UNIVERSIDAD FRANCISCO DE VITORIA

ESCUELA POLITÉCNICA SUPERIOR

GRADO EN INGENIERÍA EN SISTEMAS INDUSTRALES

PROYECTO FINAL DE GRADO

Diseño Hardware-Software de un dispositivo para monitorización con transmisión de datos por LORA

Luis Rubio Fuentes

Convocatoria de junio de 2023

CALIFICACIÓN DEL PROYECTO FINAL DE GRADO

CUALITATIVA:	
NUMÉRICA:	

Conforme Presidente:	Conforme Secretario:
Fdo.:	Fdo.:

Conforme Vocal:	Conforme Vocal:	Conforme Vocal:
Fdo.:	Fdo.:	Fdo.:

Lugar y fecha: Pozuelo de Alarcón, a ____ de _____ de 202__

“Camina lento; no te apresures; que a donde tienes que llegar es a ti mismo.”

José Ortega y Gasset

Dedico este trabajo de fin de grado a:

Mis padres Ana y Luis quienes con su paciencia y esfuerzo permitieron que lograra una de mis grandes metas (la cual pensé que nunca llegaría tras mi cambio de carrera). Gracias por ser un ejemplo no solo durante estos últimos 4 años de carrera universitaria, sino durante estos 24 años de vida que lleváis a mi lado siendo un reflejo de los mejores valores que podría tener una persona y que, espero reflejar en mi futura vida. Fuisteis, sois y seréis siempre la mayor y mejor referencia que podía tener.

Mis hermanas Ana y Marta por su apoyo y cariño incondicional a lo largo de este recorrido llamado vida universitaria, un camino que, aunque parezca que nunca llega, siempre me hicieron creer que iba a llegar antes de lo esperado. Gracias por ser una parte muy grande de ese fuego que llevamos dentro y nos guía desde niños. A toda mi familia y amigos, cuyos consejos y dedicatorias calaron en mí lo suficiente como para sentirme siempre apoyado en la consecución de esta meta.

Mi amigo Enrique, que siempre confío en mí incluso en momentos en los que ni yo mismo me veía capaz de volver a ser ingeniero, que siempre me ha acompañado, apoyado y motivado a lo largo de mi carrera no solo universitaria, sino de la vida. Gracias Kike.

Mi pareja Bel, que me acompaña desde exactamente la mitad de esta travesía académica y que siempre se ha encargado de hacerme sentir la persona más capaz del mundo sin importar todo lo demás. Gracias Bel por recordarme cosas que pensé que no existían, y por hacerme ver que en la vida no todo es frío y matemático, sino humano y aleatorio. Esta es sin duda mi mayor lección tras estos años.

Agradecimientos

Si bien es cierto que no puedo dejar pasar este apartado para agradecer a toda la gente que ha hecho posible este trabajo gracias a su esfuerzo y dedicación a la docencia en esta institución a lo largo de estos cuatro años. Gracias a todos los profesores que se han esforzado por enseñarme esos conocimientos que eran completamente un mundo desconocido para mí. Gracias en especial a aquellos profesores que siempre han puesto de su mano para resolverme las dudas, independientemente del grado de facilidad de estas e incluso independientemente de si era su horario libre o no.

Gracias a todos.

Resumen

En este proyecto se propone el desarrollo de un dispositivo electrónico para la transmisión de datos en tiempo real a través de LoRa (Long Range). En este trabajo se dispone un sistema capaz de monitorizar y transmitir datos de humedad, temperatura, nivel de luminosidad y batería para el control periódico en un determinado ecosistema.

En primer lugar, el sistema permite monitorizar remotamente los parámetros ambientales de un ecosistema para así poder actuar sobre el mismo con diversos fines. En segundo lugar, el dispositivo se encuentra formado por un controlador RAK 3172 basado en un chip STM32WLE5CC el cual es capaz de transmitir y recibir datos a través de la red LoRa.

Para la consecución de este proyecto vamos a partir de un estudio exhaustivo de las tecnologías disponibles, seguido de un diseño y fabricación de la placa base, y finalizando con una programación e implementación de un prototipo funcional.

Palabras claves

IoT, RAK3172, Monitorización, Sensores, Kicad 6.0, TTN

Abstract

This project proposes the development of an electronic device for real-time data transmission through LoRa (Long Range). This work provides a system capable of monitoring and transmitting data on humidity, temperature, brightness level and battery for periodic control in a given ecosystem.

Firstly, the system allows to remotely monitor the environmental parameters of an ecosystem in order to act on it for different purposes. Secondly, the device consists of a RAK 3172 controller based on a STM32WLE5CC chip which is able to transmit and receive data through the LoRa network.

For the achievement of this project we will start with an exhaustive study of the available technologies, followed by the design and fabrication of the motherboard, and ending with the programming and implementation of a working prototype.

Keywords

IoT, RAK3172, Monitoring, Sensors, Kicad 6.0, TTN

Índice de Contenidos

1. Introducción 1

2. Investigación previa 3

2.1. Estado del arte en cuanto a dispositivos IoT 3

2.1.1. Prismab Link 3

2.1.2. RAK3172-BME680 Environmental monitoring 5

2.2. Estado del arte en cuanto a comunicaciones LoRa 6

2.2.1. Secmotic y LORA 6

2.2.2. TTN Mapper 7

2.3. Fundamentos 8

2.3.1. Circuitos eléctricos 8

2.3.2. Ley de Ohm 8

2.3.3. Potencia y energía de las resistencias 9

2.3.4. Análisis nodal 9

2.3.5. Ley de Kirchhoff 10

2.3.6. Técnicas de programación 11

3. Objetivos 13

3.1. Objetivo general 13

3.2. Objetivos específicos 13

4. Plan de Desarrollo del Proyecto 15

4.1. Metodología 15

4.2. Tecnologías 16

4.3. Plan de trabajo 25

4.4. Recursos. 28

4.5. Condicionantes y Limitaciones 36

5. Desarrollo de la Solución Técnica 37

5.1. Elección del microcontrolador 37

5.2. Cálculos y diseño del prototipo 41

5.2.1. Sensor de nivel de luz 48

5.2.2. Sensor de temperatura y humedad 50
5.2.3. Tarjeta de memoria SD 53
5.2.4. Modelado de la PCB en Kicad 6.0 54
5.2.5. Diseño de la antena y su ruteado 55
5.2.6. Autoruteado de pistas aéreas 57
5.2.7. Creación de Gerbers 58
5.2.8. Resultado del primer prototipo 60
5.2.9. Resultado del diseño del prototipo final 60
5.3. Pedido y recepción de componentes 62
5.4. montaje y soldado de la placa 65
5.5. Programación del dispositivo 66
5.5.1. Programación de prueba de LED 67
5.5.2. Programación de fotorresistor LDR 69
5.5.3. Programación de Sensor de temperatura y humedad 72
5.5.4. Programación para recopilación completa de datos 76
5.5.5. Programación para recepción de datos mediante Gateway 77

6. Resultados 91

7. Implicaciones Éticas e Impacto Social 93

8. Conclusiones 95

9. Otros Méritos del Proyecto 97

10. Bibliografía 99

Anexo A: Esquemático del prototipo PCB 103

Anexo B: Plano de la caja impresa en 3D 105

Anexo C: Código del programa final 106

Anexo D: Datasheet Prismab 120

Anexo E: Datasheet RAK3172 121

Anexo F: Datsaheet LDR Farnell 125

Anexo G: Datasheet STM32WLE5CCU6 126

Anexo H: Datasheet SI7021 128

Anexo I: Datasheet STLINK V2 131

Anexo J: Datasheet TTL AZDelivery FT232RL 132

Anexo K: Gerbers del prototipo 133

Anexo L: Facturas de componentes 137

Anexo M: Diagrama de Gantt 147

Índice de Tablas

Tabla 1: Especificaciones PrismaB Link 4
Tabla 2: Constantes del lenguaje C 12
Tabla 3: Tecnologías empleadas en el proyecto 16
Tabla 4: Requisitos de hardware software STM32CubeIDE 21
Tabla 5: Puertos de comunicación soportados 23
Tabla 6: Precios del dispositivo final y proyecto 35
Tabla 7: Links de compra de materiales 36
Tabla 8: ventajas vs desventajas de Microchip 38
Tabla 9: Ventajas vs Desventajas de Atmel 38
Tabla 10: Frecuencias en MHz existentes a nivel mundial 41
Tabla 11: Configuración de pines del RAK3172 43
Tabla 12: Pines de programación relevantes 43
Tabla 13: Características de voltaje del RAK3172 45
Tabla 14: Características de corriente del RAK3172 46
Tabla 15: Máximo y mínimo de lúmenes LDR 50
Tabla 16: Valores de lúmenes en 16 bits 50
Tabla 17: Configuración del pedido a JLCPCB 64
Tabla 18: Estados de conexión del Gateway 86

Índice de ecuaciones

Ecuación 1: Ley de Ohm *8*

Ecuación 2: Potencia eléctrica *9*

Ecuación 3: Potencia eléctrica consumida en un intervalo de tiempo *9*

Ecuación 4: Cálculo de caída de tensión *46*

Ecuación 5: Cálculo de resistencia según caída de tensión *46*

Ecuación 6: Fórmula de divisor de tensión *49*

Ecuación 7: Fórmula de Axel-Benz *49*

Ecuación 8: Cálculo de resistencia de referencia para LDR *50*

Ecuación 9: Porcentaje de humedad relativa *75*

Ecuación 10: Temperatura ambiental *75*

Índice de ilustraciones

Ilustración 1: Dispositivos conectados a nivel mundial a IoT del 2019 al 2030 1
Ilustración 2: Dispositivo IoT Prismab Link 3
Ilustración 3: Funcionamiento red SIGFOX 4
Ilustración 4: Diagrama realizado por jh83 en Github 5
Ilustración 5: Programación para mostrar datos de un sensor BME680 I2C 6
Ilustración 6: Ejemplo de sistema de comunicación LoRa 7
Ilustración 7: TTN Mapper del mundo 7
Ilustración 8: Diagrama de bloque de circuito eléctrico 8
Ilustración 9:Ejemplo de análisis nodal 10
Ilustración 10: Ley de corrientes de Kirchhoff (LKC) 10
Ilustración 11: Ley de voltajes de Kirchhoff (LKT) 11
Ilustración 12: ANSI C Keywords 11
Ilustración 13: Tipos de datos de C 12
Ilustración 14: Fases de una metodología en cascada 15
Ilustración 15: Menú principal del Software Kicad 6.0 17
Ilustración 16: Visualización del programa GerbViewer 18
Ilustración 17: Panel de control inicial de STM32CubeMx 19
Ilustración 18-Sinergías de programas de STMicroelectronics 19
Ilustración 19: Pantalla de inicio de software STM32CubeIDE 20
Ilustración 20: Explorador de proyectos en STM32CUbeIDE 20
Ilustración 21: Funciones definidas de serie en main.c 21
Ilustración 22: Localización del Debugger y diferentes herramientas del programa 21
Iluestración 23: Comentarios delimitadores de código en STM32CubeIDE 22
Ilustración 24: Pantalla de inicio de MobaXterm 22
Ilustración 25: Puertos de transmisión de datos disponibles en MobaXterm 23
Ilustración 26: Diagrama de Gantt 25
Ilustración 27: Fragmento de diagrama de Gantt para el análisis de requisitos 26
Ilustración 28: Fragmento de diagrama de Gantt para el prototipado y testeo 26
Ilustración 29: Fragmento de diagrama de Gantt para la adquisición y contacto 26
Ilustración 30: Fragmento de diagrama de Gantt para el soldado de material 27
Ilustración 31: Fragmento de diagrama de Gantt para el testeo de comunicaciones 27
Ilustración 32: Fragmento del diagrama de Gantt para el testeo de la LDR 27

Ilustración 33: Fragmento del diagrama de Gantt para el testeo de comunicación LoRa 27
Ilustración 34: Diagrama de Gantt completo 28
Ilustración 35: Diagrama de Gantt 37
Ilustración 36: Familias de PIC y AVR 38
Ilustración 37: PinMap Arduino UNO 39
Ilustración 38: ATMega 328 datasheet Features 39
Ilustración 39: Arduino HopeRF RFM95 features 40
Ilustración 40: Consumos de corriente del RAK3172 41
Ilustración 41:Configuración de pines del RAK3172 42
Ilustración 42: Diseño y definición de pines del rak3172 en Kicad 6.0 44
Ilustración 43: Bloque de diagramas del RAK3172 y STM32WLE5CCU6 44
Ilustración 44: RESET del prototipo diseñado 45
Ilustración 45: Resistencia empleada en el cálculo del RESET 46
Ilustración 46: Esquemático con el puerto de alimentación incorporado 47
Ilustración 47: Fuente de alimentación del dispositivo para el esquemático 47
Ilustración 48: Parte del esquemático referente al botón de testeo del sistema 48
Ilustración 49: Gráfica explicativa de sensor LDR 49
Ilustración 50: Ejemplificación en Kicad 6.0 para el cálculo del LDR 49
Ilustración 51: Puerto PB2 del esquemático 50
Ilustración 52: Diagrama de funcionamiento de microcontroladores 51
Ilustración 53: Esquema de conexión entre dispositivos 51
Ilustración 54: Esquemático del SI7021 52
Ilustración 55: Funcionamiento del SPI 53
Ilustración 56: Definición de pines de una tarjeta SD 53
Ilustración 57: Tarjeta SD de Sparkfun electronics 54
Ilustración 58: Vista en esquemático de la tarjeta SD 54
Ilustración 59: Visor de disposición de huellas en Kicad 6.0 55
Ilustración 60: Antenas dipolo y monopolo de 1/4 de onda 55
Ilustración 61: Antena FPC 56
Ilustración 62: Modelado de PCB correctamente dispuesto 56
Ilustración 63: Parte frontal del primer prototipo 57
Ilustración 64: Parte trasera del primer prototipo 57
Ilustración 65: Autoruteado con programa "Autoroute" 57
Ilustración 66: Pistas aéreas visualizadas e importadas a Kicad 6.0 58
Ilustración 67: Capa de contorno PCB 58

Ilustración 68: Top Layer de Cobre PCB 59
Ilustración 69: Top layer máscara anti-soldadura PCB 59
Ilustración 70: Lugares de situación de la pasta y serigrafías PCB 59
Ilustración 71: Taladros de la PCB 59
Ilustración 72: Capa de Cobre de la PCB 60
Ilustración 73: Esquemático del prototipo enviado a fabricación 61
Ilustración 74: Esquemáticos de puertos J2, J4 y J5 61
Ilustración 75: Pistas aéreas del dispositivo final 62
Ilustración 76: Visualización 3D delantera del dispositivo final 62
Ilustración 77: Visualización 3D trasera del dispositivo final 62
Ilustración 78: Gerber delantero del prototipo final 63
Ilustración 79: Gerber trasero del prototipo final 63
Ilustración 80: Especificaciones Stencil para soldadura SMD 64
Ilustración 81: Visualización del Stencil para soldadura SMD 65
Ilustración 82: Configuración de STM32WLE5CCU6 67
Ilustración 83: Conexión de cables para la programación del LED 68
Ilustración 84: Configuración necesaria para el parpadeo del LED 68
Ilustración 85: Ejemplo de visualización en main.h del PIN A8_LED 69
Ilustración 86: Código main.c que permite el parpadeo del LED cada 1000 ms 69
Ilustración 87: Configuración para programación del fotorresistor 69
Ilustración 88: Configuración Pin PB3 para LDR 70
Ilustración 89: Configuración puerto Usart 1 70
Ilustración 90: Código para leer el fotorresistor LDR 72
Ilustración 91: Ejemplo de programa lector de luminosidad 72
Ilustración 92: Configuración de sensor SI7021 73
Ilustración 93: Configuración de pines del sensor SI7021 73
Ilustración 94: Código de declaración del fichero si7021.h 74
Ilustración 95: Código para leer la humedad relativa y la temperatura 75
Ilustración 96: Secuencia de recepción de datos del sensor SI7021 75
Ilustración 97: Código para visualizar los datos en la terminal 76
Ilustración 98: Visualización final de datos del sensor Si7021 en la terminal 76
Ilustración 99:Código necesario para la escritura de todos los parámetros medidos 76
Ilustración 100: Visualización del programa completo 77
Ilustración 101: Esquema de funcionamiento de LoRa del dispositivo 77
Ilustración 102: Configuración de pines del programa 78

Ilustración 103: Configuración de HSC y LSC *78*
Ilustración 104: Configuración de reloj real *79*
Ilustración 105: Constantes y valores del esqueleto de código *79*
Ilustración 106: Configuración para la lectura del nivel de batería del dispositivo *79*
Ilustración 107: Configuración de claves OTTA *80*
Ilustración 108: Activación de módulo de frecuencia *80*
Ilustración 109: Configuración de LoRaWAN *80*
Ilustración 110: Configuración de frecuencia según región *81*
Ilustración 111: Configuraciones de la plataforma *81*
Ilustración 112: Parámetros de conexión LoRa *82*
Ilustración 113: Configuración del reloj del micro *82*
Ilustración 114: Configuración de pines de salida y CTRL *83*
Ilustración 115: Configuración de pines del si7021 ..... *83*
Ilustración 116: Configuración de USART *83*
Ilustración 117: Código implementado en radio.board_if.h *83*
Ilustración 118: Código implementado en radio_board_if.c *84*
Ilustración 119: Configuración de timers *84*
Ilustración 120: Programación para envío de datos de sensores *85*
Ilustración 121: Configuración del Gateway en TTN *85*
Ilustración 122: Configuración del dispositivo en TTN *86*
Ilustración 123: Estado de conexión del Gateway *87*
Ilustración 124: Flujo del programa para la recepción de datos *87*
Ilustración 125: Envío de paquete de datos del dispositivo PCB *88*
Ilustración 126: Visualización a través de node-RED *88*
Ilustración 127: Visualización a través de myopentech *88*
Ilustración128: Carcasa en Autodesk Inventor 2024 y Ultimaker Cura *89*
Ilustración 129: Diseños para la visualización de datos en dispositivos web *92*

Índice de fotografías

Fotografía 1: Interior de un dispositivo Prismab Link 4
Fotografía 2: BOSCH BME680 5
Fotografía 3: AZDelivery 3X Adaptador Compatible con FT232RL, USB a TT 3.3V-5V 23
Fotografía 4: ejemplo de soldadura SMD en PCB 24
Fotografía 5: Resistencia de 10 K 29
Fotografía 6: Condensador 0,1 uF 29
Fotografía 7: Portapilas doble AA 29
Fotografía 8: Resistencia de 22 K 29
Fotografía 9: Resistencia 100 Ohms 30
Fotografía 10: Condensador de 10 uF 30
Fotografía 11: Micropulsador SW074 de Reset 30
Fotografía 12: Sensor SI7021 30
Fotografía 13: Switch de encendido/apagado 31
Fotografía 14: Módulo RAK3172 31
Fotografía 15: Conversor IPEX antena 31
Fotografía 16: Antena 863-870 MHz 31
Fotografía 17: Antena Molex 868-915 MHz 32
Fotografía 18: PCB + PLaca de SMD 32
Fotografía 19: LED 32
Fotografía 20: LDR 32
Fotografía 21: Pinzas Antiestáticas 32
Fotografía 22: Flux 33
Fotografía 23: Bobina de Estaño 33
Fotografía 24: AZDelivery USB a TTL 3,3-5 V 33
Fotografía 25: Gateway LoRa 33
Fotografía 26: ST-Link V2 34
Fotografía 27: Jeringuilla de Estaño 34
Fotografía 28: Plancha de soldado 34
Fotografía 29: Sensor Si7021 52
Fotografía 30: PCB + Stencil recibidos 65
Fotografía 31: Parte frontal del dispositivo final 66
Fotografía 32: Parte trasera del dispositivo final 66
Fotografía 33: Estado de conexión del Gateway 86

Fotografía 34: Carcasa impresa en 3D con impresora Ender 3 ..... *89*

Fotografía 35: Gateway con Raspberry Pi y concentrador ic880a *91*

Fotografía 36: Dispositivo final creado..... *91*

Fotografía 37: Caja impresa en 3D para implementar el dispositivo con protoboard *92*

Lista de Acrónimos

Acrónimo	Significado
IoT	Internet Of Things
TTN	The Things Network
LoRa	Long Range
UNB	Ultra Narrow Band
OTAA	Over The Air Authentication
ODS	Objetivos de Desarrollo Sostenible
IP	Internet Protocol
SMD	Surface Mount Technology
USART	Universal Synchronous and Asynchronous serial Receiver and Transmitter
PCB	Printed Circuit Board
RAM	Random Access Memory
IDE	Integrated Developmnet Environment
LPWAN	Low Power Wide Area Network
ADC	Analog to Digital Converter
RST	Reset
FPC	Flexible Printed Circuit
CLK	Clock
DMA	Direct Memory Access
EUI	Extended Unique Identifier

1. INTRODUCCIÓN

Con el paso del tiempo hemos vivido numerosos cambios tecnológicos, muchos de ellos dieron lugar a auténticas revoluciones industriales. Así pues, desde el año 2000 hasta la actualidad vivimos en una revolución industrial liderada por los sistemas físicos y cibernéticos.

Hoy en día estamos rodeados de dispositivos electrónicos los cuales se encuentran constantemente conectados a la red Internet e intercambian datos continuamente tratando de mejorar nuestras vidas o incluso tratando de salvarlas en algunos casos. Estos dispositivos constituyen el ya conocido por todos como Internet of things (IoT).

La tecnología descrita como IoT ha evolucionado enormemente a lo largo de los años, y es que podemos observar que el número de dispositivos conectados ha aumentado desde los 8,9 miles de millones del año 2018 hasta los 15,1 miles de millones planeados para el año 2023. Incluso podemos observar que para el año 2030 casi se llegue a duplicar dicha cifra. (Fernández, 2022)

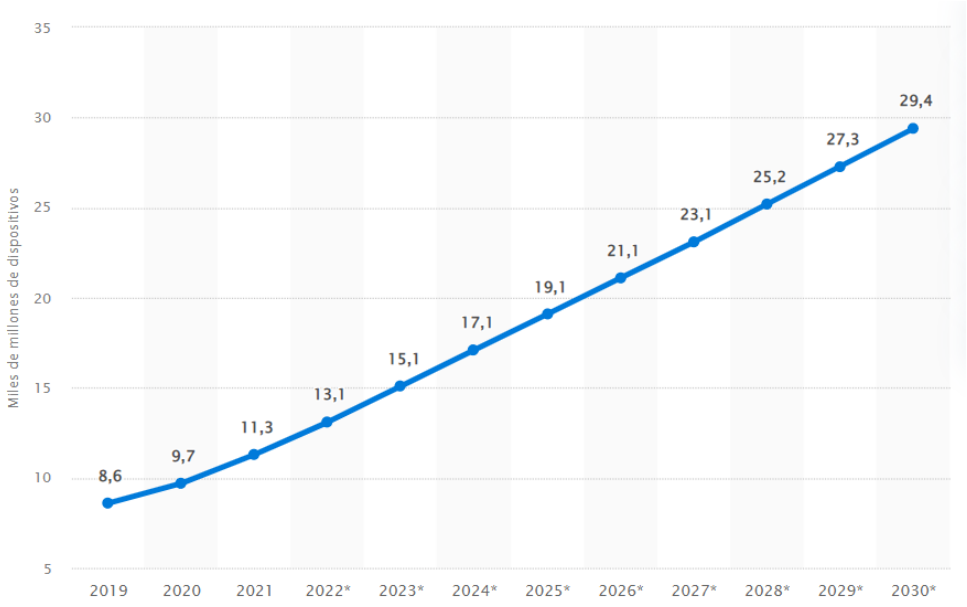


Ilustración 1: Dispositivos conectados a nivel mundial a IoT del 2019 al 2030

Gracias a la existencia del IoT podremos utilizar desde la comodidad de nuestras casas dispositivos capaces de transmitir datos en tiempo real, lógicamente, siempre que poseamos los conocimientos necesarios para utilizarlos de forma adecuada.

Así pues, plantearemos este trabajo como el desarrollo de un prototipo didáctico de tal forma que podamos construir un dispositivo Hardware alimentado a través de dos pilas AA de 1.5 V. Este estará formado por un controlador basado en un chip STM32WLE5CC que será programado de forma eficiente para poder monitorizar parámetros ambientales y enviarlos a una red LoRa pública como es TheThingsNetwork. O bien a un servidor LoRa local para poder analizarlos y a su vez recibir comandos que puedan ser ejecutados por el microcontrolador.

El principal factor motivante de este proyecto es que presenta un gran desafío para lo aprendido a lo largo de estos años. Cierto es, que puede verse como un proyecto más orientado a la ingeniería en electrónica y de telecomunicaciones, y, de hecho, con tan solo buscar en internet las palabras “diseño” y “sistema LoRa” vamos a vislumbrar numerosos proyectos de final de carrera de esta rama. No obstante, considero que estamos en una edad fundamental para formarnos y descubrir aquello a lo que vamos a orientar nuestro futuro, y es esa una de las razones que han motivado este proyecto.

El proyecto me brinda la posibilidad de familiarizarme con un sector relacionado con la electrónica y automática el cual no descarto en mi futuro profesional. Este sector siempre ha suscitado en mí una enorme curiosidad, hecho por el cual obtuve la certificación de Arduino Fundamentals hace un año.

Por otro lado, reconozco que este proyecto me ha atraído personalmente ya que provengo de una familia del sector agrícola-ganadero, un sector en el cual he podido trabajar desde los 16 años y he podido observar el avance de la ciencia y la tecnología. Hace ya casi 5 años que vi por primera vez un sistema de monitorización de estado de ecosistema agrícola basado en redes privadas Sigfox. También, he visto por desgracia cómo muchas empresas se aprovechan de este sector inflando desorbitadamente los precios para estos profesionales agrícolas-ganaderos los cuales no tienen un conocimiento de la materia y, por lo tanto, le expone a un estado indefenso ante cualquier propuesta de campo + IoT.

Así pues, para finalizar me gustaría recalcar que mi motivación no es solo la oportunidad de formarme en un campo algo desconocido para mí como es la electrónica + IoT a nivel de desarrollo, sino también, demostrar que se pueden desarrollar sistemas de monitorización a precios asequibles para sectores como el agrícola -ganadero.

2. INVESTIGACIÓN PREVIA

En las siguientes secciones se realizará una investigación sobre las distintas soluciones existentes tanto en el mercado como en cuanto a prototipos personales de dispositivos IoT. Dada la índole del trabajo, el estudio e investigación irán relacionados con dispositivos encargados de medir diferentes parámetros de determinados ecosistemas.

El propósito de esta investigación previa es adquirir un conocimiento detallado de los diversos diseños, tecnologías y metodologías utilizados en la creación de proyectos de la misma índole, con el fin de obtener una perspectiva más amplia del sector. De esta forma, se busca obtener conclusiones generales que contribuyan a la concepción y diseño adecuado del dispositivo en cuestión.

2.1. ESTADO DEL ARTE EN CUANTO A DISPOSITIVOS IOT

2.1.1. Prismab Link

Prismab Link es un dispositivo IoT de la compañía prismaB agrotech, el dispositivo ofrece una solución sencilla para el estudio exhaustivo de condiciones climatológicas en empresas agrícolas. La mayor ventaja de este dispositivo es que consta de diferentes sensores los cuales no requieren ningún tipo de conocimiento tecnológico por parte del comprador, simplemente se conectan y ya retransmiten datos a la nube.

Ilustración 2: Dispositivo IoT Prismab Link

Este dispositivo emplea un sistema de transmisión de datos llamado SIGFOX. Esta es una red de conectividad celular enfocada al empleo de IoT, la especial característica de esta es el elevado consumo de energía por parte de los dispositivos conectados.

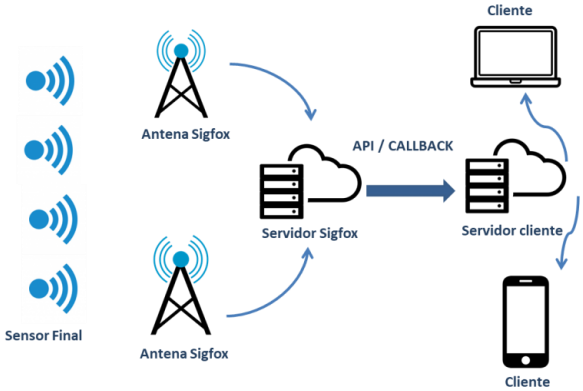


Ilustración 3: Funcionamiento red SIGFOX

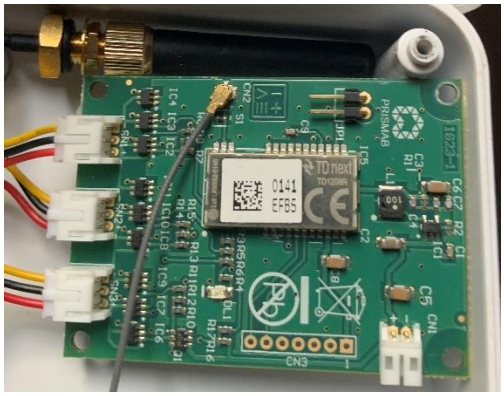
El empleo de esta red es debido a su alto rango de alcance en campo abierto (20 Km) e incluso su elevada precisión en urbanización (1,5 m). Su banda es de tipo UNB (Ultra Narrow Band) consiguiendo así unas transferencias de datos de 10 a 10000 bits por segundo con las consecuentes bajas velocidades. (Dset Energy, s.f.)

El dispositivo presenta unas características muy positivas las cuales servirán de referencia como parámetros deseados a nivel de futuro desarrollo de la solución con el RAK3172. Estas características son: (Dset Energy, s.f.)

Frecuencia de envío de datos	Autonomía media	Consumo en reposo	Consumo en operación / voltaje	Alimentación	Tipo de protección	Rango de temperatura de trabajo	Peso	Dimensiones
1 Hora	4 años	1,8µA	51mA / 3,3V	4 x baterías AA (1,5v)	IP67	-20ºC a 54ºC	290g	100 x 100 x 40 mm

Tabla 1: Especificaciones PrismaB Link

Dado que contamos con el producto en persona podremos observar la presencia de un microcontrolador TD next TD1208R con procesador ARM Cortex-M3. Este dispositivo es muy empleado para soluciones IoT con redes SIGFOX.



Fotografía 1: Interior de un dispositivo Prismab Link

2.1.2. RAK3172-BME680 Environmental monitoring

Este dispositivo trata de ofrecer al usuario una solución muy simplificada para poder monitorizar parámetros ambientales a través del sensor BOSCH BME680. El BME680 está capacitado para ofrecer al usuario los siguientes parámetros:

a) Temperatura

b) Humedad

c) Presión barométrica

d) Gas

Fotografía 2: BOSCH BME680

La solución propuesta presenta una combinación del RAK3172 y del sensor ya descrito posteriormente. Las medidas son transferidas vía LoRaWAN a través del puerto número 10 del RAK3172 con un intervalo de serie de 10 minutos tal y como podemos ver en la ilustración:

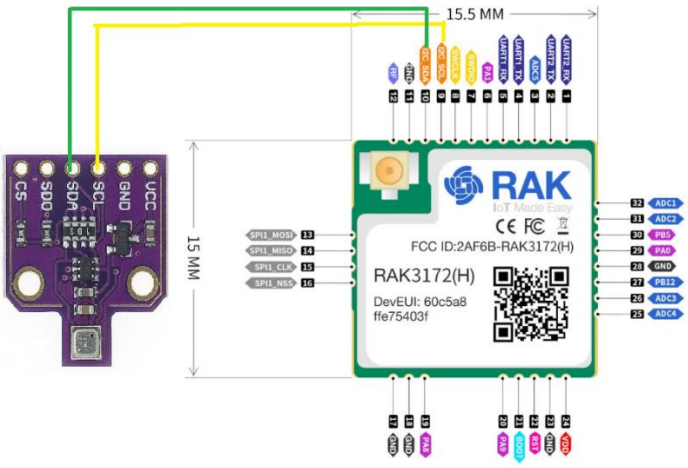


Ilustración 4: Diagrama realizado por jh83 en Github

Este sistema ofrece al usuario la oportunidad de poder familiarizarse con el envío de datos I2C a través de sistemas LoRaWAN gracias a la facilidad y simpleza del sistema propuesto por el fabricante BOSCH GROUP.

La programación ofrecida por el usuario de esta solución no es muy compleja, sin embargo, como hemos mencionado anteriormente nos presenta una forma muy útil de programación de sensores I2C con RAK3172. Como se puede observar en la siguiente ilustración, el código de este sensor es muy sencillo y su información será mostrada a través de una pequeña pantalla LCD a través de un Arduino con ADAFRUIT.

```
if (read_success) {
  message.addTemperature(bme.temperature);
  message.addHumidity(bme.humidity);
  message.addRawFloat(bme.pressure);
  message.addRawFloat(bme.gas_resistance);

  Serial.print("Temperature: ");
  Serial.println(bme.temperature);
  Serial.print("Humidity: ");
  Serial.println(bme.humidity);
  Serial.print("Pressure: ");
  Serial.println(bme.pressure / 100.0);
  Serial.print("Gas Resistance: ");
  Serial.println(bme.gas_resistance / 1000.0);
}
```

Ilustración 5: Programación para mostrar datos de un sensor BME680 I2C

Cabe recalcar que previamente a la respuesta de visualización de datos se ha realizado un código que permita corroborar que el sistema está enviando de forma correcta paquetes de datos al Gateway Lora través de OTAA (Over The Air Authentication).

2.2. ESTADO DEL ARTE EN CUANTO A COMUNICACIONES LORA

2.2.1. Secmotic y LORA

La importancia de la red Lora es muy elevada, como ya se ha mencionado en este trabajo, la implementación de esta tecnología en hogares resulta muy fructífera. Sin embargo, qué puede originar a larga escala si la expandimos a ciudades.

La respuesta a esto son las conocidas como Smart cities, son ciudades las cuales están orientadas al empleo de comunicaciones de dispositivos a través de LoRa para experimentar una mejora en términos de eficiencia, agilidad y ahorro. Secmotic es una empresa de Lanzarote la cual se encarga de la incorporación de la tecnología LoRa para lograr el cumplimiento de los Objetivos de Desarrollo Sotenible y la Agenda Urbana 2030.

Tras lo dispuesto, esta empresa ha comenzado con la instalación de Gateways en la isla para el posterior desarrollo de comunicaciones por parte de los ciudadanos, permitiendo así el análisis de datos y toma de decisiones por parte de los efectivos de la ciudad como centrales de energía, puestos de suministro eléctrico etc.

Secmotic plantea en un proyecto la instalación de Gateways marítimos situados en el puerto que permitan un puerto inteligente.

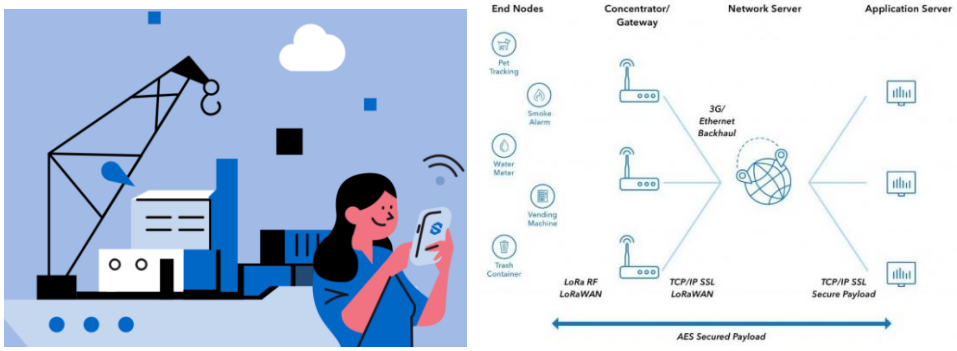


Ilustración 6: Ejemplo de sistema de comunicación LoRa

2.2.2. TTN Mapper

La conexión LoRa tiene un elevado potencial, como vemos en el trabajo esta depende de unos dispositivos los cuales necesitan conectarse a Gateways que permitan incluir los datos en la web para poder visualizarlos a través de la red. La transmisión de datos se suele realizar al igual que en el presente proyecto, un dispositivo transmite datos recolectados dentro de un área de alcance de un concentrador con una raspberry PI que están conectados mediante una IP fija al router del hogar.

Una vez realizada esa conexión se establece mediante diversos métodos una interconexión de los datos recibidos y la web para ser visualizados. El precio de un Gateway es algo elevado en la actualidad, por eso hay personas que ponen a disposición de los usuarios sus propios Gateway para así favorecer a la creación de una comunidad de dispositivos IoT más polivalente y de mayor tamaño, fomentando así también el potencial de esta herramienta.

Tras lo dispuesto, en el presente trabajo es cierto que se va a disponer de un Gateway personal, sin embargo, uno de los factores a revisar a largo plazo es el de incorporar mi propio Gateway a la comunidad y que pueda ser visible a través del TTN Mapper. Este TTN Mapper permite visualizar todos los Gateway que disponemos a nuestro alrededor en la actualidad.

