESP.getCycleCount() + ESPtimer); //80Mhz -> 128us
   interrupts();
  #elif defined( __AVR_ATtiny25__ ) || defined( __AVR_ATtiny45__ ) ||
defined( __AVR_ATtiny85__ )
```

```
    /*
    Timer 1 is used with a ATtiny85.
    http://www.atmel.com/Images/Atmel-2586-AVR-8-bit-Microcontroller-ATtiny25-ATtiny45-
ATtiny85_Datasheet.pdf page 88
    How to find the correct value: (OCRxA +1) = F_CPU / prescaler / 1953.125
    OCR1C is 8 bit register
    */

    #if F_CPU == 1000000UL
      TCCR1 = _BV(CTC1) | _BV(CS12); // 1/8 prescaler
      OCR1C = (64 >> speedFactor) - 1;
    #elif F_CPU == 8000000UL
      TCCR1 = _BV(CTC1) | _BV(CS12) | _BV(CS11) | _BV(CS10); // 1/64 prescaler
      OCR1C = (64 >> speedFactor) - 1;
    #elif F_CPU == 16000000UL
      TCCR1 = _BV(CTC1) | _BV(CS12) | _BV(CS11) | _BV(CS10); // 1/64 prescaler
      OCR1C = (128 >> speedFactor) - 1;
    #elif F_CPU == 16500000UL
      TCCR1 = _BV(CTC1) | _BV(CS12) | _BV(CS11) | _BV(CS10); // 1/64 prescaler
      OCR1C = (132 >> speedFactor) - 1;
    #else
    #error "Manchester library only supports 1mhz, 8mhz, 16mhz, 16.5Mhz clock speeds on
ATtiny85 chip"
    #endif

    OCR1A = 0; // Trigger interrupt when TCNT1 is reset to 0
    TIMSK |= _BV(OCIE1A); // Turn on interrupt
    TCNT1 = 0; // Set counter to 0

  #elif defined( __AVR_ATtiny2313__ ) || defined( __AVR_ATtiny2313A__ ) ||
defined( __AVR_ATtiny4313__ )

    /*
    Timer 1 is used with a ATtiny2313.
    http://www.atmel.com/Images/doc2543.pdf page 107
    How to find the correct value: (OCRxA +1) = F_CPU / prescaler / 1953.125
    OCR1A/B are 8 bit registers
    */

    #if F_CPU == 1000000UL
      TCCR1A = 0;
      TCCR1B = _BV(WGM12) | _BV(CS11); // reset counter on match, 1/8 prescaler
      OCR1A = (64 >> speedFactor) - 1;
    #elif F_CPU == 8000000UL
      TCCR1B = _BV(WGM12) | _BV(CS12) | _BV(CS11) | _BV(CS10); // 1/64 prescaler
      OCR1A = (64 >> speedFactor) - 1;
    #else
    #error "Manchester library only supports 1mhz, 8mhz clock speeds on ATtiny2313 chip"
    #endif

    OCR1B = 0; // Trigger interrupt when TCNT1 is reset to 0
    TIMSK |= _BV(OCIE1B); // Turn on interrupt
    TCNT1 = 0; // Set counter to 0
```

```
  #elif defined( __AVR_ATtiny24__ ) || defined( __AVR_ATtiny24A__ ) ||
defined( __AVR_ATtiny44__ ) || defined( __AVR_ATtiny44A__ ) || defined( __AVR_ATtiny84__ ) ||
defined( __AVR_ATtiny84A__ )

    /*
    Timer 1 is used with a ATtiny84.
    http://www.atmel.com/Images/doc8006.pdf page 111
    How to find the correct value: (OCRxA +1) = F_CPU / prescaler / 1953.125
    OCR1A is 8 bit register
    */

        TCCR1A = 0;

    #if F_CPU == 1000000UL
      TCCR1B = _BV(WGM12) | _BV(CS11); // 1/8 prescaler
      OCR1A = (64 >> speedFactor) - 1;
    #elif F_CPU == 8000000UL
      TCCR1B = _BV(WGM12) | _BV(CS11) | _BV(CS10); // 1/64 prescaler
      OCR1A = (64 >> speedFactor) - 1;
    #elif F_CPU == 16000000UL
      TCCR1B = _BV(WGM12) | _BV(CS11) | _BV(CS10); // 1/64 prescaler
      OCR1A = (128 >> speedFactor) - 1;
    #else
    #error "Manchester library only supports 1mhz, 8mhz, 16mhz on ATtiny84"
    #endif