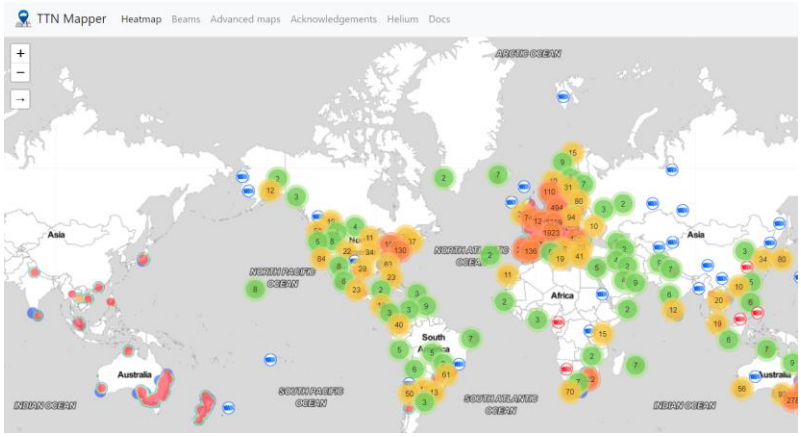


Ilustración 7: TTN Mapper del mundo

2.3. FUNDAMENTOS

Para el desarrollo de este proyecto se han empleado aspectos de diversa índole, ya sean tanto técnicos como científicos, pasando por metodológicos para así poder alcanzar la solución óptima.

Dada la temática eléctrica y electrónica de este proyecto se van a emplear numerosas leyes y principios vistos a lo largo de la carrera, no obstante, es importante recabarlos todos en este apartado para así facilitar la comprensión lectora de la memoria propuesta.

2.3.1. Circuitos eléctricos

En primer lugar, se debe conocer el funcionamiento básico de los circuitos eléctricos. "Un circuito es un conjunto de elementos combinados de tal forma que existe la posibilidad de que se origine una corriente eléctrica. Existen unos elementos denominados activos o también fuentes o generadores que suministran energía eléctrica y otros elementos denominados pasivos, que disipan o almacenan este tipo de energía. Podemos simbolizar un circuito eléctrico por el bloque dispuesto en la siguiente ilustración, que está sometido a unos estímulos de entrada denominados excitaciones, debidas a las acciones de las fuentes, originando unas respuestas en el circuito, que son las tensiones y corrientes que aparecen en la red." (Fraile Mora, 2012)

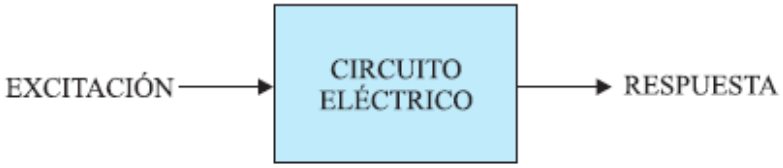


Ilustración 8: Diagrama de bloque de circuito eléctrico

2.3.2. Ley de Ohm

En términos matemáticos, la ley de Ohm se expresa como Ecuación 1, siendo I la corriente eléctrica, V el voltaje aplicado al conductor y R es la resistencia del conductor. Todas ellas medidas en Amperios, Voltios y Ohmios respectivamente. (Ruíz Vázquez, Arbelaitz Gallego, Etxeberria Uztarroz, & Ibarra Lasa, 2004)

$$I = \frac{V}{R}$$

Ecuación 1: Ley de Ohm

De aquí se puede concluir que, si se mantiene constante la resistencia del conductor y se aumenta el voltaje aplicado, la corriente eléctrica también aumentará proporcionalmente.

2.3.3. Potencia y energía de las resistencias

La potencia eléctrica de una resistencia es la cantidad de energía eléctrica que se consume por unidad de tiempo. La potencia es importante dado que nos permite saber cuánta energía se está utilizando en un circuito eléctrico. También nos ayuda a entender la capacidad de una resistencia para disipar energía y evitar sobrecalentamientos que puedan dañar el circuito o el propio componente.

En términos matemáticos, la potencia se expresa como Ecuación 2, siendo I la corriente eléctrica, R la resistencia y V el voltaje. Todos medidos en Amperios, Ohmios y Voltios respectivamente.

$$p = R \cdot I^2 = \frac{V^2}{R}$$

Ecuación 2: Potencia eléctrica

Es importante conocer con respecto a la potencia el hecho de la disipación de energía de esta con respecto al paso del tiempo. "Basándonos en la expresión de la potencia, podemos calcular la energía que absorberá una resistencia en un intervalo de tiempo determinado. Dicha energía se convierte en calor; es el denominado efecto Joule." (Ruíz Vázquez, Arbelaitz Gallego, Etxeberria Uztarroz, & Ibarra Lasa, 2004)

$$p(t) = \frac{dW(t)}{dt} \rightarrow W(t_1, t_2) = \int_{t1}^{t2} p(t) \cdot dt = \int_{t1}^{t2} R \cdot I^2(t) \cdot dt$$

$$Si\ I(t) = cte \rightarrow W(t_1, t_2) = [R \cdot I^2 \cdot t_2 - t_1]\, J\ (Julios)$$

$$= 0{,}24 \cdot [R \cdot I^2 \cdot t_2 - t_1] cal\ (Calorias)$$

Ecuación 3: Potencia eléctrica consumida en un intervalo de tiempo

2.3.4. Análisis nodal

Un nodo eléctrico es el punto de un circuito en el cual se unen tres elementos o más. Para analizar estos nodos y obtener sus propiedades eléctricas emplearemos el análisis nodal.

Según (Ruíz Vázquez, Arbelaitz Gallego, Etxeberria Uztarroz, & Ibarra Lasa, 2004) "se han de realizar los siguientes pasos para el análisis nodal:

1- Identificar los nodos del circuito.

2- Escoger un nodo como nodo de referencia. Lo más conveniente es escoger el nodo en el cual se conectan más ramas.

3- Definir las tensiones de los nodos en el esquema del circuito, para todos los nodos excepto para el de referencia. La tensión de un nodo es la diferencia de potencia entre ese nodo particular y el nodo de referencia. Si hay N nodos, habrá (N-1) tensiones de nodo desconocidas.

4- Escribir las expresiones de las corrientes de todas las ramas. En el caso de los generadores de corriente, sus corrientes de rama son conocidas; por el contrario, en el caso de las resistencias, las obtendremos en función de las tensiones de los nodos, aplicando la ley de Ohm.

5- Aplicar la ley de Kirchoff de las corrientes (LKC) en los nodos, si hay N nodos, se deben escribir N-1 ecuaciones (una por cada nodo, exceptuando el nodo de referencia).

6- Resolver el sistema de ecuaciones obtenido y, posteriormente, calcular los valores de todas las tensiones y corrientes.

“

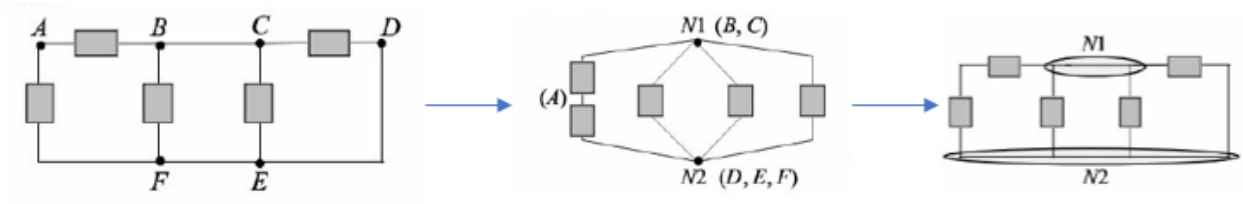


Ilustración 9:Ejemplo de análisis nodal

2.3.5. Ley de Kirchhoff

La Ley de Kirchhoff se utiliza para analizar y calcular los valores de las corrientes y los voltajes en un circuito eléctrico. Existen dos leyes fundamentales:

a) **Ley de corrientes de Kirchhoff o Ley de conservación de corriente:** establece que, en cualquier nodo de un circuito eléctrico, la suma algebraica de las corrientes que entran y salen de ese nodo es cero.

$$i_1(t) + i_5(t) + i_3(t) = i_2(t) + i_4(t)$$

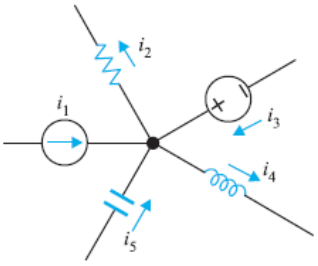


Ilustración 10: Ley de corrientes de Kirchhoff (LKC)

b) **Ley de voltajes de Kirchhoff o Ley de las mallas de Kirchhoff:** establece que la suma algebraica de los voltajes en cualquier camino cerrado de un circuito eléctrico es igual a cero.

$$U_1(t) + U_5(t) + U_2(t) = U_3(t) + U_4(t)$$

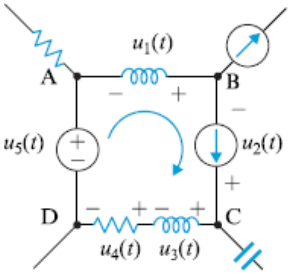


Ilustración 11: Ley de voltajes de Kirchhoff (LKT)

2.3.6. Técnicas de programación

El lenguaje de programación C fue creado en el año 1972 gracias a Brian Kernighan y Dennis Ritchie que lo implementaron en un DEC PDP-11. Obviamente con el paso de los años empleamos entornos de programación que nos permiten crear, editar y simular el código programado.

El lenguaje C es algo que conocido gracias a los conocimientos adquiridos a lo largo de la carrera con asignaturas como electrónica digital y procesadores o Introducción a la programación. No obstante, yo poseía conocimientos previos debido a que estudiaba Ingeniería informática hace unos años.

El lenguaje C presenta un grupo de palabras limitado con el standard ANSI-c como se puede observar en la siguiente ilustración:

auto	break	case	char
const	continue	default	do
double	else	enum	extern
float	for	goto	if
int	long	register	return
short	signed	sizeof	static
struct	switch	typedef	union
unsigned	void	volatile	while

Ilustración 12: ANSI C Keywords

Una vez conocidas las diferentes palabras de ANSI-C debemos de conocer los tipos de datos que admite este lenguaje. Los tipos de dato como su propio nombre indican nos van a indicar el tipo de dato que se almacenará en las variables previamente definidas. Esto es muy importante dado que si no se declaran variables el programa no podrá almacenar datos.

Nombre	Descripción	Tamaño*	Rango de valores*
char	Carácter o entero pequeño	1byte	con signo: -128 to 127 sin signo: 0 a 255
short int (short)	Entero corto	2bytes	con signo: -32768 a 32767 sin signo: 0 a 65535
int	Entero	4bytes	con signo: -2147483648 a 2147483647 sin signo: 0 a 4294967295
long int (long)	Entero largo	8bytes	con signo: -2147483648 a 2147483647 sin signo: 0 a 4294967295
bool	Valor booleano. Puede tomar dos valores: verdadero o falso	1byte	true o false
float	Número de punto flotante	4bytes	3.4e +/- 38 (7 digitos)
double	De punto flotante de doble precisión	8bytes	1.7e +/- 308 (15 digitos)
long double	Long de punto flotante de doble precisión	8bytes	1.7e +/- 308 (15 digitos)

Ilustración 13: Tipos de datos de C

Por último, queda por definir el uso de constantes y funciones en el programa. Las constantes se declaran variables que no podrán ser modificadas a lo largo de la ejecución del programa. Por su parte, las funciones permiten realizar llamadas a estas mismas para poder realizar ciertas operaciones programadas permitiendo así un aligeramiento del código y un código más intuitivo y sencillo para el usuario.

Las constantes existentes en C son:

Tipo de dato	Ejemplo
Double	-127.82
Float	25.87
Unisgned int	2U
Short int	200
Long int	6400L
Int	245
Char	‘L’

Tabla 2: Constantes del lenguaje C

Las funciones en C se declaran siguiendo una estructura en la que existe cuerpo y encabezado. Para clarificar esto muestro un ejemplo donde se ve el encabezado formado por un tipo, nombre y parámetros formales, y el cuerpo de la función de ejemplo:

```
Double calcular_peso (double num1, num2)
{
Double resultado;
resultado = (num1 + num2) / 2;
Return resultado;
}
```

3. OBJETIVOS

Para la consecución de este proyecto también se debe indagar en numerosas tecnologías, herramientas y métodos de programación no vistos a lo largo de la carrera. No obstante, esto no es ningún impedimento gracias la abundante cantidad de recursos y fuentes de información existentes en internet y bibliotecas.

La sobreinformación es uno de los grandes problemas a la hora de abarcar un proyecto, así pues, es fundamental establecer una serie de objetivos que permitan llegar al usuario a una solución óptima a través de la definición de diferentes tipologías de objetivos. En este proyecto se tratará de abarcar tipologías generales y específicas.

3.1. OBJETIVO GENERAL

El objetivo general de este proyecto es el diseño y construcción de un dispositivo hardware-software capacitado para la monitorización de parámetros ambientales de un determinado ecosistema. Dicha solución ha de ser capaz de transmitir y recibir datos mediante tecnología LoRa para permitir el acceso remoto a los usuarios.

Antes de proseguir con los objetivos específicos hay que aclarar que es un desafío realizar este dispositivo utilizando un microcontrolador tan distinto con respecto al Arduino, tanto en arquitectura hardware como en su sistema de programación. Como se puede observar en internet, muchas de las soluciones semejantes se realizan utilizando esta placa y su entorno de programación propio.

3.2. OBJETIVOS ESPECÍFICOS

Los diferentes objetivos específicos necesarios para lograr la consecución del objetivo general son:

- Estudio exhaustivo de tecnología LoRaWAN.
- Configuración de los diferentes periféricos del sistema LoRaWAN.
- Estudio de programas de Software para realizar el diseño del dispositivo.
- Análisis de ruteado del dispositivo.

- Compra de componentes y envío del dispositivo a fabricación.
- Aprendizaje y desarrollo de tecnologías de soldado para microcontroladores.
- Programación del dispositivo.
- Testeo de la solución en un entorno real.
- Establecimiento de conexión a red LoRa a través de Gateway.
- Representación de datos a través de una interfaz básica.

4. PLAN DE DESARROLLO DEL PROYECTO

4.1. METODOLOGÍA

La metodología más adecuada para realizar el trabajo es la de cascada. Esta metodología consiste en la resolución de un problema mediante la definición de objetivos con una vida útil programada. Esta suele ser empleada para desarrollos de software y su etimología procede de su transcurso en sentido descendente desde la implementación hasta la resolución.

Comúnmente la metodología de cascada consta de diferentes fases como podemos ver en la Ilustración 14: Fases de una metodología en cascada:

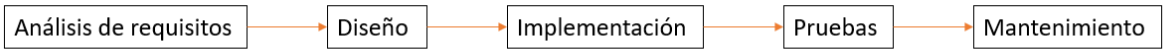


Ilustración 14: Fases de una metodología en cascada

Como se puede observar la consecución de cada uno de los bloques desencadena el comienzo del siguiente. Para este proyecto se deben de definir cada uno de los diferentes apartados de trabajo dentro de cada una de las fases. Estas fases son:

a) **Análisis de requisitos:** La consecución de esta es fundamental para el futuro desarrollo del proyecto. En este apartado se definen todas y cada una de las sub-fases necesarias para conseguir este análisis de requisitos.

    a. **Estudio de propuesta**

    b. **Documentación técnica necesaria**

b) **Diseño:** Este proyecto se fundamenta en el desarrollo de una placa de origen electrónico. Esto es de elevada importancia ya que cualquier fallo en el prototipado y desarrollo del hardware del dispositivo puede originar un retraso en la realización de la placa y con total seguridad en un proyecto descartado.

    a. **Recopilación y estudio de datasheets**

    b. **Prototipado de la solución**

c. **Testeo en software de la placa**

d. **Adquisición de materiales**

e. **Contacto con proveedores**

c) **Implementación:** el apartado de implementación se inicia tras la obtención de todos los materiales por parte de los diferentes proveedores electrónicos y eléctricos consultados. Además, la implementación ha de incluir el soldado de cada componente y para realizarlo se ha de disponer de una plancha de soldado SMD.

d) **Pruebas:** Una vez la placa ha sido totalmente montada se realizan los diferentes testeos del prototipo. Aunque será dispuesto con posterioridad en este documento, la placa sufrirá numerosas pruebas antes de alcanzar su diseño final. Por otra parte, cabe recalcar que el testeo se realizará con cada sensor propuesto ya que la disposición de estos es dependiente de los tiempos de envío de las diferentes empresas.

a. **Testeo de comunicación con la placa**

b. **Testeo de cada sensor implementado**

c. **Testeo de comunicación LoRa**

e) **Mantenimiento:** La fase final del proyecto involucra escasos apartados debido a que el prototipo solo debe de tener el mantenimiento de actualización de software y de cambio de baterías.

a. **Búsqueda de software reciente**

b. **Chequeo periódico de los datos**

4.2. TECNOLOGÍAS

El desarrollo de este proyecto se considera multidisciplinar, se van a emplear aplicaciones tan básicas como visores de gerbers online, hasta programas de desarrollo avanzado de código. Así pues, el esquema de tecnologías empleado por orden cronológico será:

Tecnologías						
Kicad 6.0	Gerb View	Stm32Cube Mx	Stm32CubeIDE	MobaXterm	Usart	Lora

Tabla 3: Tecnologías empleadas en el proyecto

Cada una de estas tecnologías es escogida dependiendo de los criterios y justificaciones recopilados a continuación:

a) **Kicad 6.0:** Kicad es un programa de código abierto especializado en la automatización de diseños electrónicos. Este programa fue lanzado en 1992 y programado en el lenguaje C++.

El programa presenta numerosas funcionalidades dentro de su página de inicio como se puede observar:

Ilustración 15: Menú principal del Software Kicad 6.0

El programa cuenta con 9 funciones diferentes para la realización de proyectos de origen eléctrico. Estas funciones son:

1. **Editor de esquemas:** Permite editar el esquema del proyecto a realizar y situado en el recuadro rojo de la ilustración 11.
2. **Editor de símbolos:** Este editor capacita la formación y edición de librerías de símbolos del sistema, ya sea a nivel de un solo proyecto o a nivel general.
3. **Editor de placas:** Una vez realizado el esquemático del proyecto este editor permite al usuario modificar el diseño de la placa de este.
4. **Editor de huellas:** Pone a disposición del usuario la visualización y edición de la librería de huellas del proyecto en particular o bien del programa en general.
5. **Visor Gerber:** Este visor habilita la visualización de modelos Gerber.
6. **Conversor de imágenes:** Convierte imágenes de mapa de bits en símbolos de esquema o huellas de placa.
7. **Herramientas de cálculo:** Muestra herramientas para el cálculo de resistencias, corrientes, capacidades etc.
8. **Editor de hoja de trabajo:** Permite al usuario editar los bordes y bloques de títulos de hojas de dibujo para el consecuente uso en esquemas y diseños de placas.

9. **Administrador de complementos de contenido:** Administra los paquetes descargables online de Kicad y repositorios de terceros como por ejemplo GitHub.

El motivo de incorporación de esta solución al proyecto es debido a que su interfaz es muy intuitiva, su ligereza/limpieza/eficacia de procesado de información, y por su condición no económica de adquisición.

b) GerbView: Tras la lectura del apartado anterior se observa que el Kicad 6.0 ya dispone de visualizador de Gerbers, sin embargo, para la realización de este proyecto se ha empleado el programa llamado "Gerber Viewer" que permite la visualización de este tipo de archivos. Aunque más adelante se dispondrá la solución técnica más detallada el motivo de elección de este programa y no del Kicad ya dispuesto es debido a que lo recomendaba e implementaba en su web el fabricante de PCB escogido (JLCPCB).

La visualización de este programa dependiendo de la vista y sección de la placa que desee el usuario visualizar será la siguiente:

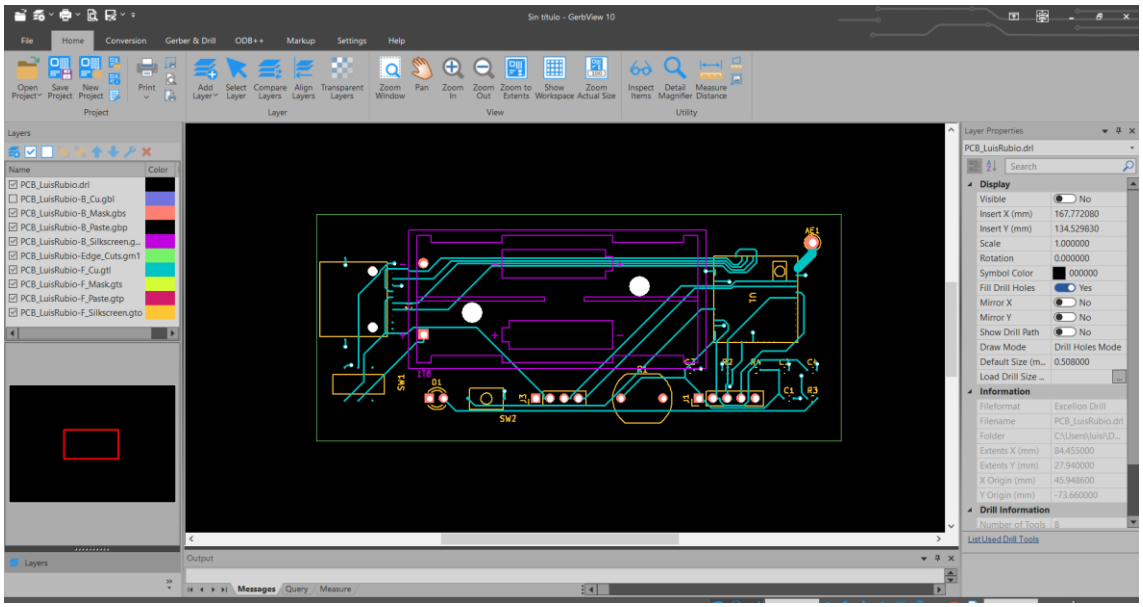


Ilustración 16: Visualización del programa GerbViewer

c) STM32CubeMx: Herramienta de interfaz gráfica empleada para la programación del microcontrolador. En el proyecto a realizar se programará un STM32WLE5CCU6 con arquitectura LoRa incorporada. Posteriormente se detallará la programación y configuración de este programa para su consecuente implementación técnica, sin embargo, debemos de percatarnos de que en la solución final este STM32CubeMx se encuentra en el interior del software STM32CubeIDE.

Gracias al STM32CubeMx se generarán en nuestro proyecto las librerías necesarias para la implementación de los distintos sensores. Además, se debe recalcar que este software es muy intuitivo ya que permite determinar a través de la selección de los diferentes tipos de protocolos de transmisión de datos dónde se encontrarán

situados los pines consecuentes. Esto puede observarse en la Ilustración 17: Panel de control inicial de STM32CubeMx:

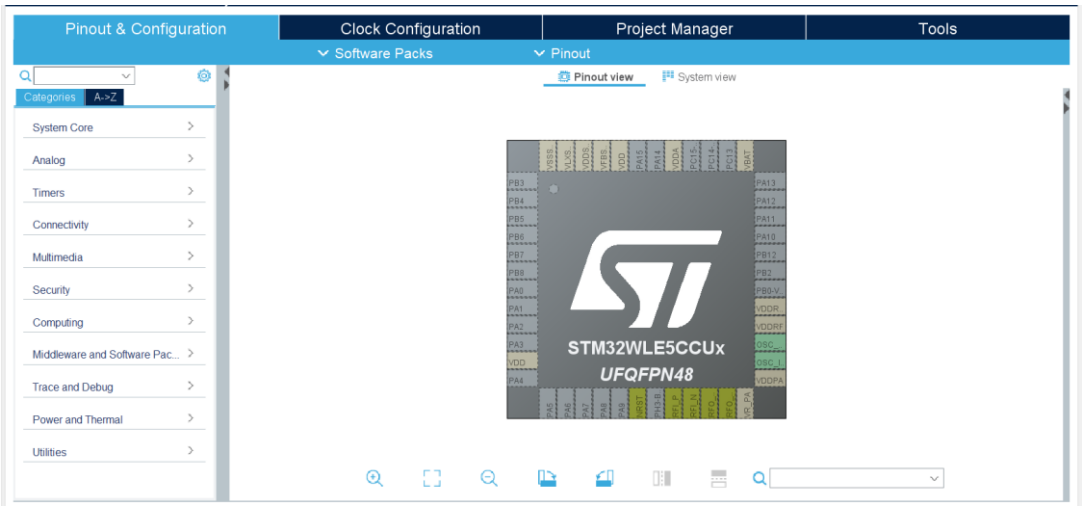


Ilustración 17: Panel de control inicial de STM32CubeMx

Es de importante relevancia destacar que tras la elección del microprocesador en el programa puede darse la situación de conflicto entre las definiciones de los pines empleados en nuestro esquemático con respecto al datasheet del Rak3172 y los pines reales de configuración del STM32WLE5CCU6.

El programa STM32CubeMx es la mejor solución para la programación del microcontrolador ya que su interoperabilidad con el microcontrolador es completa debido a que proceden del mismo desarrollador (STMicroelectronics).

Un esquema simplificado que represente la sinergia entre los programas de STM32 sería el mostrado en la Ilustración 18-Sinergías de programas de STMicroelectronics:

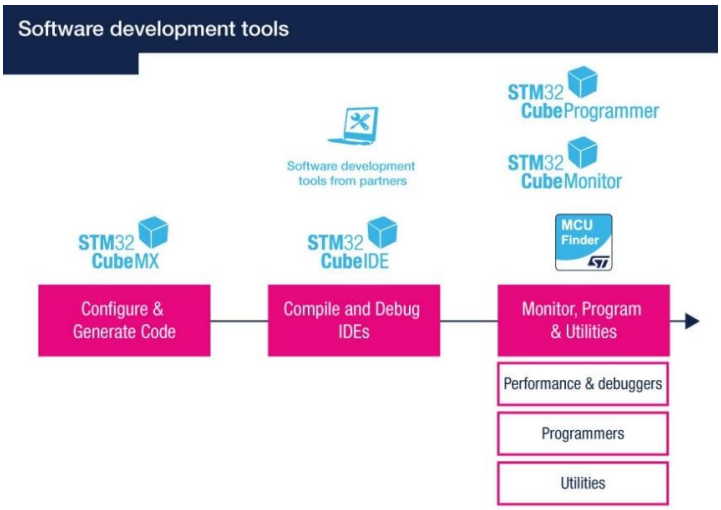


Ilustración 18-Sinergías de programas de STMicroelectronics

d) STM32CubeIDE: El programa descrito pertenece a la misma empresa mencionada anteriormente (STMicroelectronics). STM32CubeIDE se presenta como una plataforma avanzada de desarrollo de software para lenguaje C y C++. Como se ha mencionado con anterioridad, este programa cuenta con el STM32CubeMx incorporado.

Cada programa desarrollado en este compilador cuenta con dos localizaciones de ficheros de relevante importancia, además presenta en su pantalla de inicio un explorador de proyectos intuitivo y sencillo de emplear como se puede observar en la Ilustración 19: Pantalla de inicio de software STM32CubeIDE:

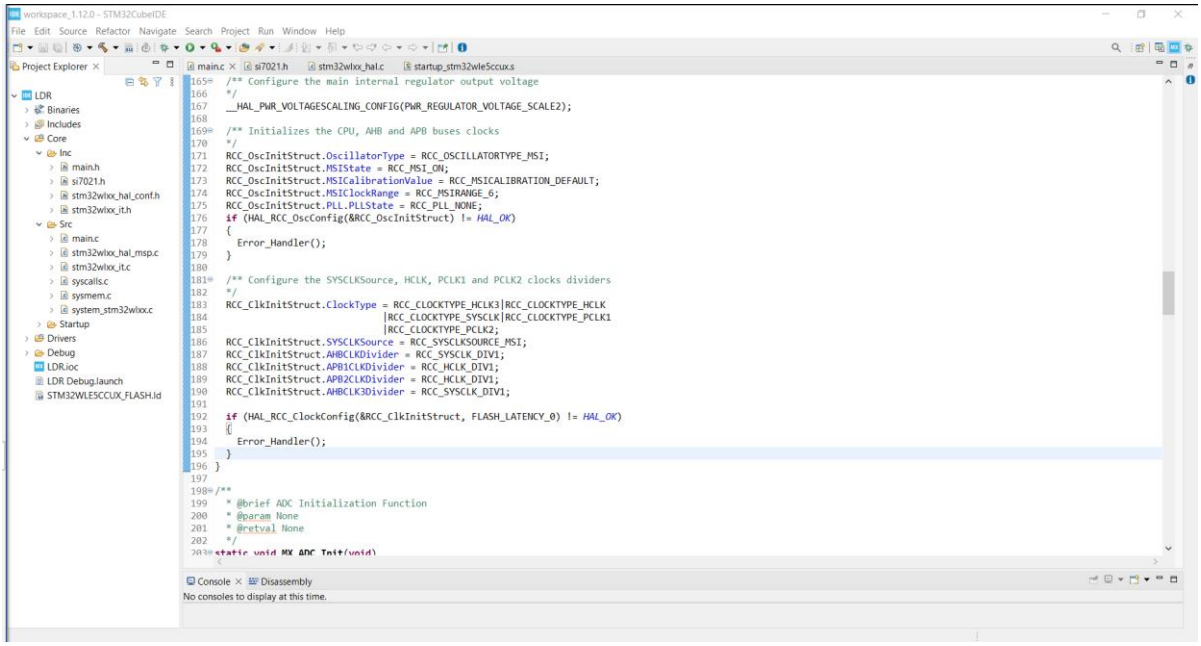


Ilustración 19: Pantalla de inicio de software STM32CubeIDE

Como se ha mencionado en el anterior párrafo las localizaciones de ficheros más importantes sobre los diferentes proyectos de programación realizados en el programa son:

- **Inc:** Incluye todos los ficheros empleados en el programa principal “main.c”.
- **Src:** Incluye los ficheros de programación principal.

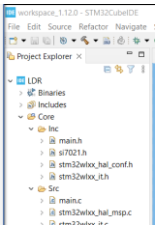


Ilustración 20: Explorador de proyectos en STM32CUbeIDE

Debido a la gran relevancia de este programa para la programación e implementación del dispositivo final han de ser mencionadas los requisitos técnicos mínimos a nivel de hardware recomendadas por los desarrolladores.

	Memoria RAM	Espacio en disco duro	S.O
Mínimo	2 Gbytes	6 Gbytes	Windows 10
Recomendado	4 Gbytes	15 Gbytes	Windows 11

Tabla 4: Requisitos de hardware software STM32CubeIDE

En caso de que el proyecto sea virgen y no se le asigne ninguna configuración desde el STM32CubeMx entonces sólo contará con el "main.c" típico de los entornos de programación. Sin embargo, dado que se está programando un microcontrolador el programa viene determinado con numerosos espacios para rellenar con código por el usuario, además de las funciones prestablecidas de serie para el control básico de programación del STM32WLE5CCU6. Algunas de estas funciones son:

```
int main(void){

}

  while (1){

  }
  void SystemClock_Config(void){

  }
  void Error_Handler(void){

  }
  void assert_failed(uint8_t *file, uint32_t line){

  }
```

Ilustración 21: Funciones definidas de serie en main.c

Estas funciones son predefinidas para el control del reloj del micro y también para la gestión de errores durante la compilación. A raíz de la definición de pines y funciones/protocolos de comunicación de estos, el programa implementará por sí solo el código consecuente de los parámetros definidos en el STM32CUbeMx.

STM32CubeIDE no es solo un programa para la redacción de código, presenta numerosas funciones típicas de cualquier compilador como por ejemplo el Debugger:

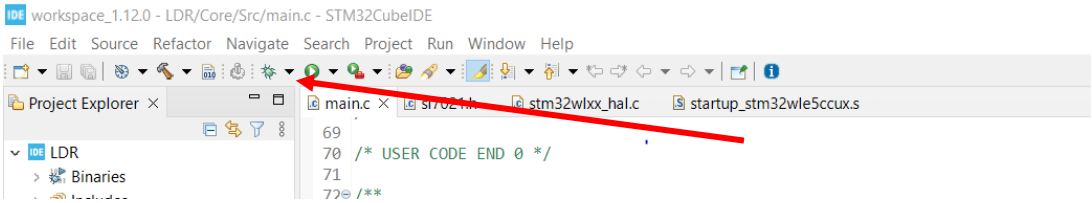


Ilustración 22: Localización del Debugger y diferentes herramientas del programa

Es de vital importancia ceñirse a los espacios que el programa reserva al usuario para la programación ya que cada cambio realizado en el STM32CubeMx eliminará todo código que no se encuentre situado entre los comentarios de "/*USER CODE BEGIN X*/ y de /*USER CODE END X*/. Lógicamente las variables "X" varían en función del número de códigos realizados. Como ejemplo puede verse la ilustración Iluestración 23: Comentarios delimitadores de código en STM32CubeIDE :

```
70 /* USER CODE END 0 */
71
72 /**
73   * @brief  The application entry point.
74   * @retval int
75   */
76 int main(void)
77 {
78   /* USER CODE BEGIN 1 */
79
80   /* USER CODE END 1 */
81
82   /* MCU Configuration--------------------------------------------------------*/
83
84   /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
85   HAL_Init();
86
87   /* USER CODE BEGIN Init */
88
89   /* USER CODE END Init */
90
91   /* Configure the system clock */
92   SystemClock_Config();
93
```

Ilustración 23: Comentarios delimitadores de código en STM32CubeIDE

El motivo de elección de este entorno de programación es la facilidad de uso, la perfecta interoperabilidad que tiene con otros programas empleados y la versatilidad de este para ejecutar el código.

e) **MobaXterm:** Esta herramienta es una terminal mejorada de Windows que incluye varias herramientas que permiten la computación de datos remota. La pantalla de inicio es elevadamente sencilla de entender y su apariencia es la siguiente:

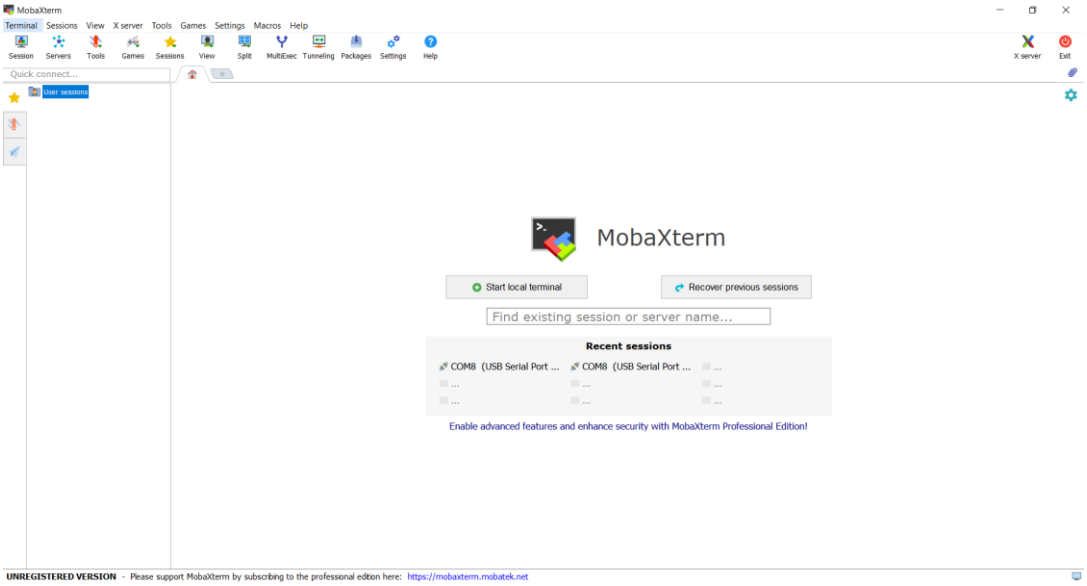


Ilustración 24: Pantalla de inicio de MobaXterm

El motivo fundamental de elección de esta terminal fue la recomendación directa de mi tutor del TFG Diego Gachet. Se ha de recalcar que los datos monitorizados serán transmitidos por USART y no serán elevadamente complejos. Así pues, esta terminal presenta un elevado potencial de utilización, además de gran intuitivita para el usuario con escaso conocimiento en manipulación de terminales.