    TIMSK1 |= _BV(OCIE1A); // Turn on interrupt
    TCNT1 = 0; // Set counter to 0

  #elif defined(__AVR_ATmega32U4__)

    /*
    Timer 3 is used with a ATMega32U4.
    http://www.atmel.com/Images/doc7766.pdf page 133
    How to find the correct value: (OCRxA +1) = F_CPU / prescaler / 1953.125
    OCR3A is 16 bit register
    */
    TCCR3A = 0;         // 2016, added, make it work for Leonardo
    TCCR3B = 0;         // 2016, added, make it work for Leonardo
    TCCR3B = _BV(WGM32) | _BV(CS31); // 1/8 prescaler
    #if F_CPU == 1000000UL
      OCR3A = (64 >> speedFactor) - 1;
    #elif F_CPU == 8000000UL
      OCR3A = (512 >> speedFactor) - 1;
    #elif F_CPU == 16000000UL
      OCR3A = (1024 >> speedFactor) - 1;
    #else
    #error "Manchester library only supports 1mhz, 8mhz, 16mhz on ATMega32U4"
    #endif

    TCCR3A = 0; // reset counter on match
    TIFR3 = _BV(OCF3A); // clear interrupt flag
    TIMSK3 = _BV(OCIE3A); // Turn on interrupt
```

```
    TCNT3 = 0; // Set counter to 0

  #elif defined(__AVR_ATmega8__)

    /*
    Timer/counter 1 is used with ATmega8.
    http://www.atmel.com/Images/Atmel-2486-8-bit-AVR-microcontroller-ATmega8_L_datasheet.pdf
page 99
    How to find the correct value: (OCRxA +1) = F_CPU / prescaler / 1953.125
    OCR1A is 16 bit register
    */

    TCCR1A = _BV(WGM12); // reset counter on match
    TCCR1B =  _BV(CS11); // 1/8 prescaler
    #if F_CPU == 1000000UL
      OCR1A = (64 >> speedFactor) - 1;
    #elif F_CPU == 8000000UL
      OCR1A = (512 >> speedFactor) - 1;
    #elif F_CPU == 16000000UL
      OCR1A = (1024 >> speedFactor) - 1;
    #else
    #error "Manchester library only supports 1Mhz, 8mhz, 16mhz on ATMega8"
    #endif
    TIFR = _BV(OCF1A);  // clear interrupt flag
    TIMSK = _BV(OCIE1A); // Turn on interrupt
    TCNT1 = 0; // Set counter to 0

  #else // ATmega328 is a default microcontroller


    /*
    Timer 2 is used with a ATMega328.
    http://www.atmel.com/dyn/resources/prod_documents/doc8161.pdf page 162
    How to find the correct value: (OCRxA +1) = F_CPU / prescaler / 1953.125
    OCR2A is only 8 bit register
    */

    TCCR2A = _BV(WGM21); // reset counter on match
    #if F_CPU == 1000000UL
      TCCR2B = _BV(CS21); // 1/8 prescaler
      OCR2A = (64 >> speedFactor) - 1;
    #elif F_CPU == 8000000UL
      TCCR2B = _BV(CS21) | _BV(CS20); // 1/32 prescaler
      OCR2A = (128 >> speedFactor) - 1;
    #elif F_CPU == 16000000UL
      TCCR2B = _BV(CS22); // 1/64 prescaler
      OCR2A = (128 >> speedFactor) - 1;
    #else
    #error "Manchester library only supports 8mhz, 16mhz on ATMega328"
    #endif
    TIMSK2 = _BV(OCIE2A); // Turn on interrupt
    TCNT2 = 0; // Set counter to 0
  #endif
```

```
} //end of setupReceive

void MANRX_BeginReceive(void)
{
  rx_maxBytes = 2;
  rx_data = rx_default_data;
  rx_mode = RX_MODE_PRE;
}

void MANRX_BeginReceiveBytes(uint8_t maxBytes, uint8_t *data)
{
  rx_maxBytes = maxBytes;
  rx_data = data;
  rx_mode = RX_MODE_PRE;
}

void MANRX_StopReceive(void)
{
  rx_mode = RX_MODE_IDLE;
}

uint8_t MANRX_ReceiveComplete(void)
{
  return (rx_mode == RX_MODE_MSG);
}

uint8_t MANRX_GetMessage(void)
{
  return (((int16_t)rx_data[0]) << 8) | (int16_t)rx_data[1];
}