Esta terminal ofrece una gran variedad de protocolos de comunicación que permitan la correcta monitorización de los parámetros deseados. Los puertos de comunicación soportados son los siguientes:

• **SSH**	• **Telnet**	• **Rsh**
• **Xdmcp**	• **RDP**	• **VNC**
• **FTP**	• **SFTP**	• **Serial**
• **File**	• **Shell**	• **Browser**
• **Mosh**	• **Aws S3**	• **WSL**

Tabla 5: Puertos de comunicación soportados

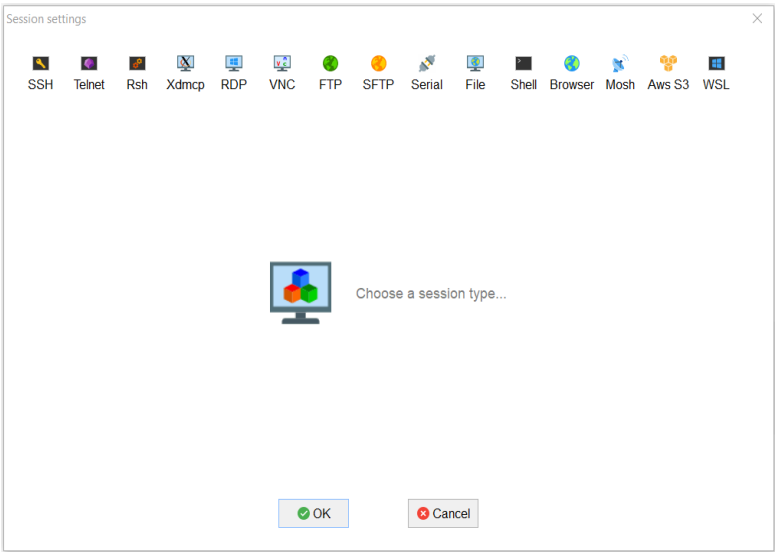
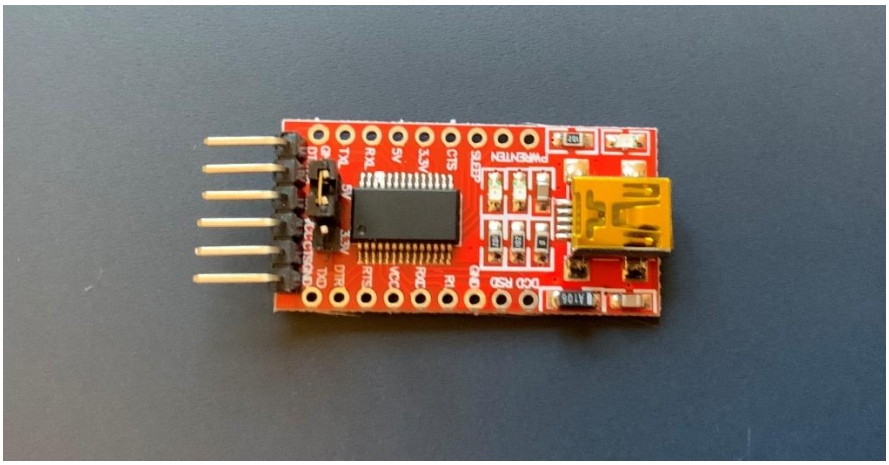


Ilustración 25: Puertos de transmisión de datos disponibles en MobaXterm

f) **Tecnología USART:** Dada su implementación en el apartado de tecnología de este proyecto se ha de conocer su función y definición. La tecnología USART (Universal Synchronous and Asynchronous serial Receiver and Transmitter) es una tecnología empleada para la comunicación serial. En especial esta tecnología es implementada en el proyecto dada su elevada flexibilidad.

El dispositivo encargado de esto es el dispuesto en la Fotografía 3: AZDelivery 3X Adaptador Compatible con FT232RL, USB a TT 3.3V-5V :



Fotografía 3: AZDelivery 3X Adaptador Compatible con FT232RL, USB a TT 3.3V-5V

El esquema de conexión de este adaptador TTL a USB para llevar a cabo la solución del dispositivo se detallará en los siguientes apartados del trabajo, no obstante, al final de este documento en la sección de anexos se encuentra adjunto el datasheet del dispositivo. Generalmente se conectarán intercambiados los puertos del dispositivo y de la PCB de tal forma que quede una interconexión como la siguiente:

"UART RX" del AZDelivery → "UART TX" de la PCB

"UART TX" del AZDelivery → "UART RX" de la PCB

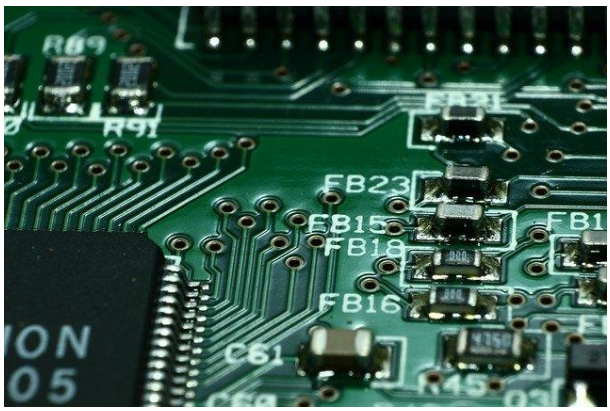
Las principales características de esta tecnología vistas comúnmente en un ATmega8 por ejemplo son:

- **Operación Full Duplex**
- **Registros de transmisión y recepción independientes**
- **Operación síncrona y asíncrona.**

g) **Tecnología LoRa:** Esta tecnología se ha mencionado con anterioridad en el trabajo, no obstante, dado que es uno de los factores fundamentales para la consecución de este trabajo se va a definir como la tecnología de comunicación de largo alcance y baja potencia capacitada para conectar dispositivos a través del Internet de las Cosas con una red amplia de baja potencia como es LPWAN (Low Power Wide Area Network".

h) **Tecnología SMD:** Las siglas hacen referencia a Surface Mount Device. Esta es la tecnología de montaje superficial empleada en aplicaciones de electrónica. La tecnología SMD permite soldar sobre las placas de circuito impreso componentes con elevada precisión dada la complejidad derivada del reducido de tamaño de sus conexiones.

Las PCB generalmente son de plomo-estaño o cobre, así pues, se necesita de una mezcla de flux y pequeñas partículas de estaño presentes en la pasta de soldadura, que se dispondrán sobre los conectores mediante plantillas diseñadas y realizadas por torneado. El proceso recibe el nombre de estarcido.



Fotografía 4: ejemplo de soldadura SMD en PCB

4.3. PLAN DE TRABAJO

En el subapartado 1 de este proyecto de fin de grado se ha desarrollado la metodología que se va a emplear para la consecución de este dispositivo basado en LoRa. Para la correcta consecución de esto se debe desarrollar un plan de trabajo acorde a la metodología planteada para así poder lograr un proyecto con la menor aparición posible de inconvenientes durante el desarrollo.

Todas las tareas descritas en los siguientes párrafos se corresponden de manera global con cada una de las fases de la metodología en cascada propuestas previamente. Lógicamente, se debe de dar un detallado ligeramente más técnico en este apartado, pero sin ser realmente lo más determinante, ya que lo realmente relevante es la explicación de los hitos alcanzados y que pueden vislumbrarse en el diagrama de Gantt dispuesto en la ilustración:

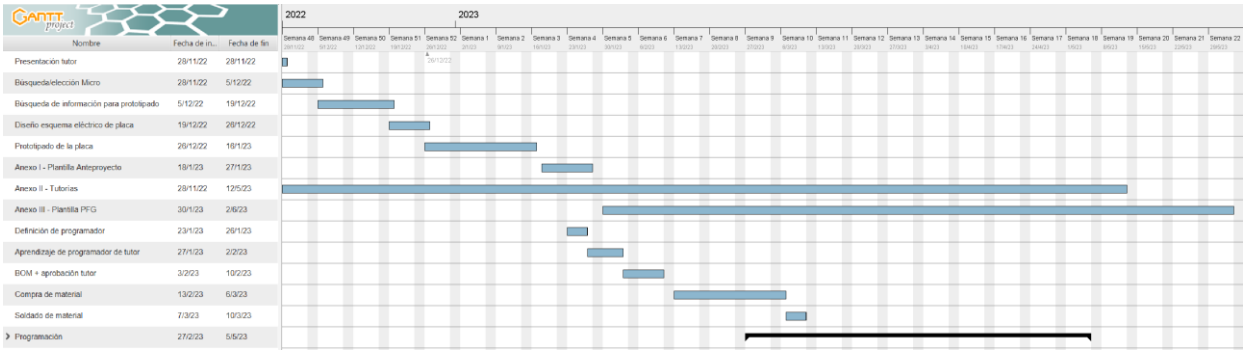


Ilustración 26: Diagrama de Gantt

Para el **análisis de requisitos** se van a plantear en función de sus dos sub-fases las siguientes tareas:

a) **Estudio de propuesta y documentación técnica necesaria:** Este estudio corresponde a las dos tareas del diagrama de Gantt de "Búsqueda/elección Micro" y "Búsqueda de información". Dado que con la consecución de una comienza el inicio de la otra tarea las fechas de inicio y final comprenden el intervalo del 28 de noviembre del 2022 al 19 de diciembre del 2022.

   Posteriormente se entrará en mayor detalle de cada tarea, sin embargo, se ha creído consecuente realizar una breve definición de cada una de estas tareas para así favorecer la futura asimilación al lector del proyecto de fin de grado.

   Ambas tareas descritas hacen referencia a la elección de un microcontrolador adecuado para realizar la solución propuesta en el documento aportado por parte de la universidad y nombrado como "Documento4". Además, comienza también la búsqueda de información con respecto a sensores, protocolos de comunicación, microcontroladores etc. Se ha de recalcar que la tarea de búsqueda de información para un trabajo del calibre de un final de carrera es continua, sin embargo, se ha decidido que por comodidad y concordancia con el desarrollo y redacción de este proyecto se establezca esta dicotomía en tanto en cuanto a términos.

En la siguiente ilustración se observa el apartado del diagrama de Gantt referente a lo mencionado:

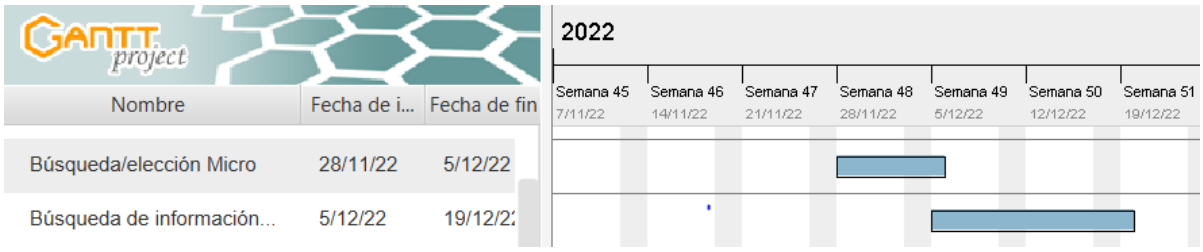


Ilustración 27: Fragmento de diagrama de Gantt para el análisis de requisitos

Para el **diseño** se van a plantear en función de sus cinco subfases las siguientes tareas:

b) **Recopilación y estudio de datasheets: Esta** recopilación está íntimamente relacionada con las anteriores del análisis de requisitos. Las tareas que se encuentran en el diagrama de Gantt que repercuten a este apartado son las mismas que antes y por ende no se van a repetir de nuevo.

c) **Prototipado de la solución y testeo en software de la placa**: Para la consecución de esta subfase se realizarán las tareas de "Diseño esquema eléctrico" y "Prototipado de la placa" comprendidas entre el 19 de diciembre de 20222, el 16 de enero del 2023. Básicamente estas tareas incumben tanto el manejo de programas como Kicad 6.0 y GerberViewer.

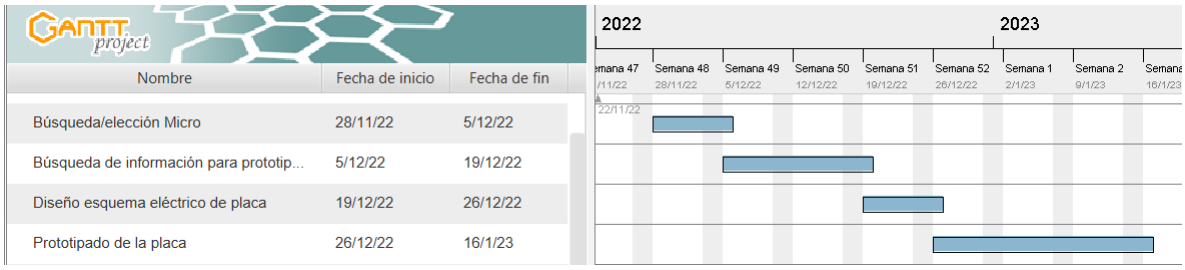


Ilustración 28: Fragmento de diagrama de Gantt para el prototipado y testeo

d) **Adquisición de materiales y contacto con proveedores:** Esta subfase se centra en las tareas de "BOM + Aprobación tutor" y "Compra de material". Estas tareas comprenden el período del 3 de febrero del 2023 al 6 de marzo del 2023. Dada la obviedad de estos apartados no se cree consecuente describirlas con excesivo detalle.

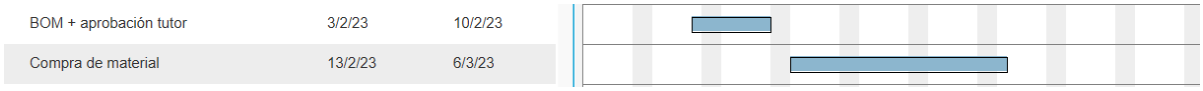


Ilustración 29: Fragmento de diagrama de Gantt para la adquisición y contacto

Para la **implementación** se van a plantear en función de su subfase la siguiente tarea**:**

e) Soldado de material: Dicha subfase se centra como su propio nombre indica en el soldado del material a la placa. En este caso se soldará con SMD, debido a la imposibilidad de soldar a mano y gracias a mi tutor se ha dispuesto durante una semana de una placa para soldar ya que los pines eran demasiado pequeños para ser soldados a mano. La tarea correspondiente a este apartado es la de "Soldado de material" y sucede entre el 7 de marzo del 2023 y el 10 de marzo del mismo año.

Ilustración 30: Fragmento de diagrama de Gantt para el soldado de material

Para las **pruebas** se van a plantear en función de sus tres subfases las siguientes tareas**:**

f) Testeo de comunicación con la placa: En este apartado se realiza una simple programación de un código para comprobar que el código y entorno de programación se ha aprendido de correcta manera. Además, el usuario se familiariza con la conexión y compilación vía USB con el dispositivo STLINK-V2. La tarea de esta subfase (al igual que las del resto de la fase de "pruebas" se encuentra en el apartado de programación de color negro del diagrama de Gantt. La tarea que incumbe a este testeo es la de "Programación de LED de testeo" y confluye entre el 27 de febrero y el 3 de marzo del 2023.

Ilustración 31: Fragmento de diagrama de Gantt para el testeo de comunicaciones

g) Testeo de cada sensor implementado: Dado el interés de medir la luminosidad de una sala, su temperatura y su humedad se van a plantear 2 programas diferentes que permitan estas tareas de forma separada. En primer lugar, se realiza la tarea de "Programación LDR" del 10 de marzo al 17 de marzo en la que se trata de obtener el valor de un fotorresistor LDR y convertirlo de valor analógico a digital con el comando HAL_ADC_PollForConversion("argumento"," Valor"). En segundo lugar, se realiza la tarea de "Programación de sensor si7021" del 17 de marzo al 27 de marzo en la que se trata de obtener y convertir siguiendo el propio datasheet del LDR los valores recomendados para la conversión, y el lugar de almacenamiento de los bits según datos recopilados.

Ilustración 32: Fragmento del diagrama de Gantt para el testeo de la LDR

h) Testeo de comunicación LoRa: Se trata de una de las tareas finales del proyecto, una vez conseguida la lectura de los datos a través de LoRa se finalizará el proyecto dado que la monitorización en un entorno más visual y "comercial" no se busca en este proyecto. La tarea encargada de esto es "Programación para Gateway y transmisión" y comprende del 28 de marzo al 5 de mayo.

Ilustración 33: Fragmento del diagrama de Gantt para el testeo de comunicación LoRa

Para el **mantenimiento** se van a plantear en función de sus dos subfases las siguientes tareas**:**

i) Búsqueda de software reciente: Aunque en el diagrama de Gantt no se ven reflejadas, esta subfase está destinada a la continua actualización del dispositivo en términos de software. Sin embargo, bien es cierto que, si se da la posibilidad de finalizar el proyecto de fin de grado antes de lo esperado, se planteará un diseño de hardware actualizado y una impresión 3D de una envoltura exterior para situarlo en un ecosistema.

j) **Chequeo periódico de datos:** Dicha tarea tampoco se verá reflejada en el diagrama de Gantt ya que es algo más dependiente del usuario final y en este caso no se considera abarcar dicho período de tiempo. Sin embargo, es cierto que el objetivo de esta tarea sería el de asegurar la continua transmisión de datos y su consecuente no pérdida del dispositivo dando cabida a posibles futuras mejoras como una emisión de un mensaje vía Telegram de que hay un problema de transmisión de datos.

A continuación, se dispone una imagen completa del Diagrama de Gantt, si su visualización no es correcta debido a su dimensionado se encontrará en el Anexo de este documento donde podrá observarse de forma correcta.

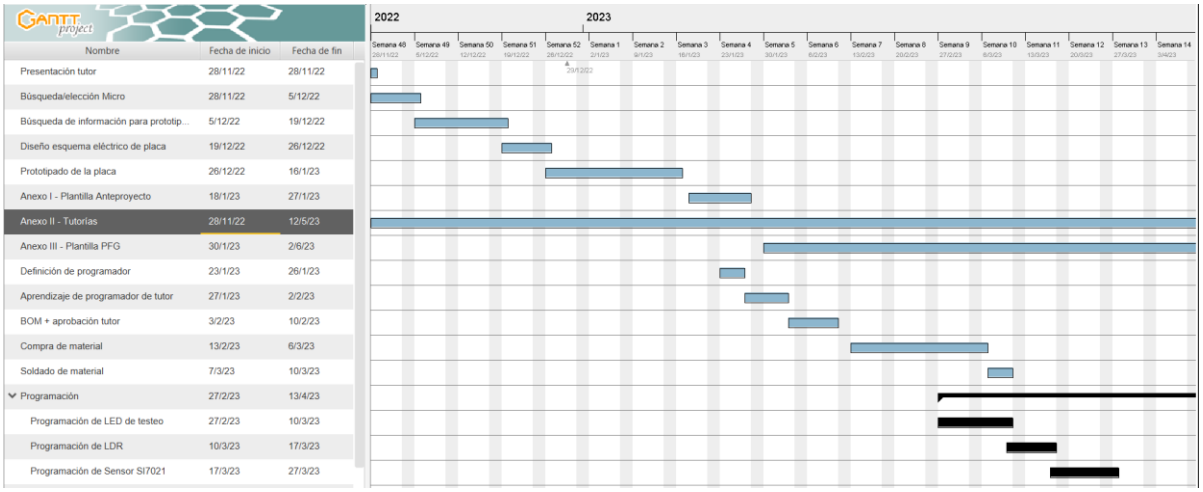


Ilustración 34: Diagrama de Gantt completo

4.4. Recursos.

La realización del presente proyecto no ha sido excesivamente cara, sin embargo, se ha de mencionar el hecho de que el precio va a ser bastante más elevado de lo planeado en el primer anteproyecto debido a que la mayoría de los componentes electrónicos son de origen chino por razones económicos y esto significa tiempos de espera para los productos demasiado elevados. Así pues, se han obtenido productos en empresas españolas o europeas cuyos tiempos de obtención de productos era menor pero que presentan mayores costes en los portes por la fragilidad de estos componentes.

Tras el siguiente párrafo se insertarán imágenes y breves descripciones sobre los diferentes componentes a nivel hardware que han sido adquiridos para la realización de este proyecto. Sin embargo, si se desea mayor nivel de detalle se expondrán las tablas posteriormente en un Excel el cual se encontrará adjunto a este trabajo.

Los componentes de hardware adquiridos son los siguientes:

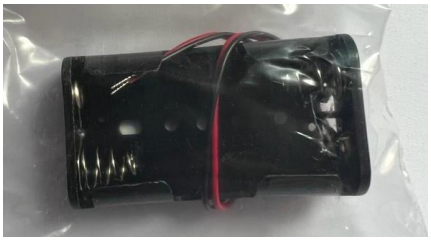
- **Resistencia de 10 KΩ:** La resistencia es conectada al puerto Reset y a la alimentación VDD en caso de que se desee usar al emplear el dispositivo.

Fotografía 5: Resistencia de 10 K

- **Condensador 0,1 uF:** El condensador de 0,1 uF es fundamental ya que se encarga de compensar la alimentación de baterías AA elegida para este dispositivo.

Fotografía 6: Condensador 0,1 uF

- **Portapilas AA:** El portapilas de dos pilas AA se encarga de alimentar el dispositivo, en la parte más técnica del trabajo se detallará la explicación y motivo de la elección de este tipo de solución.



Fotografía 7: Portapilas doble AA

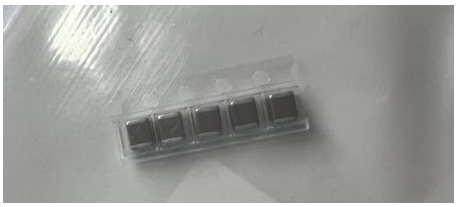
- **Resistencia de 22 KΩ:** La resistencia de 22 K desempeña la función de evitar la saturación del fotorresistor LDR empleado. Su valor calculado es fundamental para el correcto funcionamiento del dispositivo en cuanto al nivel de luminosidad recibido.

Fotografía 8: Resistencia de 22 K

- **Resistencia de 100 Ω:** La resistencia de 100 Ohms es la primera encargada de asegurar el correcto encendido y apagado del LED del dispositivo.

Fotografía 9: Resistencia 100 Ohms

- **Condensador de 10 uF:** Los condensadores de 10 uF se encargan al igual que el de 100 nF de compensar la corriente de alimentación de entrada al microcontrolador.



Fotografía 10: Condensador de 10 uF

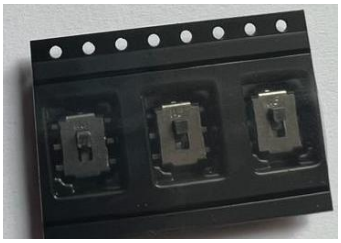
- **Micropulsador de 2 pines SW074:** El micropulsador de 2 pines se emplea para el apartado de Reinicio del dispositivo.

Fotografía 11: Micropulsador SW074 de Reset

- **Sensor temperatura + Humedad SI7021:** El Si7021 es un sensor de Sparkfun encargado de medir tanto la temperatura ambiental como el nivel de humedad.

Fotografía 12: Sensor SI7021

- **Pulsador Switch Slide:** El pulsador de deslizamiento tiene la función de encender y apagar el dispositivo final, permite el sumini9stro y corte de alimentación.



Fotografía 13: Switch de encendido/apagado

- **RAK3172:** El dispositivo LoRa encargado del funcionamiento del dispositivo y la consecuente transmisión de datos.

Fotografía 14: Módulo RAK3172

- **Convertidor IPEX:** El convertidor IPEX permite emplear antenas que no son IPEX como la comprada de la marca molex, para poder emplear las de la marca RAK que son de tipo antena dipolo omnidireccional.

Fotografía 15: Conversor IPEX antena

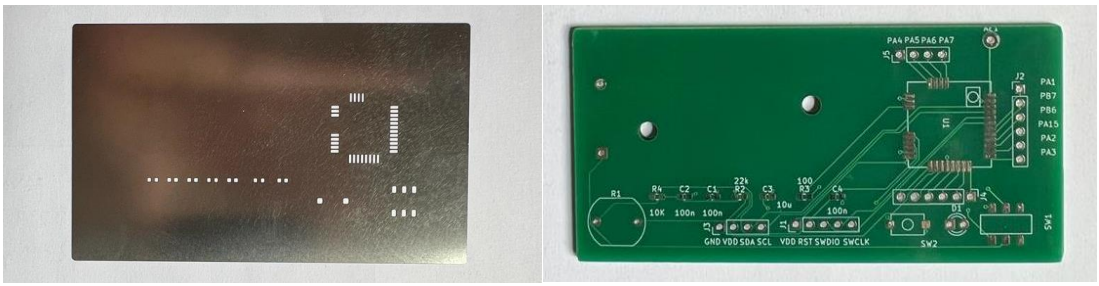
- **Antena RAK Wireless:** La antena es la encargada de permitir la transmisión de datos, nuestra frecuencia deseada al ser para Europa es de 863 MHz – 870 MHz.

Fotografía 16: Antena 863-870 MHz

- **Antena Molex:** Esta antena fue adquirida con posterioridad, su frecuencia es de 863 MHz – 870 MHz y es del tipo FPC (Flexible Printed Circuit).

Fotografía 17: Antena Molex 868-915 MHz

- **JLCPCB:** Esta es la PCB impresa y previa al soldado, lógicamente es la parte fundamental del proyecto. Incluye la Placa SMD para poder soldar con placa caliente.

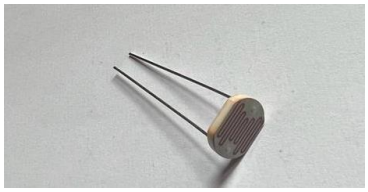


Fotografía 18: PCB + PLaca de SMD

- **LED:** Es un simple LED cuya función es la de control de la placa al programarla, es de elevada utilidad ya que permite detectar la carga completa del programa en la PCB.

Fotografía 19: LED

- **LDR:** La LDR de Farnell indica el índice de luminosidad del ecosistema presente.



Fotografía 20: LDR

- **Juego de pinzas antiestáticas:** Pinzas requeridas para el soldado de pequeños componentes.

Fotografía 21: Pinzas Antiestáticas

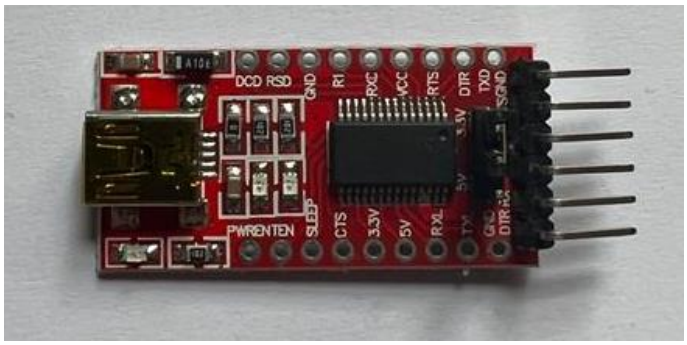
- **JBC FL 9582 - Flux - 15 ml:** Flux empleado para el soldado correcto del dispositivo.

Fotografía 22: Flux

- **MBO - Soldadura de Estaño:** Bobina de estaño empleada en el soldado de los dispositivos de la PCB.

Fotografía 23: Bobina de Estaño

- **AZDelivery USB a TTL:** Este adaptador de amplitud de 3,3 V a 5 V permite mediante las salidas UART_TX y UART_RX ver las mediciones obtenidas por la placa en una terminal.



Fotografía 24: AZDelivery USB a TTL 3,3-5 V

- **Gateway LoRa:** El Gateway LoRa es el encargado de realizar la interconexión entre nuestro dispositivo PCB y nuestro ordenador/dispositivo electrónico a través de la recepción de datos mediante conexión a internet a través del rúter de nuestra vivienda. Nuestro Gateway consta de una Raspberry Pi y un concentrador IC880a

Fotografía 25: Gateway LoRa

- **STLInk V2:** Programador empleado para conectar la placa al ordenador y poder programarla.

Fotografía 26: ST-Link V2

- **MBO - Jeringuilla de Estaño:** Estaño en crema que facilita la soldadura de componentes de alta densidad de pines.

Fotografía 27: Jeringuilla de Estaño

- **Plancha de soldado:** Esta plancha es la empleada para el soldado SMD, no está en el apartado costes dado que fue prestada por Juan Félix Mateos.

Fotografía 28: Plancha de soldado

En el Anexo del trabajo se dispondrán las diferentes facturas de obtención de productos con sus consecuentes pesos, no obstante, y dado la importancia para este proyecto se van a insertar a continuación tablas de precios de los diferentes productos necesarios para la obtención del dispositivo final.

Nombre	Cantidad	Necesarios	Precio	Precio proyecto
Resistor 10 K	5000	1	6,95	0,00139
Condensador 0,1 uF	25	3	3,96	0,1584
Portapilas AA	2	1	1,72	0,86
Resistor 22 K	5000	1	7,63 €	0,001526 €
Resistor 100	5000	1	9,92 €	0,001984 €
Condensador 10 uF	5	1	5,33 €	1,06600 €
Micropulsador 2 pines SW074	4	1	13,78 €	11,39 €
Sensor SI7021	2	1	20,92 €	10,46 €
Switch slide	3	1	7,53 €	2,51 €
RAK3172	2	1	11,98 €	5,99 €
Convertidor antena IPEX	1	1	5,00 €	5,00 €
Antena RAK3172	1	1	7,00 €	7,00 €
Envío RAK Store	NO APLICA	NO APLICA	25,59 €	25,59 €
Envío Mouser	NO APLICA	NO APLICA	40,00 €	20,00 €
Envío Farnell	NO APLICA	NO APLICA	28,00 €	14,00 €
Envío RS components	NO APLICA	NO APLICA	14,00 €	7,00 €
Antena Molex	3	1	7,44 €	2,48 €
JLC PCB	1	1	26,93 €	26,93 €
LED	2	1	1,34 €	0,67 €
LDR	4	1	8,44 €	2,11 €
Juego 6 pinzas antiestáticas	1	1	10,94 €	10,94 €
JBC FL 9582 - Flux - 15 ml	1	1	9,24 €	9,24 €
MBO - Soldadura de Estaño	1	1	10,25 €	10,25 €
AZDelivery USB a TTL	3	1	15,99 €	5,33 €
STLInk V2	1	1	15,00 €	15,00 €
MBO - Jeringuilla de Estaño	1	1	18,66 €	18,66 €
TOTAL			333,52 €	212,62 €

Tabla 6: Precios del dispositivo final y proyecto

La tabla dispuesta previamente presenta 2 precios muy diferentes debido a que por decisión propia se han considerado los siguientes condicionantes:

a) Se considera que lo más adecuado es establecer una variable llamada "**Precio",** la cual corresponde al precio de absolutamente todos y cada uno de los productos comprados, independientemente de si se han usado o de si sobraban.

b) Por otro lado, se tiene la variable "**Precio proyecto**", corresponde con un precio de proyecto más aproximado al coste total necesario para fabricar un solo dispositivo.

Si bien es cierto que parece que es un proyecto elevado, no obstante, hay que recalcar que si quitásemos el IVA y si no tuviésemos la exigencia de tiempo tan limitada se puede afirmar que el precio para realizar una sola PCB con Lora integrada apenas superaría los 100 € por unidad.

Además, se añade una tabla adicional con los consecuentes links para la consecución de los dispositivos dispuestos anteriormente:

Nombre	Links
Resistor 10 K	https://es.rs-online.com/web/p/resistencias-de-montaje-en-
Condensador 0,1 uF	https://es.rs-online.com/web/p/condensadores-ceramicos
Portapilas AA	https://es.rs-online.com/web/p/portapilas/1854604
Resistor 22 K	https://es.rs-online.com/web/p/resistencias-de-montaje-en
Resistor 100	https://es.rs-online.com/web/p/resistencias-de-montaje-en
Condensador 10 uF	https://es.rs-online.com/web/p/condensadores-ceramicos
Micropulsador 2 pines SW074	https://www.dieltron.com/sw074-dtsh31nb-oem-dieltron
Sensor SI7021	https://www.mouser.es/ProductDetail/474-SEN-13763
Switch slide	https://www.mouser.es/ProductDetail/710-452403012014
RAK3172	https://store.rakwireless.com/products/wisduo-lpwan-
Convertidor antena IPEX	https://store.rakwireless.com/products/ipex-to-sma-connector
Antena RAK3172	https://store.rakwireless.com/products/blade-antenna?
Antena Molex	https://www.mouser.es/ProductDetail/538-211140-0100
JLC PCB	https://jlcpcb.com/
LED	https://www.mouser.es/ProductDetail/VCC/4304H1?
LDR	https://es.farnell.com/advanced-photonix/nsl-4962/ldr
Juego 6 pinzas antiestáticas	https://www.electronicaembajadores.com/es/Productos/
JBC FL 9582 - Flux - 15 ml	https://www.electronicaembajadores.com/es/Productos
MBO - Soldadura de Estaño	https://www.electronicaembajadores.com/es/Productos
AZDelivery USB a TTL	https://www.amazon.es/dp/B0753GY7FR?p
STLInk V2	https://www.amazon.es/dp/B07RKVM7H8?psc=1&ref=p
MBO - Jeringuilla de Estaño	https://www.electronicaembajadores.com/es/Producto

Tabla 7: Links de compra de materiales

4.5. Condicionantes y Limitaciones

Condicionantes a la hora del diseño del dispositivo ha habido menos de los esperados, el hecho de que muchos componentes dependan de cálculos previos hace que el diseño final del dispositivo sea viable y no contenga futuras modificaciones que hagan desechar componentes. Sin embargo, algo que sucedió en el período comprendido entre el anteproyecto y el proyecto final fue el desabastecimiento de algunos componentes como por ejemplo el sensor HTU21D que fue sustituido por el SI7021 de Sparkfun.

La mayor limitación del proyecto fue el consecuente soldado de los componentes. Obviamente, el soldado de piezas estándar con estaño y cautín no supuso ningún problema, sin embargo, para realizar el soldado del RAK3172 necesitábamos una estación de soldadura la cual rondaba los 150 euros y de la cual lógicamente no disponíamos (ni a nivel universitario). Gracias a Félix Mateos se realizó este apartado del trabajo como se ha mencionado en el apartado anterior del presente documento.

Por último, hay que mencionar que el mayor miedo y limitación al que se encontraba expuesto uno en este trabajo era el hecho de romper la placa mientras se programaba debido a una manipulación indebida. Este hecho podría suponer el fin de la propuesta en cuanto a hardware y quién sabe si la no presentación a tiempo de este proyecto dada la complicación de adquisición de materiales de nuevo.

5. DESARROLLO DE LA SOLUCIÓN TÉCNICA

El presente proyecto ha sido desarrollado en varias fases. Dado que se han mencionado en anteriores apartados las diferentes metodologías y procesos de trabajo que se han llevado a cabo. Se cree consecuente gestionar el apartado 5 de Desarrollo de la Solución Técnica a medida que se vaya implementando la solución para así favorecer el consecuente entendimiento del diseño del dispositivo LoRa.

Conforme lo previsto, se dispondrá el proyecto en el consecuente orden ilustrado en el flujograma dispuesto a continuación:

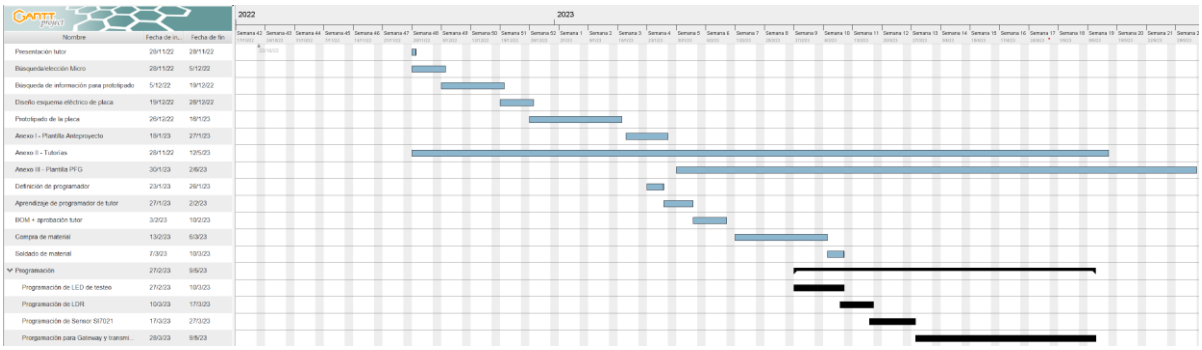


Ilustración 35: Diagrama de Gantt

5.1. ELECCIÓN DEL MICROCONTROLADOR

Una vez realizada la primera reunión con el tutor, y tras haber presentado la iniciativa del trabajo de fin de grado me dispongo a encontrar el mejor microcontrolador de bajo consumo que se adapte a los requisitos propuestos y que además posea el menor consumo posible.

Los microcontroladores para comparar serán:

a) Microchip

Aunque no se va a definir lo básico en cuanto a los microcontroladores, bien es cierto que los Microchip se consideran la mejor forma de iniciarse en el mundo de la programación/diseño de microcontroladores. Estos cuentan con una serie de ventajas y desventajas:

Ventajas	Desventajas
Sencillos de programar	Precio más elevado
Recomendados para principiantes	Problemas de comunicación con ordenadores u operaciones de lectura/escritura
Elevada disponibilidad de documentación	Presentan más ruido eléctrico
Gran variedad de dispositivos en el mercado	Algunos compiladores cuestan dinero
Mejores herramientas de desarrollo (compiladores, IDE, Debuggers)	

Tabla 8: ventajas vs desventajas de Microchip

b) Atmel

Ventajas	Desventajas
Mayor velocidad siempre que la velocidad del CLK sea la misma que para el microchip	Menor información disponible
Bajos precios para space-saving packages (MLF)	Más desconocidos que los microchips
Precio menor vs microchip	Presentan a veces peor calidad
Arquitectura eficiente en cuanto a código	Precio del programador recomendado

Tabla 9: Ventajas vs Desventajas de Atmel

Las familias de ambos existentes son:

Ilustración 36: Familias de PIC y AVR

Una vez comprendidas estas ventajas y desventajas de los modelos microchip y Atmel se procede a buscar un microcontrolador con una solución LoRa integrada que permita diseñar el dispositivo hardware/software buscado. Así pues, valoraremos las siguientes opciones:

c) Arduino Uno con RF95W

Es una opción muy interesante cuando se tienen conocimientos del entorno de programación y diseño de esta herramienta, como, por ejemplo, la certificación Arduino

Foundamentals. Desde otro punto de vista este Arduino Uno emplea un ATmega 328 el cual al ser tecnología Atmel parte de todas las ventajas descritas previamente. Se puede observar que este microcontrolador posee un amplio rango de voltaje (1.8 V – 5.5 V) por lo que permite emplear diversos sensores y a la vez presenta seis modos de reposo o "sleep modes" que permitirán reducir lo máximo el consumo del sistema y que dure la batería el mayor tiempo posible.

El datasheet y pin map es el siguiente:

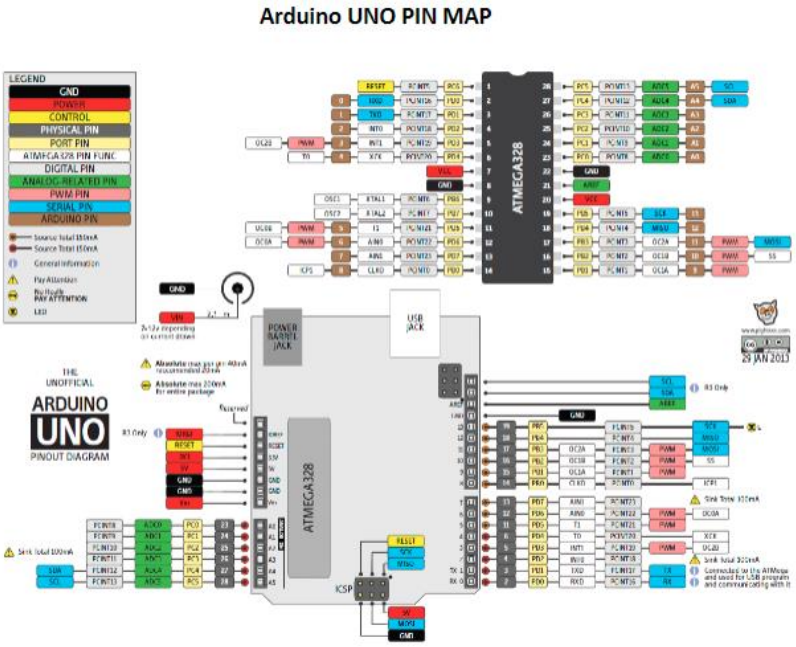


Ilustración 37: PinMap Arduino UNO

Features

- High Performance, Low Power Atmel® AVR® 8-Bit Microcontroller Family
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32KBytes of In-System Self-Programmable Flash program memory
  - 256/512/512/1KBytes EEPROM
  - 512/1K/1K/2KBytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C[1]
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
- Atmel® QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix® acquisition
  - Up to 64 sense channels
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
    - Temperature Measurement

Ilustración 38: ATMega 328 datasheet Features

Para la implementación del Arduino Uno se requiere además de un transceptor que será el encargado de actuar de receptor y de transmisor para la conectividad LoRa, en este caso se requiere de un HopeRF RFM95.

La elección del HopeRF RFM95 es debido a que es el más empleado para aplicaciones IoT de medición de datos, y a que tiene una elevada eficiencia como se puede observar en el consecuente datasheet:

KEY PRODUCT FEATURES

- LoRa™ Modem.
- 164 dB maximum link budget.
- +20 dBm - 100 mW constant RF output vs. 3.3V supply.
- +14 dBm high efficiency PA.
- Programmable bit rate up to 300 kbps.
- High sensitivity: down to -144 dBm.
- Bullet-proof front end: IIP3 = -12.5 dBm.
- Excellent blocking immunity.
- Low RX current of 10.3 mA, 200 nA register retention.
- Fully integrated synthesizer with a resolution of 61 Hz.
- FSK, GFSK, MSK, GMSK, LoRa™ and OOK modulation.
- Built-in bit synchronizer for clock recovery.
- Preamble detection.
- 127 dB Dynamic Range RSSI.
- Automatic RF Sense and CAD with ultra-fast AFC.
- Packet engine up to 256 bytes with CRC.
- Built-in temperature sensor and low battery indicator.
- Modue Size：16*16mm

APPLICATIONS

- Automated Meter Reading.
- Home and Building Automation.
- Wireless Alarm and Security Systems.
- Industrial Monitoring and Control
- Long range Irrigation Systems

Ilustración 39: Arduino HopeRF RFM95 features

d) RAK3172

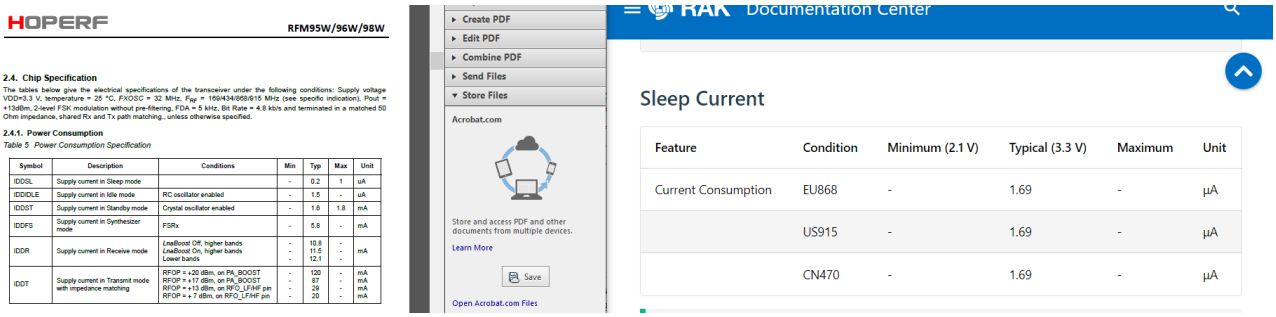
Esta es la opción finalmente elegida debido a que pese a ofrecer un máximo de 15 km con una antena muy optimizada para el rango de recepción de datos, se ha visto que puede desempeñar elevados rangos de recepción de datos si se configura adecuadamente. Sus especificaciones son:

Product Features

- LoRa module for Smart City, Smart Agriculture, Smart Industry
- Compact Form Factor: 15 x 15.5 x 3.5 mm 32 Pin Stamp Pad for PCB SMT mounting I/O ports: UART/I2C/SPI/ADC/GPIO
- Temperature range: -40 °C to +85 °C
- Supply voltage: 2.0 ~ 3.6 V
- Frequency range: 150 MHz to 960 MHz
- Low-Power Wireless Systems with 7.8 kHz to 500 kHz Bandwidth
- Ultra-Low Power Consumption 1.69 µA in sleep mode
- Core: ARM 32-bit Cortex – M4 with MPU Up to 256 KB flash memory with ECC
- 64 KB RAM
- RUI3 API Compatible
- AT and Binary Commands Support
- Supported bands: (EU433, CN470, RU864, IN865, EU868, AU915, US915, KR920, and AS923)

Tras una búsqueda exhaustiva se ha llegado a la conclusión que el segundo problema que puede presentar el RAK3172 es el excesivo consumo de batería, sin embargo, con unas simples modificaciones y teniendo en cuenta ciertos factores se puede revertir todas estas debilidades del microcontrolador y ser finalmente la mejor solución.

El consumo de corriente de este RAK3172 es el mostrado a continuación en la ilustración:



HOPERF RFM95W/96W/98W

2.4. Chip Specification

The tables below give the electrical specifications of the transceiver under the following conditions: Supply voltage VDD=3.3 V, temperature = 25 °C, *FXOSC* = 32 MHz, F_{RF} = 169/434/868/915 MHz (see specific indication), Pout = +13dBm, 2-level FSK modulation without pre-filtering, FDA = 5 kHz, Bit Rate = 4.8 kb/s and terminated in a matched 50 Ohm impedance, shared Rx and Tx path matching., unless otherwise specified.

2.4.1. Power Consumption

Table 5 Power Consumption Specification

Symbol	Description	Conditions	Min	Typ	Max	Unit
IDDSL	Supply current in Sleep mode		-	0.2	1	uA
IDDIDLE	Supply current in Idle mode	RC oscillator enabled	-	1.5	-	uA
IDDST	Supply current in Standby mode	Crystal oscillator enabled	-	1.6	1.8	mA
IDDFS	Supply current in Synthesizer mode	FSRx	-	5.8	-	mA
IDDR	Supply current in Receive mode	*LnaBoost* Off, higher bands *LnaBoost* On, higher bands Lower bands	- - -	10.8 11.5 12.1	- - -	mA
IDDT	Supply current in Transmit mode with impedance matching	RFOP = +20 dBm, on PA_BOOST RFOP = +17 dBm, on PA_BOOST RFOP = +13 dBm, on RFO_LFHF pin RFOP = + 7 dBm, on RFO_LFHF pin	- - - -	120 87 29 20	- - - -	mA mA mA mA

Create PDF
Edit PDF
Combine PDF
Send Files
Store Files
Acrobat.com
Store and access PDF and other documents from multiple devices.
Learn More
Save
Open Acrobat.com Files

RAK Documentation Center

Sleep Current

Feature	Condition	Minimum (2.1 V)	Typical (3.3 V)	Maximum	Unit
Current Consumption	EU868	-	1.69	-	µA
	US915	-	1.69	-	µA
	CN470	-	1.69	-	µA

Ilustración 40: Consumos de corriente del RAK3172

Por último, se ha de valorar la incorporación de la antena al diseño final. Las antenas ofrecen tres variantes en función de la frecuencia, nosotros al pertenecer a Europa elegimos la primera de 868 MHz:

Part Number	Frequency Band	Spreading Factor	Bandwidth	Effective Bitrate	Est. Sensitivity
RFM95W	868\915MHz	6 - 12	125 - 500 kHz	.293 - 37.5 kbps	-111 to -136 dBm
RFM96W/98W	433\470MHz	6 - 12	62.5- 500 kHz	.1465 - 37.5 kbps	-112 to -140dBm
RFM98W	169MHz	6 - 12	31.25- 125 kHz	73.24- 9375 bps	-118 to -143 dBm

Tabla 10: Frecuencias en MHz existentes a nivel mundial

5.2. CÁLCULOS Y DISEÑO DEL PROTOTIPO

En el apartado 5.1 se determina que la solución a desarrollar está basada en el RAK3172. Entre los múltiples motivos de elección se encuentra el hecho de que esto es un proyecto de fin de carrera y como tal ha de ser desafiante para el alumno, a la par que competitivo a nivel de reflejar los conocimientos adquiridos a lo largo de estos cuatro años sin olvidar la importancia de realizar un trabajo innovador y único.

En primer lugar, para la realización de la PCB basada en un microchip Rak 3172 debemos de conocer las conexiones y usos de los que dispone este:

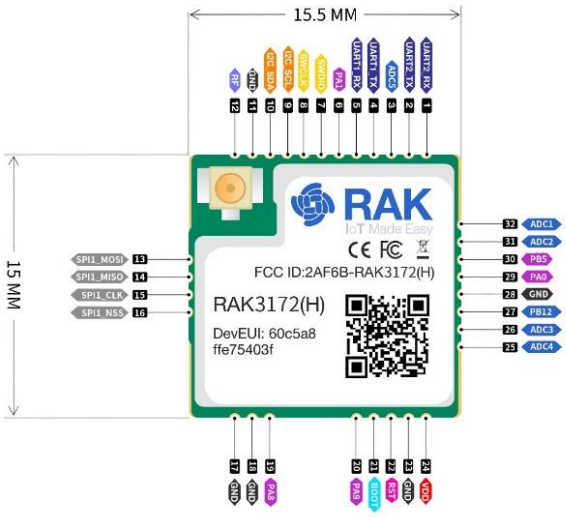


Ilustración 41:Configuración de pines del RAK3172

El diseño de la PCB se realizará en primer lugar en el programa Kicad 6.0. Por lo tanto, de todos los puertos dispuestos en la anterior ilustración es fundamental saber cuáles son entradas, cuáles salidas y cuáles son bidireccionales para así poder diseñar el primer modelo de la forma más correcta posible.

Nº de pin	Nombre	Tipo	Descripción
1	PA3/UART2_RX	I	Reserved - UART2/LPUART1 Interface (AT Commands and FW Update)
2	PA2/UART2_TX	O	Reserved - UART2/LPUART1 Interface (AT Commands and FW Update)
3	PA15/PIN_A4	I/O	GPIO and ADC
4	PB6/UART1_TX	O	UART1 Interface
5	PB7/UART1_RX	I	UART1 Interface
6	PA1	I/O	GPIO only
7	PA13/SWDIO		Reserved - SWD debug pin (SWDIO)
8	PA14/SWCLK		Reserved - SWD debug pin (SWCLK)
9	PA12/I2C_SCL	I/O	GPIO and I2C (SCL)
10	PA11/I2C_SDA	I/O	GPIO and I2C (SDA)
11	GND		Ground connections
12	RF		RF Port (only available on RAK3172 No-IPEX connector variant)
13	PA7/SPI1_MOSI	I/O	GPIO and SPI (MOSI)
14	PA6/SPI1_MISO	I/O	GPIO and SPI (MISO)
15	PA5/SPI1_CLK	I/O	GPIO and SPI (CLK)
16	PA4/SPI1_NSS	I/O	GPIO and SPI (NSS)
17	GND		Ground connections
18	GND		Ground connections
19	PA8	I/O	GPIO only

20	PA9	I/O	GPIO only
21	BOOT0		Boot0 mode enable pin - high active
22	RST		MCU Reset (NRST)
23	GND		Ground connections
24	VDD		VDD - Voltage Supply
25	PA10/PIN_A3	I/O	GPIO and ADC
26	PB2/PIN_A2	I/O	GPIO and ADC
27	PB12	I/O	10 kΩ internally pulled-up for high freq variant or pulled-down for low freq variant
28	GND		Ground connections
29	PA0	I/O	GPIO only
30	PB5	I/O	GPIO only
31	PB4/PIN_A1	I/O	GPIO and ADC
32	PB3/PIN_A0	I/O	GPIO and ADC (3º pin del puerto B)

Tabla 11: Configuración de pines del RAK3172

Es fundamental advertir al usuario diseñador que antes de realizar el modelo en Kicad 6.0 se ha de documentar bien sobre el microchip empleado, ya que, aunque en el datasheet oficial pongan los tipos que son (entrada, salida o bidireccional) bien es cierto que, si se consulta el datasheet del STM32WLE5CCU6 algunas de ellas varían. Esta es la explicación del por qué en el esquema realizado algunas no son iguales al datasheet de RAK.

Algunas anotaciones aprendidas con respecto al estudio de este RAK3172 es que hay ciertos pines de elevada importancia:

1. **Pines de programación:** Estos se emplean para programar el dispositivo, en este caso uno es para programar según el estándar SWD para ARM (pin 7) y el otro según SWC para otros tipos. En nuestro caso nos interesa el pin 7 ya que el STM32W es ARM.

7	PA13/SWDIO		Reserved - SWD debug pin (SWDIO)
8	PA14/SWCLK		Reserved - SWD debug pin (SWCLK)

Tabla 12: Pines de programación relevantes

2. **Pines genéricos a la par que específicos:** En este caso como podemos observar en la notación de los pines, estos poseen dos funciones (ejemplo de pin 8):
    a. **PA14:** Pin genérico 14 de entrada y salida
    b. **SWCLK:** Actúa como reloj del microcontrolador
3. **Pines de tierra:** Es fundamental recalcar que en la nomenclatura de Kicad 6.0 las tierras tanto las alimentaciones son entradas de potencia.
4. **RESET:** Definidos como bidireccionales, pero bien es cierto que generalmente es considerado de entrada.

Conforme a lo tratado previamente, el primer diseño del RAK 3172 en Kicad posee el siguiente aspecto:

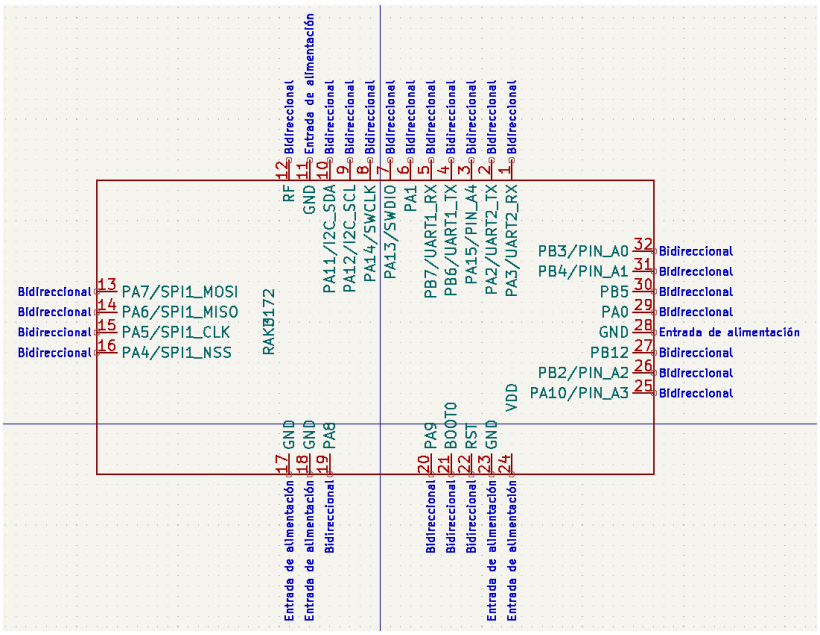


Ilustración 42: Diseño y definición de pines del rak3172 en Kicad 6.0

Como se ha informado previamente, en el interior de este RAK3172 existe un STM32WLE5CCU6 con el consiguiente bloque de diagramas de representación:

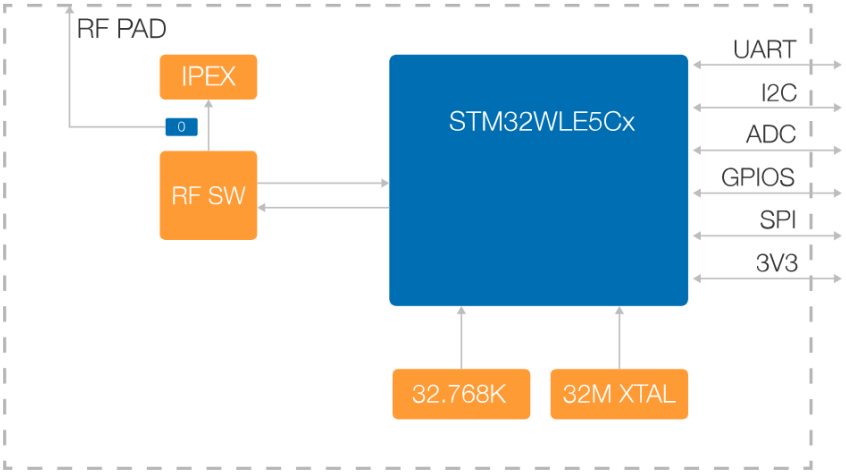


Ilustración 43: Bloque de diagramas del RAK3172 y STM32WLE5CCU6

A continuación, se reconocen las conexiones mínimas que requiere este RAK 3172 para funcionar, estas mismas serán las de RESET, VDD y GND (puertos 22,24 y 23 respectivamente).

De estas tres conexiones es fundamental tras estudiar el datasheet que el RESET debe de estar conectado a nuestra fuente de alimentación a través de una resistencia de 10 k y con un condensador de 100 nF. Así pues, en Kicad se mostrará así tras el diseño:

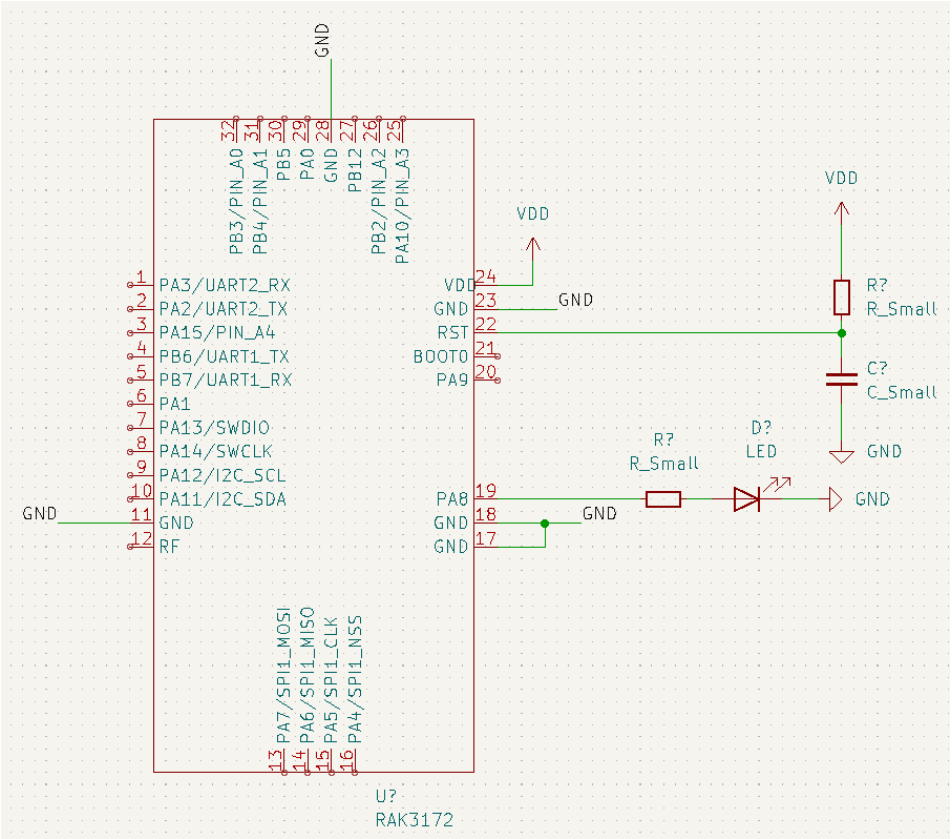


Ilustración 44: RESET del prototipo diseñado

En la ilustración anterior se puede observar cómo las resistencias irán colocadas con sus respectivas uniones a tierra y a la fuente de alimentación, sin embargo, falta determinar los valores de estas resistencias y condensadores. Para los componentes del RESET no hacen falta cálculos, ya que en la web de RAK puede observarse el esquemático de una pequeña placa para pruebas diseñada por la marca. No obstante, para el PA8 dispuesto de un LED nos bastará con una sencilla Ley de OHM teniendo en cuenta el datasheet del STM32WLE5Cx:

Table 22. Voltage characteristics(1)

Symbol	Ratings	Min	Max	Unit
$V_{DDX} - V_{SS}$	External main supply voltage (including V_{DD}, V_{DDA}, V_{DDRF}, V_{DDSMPS}, V_{BAT}, V_{REF+})	–0.3	3.9	V
V_{IN}(2)	Input voltage on FT_xx pins	V_{SS} - 0.3	min (V_{DD}, V_{DDA}, V_{DDRF}, V_{DDSMPS}) + 3.9(3)(4)	
	Input voltage on TT pins		3.9	
	Input voltage on any other pin		3.9	
$\lvert\Delta V_{DDx}\rvert$	Variations between different V_{DDX} power pins of the same domain	-	50	mV
$\lvert V_{SSx}-V_{SS}\rvert$	Variations between all the different ground pins(5)	-	50	
$V_{REF+} - V_{DDA}$	Allowed voltage difference for $V_{REF+} > V_{DDA}$	-	0.4	V

1. All main power (VDD, VDDRF, VDDA, VBAT) and ground (VSS) pins must always be connected to the external power supply, in the permitted range.

Tabla 13: Características de voltaje del RAK3172

Table 23. Current characteristics

Symbol	Ratings	Max	Unit
ΣIV_{DD}	Total current into sum of all V_{DD} power lines (source)(1)	130	mA
ΣIV_{SS}	Total current out of sum of all V_{SS} ground lines (sink)(1)	130	
$IV_{DD(PIN)}$	Maximum current into each VDD power pin (source)(1)	130	
$IV_{SS(PIN)}$	Maximum current out of each VSS ground pin (sink)(1)	100	
$I_{IO(PIN)}$	Output current sunk by any I/O and control pin, except FT_f	20	
	Output current sunk by any FT_f pin	20	
	Output current sourced by any I/O and control pin	20	
$\Sigma I_{IO(PIN)}$	Total output current sunk by sum of all I/Os and control pins(2)	100	
	Total output current sourced by sum of all I/Os and control pins(2)	100	
$I_{INJ(PIN)}$(3)	Injected current on FT_xx, TT and RST pins, except PB0	–5 / +0(4)	
	Injected current on PB0	-5/0	
$\Sigma \lvert I_{INJ(PIN)} \rvert$	Total injected current (sum of all I/Os and control pins)(5)	25	

Tabla 14: Características de corriente del RAK3172

Con lo dispuesto se realiza el siguiente cálculo:

$$A\ 20\ mA\ y\ 1.8V\ de\ Led\ la\ caída\ de\ tensión\ debe\ de\ ser \rightarrow 3 - 1.8 = 1.2\ V$$

Ecuación 4: Cálculo de caída de tensión

$$V = I \cdot R \rightarrow R = \frac{1.2\ V}{0.02\ A} = 60\ \Omega \approx 68\ \Omega \approx 100\ \Omega\ (por\ comodidad)$$

Ecuación 5: Cálculo de resistencia según caída de tensión

De esta forma la resistencia empleada por facilidad de adquisición en el mercado será de 100 Ohmios. El esquemático del dispositivo quedará según lo dispuesto en la ilustración Ilustración 45: Resistencia empleada en el cálculo del RESET:

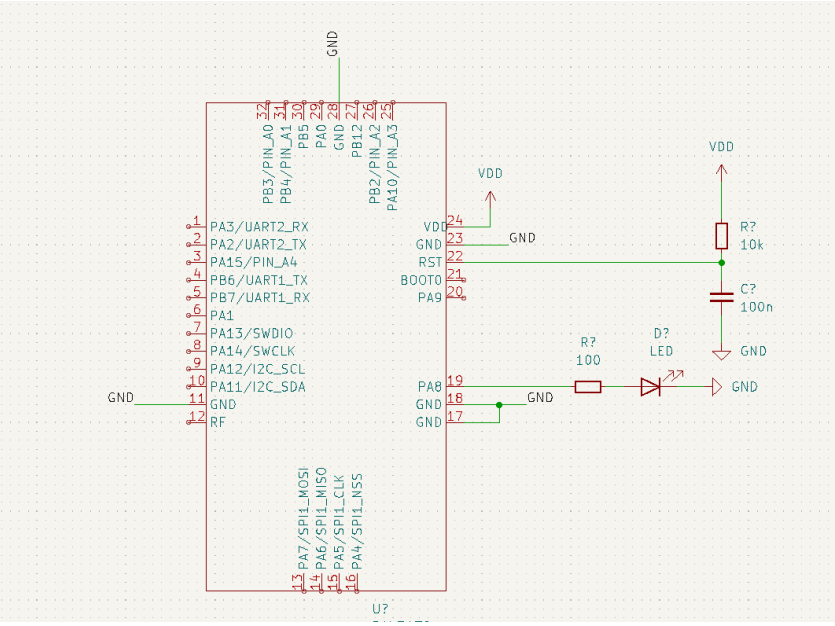


Ilustración 45: Resistencia empleada en el cálculo del RESET

Una vez se ha realizado el cálculo de la resistencia del Reset se incorpora la antena. Además, se incorpora un elemento fundamental conocido como puerto de alimentación, que debe incluir los pines de programación, de tierra de RESET y de alimentación valga la redundancia.

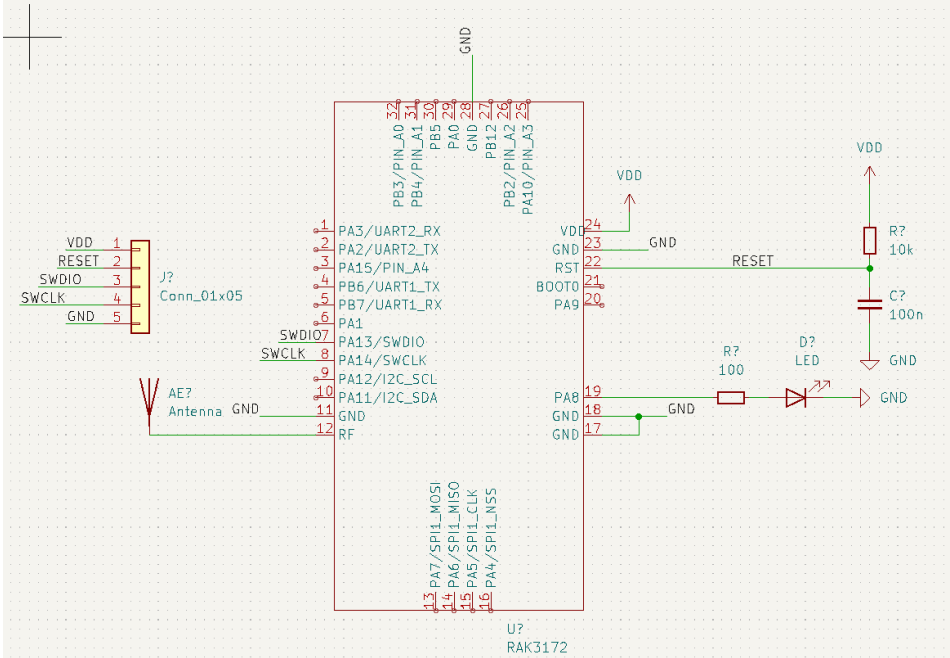


Ilustración 46: Esquemático con el puerto de alimentación incorporado

Para alimentar el microcontrolador se emplean baterías siguiendo los requisitos del trabajo. Se emplearán baterías alcalinas de 1,5 V. Bien es cierto que se podrían emplear reguladores como los RDOUT los cuales son de caída y tensión y disipan el exceso de tensión en forma de calor. O bien los BUSS CONVERTER que presentan el problema de no poder aportar demasiada corriente, y, por lo tanto, se tendrían que incorporar dispositivos extra más costosos y que por lo tanto hiciesen nuestro dispositivo menos óptimo.

La alimentación poseerá las propias baterías ya mencionadas, un interruptor de 2 polos y, por supuesto, unos condensadores de desacoplo para poder evitar tanto los picos de tensión como el ruido de la señal de corriente. Aunque podrían calcularse los valores de los condensadores, he optado por poner condensadores de 100 nF y 1 uF para que reaccionen rápido ante las variaciones de tensión y tengamos una vida de dispositivo mayor.

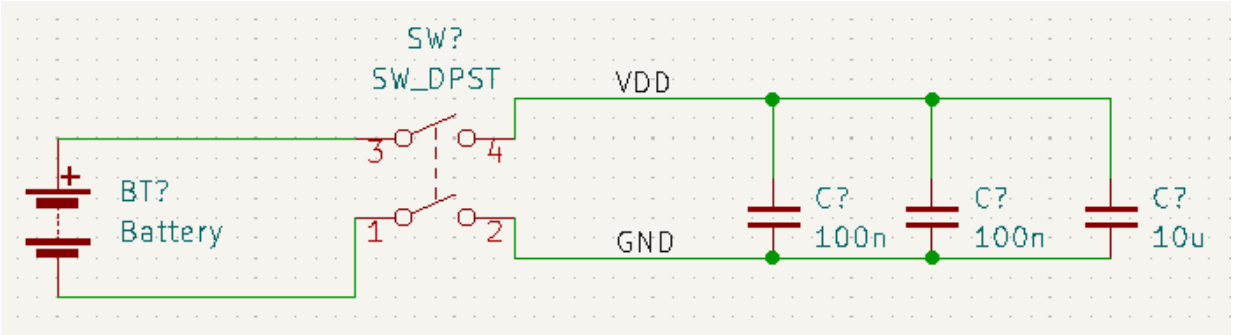


Ilustración 47: Fuente de alimentación del dispositivo para el esquemático

A continuación, se insertará un botón para el testeo del sistema o UPLINK. El botón es un sensor digital ya que sólo presenta dos estados: Pulsado o no pulsado. Si le pulsamos se leerá

el valor 0 y al no tenerlo pulsado para que no quede al “aire” necesitamos que tenga un valor definido. De esta forma se van a emplear unas resistencias llamadas de Pull-up o de Pull-down que son de 10 k. En nuestro caso, cada pin del RAK 3172 trae Pull-up y Pull-down que programables por lo que para el Pin 20 (PA9) no hace falta poner la resistencia de 10k mencionada.

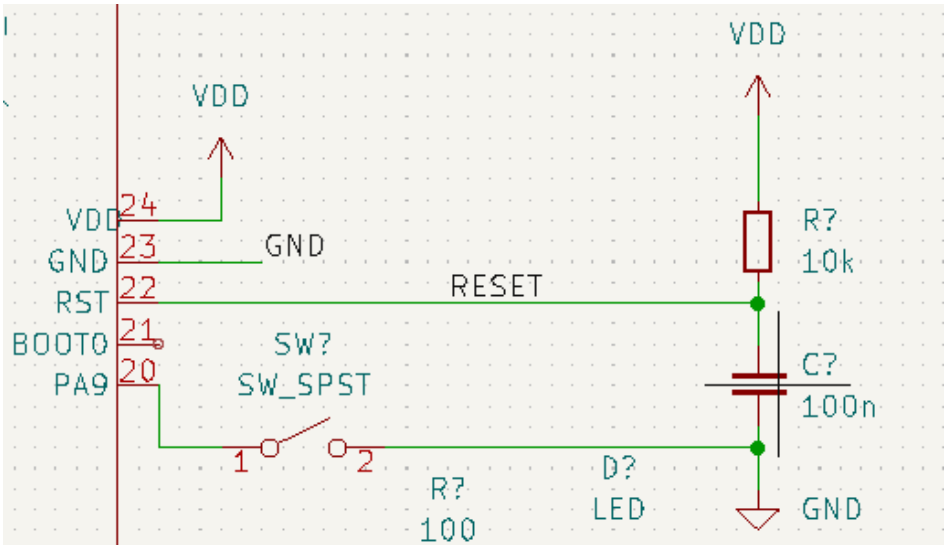


Ilustración 48: Parte del esquemático referente al botón de testeo del sistema

Una vez diseñados todos estos parámetros nos encargaremos de realizar las entradas analógicas. Estas entradas son las que emplearemos para los sensores que queremos implementar en nuestro sistema. Los microcontroladores utilizan un periférico llamado ADC (analog to digital converter), que les permite convertir una señal analógica en un valor numérico.

Es fundamental tener en cuenta que:

- Los extremos de tensión serán los de GND y VDD del sistema.
- Algunos Microcontroladores permiten establecer como extremos de tensión unos diferentes a los de VCC.

En el caso del STM32WLE5Cx tendremos una resolución del ADC de 12 bits de resolución, extensibles a 16 bits; por lo que podremos distinguir entre 4096 o 65536 valores de tensión.

5.2.1. Sensor de nivel de luz

Un sensor de luz LDR tiene un funcionamiento muy simple, su valor de resistencia disminuye según le incide más luz. Así pues, para incorporarlo a nuestro RAK3172 debemos de contar con un divisor de tensión que ya estudiamos a lo largo de la carrera.