void MANRX_SetRxPin(uint8_t pin)
{
  RxPin = pin;
  pinMode(RxPin, INPUT);
}//end of set transmit pin

void AddManBit(uint16_t *manBits, uint8_t *numMB,
               uint8_t *curByte, uint8_t *data,
               uint8_t bit)
{
  *manBits <<= 1;
  *manBits |= bit;
  (*numMB)++;
  if (*numMB == 16)
  {
    uint8_t newData = 0;
    for (int8_t i = 0; i < 8; i++)
    {
      // ManBits holds 16 bits of manchester data
      // 1 = LO,HI
      // 0 = HI,LO
```

```
      // We can decode each bit by looking at the bottom bit of each pair.
      newData <<= 1;
      newData |= (*manBits & 1); // store the one
      *manBits = *manBits >> 2; //get next data bit
    }
    data[*curByte] = newData ^ DECOUPLING_MASK;
    (*curByte)++;

    // added by caoxp @ https://github.com/caoxp
    // compatible with unfixed-length data, with the data length defined by the first byte.
        // at a maximum of 255 total data length.
    if( (*curByte) == 1)
    {
      rx_maxBytes = data[0];
    }

    *numMB = 0;
  }
}



#if defined( ESP8266 )
void ICACHE_RAM_ATTR timer0_ISR (void)
#elif defined( __AVR_ATtiny25__ ) || defined( __AVR_ATtiny45__ ) ||
defined( __AVR_ATtiny85__ )
ISR(TIMER1_COMPA_vect)
#elif defined( __AVR_ATtiny2313__ ) || defined( __AVR_ATtiny2313A__ ) ||
defined( __AVR_ATtiny4313__ )
ISR(TIMER1_COMPB_vect)
#elif defined( __AVR_ATtiny24__ ) || defined( __AVR_ATtiny24A__ ) ||
defined( __AVR_ATtiny44__ ) || defined( __AVR_ATtiny44A__ ) || defined( __AVR_ATtiny84__ ) ||
defined( __AVR_ATtiny84A__ )
ISR(TIM1_COMPA_vect)
#elif defined(__AVR_ATmega32U4__)
ISR(TIMER3_COMPA_vect)
#else
ISR(TIMER2_COMPA_vect)
#endif
{
  if (rx_mode < RX_MODE_MSG) //receiving something
  {
    // Increment counter
    rx_count += 8;

    // Check for value change
    //rx_sample = digitalRead(RxPin);
    // caoxp@github,
    // add filter.
    // sample twice, only the same means a change.
    static uint8_t rx_sample_0=0;
    static uint8_t rx_sample_1=0;
    rx_sample_1 = digitalRead(RxPin);
```

```
if( rx_sample_1 == rx_sample_0 )
{
  rx_sample = rx_sample_1;
}
rx_sample_0 = rx_sample_1;


//check sample transition
uint8_t transition = (rx_sample != rx_last_sample);

if (rx_mode == RX_MODE_PRE)
{
  // Wait for first transition to HIGH
  if (transition && (rx_sample == 1))
  {
    rx_count = 0;
    rx_sync_count = 0;
    rx_mode = RX_MODE_SYNC;
  }
}
else if (rx_mode == RX_MODE_SYNC)
{
  // Initial sync block
  if (transition)
  {
    if( ( (rx_sync_count < (SYNC_PULSE_MIN * 2) )  || (rx_last_sample == 1)  ) &&
        ( (rx_count < MinCount) || (rx_count > MaxCount)))
    {
      // First 20 bits and all 1 bits are expected to be regular
      // Transition was too slow/fast
      rx_mode = RX_MODE_PRE;
    }
    else if((rx_last_sample == 0) &&
            ((rx_count < MinCount) || (rx_count > MaxLongCount)))
    {
      // 0 bits after the 20th bit are allowed to be a double bit
      // Transition was too slow/fast
      rx_mode = RX_MODE_PRE;
    }
    else
    {
      rx_sync_count++;

      if((rx_last_sample == 0) &&
         (rx_sync_count >= (SYNC_PULSE_MIN * 2) ) &&
         (rx_count >= MinLongCount))
      {
        // We have seen at least 10 regular transitions
        // Lock sequence ends with unencoded bits 01
        // This is encoded and TX as HI,LO,LO,HI
        // We have seen a long low - we are now locked!
        rx_mode    = RX_MODE_DATA;
        rx_manBits = 0;
```

```
            rx_numMB   = 0;
            rx_curByte = 0;
          }
          else if (rx_sync_count >= (SYNC_PULSE_MAX * 2) )
          {
            rx_mode = RX_MODE_PRE;
          }
          rx_count = 0;
        }
      }
    }
    else if (rx_mode == RX_MODE_DATA)
    {
      // Receive data
      if (transition)
      {
        if((rx_count < MinCount) ||
           (rx_count > MaxLongCount))
        {
          // wrong signal lenght, discard the message
          rx_mode = RX_MODE_PRE;
        }
        else
        {
          if(rx_count >= MinLongCount) // was the previous bit a double bit?
          {
            AddManBit(&rx_manBits, &rx_numMB, &rx_curByte, rx_data, rx_last_sample);
          }
          if ((rx_sample == 1) &&
              (rx_curByte >= rx_maxBytes))
          {
            rx_mode = RX_MODE_MSG;
          }
          else
          {
            // Add the current bit
            AddManBit(&rx_manBits, &rx_numMB, &rx_curByte, rx_data, rx_sample);
            rx_count = 0;
          }
        }
      }
    }

    // Get ready for next loop
    rx_last_sample = rx_sample;
  }
#if defined( ESP8266 )
  timer0_write(ESP.getCycleCount() + ESPtimer);
#endif
}

Manchester man;
```

ANEXO D. CÓDIGO DEL ENTORNO DE VISUALIZACIÓN.

En las siguientes imágenes se muestra el código utilizado para el entorno de visualización en Processing.

Aplicacion_LiFi

```
import processing.serial.*; //Importamos la librería Serial

Serial port; //Nombre del puerto serie

PrintWriter output;  //Para crear el archivo de texto donde guardar los datos
PrintWriter datos;  //Para crear el archivo de texto donde guardar los datos


int xlogo=450;//Posición X de la imagen
int ylogo=100;//Posición Y de la imagen

int xcama1= 200; // Posición X de la cama 1
int ycama= 300; // Posicion Y de la cama 1
int xcama2= 700; // Posición X de la cama 2

int valor_cama;//Valor de la temperatura
int temperatura;

boolean boton = true;

int radio=250; //Radio del quadrado
int xcuadrado=450;  //Posición X de rect
int ycuadrado=450;  //Posición Y de rect

int valores[] = {0, 0, 0};  // Array para obtener los tres valores del puerto serie


void setup()
{
 println(Serial.list()); //Visualiza los puertos serie disponibles en la consola de abajo
 port = new Serial(this, Serial.list()[0], 9600); //Abre el puerto serie COM3

 output = createWriter("temeratura_datos.txt"); //Creamos el archivo de texto, que es guardado en la carpeta del programa
 datos = createWriter("datos.txt"); //Creamos el archivo de texto, que es guardado en la carpeta del programa


 size(900, 600); //Creamos una ventana de 900 píxeles de anchura por 600 píxeles de altura

 port.bufferUntil('\n');   // guardo datos hasta que encuentro salto de línea

}

void draw()
{
  printArray(valores);

    if(valores[0] == 3)
    {
    //Escribimos los datos de la temperatura con el tiempo (h/m/s) en el archivo de texto
    output.print(valores[1] + " ºC     ");
    output.print(hour( )+":");
    output.print(minute( )+":");
    output.println(second( ));
    output.println("Cama: " + valores[2]);
    output.println("");
    datos.println(valores[1]);
    datos.println(valores[2]);
    }
```

Aplicacion_LiFi

```
 if (boton){
   background(255);

    //Ponemos la imagen de nuestro logo
   imageMode(CENTER);//Esta función hace que las coordenadas de la imagne sean el centro de esta y no la esquina izquierda arriba
   PImage imagen=loadImage("logo.png");
   image(imagen,xlogo,ylogo,250,150);

   //Ponemos la imagen de las camas
   PImage imagen_cama=loadImage("cama.jpg");
   image(imagen_cama,xcama1,ycama, 200, 100); //Cama 1
   image(imagen_cama,xcama2,ycama, 200, 100); //Cama 2


   //Creamos un texto de color negro con la palabra Cama
   fill(0,0,0);
   PFont f = loadFont("Calibri-48.vlw");//Tipo de fuente
   textFont(f, 20);
   text("Cama 1", 120, 405);
   text("Cama 2", 670, 405);