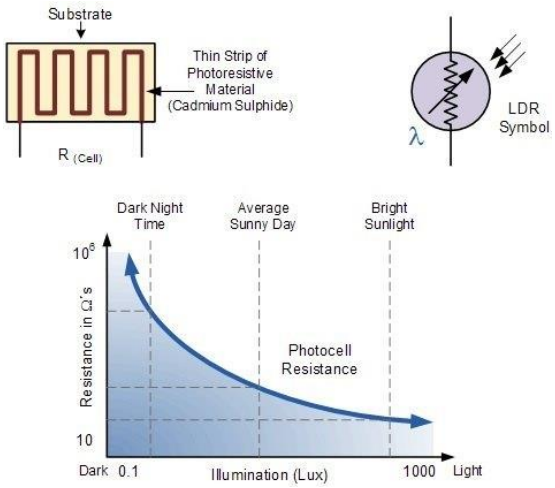


Ilustración 49: Gráfica explicativa de sensor LDR

Así pues, se procede a elegir el valor de resistencia adecuado para el sensor LDR que permita que la información captada sea lo más precisa posible. Básicamente nuestro esquema sería así:

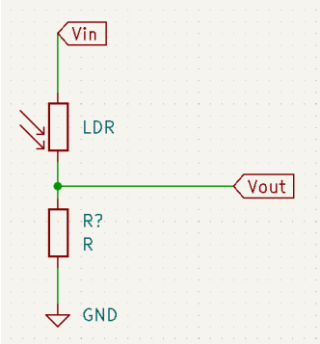


Ilustración 50: Ejemplificación en Kicad 6.0 para el cálculo del LDR

Como se desea que el sensor sea lo más preciso posible, se va a calcular la diferencia de potencial entre Vout y Vin tanto con la mayor resistencia posible ofrecida por LDR como por la menor resistencia posible ofrecida por esta.

$$V_{out} = V_{in} \cdot \frac{R}{R + R_{LDR}}$$

Ecuación 6: Fórmula de divisor de tensión

Así pues, luego se podrá crear un intervalo de valores lo más preciso posible. No obstante, emplearemos la fórmula de Axel-Benz la cual se basa en:

$$R_{ref} = \sqrt{R_{min} \cdot R_{máx}}$$

Ecuación 7: Fórmula de Axel-Benz

Electrical Characteristics (T_A=25°C, source at 2854 K)

Silonex Part No	Sensitive Material	Spectral Peak (nm)	R @ 1 ftc (Kohms) (3)	R @ 2 ftc (Kohms) (3)	R_{TYP} @ 100 ftc (ohms)	Min. R_{DARK} (Kohms) (2)	V_{MAX}(Volts) (peak AC or DC)
NSL-4912			9.3		150	560	80
NSL-4922			4.0		80	250	80
NSL-4932			1.7		50	100	80
NSL-4942	Type 4	515	0.725		30	50	80
NSL-4952	CdS		40		860	2400	320
NSL-4962			17		500	1000	320
NSL-4972			7.25		260	400	320
NSL-4982			114		2500	6800	320

$$R_{ref} = \sqrt{1000000 \cdot 500} = 22360{,}680\ \Omega \approx 22k\ \Omega$$

Ecuación 8: Cálculo de resistencia de referencia para LDR

De esta forma si el V_{in} es de 3V tendremos para el máximo de lúmenes y para el mínimo:

Mínimo de lúmenes	Máximo de lúmenes
$V_{out} = 3 \cdot \frac{500}{500 + 22000} = 0{,}06\ V$	$V_{out} = 3 \cdot \frac{1000000}{1000000 + 22000} = 2{,}94\ V$
Su rango dinámico será de 2,88 V	

Tabla 15: Máximo y mínimo de lúmenes LDR

Estos valores para 16 bits serán de:

Mínimo de lúmenes	Máximo de lúmenes
$\frac{0{,}06\ V \cdot 65535}{3\ V} = 1310{,}7\ \approx 1311$	$\frac{2{,}94\ V\ \cdot 65535}{3\ V} = 64224{,}3\ \approx 64224$

Tabla 16: Valores de lúmenes en 16 bits

Finalmente, en KICAD 6.0 quedará así conectado al puerto 26:

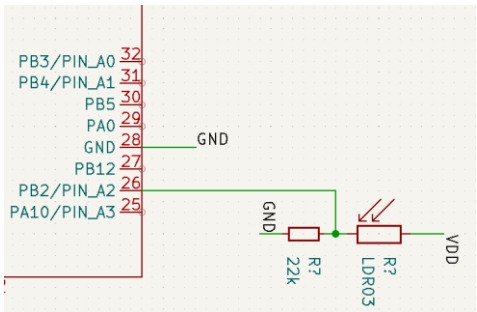


Ilustración 51: Puerto PB2 del esquemático

5.2.2. Sensor de temperatura y humedad

Antes de mencionar este sensor, es interesante tener en cuenta los protocolos serie más comunes en el mundo de las IoT:

a) **USART:** Universal Synchronous / Asynchronous Serial Receiver, bastante lenta

b) **I2C:** Inter Integrated Circuit, la que emplearemos y más rápida que la anterior.

c) **SPI:** Serial Perypheal Interface es la más rápida.

Aunque bien es cierto que los pines de los microcontroladores vienen preparados para estas configuraciones, debemos de conocer los dos métodos de funcionamiento principales de estos en función de las salidas:

- **Push-pull:** La salida se puede poner a 0 o a 1
- **Open Drain:** La salida puede ponerse a alta impedancia o en 0 pero requieren siempre de una resistencia pull-up.

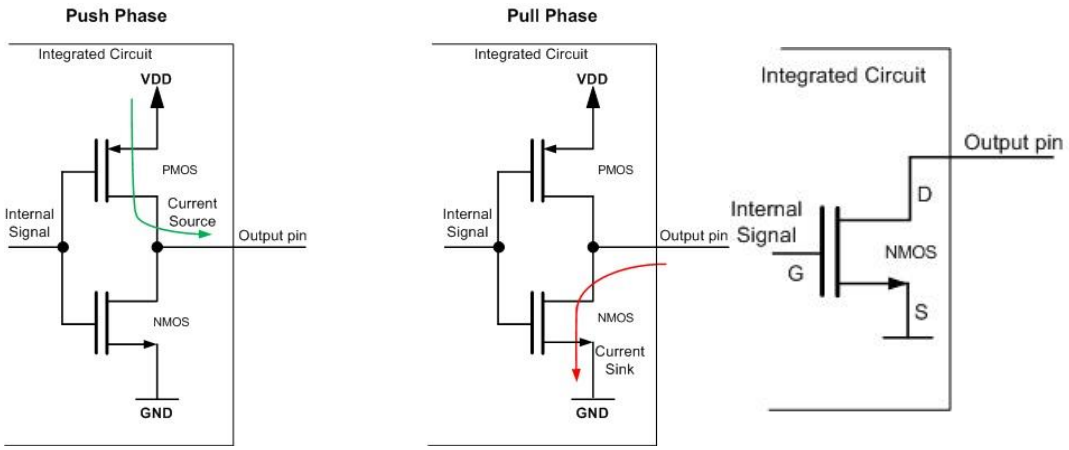


Ilustración 52: Diagrama de funcionamiento de microcontroladores

Como se observa, la ventaja del open drain es que, si necesitamos gestionar con el microcontrolador una tensión más alta de la que puede ofrecer este mismo, podríamos emplear este colector abierto. Por otro lado, si al output pin están conectados varios sensores, todos podrían estar en alta hasta que uno de ellos quiera el solo bajarse y dejar al resto en alta.

Por último, se mencionan las características y definiciones de la conexión **I2C**:

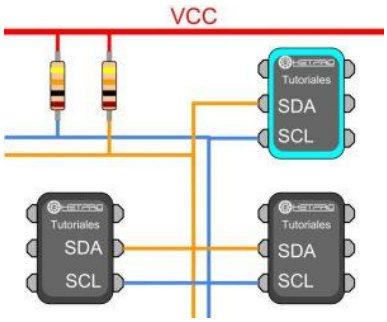


Ilustración 53: Esquema de conexión entre dispositivos

Aquí se muestra una conexión típica de tres dispositivos, el bus consiste en dos líneas llamadas, Serial DAta – SDA y Serial CLock – SCL. Es decir, Datos Seriales y Reloj Serial. En particular al bus se le conectan dos resistencias en arreglo pull-up, de entre 2.2K y 10K.

El MAESTRO I2C se encarga de controlar al cable de reloj, por sus siglas en inglés llamada SCL – Serial CLock. Además, el MAESTRO se encarga de iniciar y parar la comunicación. La información binaria serial se envía sólo por la línea o cable de datos seriales, en inglés se llama SDA – Serial Data. Dos Maestros no pueden hacer uso de un mismo puerto I2C. Puede funcionar de dos maneras, como maestro-transmisor o maestro-receptor. Sus funciones principales son:

- Iniciar la comunicación – S
- Enviar 7 bits de dirección – ADDR
- Generar 1 bit de Lectura ó Escritura – R/W
- Enviar 8 bits de dirección de memoria
- Transmitir 8 bits de datos –
- Confirmar la recepción de datos – ACK – ACKnowledged
- Generar confirmación de No-recepción, NACK – No-ACKnowledged
- Finalizar la comunicación

El ESCLAVO I2C, generalmente suele ser un sensor. Este elemento suministra de la información de interés al MAESTRO. Puede actuar de dos formas: esclavo-transmisor ó esclavo-receptor. Un dispositivo I2C esclavo, no puede generar a la señal SCL. Sus funciones principales son:

- Enviar información en paquetes de 8 bits.
- Enviar confirmaciones de recepción, llamadas ACK

Así pues, el sensor empleado será el SI7021 de 4 pines:

Fotografía 29: Sensor Si7021

Este sensor tiene dos resistencias en paralelo, por lo que las resistencias de pull up vienen incluidas en el propio sensor y es por eso por lo que en Kicad 6.0 quedará simplemente conectado a cada entrada de datos SDA y SCL.

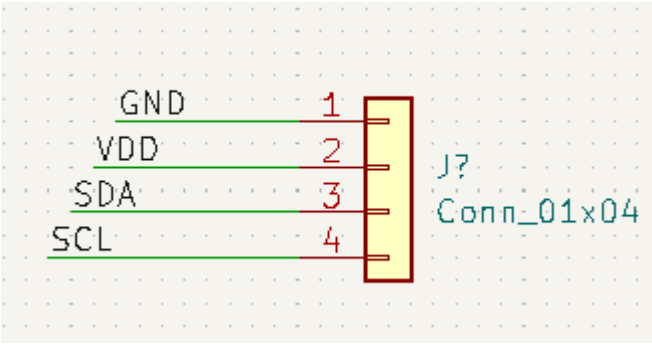


Ilustración 54: Esquemático del SI7021

5.2.3. Tarjeta de memoria SD

El último tipo de protocolo de comunicación por estudiar y el cual es fundamental para implementar una tarjeta SD a nuestra PCB es el SPI.

Las siglas provienen de Serial Peripherical Interface (SPI) el cual es bastante más rápido que el I2C y es empleado para frecuencias de unos pocos MHz. EL SPI tiene 3 pines compartidos:

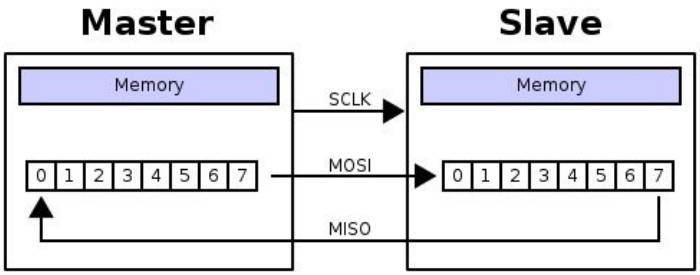


Ilustración 55: Funcionamiento del SPI

Cada uno de estos pines hará referencia a:

a) SCLK: Empleado para la señal del reloj

b) MOSI: Master Output Slave Input

c) MISO: Master input Slave Output

Además de estos 3 pines empleados, debemos de añadir uno adicional para la elección del dispositivo que quiero incorporar. O bien si tengo 2 sensores a elegir, pues dispondré de un total de 5 pines. El pin de selección es:

- SS: Slave Select activados por lógica inversa por lo que se activan al estar en 0.

Antes de incorporarlo a nuestro diseño debemos de recordar que las SD pueden funcionar con el modo SD (más rápido y más pines) y el modo SPI (lento y menos pines). Además, hay que tener mucho cuidado con el voltaje que recibe nuestra SD ya que soporta como máximo 3,3 V.

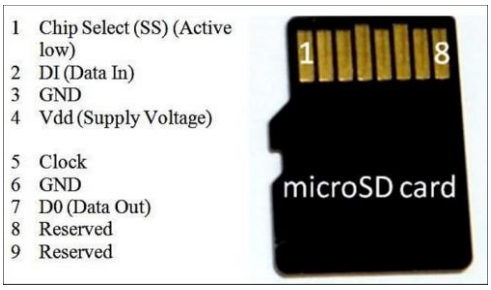


Ilustración 56: Definición de pines de una tarjeta SD

Usaremos los pines 9 y 10 dado que, aunque el RAK 3172 solo dispone de dos puertos I2C, podremos situar todos los sensores hasta el momento en esos pines mencionados para así dejar libre los pines 13 y 31.

Ilustración 57: Tarjeta SD de Sparkfun electronics

En Kicad 6.0 se verá de la siguiente forma:

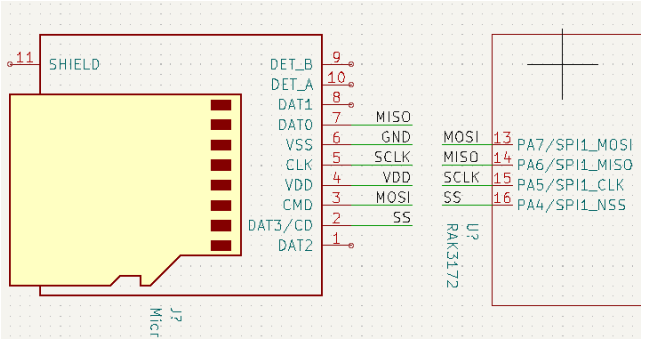


Ilustración 58: Vista en esquemático de la tarjeta SD

Como se podrá observar en el prototipo final, esta tarjeta SD fue excluida del diseño debido en primer lugar a la dificultad de incorporación de la misma, y en segundo lugar, debido a la complejidad de la misma para la consecuente programación por parte del usuario. No obstante, se ha considerado oportuno introducirlo en este proyecto pese a su no incorporación dado que puede ser usado en una implementación futura del prototipo mejorado.

5.2.4. Modelado de la PCB en Kicad 6.0

Tras la finalización del esquemático en Kicad 6.0 se procede el modelado de esta. Este software permite (además de realizar complejos esquemáticos) diseñar la PCB. Cabe mencionar que entre el modelado de la PCB y la realización del esquemático se han diseñado las huellas relevantes de los componentes empleados para su consecuente visualización en el visor 3D.

En primera instancia, como se puede observar en la ilustración Ilustración 59: Visor de disposición de huellas en Kicad 6.0 los componentes se encuentran obviamente desordenados y requieren de la interfaz humana para su recolocación:

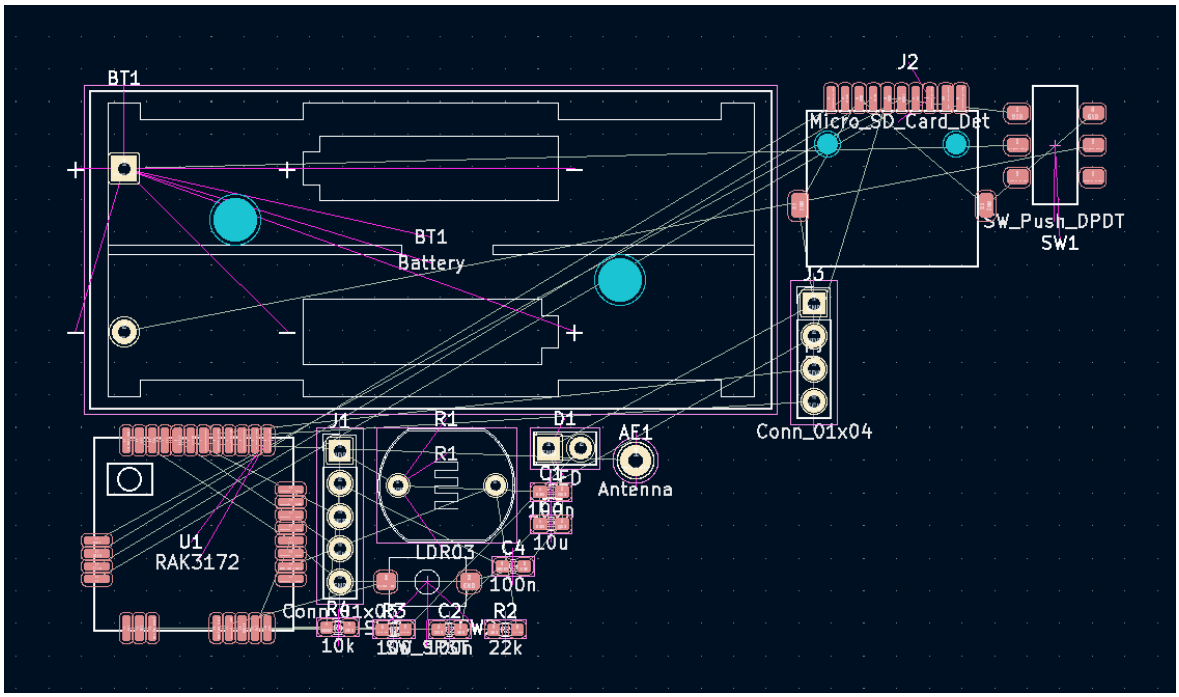


Ilustración 59: Visor de disposición de huellas en Kicad 6.0

5.2.5. Diseño de la antena y su ruteado

Antes de modificar este diseño mencionado. El usuario ha de percatarse que la hora de diseñar la PCB hay dos factores que influyen negativamente en el tamaño de la misma. Obviamente se desea que la placa sea lo más pequeña posible para así escatimar costes y que el dispositivo consuma menos energía y aumente su duración. Así pues, debemos de prestar especial atención tanto a la antena como a la batería del dispositivo.

La batería podemos modificarla poco, esto es debido a que la elección de esta está predefinida de antes y es tan simple que no merece la pena estudiar demasiado su modificación.

Por otra parte, tenemos la antena. Los tipos de antena pueden variar, generalmente tenemos antenas dipolo o monopolo de ¼ de onda.

Ilustración 60: Antenas dipolo y monopolo de 1/4 de onda

Como podemos observar la antena monopolo ¼ es bastante más pequeña que el dipolo, esto es debido a que eliminamos uno de los polos de esta aparentemente. No obstante, lo que realmente ocurre es que necesitamos en nuestra PCB un plano de masa que juegue de polo ausente de la misma. El plano de masa es del orden de ¼ de onda, si colocásemos la antena en línea con el circuito impreso reduciríamos el tamaño final de la PCB.

Si por ejemplo queremos 868MHz la antena es de unos 86mm aproximadamente.

Debemos de recordar que en ocasiones es imposible situar antenas de este tipo en los dispositivos electrónicos debido al requerimiento de espacio, por lo que se emplearán unas antenas flexibles imprimibles (FPC-Flexible Printed Circuit) que actúan como dipolos y no se ven afectadas por el tamaño del plano de masa. Además, estas antenas necesitan de filtros tipo pi compuestos por dos condensadores y un inductor.

Ilustración 61: Antena FPC

Tras todos estos conceptos adquiridos, recolocamos la PCB y nos queda de la siguiente forma:

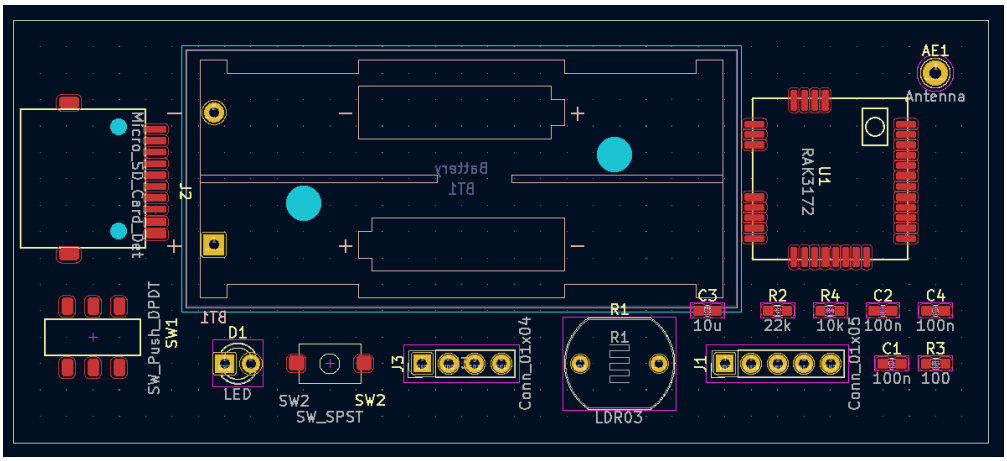


Ilustración 62: Modelado de PCB correctamente dispuesto

Kicad 6.0 ha demostrado ser una poderosa herramienta, hecho que se refuta en cuanto a la comodidad de su herramienta de visualizado 3D. Esta herramienta permite visualizar de forma muy precisa tanto la apariencia de ciertos componentes como las huellas y pistas de ruteado de las mismas. Se ha de recalcar que por comodidad hay ciertos componentes que no se han diseñado ya que se ha creído consecuente que con el esquemático y la huella era suficiente para mandar la pista a imprimir.

La visualización 3D frontal de nuestro primer prototipo será la mostrada en la ilustración Ilustración 63: Parte frontal del primer prototipo :

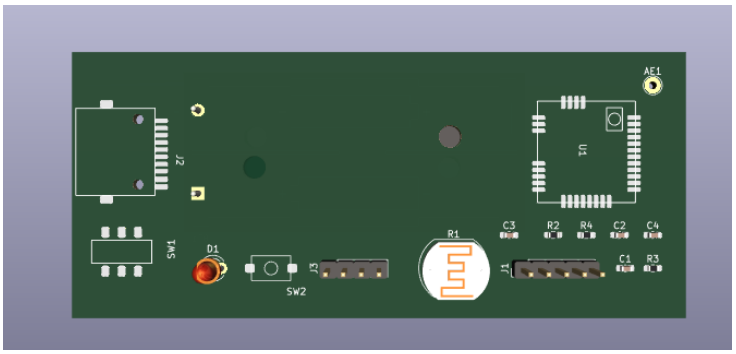


Ilustración 63: Parte frontal del primer prototipo

Además, la parte trasera se vislumbra en la ilustración Ilustración 64: Parte trasera del primer prototipo:

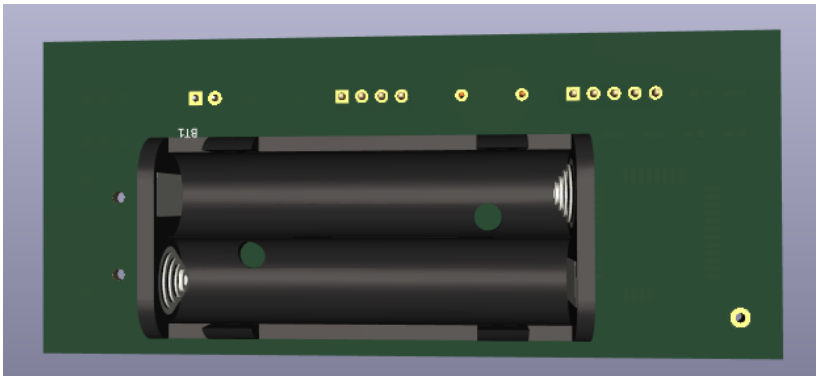


Ilustración 64: Parte trasera del primer prototipo

5.2.6. Autoruteado de pistas aéreas

A continuación, se procede a realizar la modificación de las líneas aéreas de la placa. Para ello se ejecuta el programa autoroute que realizará automáticamente dichas conexiones.

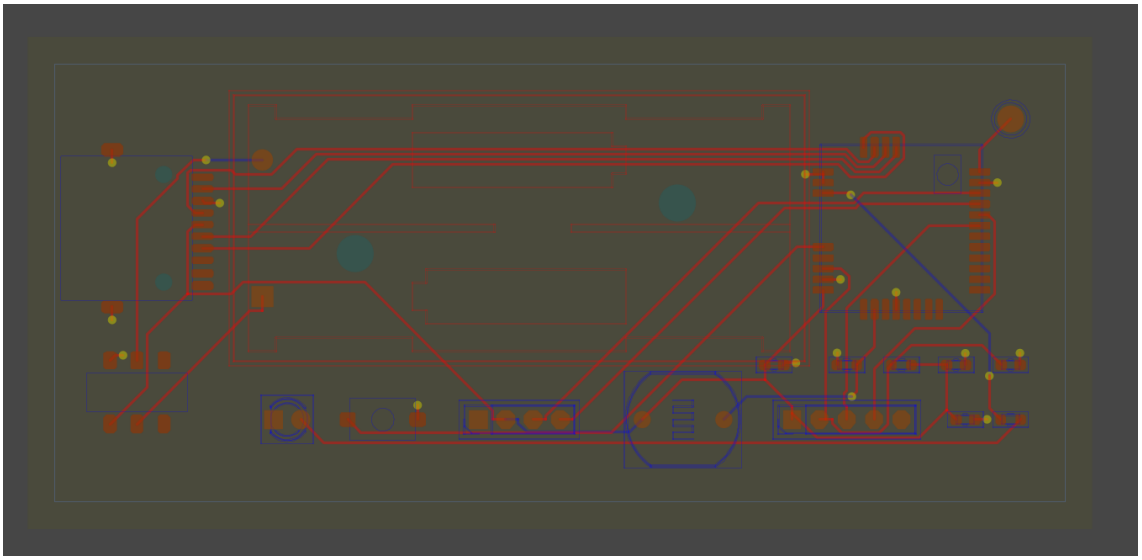


Ilustración 65: Autoruteado con programa "Autoroute"

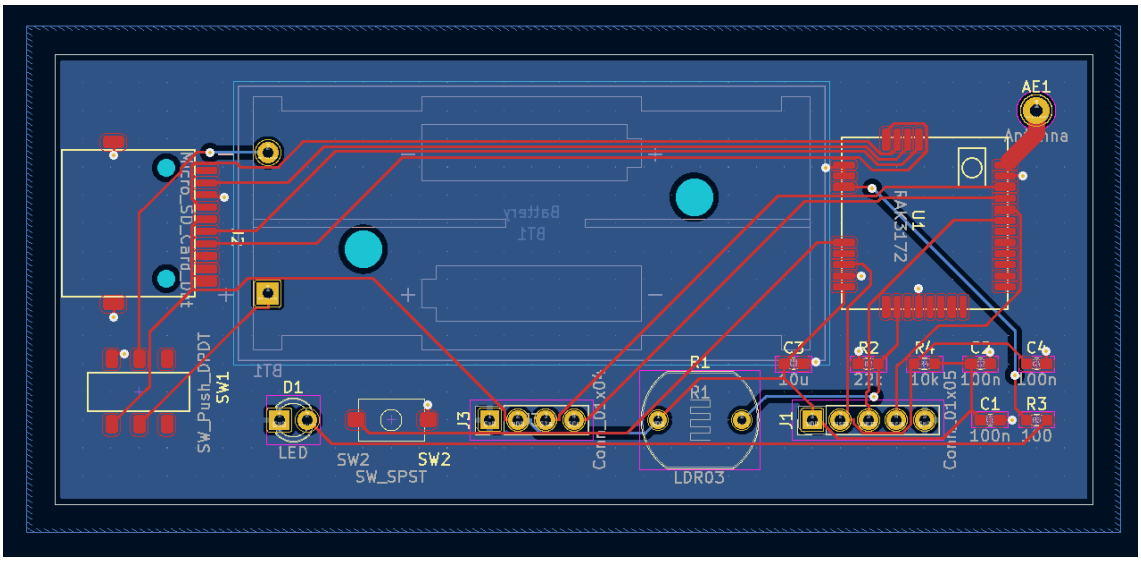


Ilustración 66: Pistas aéreas visualizadas e importadas a Kicad 6.0

5.2.7. Creación de Gerbers

Del programa Autoroute se procede a exportarlo como archivo Specctra. A continuación, se importa en Kicad 6.0 y se corrigen errores gracias al detector automático de errores del propio Kicad 6.0. Dado que en el primer caso sólo se detectan problemas de serigrafia, serán obviados ya que el fabricante de la PCB al recibir el archivo gerber simplemente recortará estas serigrafías.

Los archivos Gerbers que se generarán automáticamente contendrán los 6 apartados fundamentales para la correcta interpretación del modelo en fábrica. Este proceso será redactado posteriormente, la empresa encargada del mismo será JLCPCB.

Cada archivo Gerber será visualizado con la herramienta GerberViewer y se corresponden en nuestra placa con:

1- **Capa de contorno:**

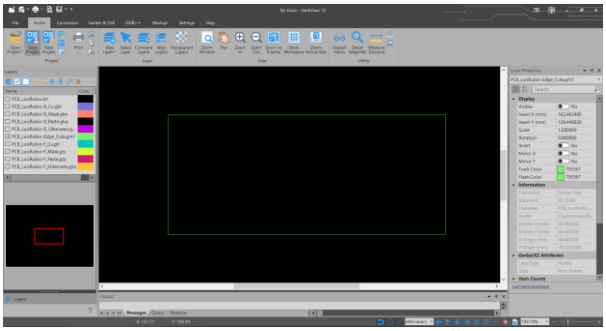


Ilustración 67: Capa de contorno PCB

2- **Top Layer de Cobre:**

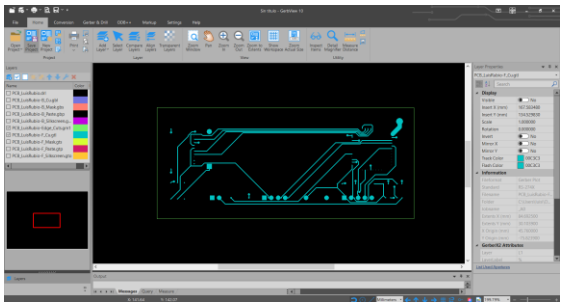


Ilustración 68: Top Layer de Cobre PCB

3- Top Layer de máscara anti-soldadura:

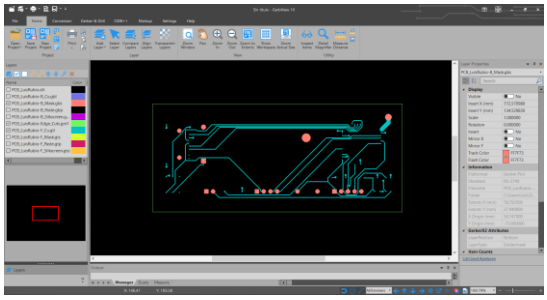


Ilustración 69: Top layer máscara anti-soldadura PCB

4- Lugares de situación de la pasta y serigrafías:

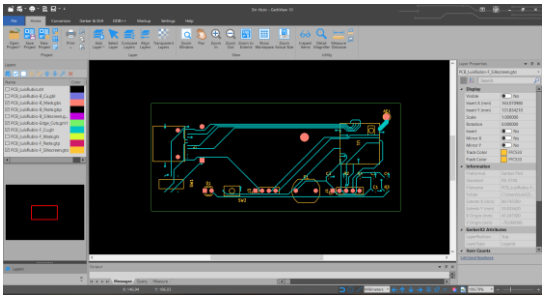


Ilustración 70: Lugares de situación de la pasta y serigrafías PCB

5- Taladros

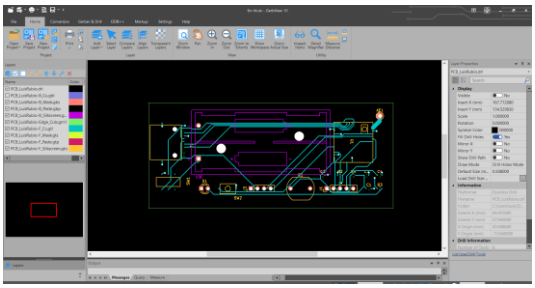


Ilustración 71: Taladros de la PCB

6- Cobre

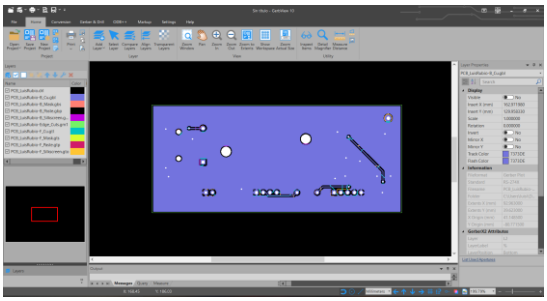


Ilustración 72: Capa de Cobre de la PCB

5.2.8. Resultado del primer prototipo

En el presente apartado no se llega a presentar una realización completa del prototipo debido a que antes de enviarlo a nuestro fabricante de placas (JLCPCB) ha de recibir la confirmación de diseño por parte del tutor del TFG.

Tras lo dispuesto previamente, el tutor Diego Gachet realizará un examen exhaustivo del prototipo para saber si es apto para fabricación. Lógicamente esto permitirá un ahorro de costes ya que producir algo que no funcione puede desencadenar un retraso en el prototipo final deseado.

Tras el análisis realizado por el tutor se confirma que el prototipo es completamente viable, no obstante, recomienda unas modificaciones básicas. Estas son:

a) Incorporación de más huellas para permitir un mayor reconocimiento visual de cada pin dispuesto en la PCB. Esto ahorrará un valioso tiempo de detección de puertos del prototipo a la hora de su consecuente manipulación.

b) Implementación de diversos pines extra para poder conectar el resto de los puertos no empleados del dispositivo. Este apartado es algo que discutimos el tutor y yo ya que no terminaba de verle utilidad, sin embargo, como dijo Diego Gachet el hecho de poseer más pines daba la facilidad de programación e implementación de más sensores en el futuro. Podríamos afirmar que el objetivo buscado sería plenamente didáctico.

5.2.9. Resultado del diseño del prototipo final

Dado que las modificaciones del dispositivo requeridas son muy leves, se realizan en un plazo de apenas cuatro días y permiten de nuevo ser enviadas al tutor para aprobarlas y poder enviarlas al fabricante.

El dispositivo que será enviado a JLCPCB es el dispuesto en la siguiente ilustración:

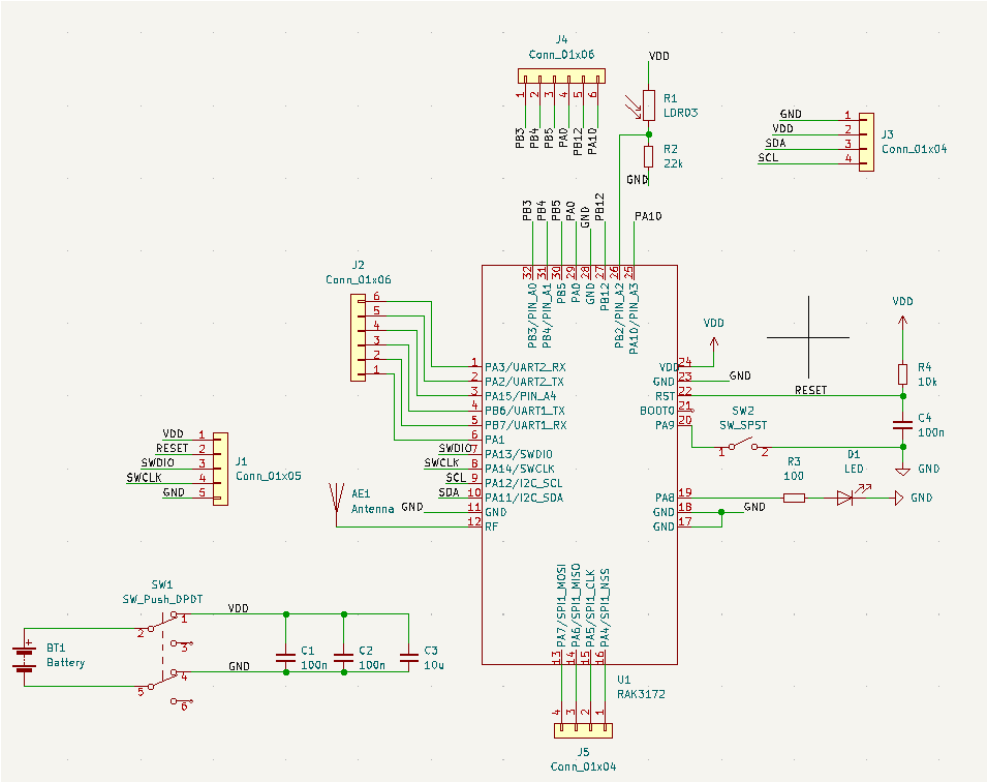


Ilustración 73: Esquemático del prototipo enviado a fabricación

Si se realiza una comparativa exhaustiva con respecto al modelo anterior nos percatamos en el esquemático de la implementación de los conectores J2, J4 y J5 que darán acceso a los puertos PA3,PA2,PA15,PB6,PB7,PB3,PB4,PB5,PA0,PB12,PA10,PA7,PA6,PA5 y PA4 respectivamente.