   //Visualizamos la temperatura con un texto
   text("Temperatura =",50,455);
   text("Temperatura =",600,455);

   text("°C",205,455);
   text("°C",755,455);

   //Dibujamos un a esfera para colorear la temperatura
   noStroke();

   if (valores[0] == 3)
   {
     if (valores[1]>= 37)
     {
       fill(255,0,0); //Color rojo (r,g,b)
       if (valores[2] == 1)
       {
         rectMode(RADIUS); //Esta función hace que Width y Height de rect sea el radio (distancia desde el centro hasta un costado).
         rect(250,450,10,10); // Cama 1
       }
       else
       {
         rectMode(RADIUS);
         rect(800,450,10,10); // Cama 2
       }
     }
     else if (valores[1] < 37)
     {
       fill(0,0,255); // Color azul (r,g,b)
         rectMode(RADIUS); //Esta función hace que Width y Height de rect sea el radio (distancia desde el centro hasta un costado).
         rect(250,450,10,10); // Cama 1
         rect(800,450,10,10); // Cama 2
     }
   }

   //Dibujamos un circulo para colorear si la cama está libre o no
   fill(255,0,0); // Color rojo (r,g,b) porque esta ocupada
```

Aplicacion_LiFi

```
118     ellipseMode(CENTER);
119     ellipse(250,400,20,20); //Temperatura Cama 1
120     ellipse(800,400,20,20); //Temperatura Cama 2
121
122     // Ponemos el valor obtenido de la temperatura
123     fill(0);
124
125     if(valores[0]==3)
126     {
127       if(valores[2]==1)
128       {
129         text(valores[1], 180, 455);
130       }
131       else if (valores[2]==2)
132       {
133         text(valores[1],730,455);
134       }
135     }
136
137   }
138
139   //Ponemos la imagen de las graficas
140   PImage graf=loadImage("grafico.png");
141   image(graf,xcama1,500, 40, 30); //Cama 1
142   image(graf,xcama2,500, 40, 30); //Cama 3
143
144 }
145
146 void mousePressed()  //Cuando clicamos con el ratón
147 {
148       if(mouseX > xcama1-40 && mouseX < xcama1+30 && mouseY > 500-40 && mouseY < 500+30)
149       {
150         if (boton) {
151           boton=false;
152           PImage historia1=loadImage("historia1.jpg");
153           image(historia1,xcama1,ycama, 350, 500);
154
155         } else {
156           boton=true;
157         }
158       }
159       else if (mouseX > xcama2-40 && mouseX < xcama2+30 && mouseY > 500-40 && mouseY < 500+30)
160       {
161          if (boton) {
162           boton=false;
163           PImage historia2=loadImage("H2.jpg");
164           image(historia2,xcama2,ycama, 400, 500);
165
166         } else {
167           boton=true;
168         }
169       }
170 }
171
172 void keyPressed() //Cuando se pulsa una tecla
173 {
174   //Pulsar la tecla E para salir del programa
175   if(key=='e' || key=='E')
```

```
  {
    output.flush(); // Escribe los datos restantes en el archivo
    output.close(); // Final del archivo
    exit();//Salimos del programa
  }
}

void serialEvent(Serial p){
  String dato = p.readStringUntil('\n'); // leemos el buffer del puerto serie

  if(dato != null){
    dato = trim(dato);
    valores = int(split(dato, ','));
  }
}
```

En las siguientes imágenes se muestra el código utilizado para el entorno de visualización en Matlab.

```
classdef clase1 < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                        matlab.ui.Figure
        Ejes                            matlab.ui.control.UIAxes
        b1                              matlab.ui.control.Button
        b2                              matlab.ui.control.Button
        RejillaCheckBox                 matlab.ui.control.CheckBox
        VariacindelaTemperaturaLabel    matlab.ui.control.Label
        Image                           matlab.ui.control.Image
        NiveldeAlertaLampLabel          matlab.ui.control.Label
        NiveldeAlertaLamp               matlab.ui.control.Lamp
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: b1
        function b1ButtonPushed(app, event)
            cla(app.Ejes,"reset");
            fileID = fopen('datos.txt','r');
            formatSpec = '%d %f';
            sizeA = [2 Inf];
            A = fscanf(fileID,formatSpec,sizeA);
            A = A';
            [m,s]=size (A);
            A = fliplr(A);