Para mayor claridad, estos puertos en el esquemático se ven conforme lo dispuesto en la ilustración Ilustración 74: Esquemáticos de puertos J2, J4 y J5:

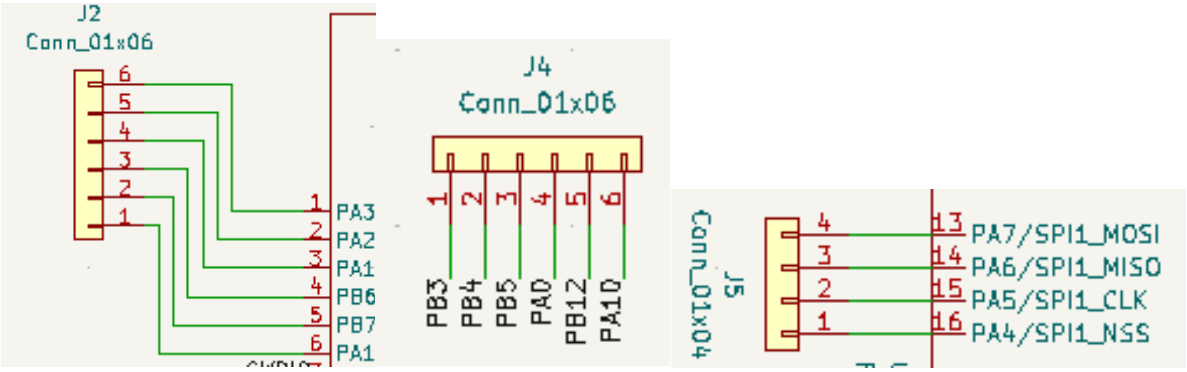


Ilustración 74: Esquemáticos de puertos J2, J4 y J5

Con la incorporación de pines descrita previamente se han de recolocar los componentes del dispositivo final y volver a auto rutearlos. Se considera que como es realizar los mismos pasos que previamente no se van a volver a explicar y por lo tanto es suficiente con incorporar las imágenes de las pistas aéreas y la consecuente visualización 3D del dispositivo.

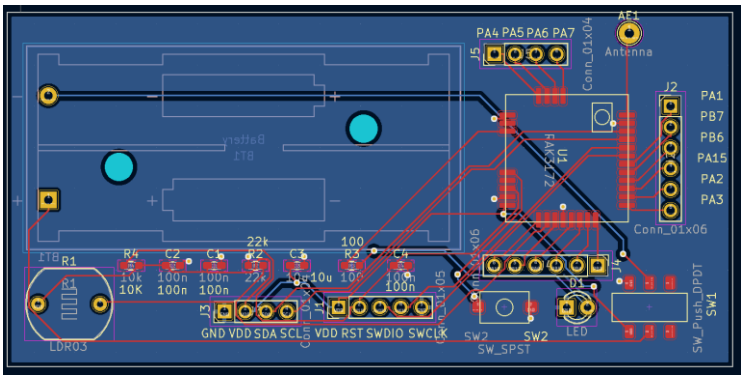


Ilustración 75: Pistas aéreas del dispositivo final

Se presenta a continuación una ilustración del Visor 3D que permita la concepción al usuario más cercana al dispositivo final que se producirá:

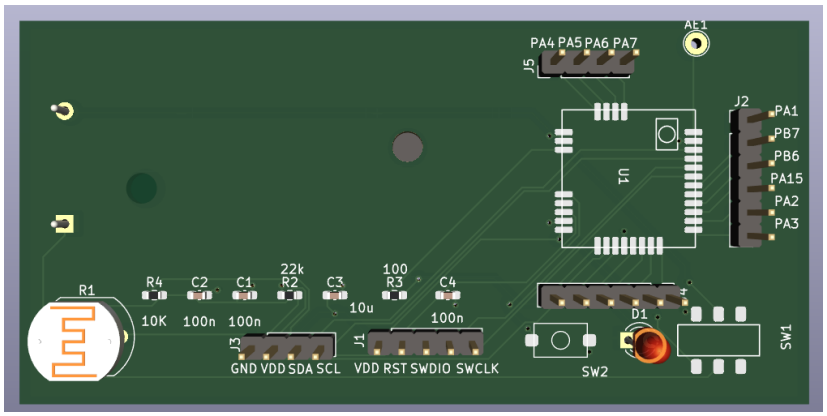


Ilustración 76: Visualización 3D delantera del dispositivo final

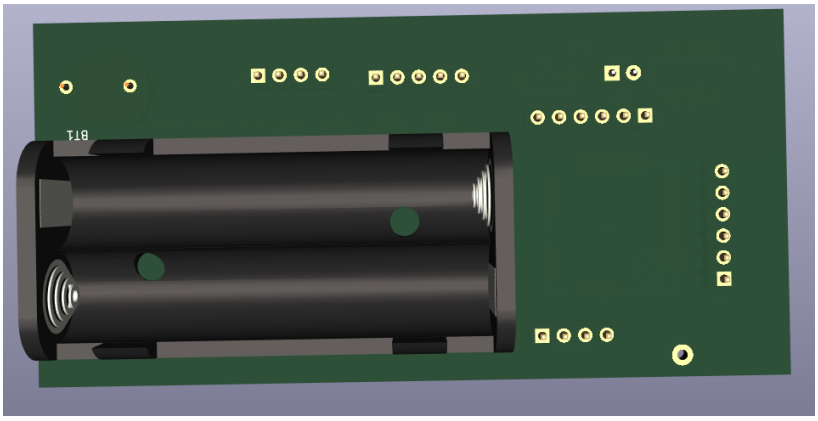


Ilustración 77: Visualización 3D trasera del dispositivo final

5.3. Pedido y recepción de componentes

Una vez el prototipo ha recibido la validación por parte del tutor se procede al pedido de los diversos componentes necesarios, esta fase del proyecto (pese a haber comprado mayoritariamente en sitios europeos y nacionales) es bastante lenta. No obstante, finalmente se reciben los componentes tras un par de semanas.

Para la fabricación de la PCB bastaba con enviar los diversos Gerbers generados al fabricante (JLCPCB en nuestro caso). Alguno de estos Gerbers (el resto se introducirán en el Anexo del trabajo) se disponen a continuación:

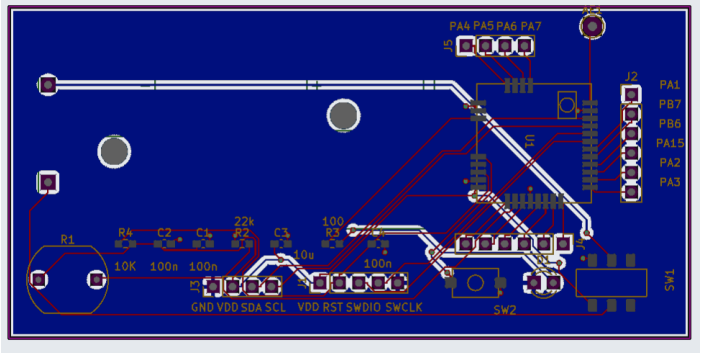


Ilustración 78: Gerber delantero del prototipo final

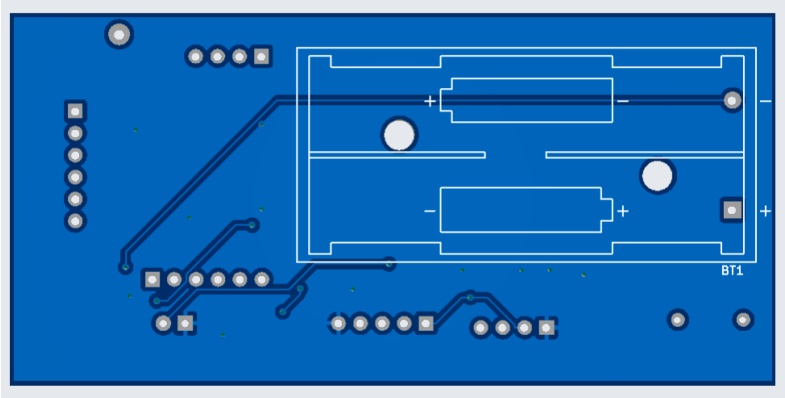


Ilustración 79: Gerber trasero del prototipo final

Además de los Gerbers el fabricante ofrece diversos capos de configuración a deseo del usuario. Alguna de estas opciones es muy curiosa como por ejemplo elección del color de la placa, incorporación automática de los componentes a través de su programa EASYEda (el cual no se ha empleado dado que sería más sencillo) e incluso la posibilidad de fabricar la placa para la soldadura SMD, llamada Stencil (opción escogida dado el ahorro de tiempo posible).

Las configuraciones empleadas en la página web son las siguientes:

Deburring/Edge rounding:	No	Outer Copper Weight:	1 oz
Gold Fingers:	No	Flying Probe Test:	Fully Test
Castellated Holes:	no	Remove Order Number:	No
4-Wire Kelvin Test:	No	Material Type:	FR4-Standard TG 135-140
Paper between PCBs:	No	Appearance Quality:	IPC Class 2 Standard
Confirm Production file:	No	Silkscreen Technology:	Ink-jet/Screen Printing Silkscreen
Package Box:	With JLCPCB logo		

Via Covering:	Tented	Surface Finish:	LeadFree HASL
Deburring/Edge rounding:	No	Outer Copper Weight:	1 oz
Gold Fingers:	No	Flying Probe Test:	Fully Test
Castellated Holes:	no	Remove Order Number:	No
4-Wire Kelvin Test:	No	Material Type:	FR4-Standard TG 135-140
Paper between PCBs:	No	Appearance Quality:	IPC Class 2 Standard
Confirm Production file:	No	Silkscreen Technology:	Ink-jet/Screen Printing Silkscreen
Package Box:	With JLCPCB logo		

Tabla 17: Configuración del pedido a JLCPCB

La mayoría de las configuraciones son muy sencillas, sin embargo, como se puede observar en el apartado dimensión se produjo una modificación automática tras el envío del pedido gracias al proveedor. El error fue de 0,02 mm en la parte ancha de la PCB motivado para permitir la correcta visualización de la huella de la antena “AE1”. JLCPCB fue escogido gracias a que ofrece este tipo de ayuda al usuario el cual se agradece si se es novato en el mundo de la electrónica. Así pues, el Stencil para soldadura SMD tiene las siguientes especificaciones:

Gerber file:	Gerbers_Y1	Build Time:	1-2 days
Framework:	No	Step Stencil:	No
Dimension:	380mm*280mm	Customer size:	100*60mm
Number:	1	Stencil Side:	Top
Electropolishing:	Sanding	Fiducials:	No Fiducial
Confirm Production file:	NO		

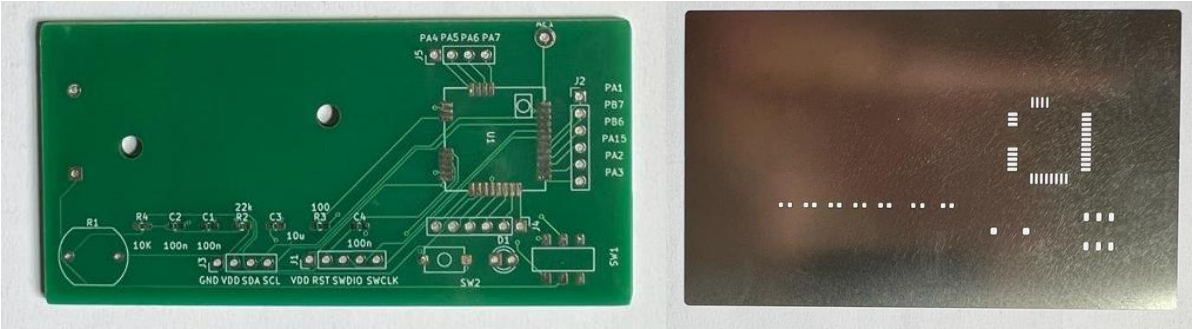
Ilustración 80: Especificaciones Stencil para soldadura SMD

Debido a que el proyecto es complejo, no se puede permitir el usuario estar sin hacer nada por lo que tras hablar en una tutoría con Diego Gachet, se determina que para ir familiarizándose con el programa y el lenguaje de programación lo óptimo es ir probando y aprendiendo a usar el software y hardware. Obviamente, no tenía ninguna placa por lo que afortunadamente Diego Gachet poseía una PCB ya soldada con un RAK3172 la cual me dejó para programar y aprender durante el intervalo de tiempo de recepción del dispositivo fabricado.

El stencil que permitirá el posterior soldado de los componentes de menor tamaño es el siguiente:

Ilustración 81: Visualización del Stencil para soldadura SMD

Finalmente, la PCB es recibida junto con el stencil, se adjunta fotografía del prototipo previo al soldado de este:



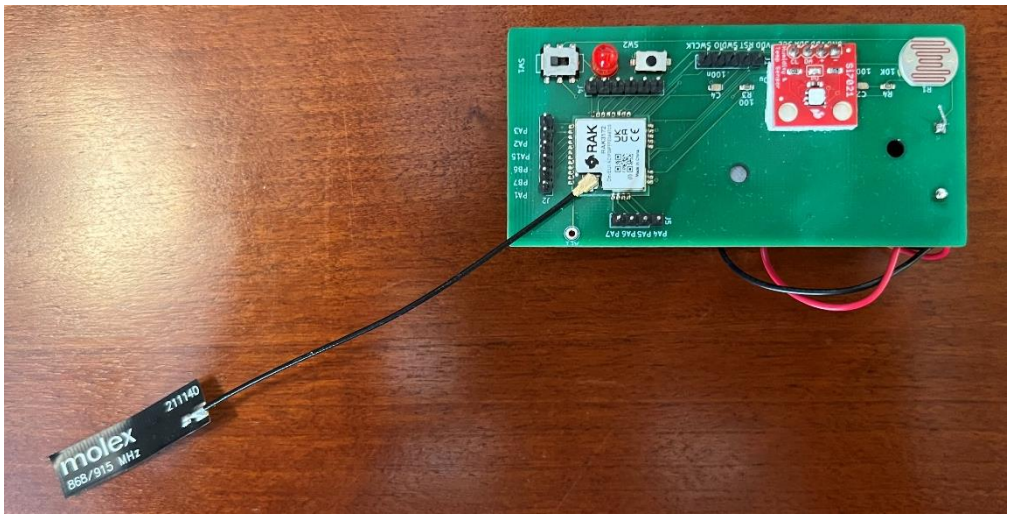
Fotografía 30: PCB + Stencil recibidos

5.4. MONTAJE Y SOLDADO DE LA PLACA

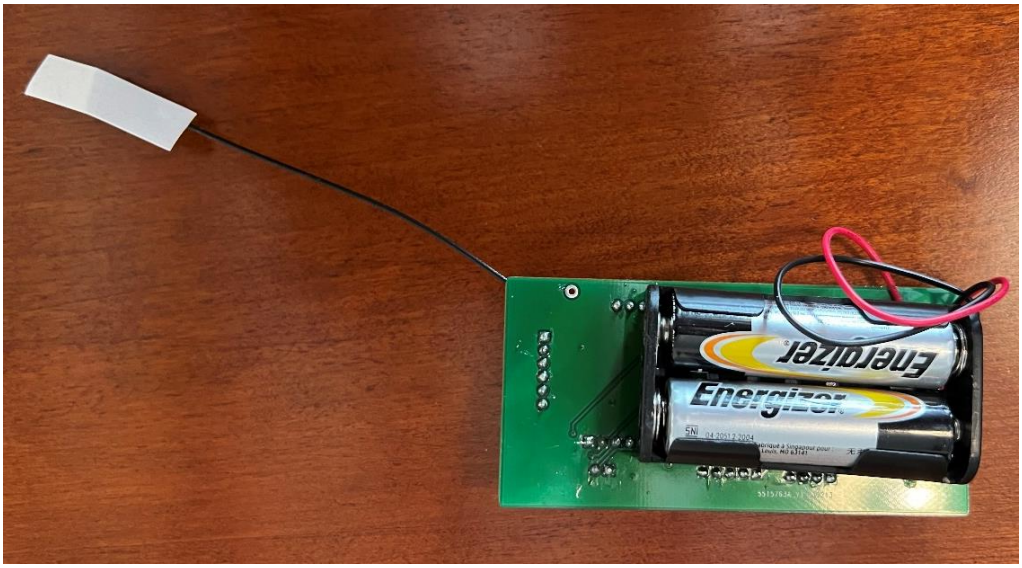
Una vez se han recibido todos los pedidos solo faltaría montar el dispositivo, aunque se describirá el proceso posteriormente, cabe recalcar que el dispositivo se ha programado 5 veces diferentes. Cada una de ellas empleadas para la diferente adaptación e implementación de los sensores del dispositivo.

El montaje de los dispositivos se ha realizado por soldadura de estaño y con la plancha de soldado junto al stencil. Dado que la plancha de soldado no era mía, mi tutor Diego Gachet fue el encargado de realizar el soldado del RAK3172 y sus diferentes resistencias y condensadores. Tras este ensamblado se soldaron los sensores de humedad y temperatura, los pines, el LED, el fotoresistor LDR y el portapilas doble AA.

Una vez soldado y montado completamente, el dispositivo se verá como se muestra en las siguientes fotografías:



Fotografía 31: Parte frontal del dispositivo final



Fotografía 32: Parte trasera del dispositivo final

5.5. PROGRAMACIÓN DEL DISPOSITIVO

La programación efectuada en el proyecto para el desarrollo del prototipo se divide en 5 iteraciones básicas:

1. **Programación de prueba de LED**
2. **Programación de fotoresistor LDR**
3. **Programación de Sensor de temperatura y humedad**
4. **Programación para recopilación completa de datos**
5. **Programación para recepción de datos mediante Gateway**

5.5.1. Programación de prueba de LED

El primer programa diseñado es muy simple, su función es la de verificar que el sistema funciona correctamente y que su programación es posible sin ningún tipo de inconveniente a la hora de realizar la conexión entre ordenador y placa PCB a través del STLink V2.

El programa emplea dos librerías básicas que permiten el parpadeo del LED durante unos milisegundos definidos. Estas son:

a) **HAL_GPIO_TogglePin:** La función posee:

   a. **Nombre de función:** Void HAL_GPIO_TogglePin (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin)

   b. **Descripción de función**: Cambia de un estado a otro el "GPIO pin" elegido.

   c. **Parámetros de función**: Posee:

      i. **GPIOx:** Donde "x" hace referencia al periférico GPIO de la familia STM32WLxx.

      ii. **GPIO_Pin:** Especifica el pin que cambiará de estado.

   d. **Parámetros devueltos**: Ninguno.

b) **HAL_Delay**: La función posee:

   a. **Nombre de función:** Void HAL_Delay (uint32_t Delay)

   b. **Descripción de función:** La función provee un mínimo retraso de la variable elegida.

   c. **Parámetros de función:** Posee:

      i. **Delay:** Especifica el valor de retraso en milisegundos.

   d. **Parámetros devueltos**: Ninguno.

En primer lugar, se configura el microcontrolador que vamos a programar. En el caso de este proyecto es el STM32WLE5CCU6 como puede observarse en la siguiente ilustración:

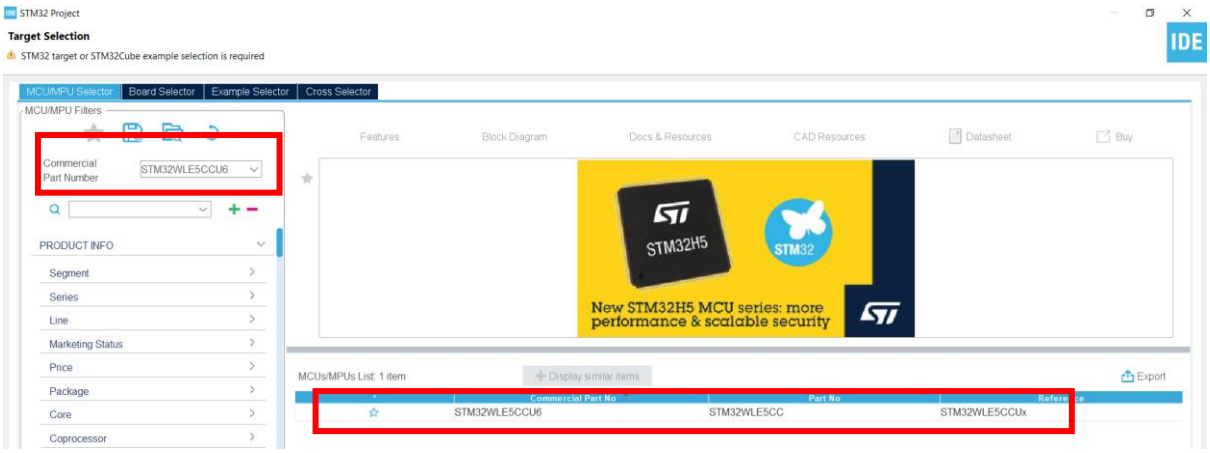


Ilustración 82: Configuración de STM32WLE5CCU6

En segundo lugar, se conecta la placa al ordenador mediante el STLink V2 quedando los cables conectados de la siguiente forma:

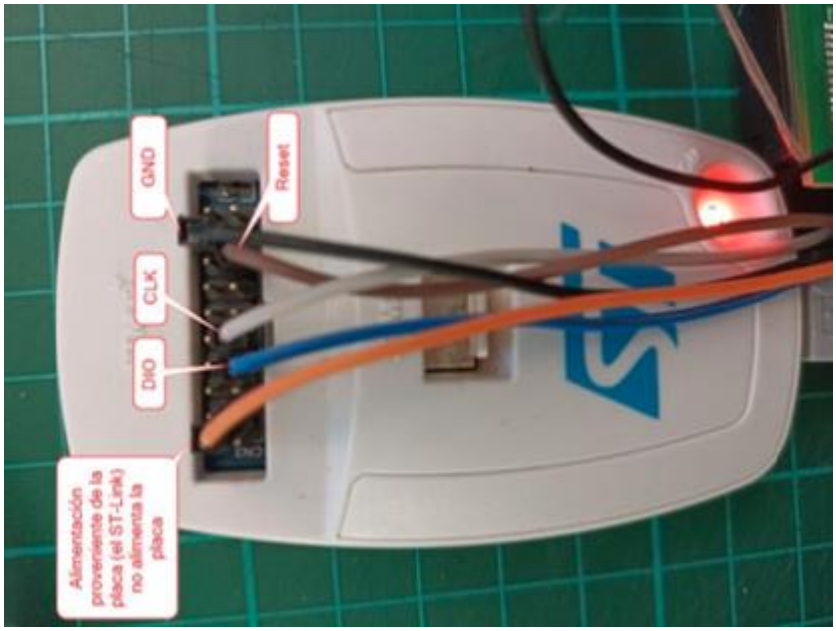


Ilustración 83: Conexión de cables para la programación del LED

En segundo lugar, se procede a configurar el procesador en el STM32CubeIde a través del STM32CubeMx que se encuentra incorporado en el interior del programa. Se configurará el Pin A8 de nuestro microcontrolador como salida, es decir, GPIO_Output. Y se le asigna el nombre LED mediante una etiqueta. En la siguiente ilustración se puede observar esta configuración descrita:

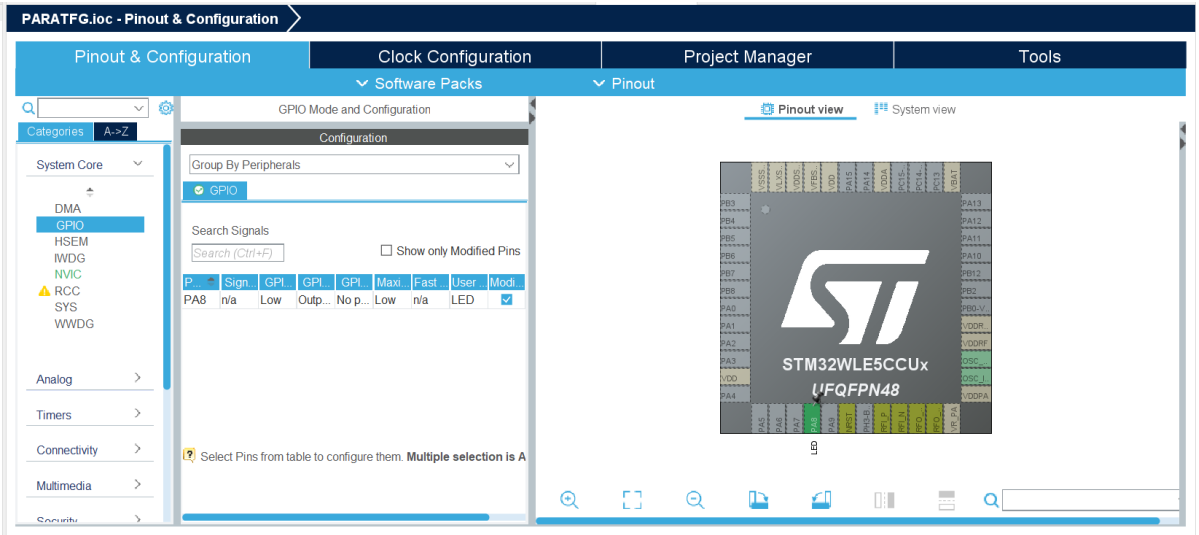


Ilustración 84: Configuración necesaria para el parpadeo del LED

En tercer lugar, una vez configurado el pin deseado se procede a generar el código presionando en la barra de herramientas el botón “Device configuration Tool Generation”. Este botón permite la generación automática de código con respecto a los parámetros escogidos en el paso previo, supone un elevado ahorro de tiempo ya que genera de manera

automática en el main.c y en el main.h el código referenciado. Un ejemplo que permite visualizar esta elección del PIN A8 como LED es el siguiente:

```
/* Private defines ----------
#define LED_Pin GPIO_PIN_8
#define LED_GPIO_Port GPIOA
```

```
/*Configure GPIO pin : LED_Pin */
GPIO_InitStruct.Pin = LED_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(LED_GPIO_Port, &GPIO_InitStruct);
```

Ilustración 85: Ejemplo de visualización en main.h del PIN A8_LED

En cuarto lugar, se programa en el main.c el siguiente código que permitirá el parpadeo del LED cada 1000 ms:

```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
      HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);
      HAL_Delay(1000);
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

Ilustración 86: Código main.c que permite el parpadeo del LED cada 1000 ms

En quinto lugar, se ejecuta el botón “RUN” y vemos que el programa funciona. No se aporta fotografía del resultado ya que no se considera relevante.

5.5.2. Programación de fotorresistor LDR

El segundo programa diseñado es algo más complejo que el anterior, una vez que se asegura el correcto funcionamiento de la PCB y su consecuente intercambio de datos con el PC se procede a recolectar datos sobre el nivel de luminosidad presente en el ambiente a través de un LDR.

En primer lugar, se configura de nuevo el pin deseado para la recolección de datos. Se nombra como ADC_IN2 y se establece que es de tipo Analog. En este caso se conecta al pin PB3. Además, se configura la salida Usart para mostrar los datos en la terminal como se explica en apartados anteriores. En la siguiente ilustración se observa esta configuración:

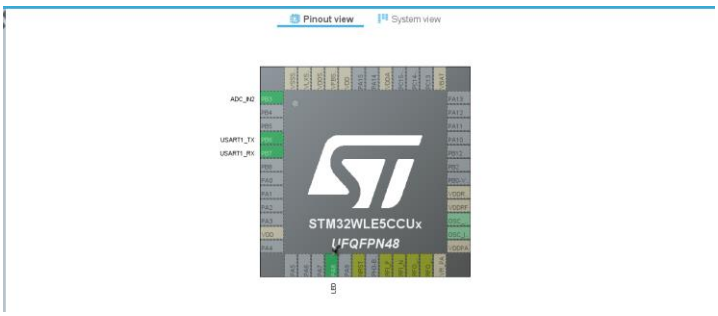


Ilustración 87: Configuración para programación del fotorresistor

En segundo lugar, se configuran tanto los puertos Usart para la visualización de datos como el puerto Analog del fotorresistor LDR. La configuración del puerto como analog es debido a que el fotorresistor recopila información en analógico, que posteriormente será convertida a digital para así poder ser mostrada al usuario. Al activar el Analog mode, se configura el pin como no pull up and no pull down. Las configuraciones que se han de realizar quedan reflejadas en las siguientes ilustraciones:

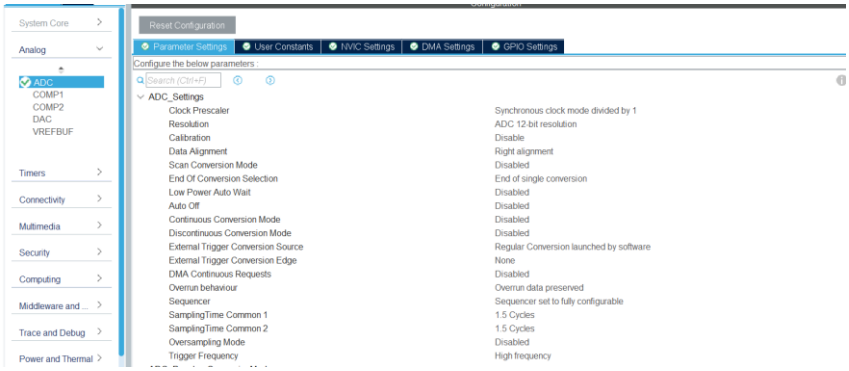


Ilustración 88: Configuración Pin PB3 para LDR

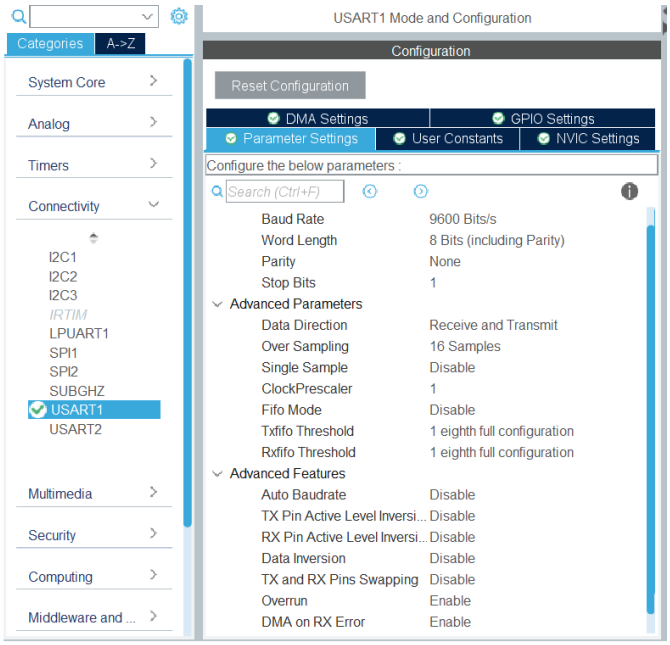


Ilustración 89: Configuración puerto Usart 1

En tercer lugar, se ejecuta el programa y se procede a la autogeneración de código por parte de las configuraciones realizadas en los anteriores dos apartados de esta programación del LDR.

En cuarto lugar, se programa en el main.c la configuración necesaria para realizar la lectura y transmisión de datos de luminosidad. Las librerías necesarias son:

a) **HAL_ADC_COnfigChannel:** La función posee:

   a. **Nombre de función:** HAL_StatusTypeDef HAL_ADC_ConfigChannel (ADC_HandleTypeDef * hadc, const ADC_ChannelConfTypeDef * pConfig)

   b. **Descripción de función**: Configura el canal al que se asigna el ADC.

c. **Parámetros de función**: Posee:

i. **Hadc:** ADC handle.

ii. **pConfig:** Indica el canal de configuración en el que se encuentra el ADC.

d. **Parámetros devueltos**: HAL: Status

b) **HAL_ADC_Start:** La función posee:

a. **Nombre de función**: HAL_StatusTypeDef HAL_ADC_Start (ADC_HandleTypeDef * hadc).

b. **Descripción de función**: Activa ADC e inicia la conversión

c. **Parámetros de función**: Posee:

i. **Hadc**: ADC handle.

d. Parámetros devueltos: HAL: Status

c) **HAL_ADC_GetValue:** La función posee:

a. **Nombre de función**: uint32_t HAL_ADC_GetValue (const ADC_HandleTypeDef * hadc)

b. **Descripción de función**: Obtiene el resultado ADC de la conversión.

c. **Parámetros de función**: Posee:

i. **Hadc**: ADC handle.

d. Parámetros devueltos: ADC: Grupo de datos convertidos

d) **HAL_ADC_PollForConversion:** La función posee:

a. **Nombre de función**: HAL_StatusTypeDef HAL_ADC_PollForConversion (ADC_HandleTypeDef * hadc, uint32_t Timeout)

b. **Descripción de función**: Espera a la completa conversión de un grupo

c. **Parámetros de función**: Posee:

i. **Hadc**: ADC handle.

ii. **Timeout**: Valor en milisegundos

d. Parámetros devueltos: HAL: Status

e) **HAL_UART_Transmit** La función posee:

a. **Nombre de función**: HAL_StatusTypeDef HAL_UART_Transmit (UART_HandleTypeDef * huart, const uint8_t * pData, uint16_t Size, uint32_t Timeout)

b. **Descripción de función**: Envía una cantidad de datos en forma de bloques

c. **Parámetros de función**: Posee:

i. **Huart:** UART handle

ii. **pData:** Puntero al buffer de datos (de u8 o u16).

iii. **Size:** Cantidad de elementos de datos para enviar (u8 o u16)

iv. **Timeout**: Valor en milisegundos

d. **Parámetros devueltos:** HAL: Status

La visualización de todas estas funciones y su uso en el main.c se puede observar en la siguiente ilustración:

```
114   while (1)
115   {
116
117       HAL_GPIO_TogglePin(LED_GPIO_Port,LED_Pin);
118       HAL_Delay(1000);
119         HAL_ADC_Start(&hadc);
120         HAL_ADC_PollForConversion(&hadc, 10000);
121         adc_val = HAL_ADC_GetValue(&hadc);
122         sprintf((char *)mensaje,"valor %u\r\n", adc_val);
123         HAL_UART_Transmit(&huart1,mensaje,sizeof(mensaje),HAL_MAX_DELAY);
124         HAL_Delay(100);
125
```

Ilustración 90: Código para leer el fotorresistor LDR

En quinto lugar, se ejecuta el programa y al abrir el realTerm para la visualización se configura una nueva sesión de tipo serial con el número de bits por segundo especificados en el primer apartado. El resultado mostrado en la terminal es el siguiente:

```
Luz: 2936
Luz: 2949
Luz: 2929
Luz: 2944
Luz: 2920
Luz: 2946
Luz: 2928
Luz: 2905
Luz: 2912
Luz: 2954
Luz: 2939
Luz: 2918
Luz: 2901
Luz: 2900
Luz: 2904
Luz: 2915
Luz: 2911
Luz: 2919
Luz: 2905
Luz: 2951
Luz: 2928
Luz: 2903
Luz: 2953
Luz: 2933
Luz: 2938
Luz: 2903
Luz: 2901
```

Ilustración 91: Ejemplo de programa lector de luminosidad

5.5.3. Programación de Sensor de temperatura y humedad

La programación del sensor Si7021 de temperatura y humedad es algo más compleja, en este caso se programa en función de la información dispuesta en el datasheet del fabricante del sensor.

En primer lugar, se configuran los pines en los que va a ir situado nuestro sensor de temperatura y humedad. En este caso se configura SCL y SDA como se indica en el propio sensor. No se van a nombrar de diferente manera a la especificada por el programa.

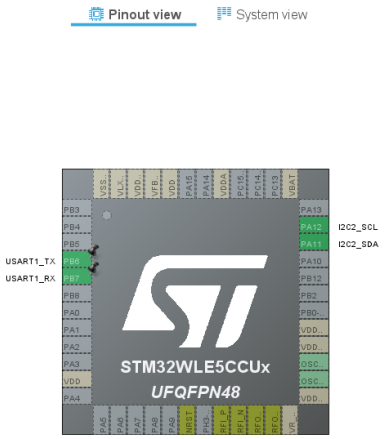


Ilustración 92: Configuración de sensor SI7021

En segundo lugar, se han de configurar los pines correspondientes al sensor, en este dispositivo son los pines PA11 y PA12 del microcontrolador. La configuración ha de ser la siguiente:

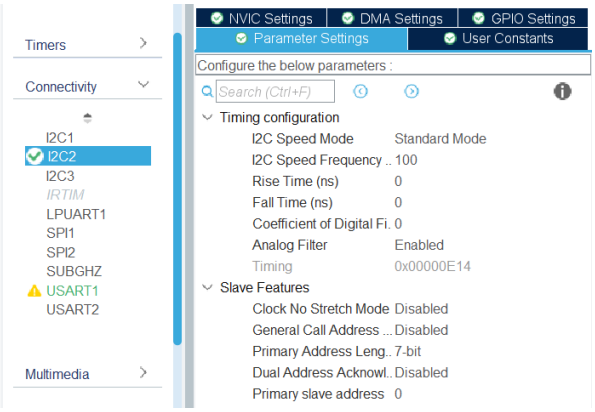


Ilustración 93: Configuración de pines del sensor SI7021

En tercer lugar, se autogenera el código necesario para la implementación de este y se procede a la programación del dispositivo. Se procede a mencionar las bibliotecas empleadas:

a) **HAL_I2C_Master_Transmit:** La función posee:

   a. **Nombre de función**: HAL_StatusTypeDef HAL_I2C_Master_Transmit (I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t* pData, uint16_t Size, uint32_t Timeout)

   b. **Descripción de función**: Transmite en modo maestro datos en forma de bloques

   c. **Parámetros de función**: Posee:

      i. **Hi2c:** Puntero a I2C_HandleTypeDef structure que contiene la información de configuración del I2C

      ii. **DevAddress:** Dirección del dispositivo de objetivo

iii. **pData:** Puntero al buffer de datos

iv. **Size:** Cantidad de elementos de datos para enviar (u8 o u16)

v. **Timeout**: Valor en milisegundos

d. **Parámetros devueltos:** HAL: Status

b) **HAL_I2C_Master_Receive:** La función posee:

a. **Nombre de función**: HAL_StatusTypeDef HAL_I2C_Master_Receive (I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t* pData, uint16_t Size, uint32_t Timeout)

b. **Descripción de función**: Recibe en modo maestro datos en forma de bloques

c. **Parámetros de función**: Posee:

i. **Hi2c:** Puntero a I2C_HandleTypeDef structure que contiene la información de configuración del I2C

ii. **DevAddress:** Dirección del dispositivo de objetivo

iii. **pData:** Puntero al buffer de datos

iv. **Size:** Cantidad de elementos de datos para enviar (u8 o u16)

v. **Timeout**: Valor en milisegundos

d. **Parámetros devueltos:** HAL: Status

En cuarto lugar, se programa el fichero si7021.h que será incorporado al main.c mediante un #include "si7021.h". En este fichero en primer lugar se definen las direcciones en las que hay que buscar la información del sensor, a continuación, se especifican las variables de almacenamiento de temperatura y humedad. Y por último se inicializa la recepción de datos del sensor. El código referente se puede observar en la siguiente ilustración:

```
4  extern I2C_HandleTypeDef hi2c2;
5  uint8_t data[3];
6  uint8_t da[1] = {0xFE};
7  uint8_t read[2]= {0xE3,0xE5};
8  uint8_t data1[3];
9  float temp,tempp;
10 float hu,humid;
11 uint16_t sum_temp;
12 uint16_t sum_humid;
13
14 void si7021_init()
15 {
16 HAL_I2C_Master_Transmit(&hi2c2,0x80,da,1,100);
17     HAL_Delay(20);
18 }
```

Ilustración 94: Código de declaración del fichero si7021.h

En quinto lugar, se realiza la función del sensor. Los lugares de almacenamiento para la temperatura y la humedad se especifican en el datasheet del dispositivo. Sin embargo, para la función void get_si7021 que enviará los datos de temperatura y humedad de tipo float se

deben de realizar las consecuentes operaciones matemáticas que permitan obtener los resultados más precisos posibles.

Las ecuaciones relevantes son:

$$\%RH = \frac{125 \cdot RH}{65536} - 6$$

Ecuación 9: Porcentaje de humedad relativa

$$Temperatura\ ºC = \frac{175{,}72 \cdot Temp_Code}{65536} - 46{,}85$$

Ecuación 10: Temperatura ambiental

A continuación, se programan estas para que el programa pueda mostrar el valor obtenido tanto de temperatura como de humedad. Se muestra en las siguientes ilustraciones el código empleado para la visualización de los parámetros mentados previamente:

```
19 void get_si7021(float *temp,float *humid)
20 {
21
22
23         HAL_I2C_Master_Transmit(&hi2c2,0x80,&read[0],1,100);/* Envía la lectura de datos de temperatura*/
24         HAL_I2C_Master_Receive(&hi2c2,0x80,data,3,1000);/* Recibe la lectura de datos de temperatura*/
25         sum_temp = (data[0]<<8) | data[1]; /* ALmcenamos los dos bytes en una variable de 16 bits*/
26         tempp    = sum_temp;
27         *temp = (-46.85 + (175.72*(tempp/65536)));
28         HAL_Delay(20);/*Parada de 20 ms para estabilizar el sensor*/
29         HAL_I2C_Master_Transmit(&hi2c2,0x80,&read[1],1,100);/* Envía la lectura de datos de humedad*/
30         HAL_I2C_Master_Receive(&hi2c2,0x80,data1,3,1000);/* Recibe la lectura de datos de humedad*/
31         sum_humid= (data1[0]<<8) | data1[1]; /* Almacenamos los dos bytes en una variable de 16 bits*/
32         hu= sum_humid;
33         *humid = (-6+(125*(hu/65536)));
34         HAL_Delay(20);
35     }
36
37 #endif
```

Ilustración 95: Código para leer la humedad relativa y la temperatura

El código es bastante sencillo, sólo llama la atención el apartado de lectura visto en la variable **sum_temp**. Este recolecta los datos que recibe así dado a que en el datasheet se nos especifica la forma en la que se están leyendo los datos del sensor Si7021. Así pues, se recolectan datos de la forma que tenemos: **"XXXXXXXX10"** en el bit menos significativo.

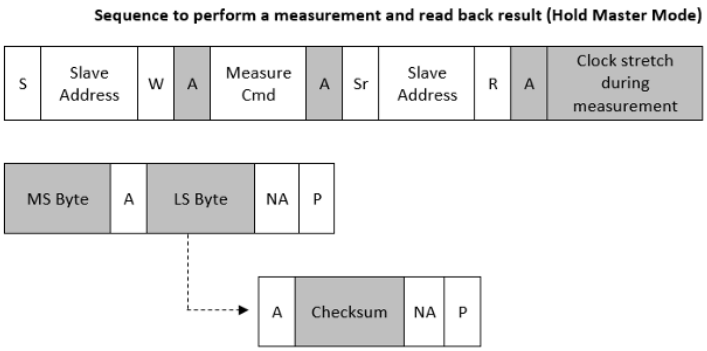


Ilustración 96: Secuencia de recepción de datos del sensor SI7021

Por último, para la escritura de los datos en el terminal se ejecuta el siguiente código:

```
while (1)
{
    HAL_GPIO_TogglePin(LED_GPIO_Port,LED_Pin);
    HAL_Delay(1000);
    HAL_ADC_Start(&hadc);
    HAL_ADC_PollForConversion(&hadc, 10000);
    adc_val = HAL_ADC_GetValue(&hadc);

    get_si7021(&temp, &humid);



     sprintf((char *)temperatura,"Temp: %.2f\r\n", temp);
     HAL_UART_Transmit(&huart1,temperatura,sizeof(temperatura),HAL_MAX_DELAY);

    sprintf((char *)humedad,"Hum: %.2f\r\n", humid);
    HAL_UART_Transmit(&huart1,humedad,sizeof(humedad),HAL_MAX_DELAY);

    HAL_Delay(100);
```

Ilustración 97: Código para visualizar los datos en la terminal

Así pues, en nuestra terminal al realizar una prueba se visualizará de la siguiente forma la lectura de datos de temperatura y humedad:

```
Temp: 21.71
Hum: 42.76

Temp: 24.30
Hum: 41.19

Temp: 24.88
Hum: 42.39

Temp: 20.77
Hum: 41.54
```

Ilustración 98: Visualización final de datos del sensor Si7021 en la terminal

5.5.4. Programación para recopilación completa de datos

Antes de comenzar con la última parte del trabajo hemos de realizar un programa que nos permita la lectura de todos los sensores, es decir, temperatura, humedad y luminosidad a la vez. Dado que he ido describiendo paso por paso previamente todos y cada uno de los pasos a implementar para cada lector individualmente, en este caso solo hemos de añadir a la estructura principal del programa el siguiente código en el archivo main.c:

```
119  while (1)
120  {
121      HAL_GPIO_TogglePin(LED_GPIO_Port,LED_Pin); /*Realiza un parpadeo del LED de la PCB*/
122      HAL_Delay(1000);
123      HAL_ADC_Start(&hadc); /*Comienza la conversión de analógica a digital en la estructura hadc predefinida*/
124      HAL_ADC_PollForConversion(&hadc, 10000);/*Espera para que la conversión se complete en 1000 ms*/
125      adc_val = HAL_ADC_GetValue(&hadc);/*Se almacena la lectura en formato digital en la variable*/
126
127      get_si7021(&temp, &humid); /*Llamada a función para obtener la temperatura y velocidad*/
128
129      sprintf((char *)mensaje,"Luz: %u\r\n", adc_val); /*Se formatea la cadena de caracteres en la variable mensaje*/
130      HAL_UART_Transmit(&huart1,mensaje,sizeof(mensaje),HAL_MAX_DELAY);/*Transmite la línea de caracteres almacenada*/
131
132
133       sprintf((char *)temperatura,"Temp: %.2f\r\n", temp);
134       HAL_UART_Transmit(&huart1,temperatura,sizeof(temperatura),HAL_MAX_DELAY);
135
136      sprintf((char *)humedad,"Hum: %.2f\r\n", humid);
137      HAL_UART_Transmit(&huart1,humedad,sizeof(humedad),HAL_MAX_DELAY);
138
139      HAL_Delay(100);
140
141
142    /* USER CODE END WHILE */
```

Ilustración 99:Código necesario para la escritura de todos los parámetros medidos

La visualización del programa tras su ejecución es la dispuesta en la siguiente ilustración:

Ilustración 100: Visualización del programa completo

5.5.5. Programación para recepción de datos mediante Gateway

El último apartado del presente trabajo consiste en la realización de la lectura de temperatura, humedad y luminosidad al igual que en apartados anteriores, pero con la gran diferencia de que se transmitirán los datos a través de LoRa.

Para comenzar se explica de forma muy sencilla el funcionamiento de recepción de datos que presenta nuestro dispositivo. Se requiere de tres dispositivos fundamentales:

- **Concentrador ic880a:** Es un concentrador de Semtech que permite construir gateways LoRa si se conecta con las Raspberry pi permitiendo recibir y transmitir datos entre los dispositivos LoRa y un servidor destinado a LoRaWAN.
- **Raspberry pi (modelo b+ v1.2 o superior):** Permite la instalación del software de Gateway para configurar nuestro concentrador ic880a como receptor de datos LoRaWAN.
- **Dispositivo LoRa con STM32WLE5CCU6 (Rak3172):** Es el transmisor de datos y nuestro proyecto de TFG.

El esquema de funcionamiento es el dispuesto en la siguiente ilustración:

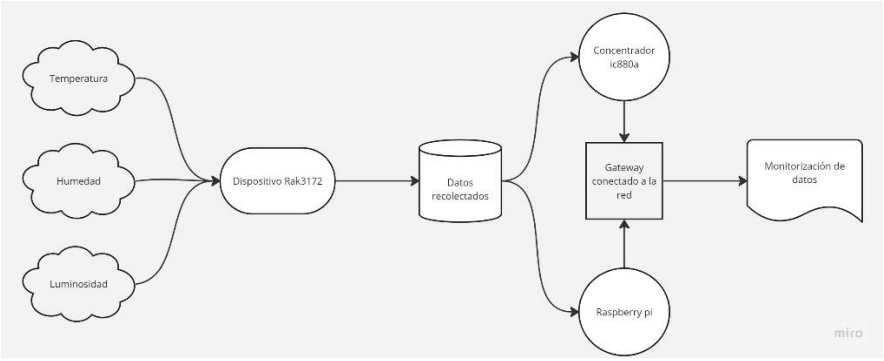


Ilustración 101: Esquema de funcionamiento de LoRa del dispositivo

Aunque parezca un proceso sencillo el simple hecho de mandar los datos monitorizados a través de LoRa dado los programas anteriores, no lo es. El código empleado y la configuración en el CubeIDEMx es bastante más compleja que la empleada previamente.

Para comenzar se ha de configurar el rak3172 según los dispositivos y sistemas de configuración que se deseen emplear. La configuración de pines es la dispuesta a continuación:

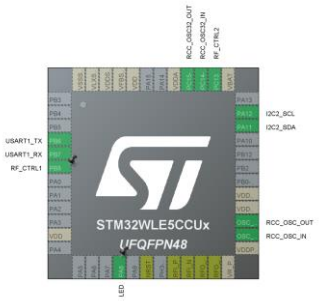


Ilustración 102: Configuración de pines del programa

En primer lugar, hemos de configurar los relojes de alta y baja velocidad de nuestro microcontrolador. Para ello elegimos la siguiente configuración:

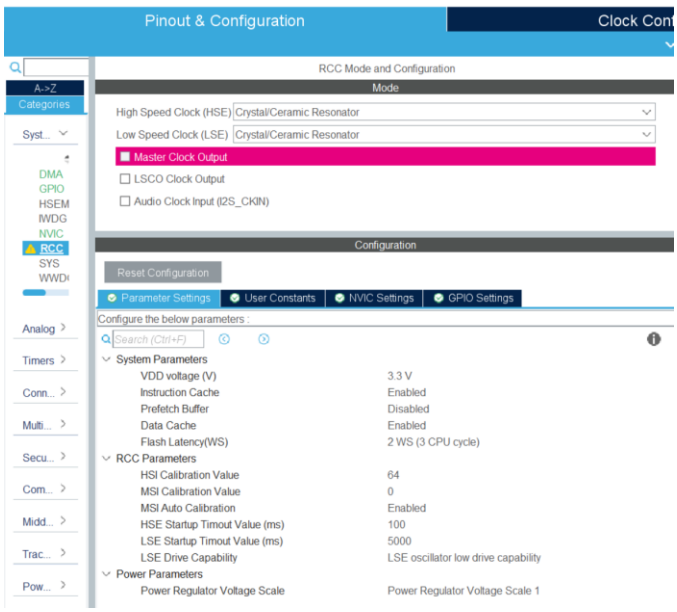


Ilustración 103: Configuración de HSC y LSC

Acto seguido se configura el reloj de tiempo real, es decir, un reloj que le indica en todo momento el tiempo que lleva funcionando el microcontrolador. Este reloj permite apagar los demás periféricos a la par que el reloj funciona solo, esto permite el funcionamiento correcto del dispositivo con el menor uso de batería posible. La configuración es la siguiente:

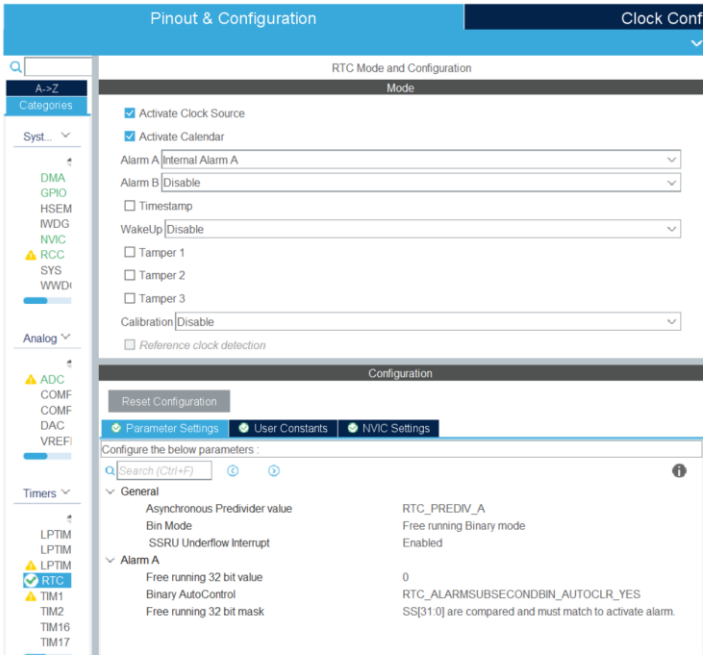


Ilustración 104: Configuración de reloj real

Es de elevada importancia la creación de unas constantes que permitan el control del esqueleto de código empleado y realizado en la comunidad The Things Network. Se emplea este esqueleto ya que permite fácilmente el empleo de transmisión LoRa. Los nombres y valores de estas constantes son los dispuestos en la ilustración:

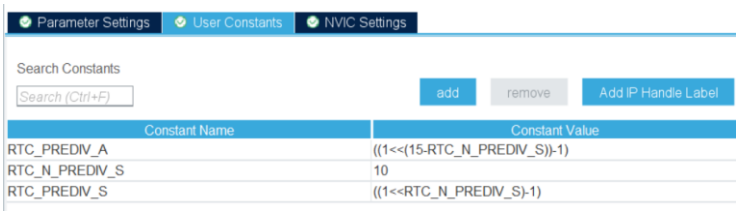


Ilustración 105: Constantes y valores del esqueleto de código

A continuación, para la lectura del nivel de abtería del microcontrolador se realiza un paso muy sencillo ya que el propio RAK3172 consta de lector de nivel de batería propio. De esta forma basta con activar en el ADC el canal de Vrefint Channel como se muestra en la ilustración:

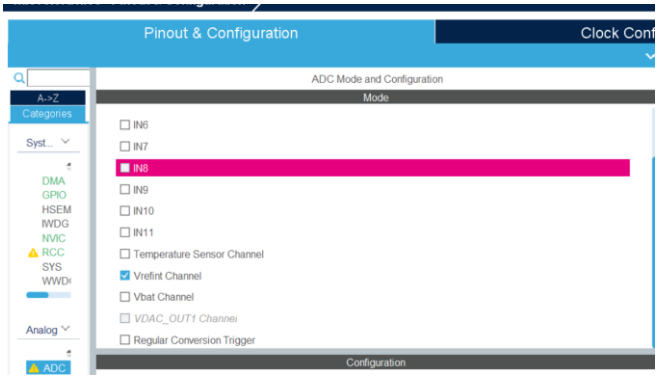


Ilustración 106: Configuración para la lectura del nivel de batería del dispositivo

Aunque se emplee un esqueleto para LoRaWAN se necesita un puerto serie UART que permita enviar mensajes que indiquen si hemos podido conectarnos por OTTA, las claves OTTA son las siguientes etc. LA configuración de USART1 debe quedar según lo dispuesto:

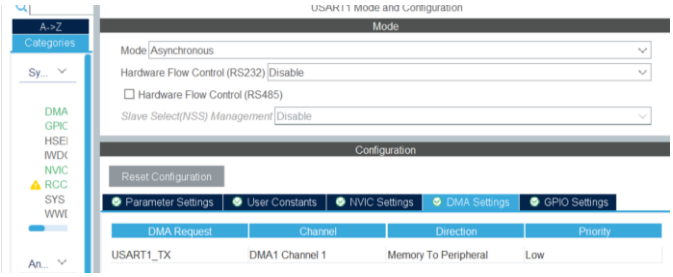


Ilustración 107: Configuración de claves OTTA

Recordemos que DMA es el Direct Memory Access y permite enviar mensajes directamente por el puerto serie sin que el microcontrolador tenga que intervenir empleando ciclos, es decir, se emplea solo el hardware puro del microcontrolador.

Como el dispositivo necesita de envío de datos a través de LoRa se ha de activar la frecuencia y su consecuente interrupción para ello. Por esto la configuración es la siguiente en el CubeIDEMx:

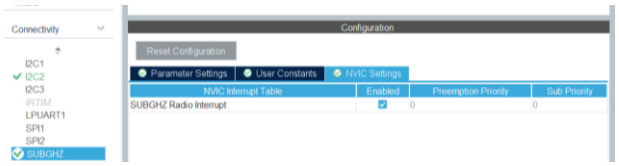


Ilustración 108: Activación de módulo de frecuencia

Ahora se ha de modificar desde el programa el módulo de LoRaWAN. Este apartado es realmente importante ya que permite modificaciones muy relevantes como si queremos enviar paquetes periódicamente o bien como respuesta a un evento (ejemplo de envío de información o pulsación de un botón). Es importante desactivar el modo de bajo consumo en caso de que al ejecutar el programa no se realicen las acciones previstas por parte del dispositivo. Por el contrario, si el programa realiza lo deseado se puede activar la casilla si problema. De consiguiente manera, la configuración es la siguiente:

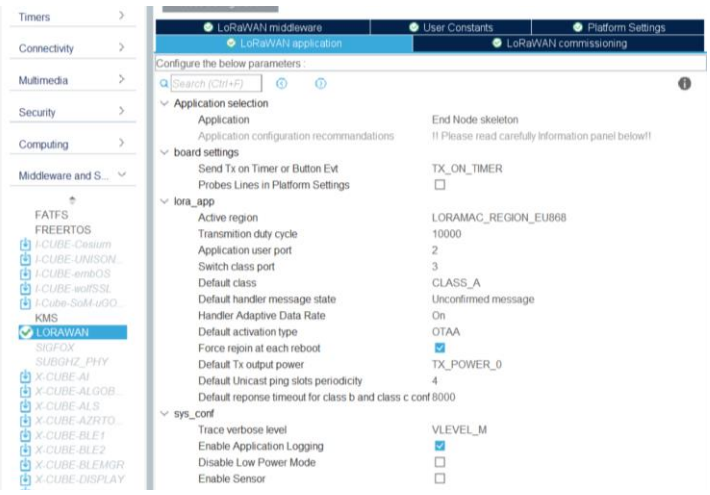


Ilustración 109: Configuración de LoRaWAN

Lo siguiente es configurar el middleware para determinar la frecuencia en la que queremos retransmitir los datos. En nuestro caso será 868 Hz dado que estamos en Europa. Además, hemos de configurar la transmisión en alta potencia dado que non tiene posibilidad de transmitir en baja potencia, esto implica que en lugar de consumir 40 mA se consumirán 80 mA. Así pues, la configuración necesaria presenta la siguiente forma:

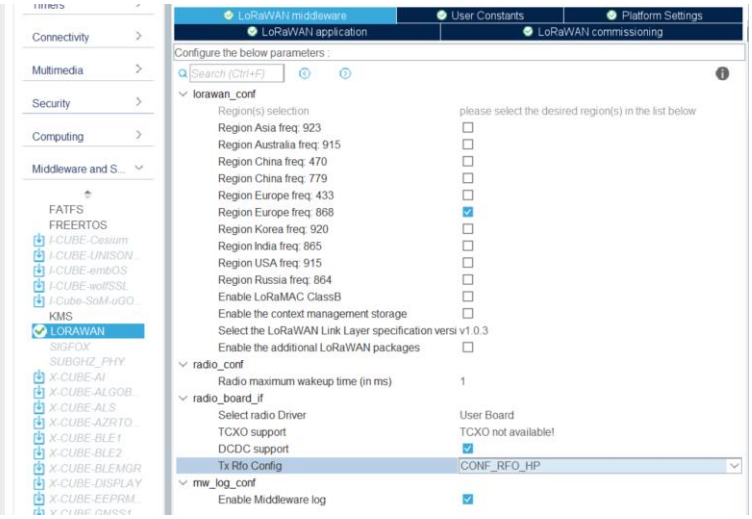


Ilustración 110: Configuración de frecuencia según región

Por último, realizamos la configuración que permite determinar el reloj que deseamos usar (programado anteriormente), la fuente de datos que se emplea (la USART) y el voltaje de lectura del nivel de batería configurado también con anterioridad. Así pues, las configuraciones de plataforma son las siguientes mostradas en la ilustración:

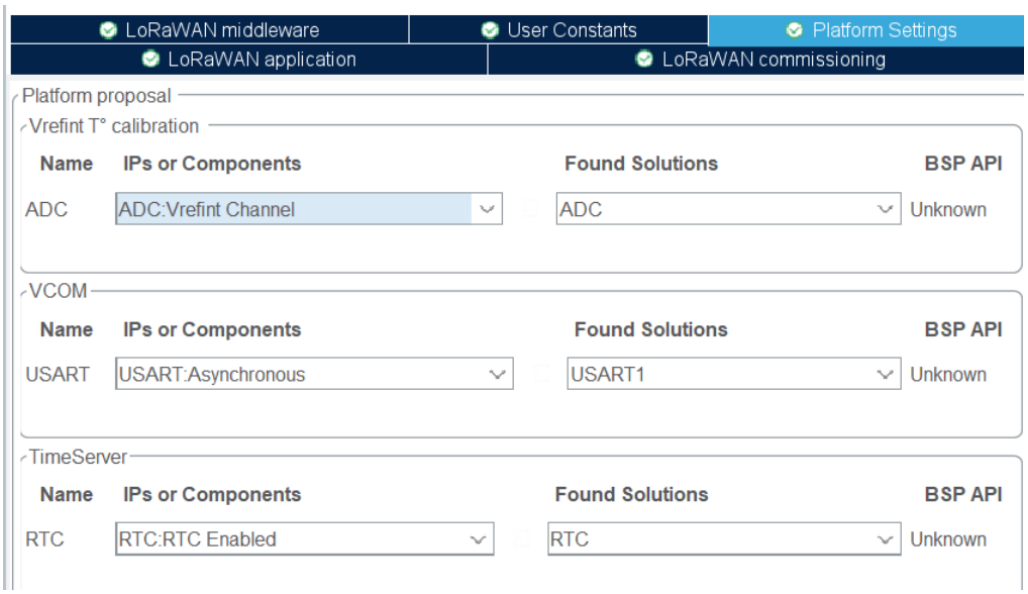


Ilustración 111: Configuraciones de la plataforma

Por último, se ha de modificar el comisionado de nuestro dispositivo, es decir el EUI y datos, claves etc. necesarios para la comunicación de nuestro dispositivo con nuestro futuro Gateway configurado en la red de The Things Network. Recalco que, aunque aporte la captura de configuración, eliminaré los datos del presente trabajo ya que pueden acarrearme violaciones de seguridad a nivel personal dado que el EUI es un código

alfanumérico de 16b dígitos que identifica de manera única a un Gateway de LoRaWAN. La configuración es la siguiente:

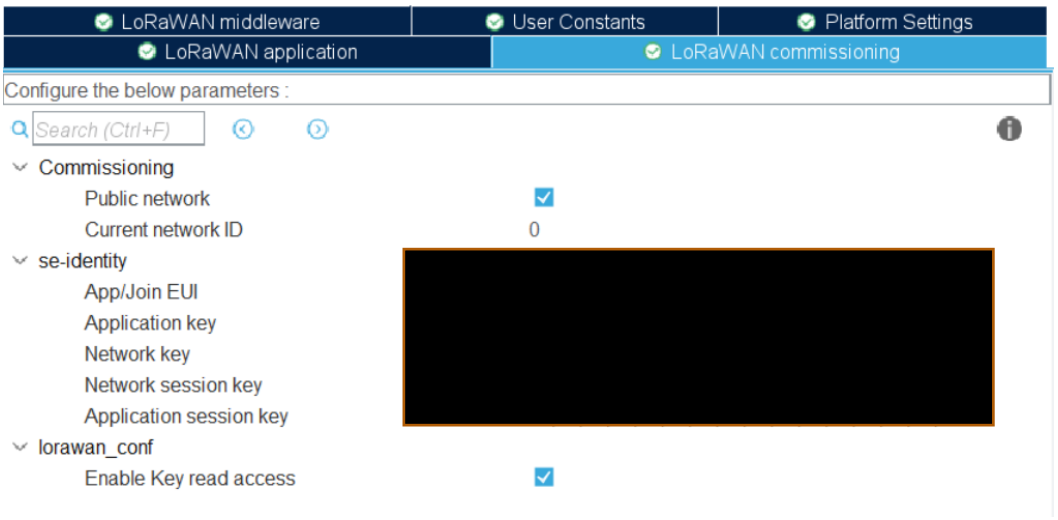


Ilustración 112: Parámetros de conexión LoRa

La configuración del reloj es de elevada importancia, así pues, nuestro dispositivo funciona con unos parámetros establecidos y que pueden ser modificados para su optimización y para la función que se desea desempeñar según usuario. EL reloj del dispositivo posee loa siguiente configuración:

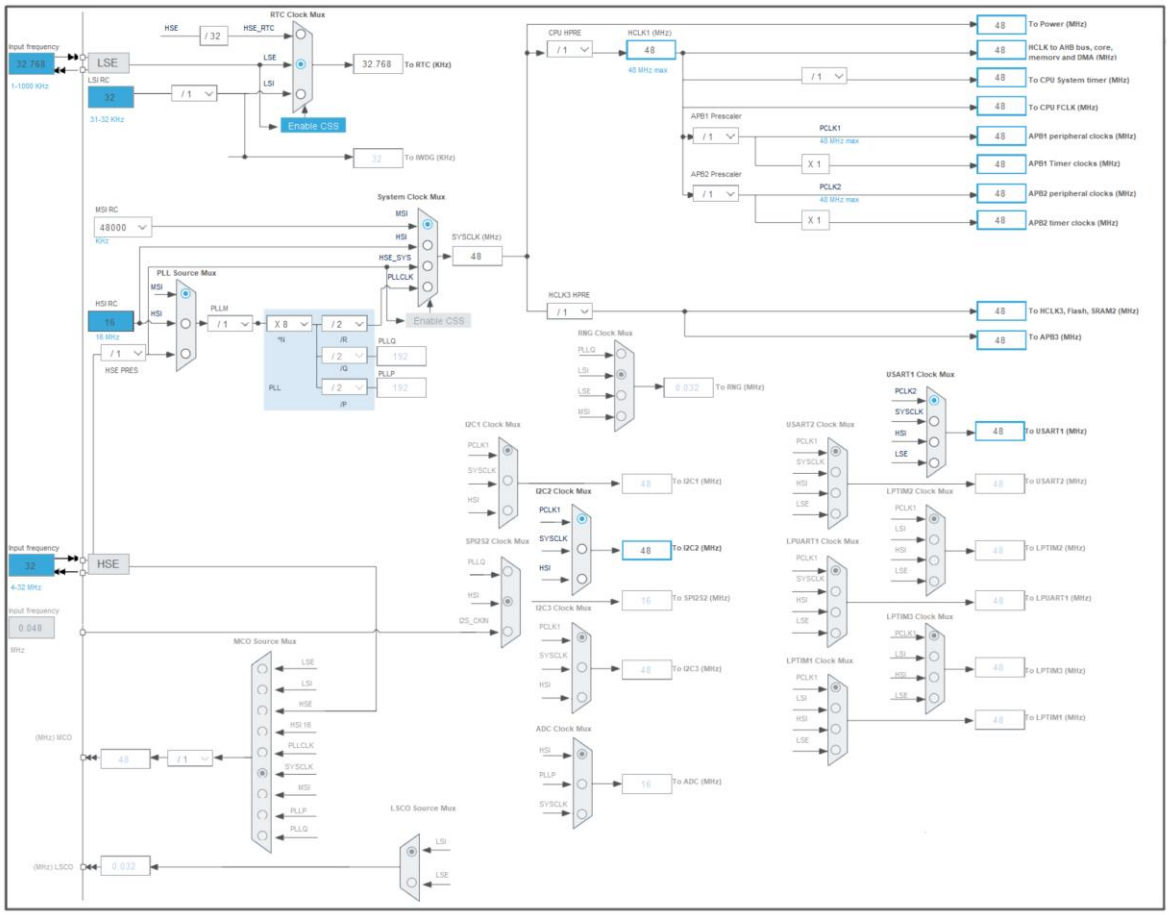


Ilustración 113: Configuración del reloj del micro

Configuramos algunos pines más dado que el esqueleto de código necesita identificarlos. Estos son el PB8 y el PC13 y se configurarán como pines de salida y con el respectivo nombre: GPIO_Output y como RF_CTRL1 y RF_CTRL2.

GPIO | I2C | RCC | USART

Search Signals

Search (Ctrl+F) — Show only Modified Pins

Pin Na...	Signal on P...	GPIO outpu...	GPIO mode	GPIO Pull-u...	Maximum o...	Fast Mode	User Label	Modified
PA8	n/a	Low	Output Pus...	No pull-up ...	Low	n/a	LED	☑
PB8	n/a	Low	Output Pus...	No pull-up ...	Low	Disable	RF_CTRL1	☑
PC13	n/a	Low	Output Pus...	No pull-up ...	Low	n/a	RF_CTRL2	☑

Ilustración 114: Configuración de pines de salida y CTRL

Por otra parte, mantenemos la configuración para el sensor de temperatura y humedad según lo dispuesto en la siguiente ilustración:

GPIO | I2C | RCC | USART

Search Signals

Search (Ctrl+F) — Show only Modified Pins

Pin Na...	Signal on P...	GPIO outpu...	GPIO mode	GPIO Pull-u...	Maximum o...	Fast Mode	User Label	Modified
PA11	I2C2_SDA	n/a	Alternate F...	Pull-up	Low	n/a		☑
PA12	I2C2_SCL	n/a	Alternate F...	Pull-up	Low	n/a		☑

Ilustración 115: Configuración de pines del si7021

Para consumir menos debemos configurar como Pull-up las salidas _TX y _RX. Así pues, el esquema será así:

GPIO | I2C | RCC | USART

Search Signals

Search (Ctrl+F) — Show only Modified Pins

Pin Na...	Signal on P...	GPIO outpu...	GPIO mode	GPIO Pull-u...	Maximum o...	Fast Mode	User Label	Modified
PB6	USART1_...	n/a	Alternate F...	Pull-up	Low	Disable		☑
PB7	USART1_...	n/a	Alternate F...	Pull-up	Low	Disable		☑

Ilustración 116: Configuración de USART

El código necesario para configurar la transmisión de datos se realiza en los archivos radio_board_if.h y radio_board_if.c. Estos se generan automáticamente al compilar el programa desde el CubeIdeMX como ya hemos visto anteriormente. Estos dos archivos al añadirles unos bloques de código definirían la función para la que sirve cada pin de nuestro dispositivo. Así pues, nuestro código necesario en radio board if.h es el siguiente:

```
90  #else
91  /* USER CODE BEGIN Exported PinMapping */
92  #warning user to provide its board definitions pins
93
94  #define RF_SW_CTRL1_PIN                    GPIO_PIN_8
95  #define RF_SW_CTRL1_GPIO_PORT              GPIOB
96  #define RF_SW_CTRL1_GPIO_CLK_ENABLE()      __HAL_RCC_GPIOB_CLK_ENABLE()
97  #define RF_SW_CTRL1_GPIO_CLK_DISABLE()     __HAL_RCC_GPIOB_CLK_DISABLE()
98
99  #define RF_SW_CTRL2_PIN                    GPIO_PIN_13
100 #define RF_SW_CTRL2_GPIO_PORT              GPIOC
101 #define RF_SW_CTRL2_GPIO_CLK_ENABLE()      __HAL_RCC_GPIOC_CLK_ENABLE()
102 #define RF_SW_CTRL2_GPIO_CLK_DISABLE()     __HAL_RCC_GPIOC_CLK_DISABLE()
103 /* USER CODE END Exported PinMapping */
104 #endif  /* USE_BSP_DRIVER  */
105
```

Ilustración 117: Código implementado en radio.board_if.h

En el radio_board_if.c disponemos lo siguiente:

```
79 #warning user to provide its board code or to call his board driver functions
80   GPIO_InitTypeDef  gpio_init_structure = {0};
81         /* Configure the Radio Switch pin */
82           gpio_init_structure.Pin   = RF_SW_CTRL1_PIN;
83           gpio_init_structure.Mode  = GPIO_MODE_OUTPUT_PP;
84           gpio_init_structure.Pull  = GPIO_NOPULL;
85           gpio_init_structure.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
86           HAL_GPIO_Init(RF_SW_CTRL1_GPIO_PORT, &gpio_init_structure);
87           gpio_init_structure.Pin = RF_SW_CTRL2_PIN;
88           HAL_GPIO_Init(RF_SW_CTRL2_GPIO_PORT, &gpio_init_structure);
89           HAL_GPIO_WritePin(RF_SW_CTRL2_GPIO_PORT, RF_SW_CTRL2_PIN, GPIO_PIN_RESET);
90           HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN, GPIO_PIN_RESET);
91   /* USER CODE END RBI_Init_2 */
92   return retcode;
93 #endif  /* USE_BSP_DRIVER  */
94 }
```

Ilustración 118: Código implementado en radio_board_if.c

Ya podemos compilar y generar todos los archivos correspondientes. Así pues, comenzaremos a continuación con la programación principal del programa. Esta es la parte del código encargado de transmitir y recibir esos datos configurados ya en pines a través de LoRaWAN.

En primer lugar, configuramos todo lo que deseamos transmitir en el fichero lora_app.c. En este fichero se crea un timer que llamará a la función OnTxTimerEvent, posteriormente arranca ese timer y así la aplicación al iniciarse pone en marcha un cronómetro de 10 segundos. Una vez que se cumplan esos 10 segundos va a llamar a esa función TxTimer y en esa función escogerá la tarea de envío asignándola prioridad 0, una prioridad que será la máxima para así tener la funcionalidad de bajo consumo. Esto nos permitirá asignar las tareas que debe realizar el dispositivo en función de una prioridad establecida. Ese código es el siguiente:

```
336
337   if (EventType == TX_ON_TIMER)
338   {
339     /* send every time timer elapses */
340     UTIL_TIMER_Create(&TxTimer, TxPeriodicity, UTIL_TIMER_ONESHOT, OnTxTimerEvent, NULL);
341     UTIL_TIMER_Start(&TxTimer);
342   }
```

Ilustración 119: Configuración de timers

Para la programación de envío de datos se muestra a continuación en la ilustración el código necesario. Se aclara que dado que vamos a poder leer los datos a través de un basic-node y a través de la app opentech se incluye algún apartado de código según mi tutor dado que yo no poseo cuenta ni permiso para esta plataforma.