            C1 = A(A(:,1)==1, :);
            [m1,n1] = size(C1);
            B = C1(:,2);
            n = size (B);

            t = 1:n;
            plot(app.Ejes,t,B);
            app.Ejes.YLim = [20 25];
            app.Ejes.XLabel.String = ('Tiempo (s)');
            app.Ejes.YLabel.String = ('Temperatura (ºC)');
            k = find (B>37);
            k1=size(k);
            if k1 ~= 0
                app.NiveldeAlertaLamp.Color = 'Red';
            else
```

```
                app.NiveldeAlertaLamp.Color = 'Green';
            end
        end

        % Button pushed function: b2
        function b2ButtonPushed(app, event)
            cla(app.Ejes,"reset");
            fileID = fopen('datos.txt','r');
            formatSpec = '%d %f';
            sizeA = [2 Inf];
            A = fscanf(fileID,formatSpec,sizeA);
            A = A';
            [m,s]=size (A);
            A = fliplr(A);

            C2 = A(A(:,1)==2, :);
            [m2,n2] = size(C2);
            B = C2(:,2);
            n = size (B);
            t = 1:n;
            plot(app.Ejes,t,B);
            app.Ejes.XLabel.String = ('Tiempo (s)');
            app.Ejes.YLabel.String = ('Temperatura (ºC)');
            k = find (B>37);
            k1=size(k);
            if k1 ~= 0
                app.NiveldeAlertaLamp.Color = 'Red';
            else
                app.NiveldeAlertaLamp.Color = 'Green';
            end
        end

        % Value changed function: RejillaCheckBox
        function RejillaCheckBoxValueChanged(app, event)

            value = app.RejillaCheckBox.Value;
            if value == 1
                app.Ejes.XGrid = 'on';
                app.Ejes.YGrid = 'on';
            else
                app.Ejes.XGrid = 'off';
                app.Ejes.YGrid = 'off';
            end
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 687 437];
            app.UIFigure.Name = 'UI Figure';

            % Create Ejes
            app.Ejes = uiaxes(app.UIFigure);
            title(app.Ejes, 'Variación de las temperaturas')
            xlabel(app.Ejes, 'Tiempo')
            ylabel(app.Ejes, 'Temperatura (ºC)')
            app.Ejes.XLim = [0 1];
            app.Ejes.XTick = [0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
            app.Ejes.Position = [29 32 491 320];

            % Create b1
            app.b1 = uibutton(app.UIFigure, 'push');
            app.b1.ButtonPushedFcn = createCallbackFcn(app, @b1ButtonPushed, true);
            app.b1.Position = [552 241 100 22];
            app.b1.Text = 'Paciente 1 ';

            % Create b2
            app.b2 = uibutton(app.UIFigure, 'push');
            app.b2.ButtonPushedFcn = createCallbackFcn(app, @b2ButtonPushed, true);
            app.b2.Position = [552 181 100 22];
            app.b2.Text = 'Paciente 2 ';

            % Create RejillaCheckBox
            app.RejillaCheckBox = uicheckbox(app.UIFigure);
            app.RejillaCheckBox.ValueChangedFcn = createCallbackFcn(app, @RejillaCheckBoxValueChanged, true);
            app.RejillaCheckBox.Text = 'Rejilla';
            app.RejillaCheckBox.Position = [575 86 54 22];

            % Create VariacindelaTemperaturaLabel
            app.VariacindelaTemperaturaLabel = uilabel(app.UIFigure);
            app.VariacindelaTemperaturaLabel.FontSize = 20;
            app.VariacindelaTemperaturaLabel.FontWeight = 'bold';
```

```
            app.VariacindelaTemperaturaLabel.Position = [208 384 274 27];
            app.VariacindelaTemperaturaLabel.Text = 'Variación de la Temperatura';

            % Create Image
            app.Image = uiimage(app.UIFigure);
            app.Image.Position = [496 331 178 95];
            app.Image.ImageSource = 'logo.png';

            % Create NiveldeAlertaLampLabel
            app.NiveldeAlertaLampLabel = uilabel(app.UIFigure);
            app.NiveldeAlertaLampLabel.HorizontalAlignment = 'right';
            app.NiveldeAlertaLampLabel.Position = [534 126 83 22];
            app.NiveldeAlertaLampLabel.Text = 'Nivel de Alerta';

            % Create NiveldeAlertaLamp
            app.NiveldeAlertaLamp = uilamp(app.UIFigure);
            app.NiveldeAlertaLamp.Position = [632 126 20 20];

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = clase1

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```