```
422 static void SendTxData(void)
423 {
424   /* USER CODE BEGIN SendTxData_1 */
425
426
427     uint16_t batteryLevel = SYS_GetBatteryLevel(); /*Almacenamos la tensión en mV en un entero de 16 bits*/
428     uint16_t LumenLevel = (float)GetLumenLevel()*1023.0/4095.0;
429     //SOLO PARA EL GTAFICO OPENTECH
430
431         LmHandlerErrorStatus_t status = LORAMAC_HANDLER_ERROR;
432         UTIL_TIMER_Time_t nextTxIn = 0;
433         AppData.Port = LORAWAN_USER_APP_PORT;
434         AppData.Buffer[0] = (uint8_t) ((batteryLevel >> 8) & 0xFF);/*Dividimos en dos bytes para su envío*/
435         AppData.Buffer[1] = (uint8_t) (batteryLevel & 0xFF);
436         AppData.Buffer[2] = (uint8_t) ((intemp >> 8) & 0xFF);
437         AppData.Buffer[3] = (uint8_t) (intemp & 0xFF);
438         AppData.Buffer[4] = (uint8_t) ((inhum) & 0xFF);
439         AppData.Buffer[5] = (uint8_t) ((512 >> 8) & 0xFF); //HUMED SUELO
440         AppData.Buffer[6] = (uint8_t) (512 & 0xFF);
441         AppData.Buffer[7] = (uint8_t) ((LumenLevel >> 8) & 0xFF);
442         AppData.Buffer[8] = (uint8_t) (LumenLevel & 0xFF);
443
```

```
        AppData.BufferSize = 9; //SI NO NO RECIBE OPENTECH
        status = LmHandlerSend(&AppData, LmHandlerParams.IsTxConfirmed, false); /*Permite enviar el mensaje de forma deseada si está en true*/

        if (LORAMAC_HANDLER_SUCCESS == status) {
            APP_LOG(TS_ON, VLEVEL_L, "SEND REQUEST\r\n");
        } else if (LORAMAC_HANDLER_DUTYCYCLE_RESTRICTED == status) {
            nextTxIn = LmHandlerGetDutyCycleWaitTime(); /*Calcula cuánto tiempo tiene que esperar para enviar el mensaje*/
            if (nextTxIn > 0) {
                APP_LOG(TS_ON, VLEVEL_L, "Next Tx in  : ~%d second(s)\r\n",
                        (nextTxIn / 1000));
            }
        }

        if (EventType == TX_ON_TIMER) {
            UTIL_TIMER_Stop(&TxTimer); /*Se adapta el timer a ese tiempo de espera para volver a intentar el envío*/
            UTIL_TIMER_SetPeriod(&TxTimer, MAX(nextTxIn, TxPeriodicity));
            UTIL_TIMER_Start(&TxTimer);
        }

        HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_SET);
        HAL_Delay(100);
        HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);
  /* USER CODE END SendTxData_1 */
}
```

Ilustración 120: Programación para envío de datos de sensores

Se prosigue para configurar en The Things Network nuestro Gateway, para ello nos hemos de crear una cuenta en caso de no poseerla y registrar nuestro Gateway introduciendo los datos aportados previamente. La configuración de la aplicación se muestra de la siguiente forma:

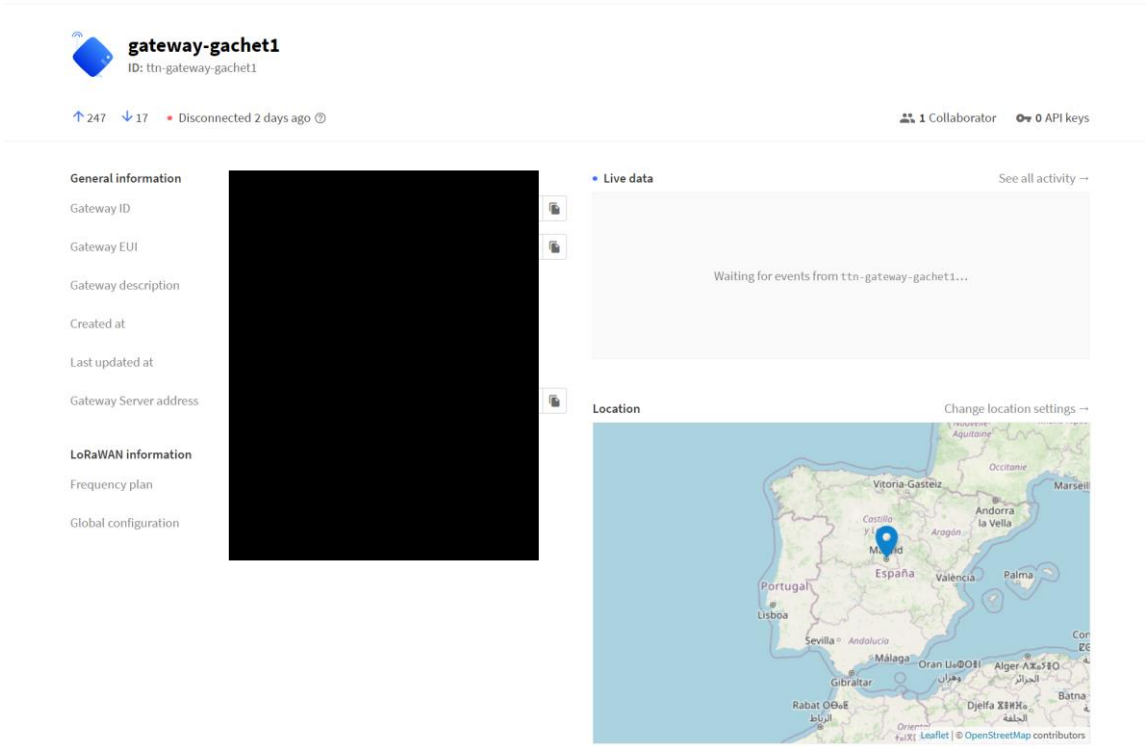


Ilustración 121: Configuración del Gateway en TTN

Por último, introducimos un dispositivo, en nuestro caso es el programado y mencionado a lo largo de este proyecto. Para ello registramos el dispositivo desde la web empleando los parámetros y configuraciones deseados por el usuario. La disposición del dispositivo según parámetros ha de ser rellenada por el usuario, en el caso dispuesto quedará de la siguiente forma:

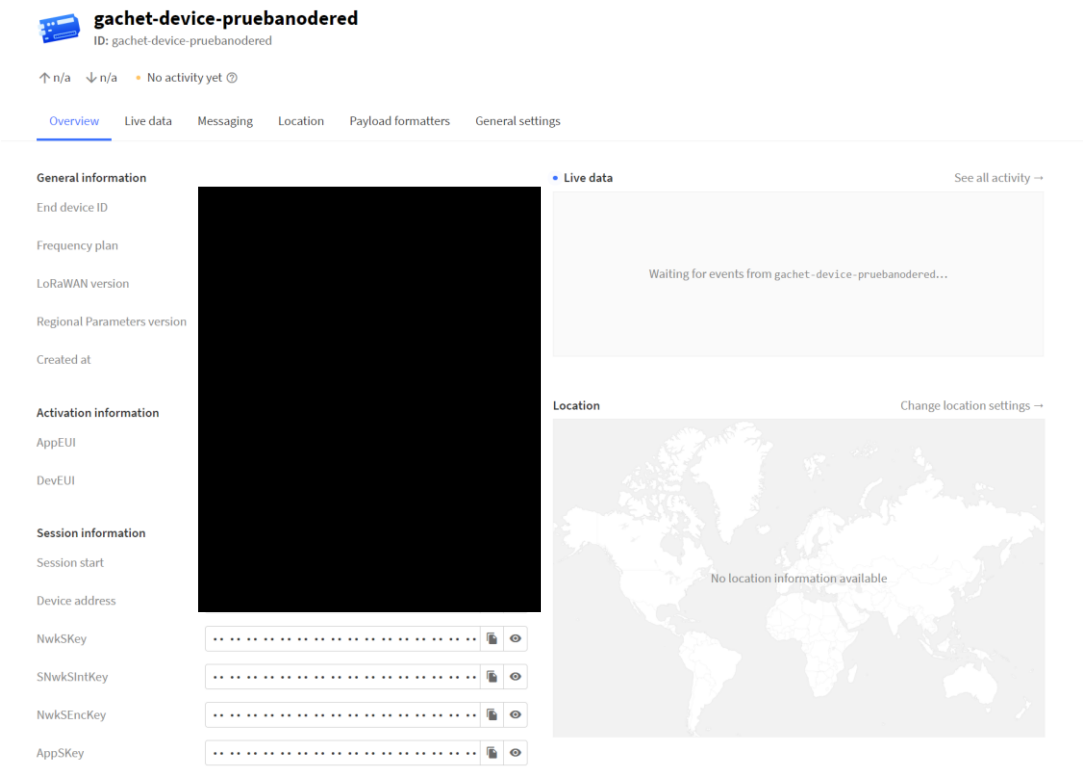


Ilustración 122: Configuración del dispositivo en TTN

Una vez se ha programado, cargado y configurado el programa realizado. Se comienza con la recepción de datos mediante LoRa. Los pasos para realizar son los descritos a continuación:

1. Conectar a la corriente el concentrador ic880a.
2. Conectar a la corriente la raspberry pi.
3. El concentrador presenta unas luces que indican los siguientes estados:

Luces	Estado
Parpadeo	No conectado
Fijas de color rojo	Conectado

Tabla 18: Estados de conexión del Gateway

Fotografía 33: Estado de conexión del Gateway

Una vez el concentrador está conectado a la red se puede comenzar con la recepción y visualización de datos. Esta visualización se puede conseguir mediante dos enlaces diferentes. El Gateway está retransmitiendo datos como se puede comprobar desde la web de TTN, se verá como se muestra a continuación:

Time	Type	Data preview
18:38:54	Receive gateway status	Metrics: { rxfw: 0, ackr: 100, txin: 0, txok: 0, rxin: 0, rxok: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:38:53	Console: Stream reconnected	The stream connection has been re-established
18:38:37	Console: Stream reconnected	The stream connection has been re-established
18:38:24	Receive gateway status	Metrics: { rxin: 3, rxok: 0, rxfw: 0, ackr: 100, txin: 0, txok: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:37:54	Receive gateway status	Metrics: { rxok: 0, rxfw: 0, ackr: 100, txin: 0, txok: 0, rxin: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:37:24	Receive gateway status	Metrics: { rxin: 0, rxok: 0, rxfw: 0, ackr: 100, txin: 0, txok: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:36:54	Receive gateway status	Metrics: { rxin: 0, rxok: 0, rxfw: 0, ackr: 100, txin: 0, txok: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:36:24	Receive gateway status	Metrics: { rxfw: 0, ackr: 100, txin: 0, txok: 0, rxin: 0, rxok: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:35:54	Receive gateway status	Metrics: { txok: 0, rxin: 1, rxok: 0, rxfw: 0, ackr: 100, txin: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:35:24	Receive gateway status	Metrics: { ackr: 100, txin: 0, txok: 0, rxin: 0, rxok: 0, rxfw: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:34:54	Receive gateway status	Metrics: { ackr: 0, txin: 0, txok: 0, rxin: 3, rxok: 0, rxfw: 0 } Versions: { ttn-lw-gateway-server: "3.25.1-SNAPSHOT-414e327a2" }
18:34:24	Connect gateway	

Ilustración 123: Estado de conexión del Gateway

La configuración más relevante será la que podamos modificar en todo momento dado que tenemos cuenta en la plataforma. La plataforma myopentech no me permite la creación de usuario por lo que para el presente trabajo se ha empleado Node-RED.

Node-RED es una herramienta para desarrolladores cuya programación es de tipo visual y permite editar y conectar nodos para establecer conexión entre ellos con el consecuente intercambio de información.

El nodo de programación realizado para el presente proyecto con la ayuda del tutor es el dispuesto en la siguiente ilustración:

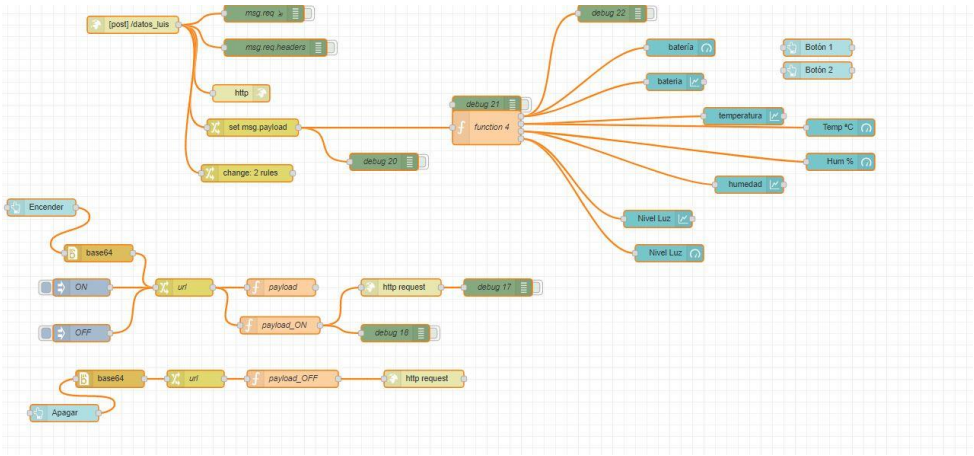


Ilustración 124: Flujo del programa para la recepción de datos

El nodo conectado a la red saldrá conectado y también es comprobable desde el enlace de TTN como se observa en la siguiente ilustración:

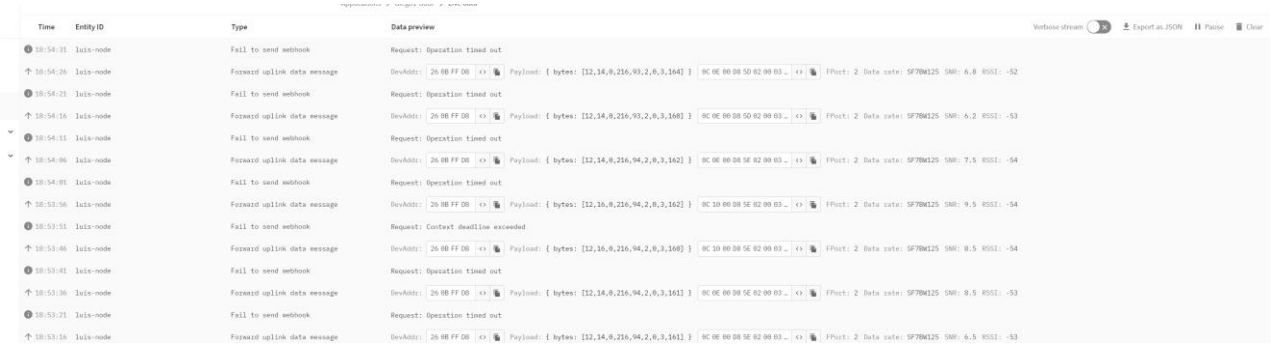


Ilustración 125: Envío de paquete de datos del dispositivo PCB

El primer enlace permite una visualización completa de los datos, esta se ha realizado bajo la plataforma basic-node. Su visualización es la siguiente dispuesta:

Ilustración 126: Visualización a través de node-RED

El segundo enlace ya aportado por el tutor y la comunidad myopentech permite una visualización algo más profesional si se permite la expresión, pero en mi caso no tengo acceso ni permiso para su modificación. Se muestra a continuación cómo se visualiza esta plataforma:

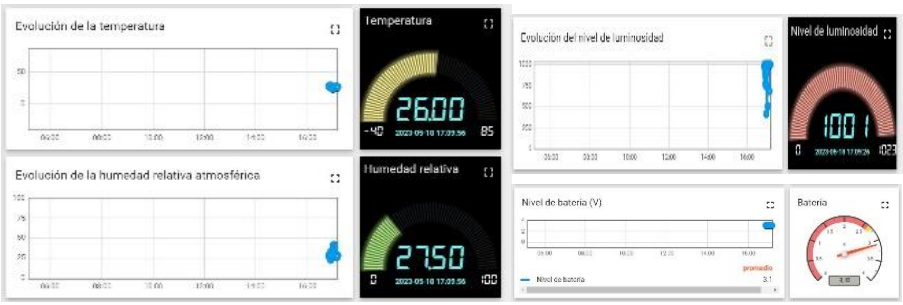


Ilustración 127: Visualización a través de myopentech

Por último, se propone un diseño realizado en Autodesk Inventor de una carcasa impresa en 3D en una impresora Ender 3 para que el dispositivo pueda incorporarse a una plataforma que se vende con los paquetes de aprendizaje de Arduino y que permite la

incorporación de una protoboard gracias al diseño de un esqueleto con un idéntico posicionamiento de los tornillos de métrica 3 con respecto a los de la placa Arduino UNO.

Los diseños con medidas se incorporan en el Anexo y en las presentes ilustraciones se muestra el resultado de la impresión:

Fotografía 34: Carcasa impresa en 3D con impresora Ender 3

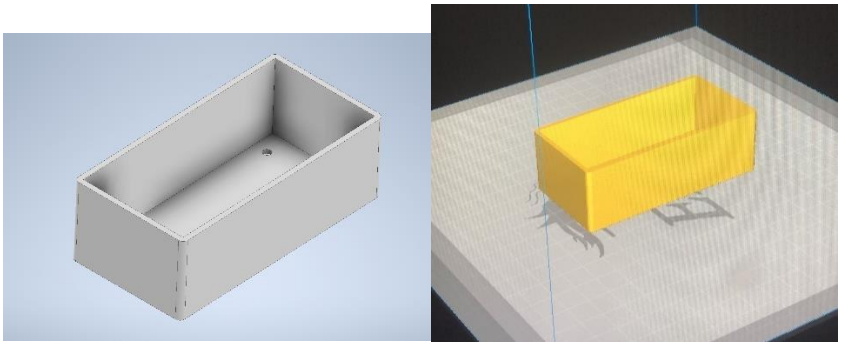


Ilustración128: Carcasa en Autodesk Inventor 2024 y Ultimaker Cura

6. RESULTADOS

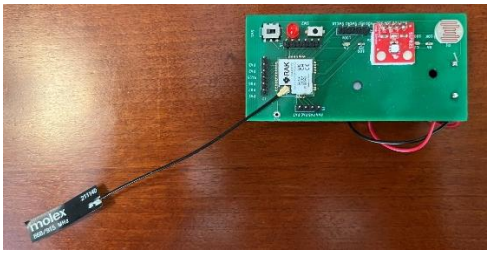
El análisis de resultados del presente proyecto es altamente positivo como se ha podido comprobar a lo largo del presente proyecto. Bien es cierto que podrían añadirse mejoras a lo largo del tiempo, pero eso es lo realmente remarcable de este dispositivo. El hecho de ofrecer una respuesta al problema ha planteado para el proyecto, y además añadir un valor de posibilidad de modificación/incorporación/mejora del dispositivo por parte del usuario.

Se han conseguido alcanzar todos y cada uno de los objetivos propuestos en el trabajo. Aunque las evidencias han sido demostradas a lo largo del apartado 5 del presente documento, adjuntaré de nuevo algunas ilustraciones y fotografías que ilustren el alcance de los consecuentes resultados.

- Creación de un dispositivo de recepción de datos mediante LoRa con un Gateway.

Fotografía 35: Gateway con Raspberry Pi y concentrador ic880a

- Diseño, testeo y creación de placa PCB capacitada para la transmisión de datos LoRa mediante RAK3172.



Fotografía 36: Dispositivo final creado

- Programación y monitorización de sensores de temperatura, humedad, luminosidad y nivel de batería.

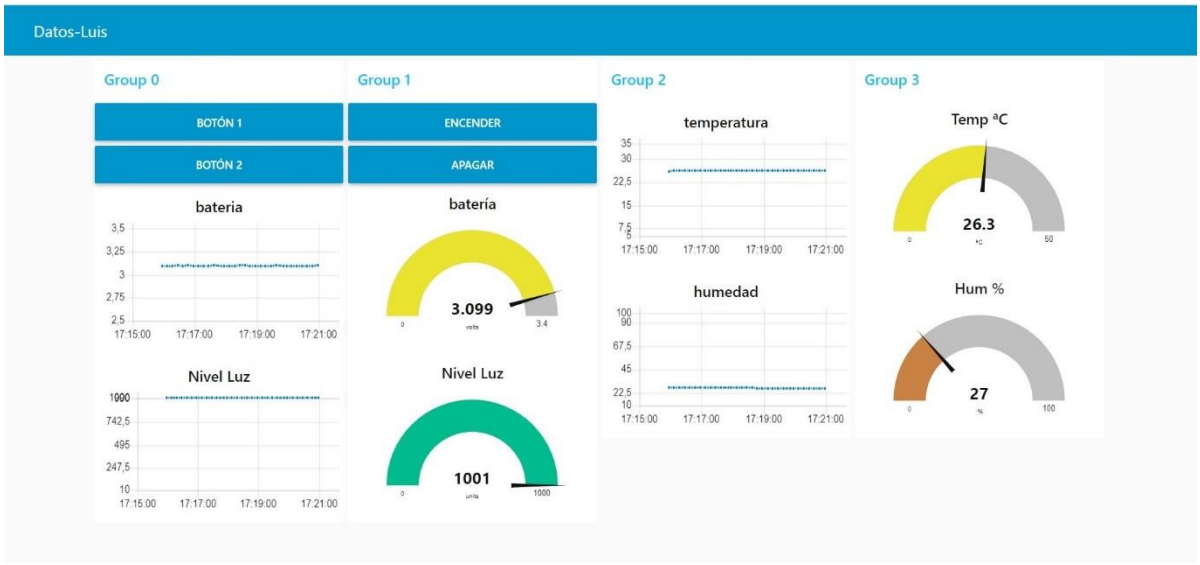
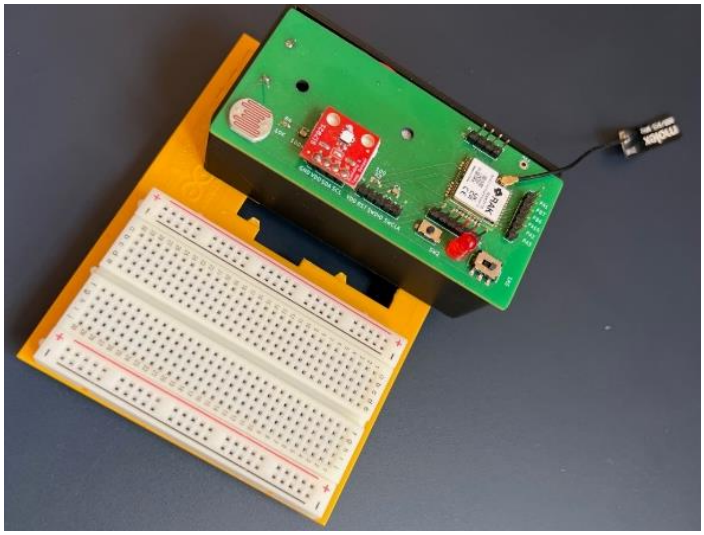


Ilustración 129: Diseños para la visualización de datos en dispositivos web

- Incorporación de una carcasa realizada en impresión 3D que permita la adaptación y fácil programación del dispositivo creado.



Fotografía 37: Caja impresa en 3D para implementar el dispositivo con protoboard

7. IMPLICACIONES ÉTICAS E IMPACTO SOCIAL

Desde el punto de vista ético y social el presente proyecto posee numerosas características de ambas tipologías. Por una parte, aunque luego sea desarrollado de una manera más detallada, el proyecto puede suponer un impacto positivo a nivel medioambiental gracias a la incorporación de este tipo de sensores en todos los barrios de Madrid, permitiendo así un mejor análisis y estudio del cambio climático a lo largo de los tiempos. Por otra parte, el proyecto realizado a gran escala podría presentar un avance para el desarrollo de aquellos continentes más subdesarrollados, permitiendo la incorporación del sensor a sus cultivos para la mejora de estos.

En primer lugar, se hablará sobre el desarrollo positivo a nivel medioambiental. Como bien estudiamos a lo largo de la carrera en la asignatura eficiencia energética y sostenibilidad se presentan 17 objetivos de desarrollo sostenible para la agenda 2030. El proyecto desarrollado en este trabajo puede contribuir a la mejora e incorporación de los siguientes objetivos ODS:

- **Industria innovación e infraestructura:** La aportación con respecto a este objetivo está íntimamente ligada con el fomento de la innovación y el progreso tecnológico. Proyectos como este presentan la posibilidad de incrementar el interés de las economías extranjeras al reflejar una comunidad cada vez más enfocada en el desarrollo de nuevas tecnologías capaces de ser modificadas e implementadas por usuarios desde sus casas como la descrita con este sensor de temperatura, humedad y luminosidad conectado a través de LoRa.
- **Ciudades y comunidades sostenibles:** La población en las ciudades se verá incrementada hasta en un 60% en el año 2030. Es fundamental el cuidado de estas ciudades por ello la implementación de dispositivos como el descrito en el trabajo tal vez pueda lograr a futuros una construcción de equipos LoRa encargados de monitorizar e informar sobre el estado de cada uno de los edificios y zonas urbanas de las ciudades. Estos datos podrían ser empleados para futuros desarrollos de edificios adaptables a las condiciones climatológicas venideras. Incluso podrían alcanzarse niveles de monitorización por sensores en casas que permitiesen una mayor gestión en cuanto a residuos u otros problemas graves de la actualidad comunitaria.

- **Acción por el clima:** El cambio climático es un hecho innegable hoy en día. El presente proyecto es un sensor de parámetros medioambientales por lo que su incorporación en las ciudades en tanto en cuanto a proliferación del control del cambio climático es obvia. Sin embargo, como ya se ha mencionado, el sensor puede incorporar otros sensores empleando los pines libres, hablamos de sensores que pueden influenciar en esta lucha contra el cambio climático ofreciendo datos sobre riqueza del aire de la zona en la que se instale, análisis de minerales de las tierras o niveles de radiación solar por metro cuadrado en las zonas de las ciudades deseadas.
- **Vida de ecosistemas terrestres:** El impacto del cambio climático no es solo observable en humanos, la fauna y flora de las ciudades también se está viendo tremendamente afectada con el paso de los años. Así pues, se podrían desarrollar dispositivos basados en la PCB descrita en este trabajo capacitados para monitorizar y alarmar a los ayuntamientos y autoridades pertinentes sobre aquellas zonas en las que se estén registrando datos alarmantes preprogramados sobre niveles de contaminación ya sea acústica, hídrica, radioactiva, lumínica etc.

Desde un punto de vista menos medioambiental, el proyecto puede presentar también un elevado impacto social. Como bien sabemos desde que tenemos prácticamente uso de razón, siempre han existido países menos desarrollados que no han tenido la misma suerte que nosotros por desgracia. Por una parte, este dispositivo de temperatura, humedad y luminosidad podría suponer un avance para ciertas partes del mundo más subdesarrolladas. Propongo dos ejemplos muy simples que bajo mi humilde opinión creo que podrían desempeñarse sin un elevado precio al respecto:

- **Proyecto educativo:** Obviamente esto son proyectos que podrían llevarse a cabo, reconozco que no hay un estudio tras ellos. Sin embargo, hay ciertas zonas en las que al menos hay un portátil por cada “x” miembros de la comunidad. Se podría fomentar la enseñanza y aprendizaje con este tipo de dispositivos implementando un plan educativo de un mes en países menos desarrollados en los que se enseñen las bases necesarias para que una vez transcurrido ese mes los niños o adultos que hayan asistido a él cuenten con diferentes prototipos a los que puedan añadir los sensores que deseen. Una vez transcurrido su diseño del prototipo tal vez podrían enviar o guardar el prototipo y ser llevado a cabo al año siguiente al volver para impartir más lecciones. Lógicamente esto serían una especie de voluntariado el cual no estaría remunerado, pero podría ser de gran ayuda para ciertas poblaciones con desigualdad de oportunidades.
- **Proyecto agrario:** Los datos recogidos por este dispositivo son de elevada importancia para los cultivos. Con la implementación de un Gateway capaz de transmitir datos por internet vía una tarjeta de red SIM o algo por el estilo podría dar cabida a un sistema de monitorización meteorológico que pudiese ayudar al cuidado y predicción de temporales no favorables en aquellos poblados de continentes menos desarrollados y que no poseen de ninguna otra fuente de alimentación que no sea su cultivo y ganadería.

8. CONCLUSIONES

Las conclusiones respecto al proyecto son las esperadas en todo momento. El hecho de que el desarrollo de nuevas tecnologías que permitan la mejora tecnológica y social de las ciudades del futuro está en auge es algo innegable. El internet de las cosas es un medio de elevada importancia y que tiene posibilidades infinitas siempre que se quiera invertir en su desarrollo e implementación.

Como hijo de un empresario relacionado con el mundo de la agricultura y ganadería (sector el cual está experimentando diariamente un avance y desarrollo sin precedentes en tanto en cuanto a incorporación de nuevas tecnologías de monitorización para cultivos y animales) me atrevo a concluir que el desarrollo de dispositivos como el dispuesto en el presente trabajo de fin de carrera serían de elevada importancia para el futuro agrícola-ganadero pero siempre y cuando se elimina la necesidad de abusar en cuanto a precios por parte de las empresas fabricantes de estos dispositivos.

El dispositivo realizado tiene una infinidad de modificaciones a nivel futuro, es mejorable partiendo de la base ya diseñada, creada y programada por mí. Es por esto por lo que afirmo que algunas empresas con más experiencia y capital dedicadas a fabricar este tipo de dispositivos llegan a aprovecharse del desconocimiento por parte del sector agrícola existente. Un sector que como podemos ver cada vez se está viendo más abandonado y obsoleto en cuanto a trabajadores que quieran dedicarse al mismo.

Mi proyecto aporta una base capaz de ampliarse fácilmente y que permitiría a cualquier persona interesada en este mundo formarse para construir el dispositivo que realmente quiera. Mi dispositivo puede presentar a futuros numerosas modificaciones y mejoras como por ejemplo reducción del tamaño de la PCB, definición de sensores más relevantes y por tanto eliminación de tiras de pines para conectarlas a protoboards, realización de una carcasa impresa en 3D, implementación de sensores más profesionales para su uso en agricultura y ganadería real, y por último acompañamiento de la PCB de un Gateway con conexión 4G que permitiese retransmitir datos desde cualquier sitio y sin necesidad de su conexión a un enchufe de la casa para establecer la conexión con el router más cercano.

9. OTROS MÉRITOS DEL PROYECTO

El proyecto ha conseguido numerosos méritos, a nivel personal pensaba que era imposible llevar a cabo esta solución en la vida real. Pensaba que con el diseño del dispositivo iba a ser suficiente dada su complejidad, sin embargo, el hecho de haber realizado un dispositivo capacitado para monitorizar humedad, temperatura, luminosidad y nivel de batería desde cero completamente y sin tecnología básica y fácil de encontrar en internet (como por ejemplo con Arduino) es en sí un mérito.

El dispositivo y su programación se pondrá a disposición pública a través de GitHub ya que seguramente a muchas otras personas pueda ayudarlas al igual que a mí en estos comienzos en el mundo del internet de las cosas desde cero.

Como he podido observar a lo largo del proyecto, esto es una comunidad emergente y que diariamente se hace más grande como he podido observar, este hecho reafirma aún más la necesidad de fomentar esta tecnología que realmente tiene un poder ingente en tanto en cuanto a mejorar las ciudades del futuro y en cuanto a fomentar el intelecto de los ingenieros/aficionados a la electrónica.

10. BIBLIOGRAFÍA

Arne Martin Holberg, AVR Project Manager, & Asmund Saetre, AVR Marketing Manager . (Febrero de 2006). *Atmel-nnovative Techniques for Extremely Low Power Consumption.* Obtenido de http://ww1.microchip.com/downloads/en/devicedoc/doc7903.pdf

Author, A. (13 de Mayo de 2018). *Comparativa de microcontroladores actuales.* Obtenido de https://nanopdf.com/download/comparativa-de-microcontroladores-actuales_pdf

CATSENSORS. (s.f.). *Aplicaciones de sensores LoRaWAN*. Obtenido de https://www.catsensors.com/es/lorawan/aplicaciones-sensores-lorawan

Dailywell Electronics CO. LTD. (s.f.). *¿Qué Medios De Interruptor De Palanca, Interruptor Basculante SPDT, DPDT Y 3PDT Están Representados? Si Solicita Un Interruptor De Palanca De 9 Pines, De Qué Tipo Es.* Obtenido de https://www.dailywell.com.tw/es/faq/Dailywell-FAQ-09.html

Dset Energy. (s.f.). *Tecnología Sigfox: Internet de las cosas*. Recuperado el 23 de Febrero de 2023, de http://productos-iot.com/sigfox-3/#:~:text=Sigfox%20es%20una%20red%20de,energ%C3%ADa%20de%20los%20dispositivos%20conectados.

Electrónica Embajadores. (s.f.). *Electrónica Embajadores*. Obtenido de https://www.electronicaembajadores.com/es

Esteban, E. V. (s.f.). *Lenguaje C.* Obtenido de Universidad de VIgo: https://informatica.uv.es/estguia/ATD/apuntes/laboratorio/Lenguaje-C.pdf

Fernández, R. (27 de Septiembre de 2022). *Statista*. Obtenido de https://es.statista.com/estadisticas/517654/prevision-de-la-evolucion-de-los-dispositivos-conectados-para-el-internet-de-las-cosas-en-el-mundo/

Fraile Mora, J. (2012). *Circuitos eléctricos.* Madrid: Pearson.

Fuentes, L. R., & Gachet Páez, D. (2023). *Nodo-Luis*. Obtenido de http://pruebanode.ddns.net:1880/

Fuentes, L. R., & Gachet Páez, D. (2023). *Nodo-Luis*. Obtenido de https://my.iotopentech.io/dashboards/b9e67620-aca2-11eb-a6e5-e986ddba53ce?state=W3sicGFyYW1zIjp7ImVudGl0eUlkIjp7ImVudGl0eVR5cGUiOiJERVZJQ0UiLCJpZCI6ImE1OWE0Y2MwLTFjOGUtMTFlZC1iMzY1LTliMGJlMzkxNzAzYSJ9LCJlbnRpdHlOYW1lIjoibm9kby1sdWlzIn19XQ%3D%3D

González, A. G. (15 de Octubre de 2014). *panamahitek*. Obtenido de https://panamahitek.com/como-funciona-el-protocolo-spi/

Hetpro. (16 de Noviembre de 2022). *Comunicación serial USART*. Obtenido de https://hetpro-store.com/TUTORIALES/comunicacion-serial/#:~:text=La%20USART%20o%20Universal%20Synchronous,Operaci%C3%B3n%20s%C3%ADncrona%20o%20as%C3%ADncrona

jh83. (23 de Agosto de 2022). *GitHub*. Obtenido de https://github.com/jh83/RAK3172-BME680

JLCPCB. (s.f.). *JLC PCB Homepage*. Obtenido de https://jlcpcb.com/

Kicad EDA. (s.f.). *Kicad*. Obtenido de https://www.kicad.org/

Martínez, J. (3 de Febrero de 2019). PIC vs AVR, Comparación objetiva. Obtenido de https://www.youtube.com/watch?v=R6Ajwi9SHmE

MobaXterm. (s.f.). *MobaXterm features*. Obtenido de https://mobaxterm.mobatek.net/

Organización de las Naciones Unidas. (2030). *Objetivos de desarrollo sostenible*. Obtenido de https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/

PRISMAB. (s.f.). *Prismab Link*. Obtenido de https://prismab.com/tecnologia/link/

Ramírez Juárez, A. R. (22 de Octubre de 2019). Obtenido de https://d1wqtxts1xzle7.cloudfront.net/63362670/Leyes_de_Kirchhoff20200519-84014-3f6yxn-libre.pdf?1589910425=&response-content-disposition=inline%3B+filename%3DLeyes_de_Kirchhoff.pdf&Expires=1677693970&Signature=TUJKrq8JY~KLH6wx3Hpr9E264blbIHBEkk5odzn~46oq

Redacción Interempresas. (18 de Noviembre de 2022). Lanzarote cuenta ya con una red pública LoRa. *Lanzarote cuenta ya con una red pública LoRa*, pág. 1.

RIVAS, C. I. (05 de Julios de 2015). *Revista Tecnología e Innovación*. Obtenido de https://www.ecorfan.org/bolivia/researchjournals/Tecnologia_e_innovacion/vol2num5/Tecnologia_e_Innovacion_Vol2_Num5_6.pdf

Ruíz Vázquez, E., Arbelaitz Gallego, O., Etxeberria Uztarroz, I., & Ibarra Lasa, A. (2004). *Análisis básico de circuitos eléctricos y electrónicos*. Madrid: Pearson.

Secmotic smart inside. (2019). *Secmotic*. Obtenido de Digital Port, plataforma de servicios para el entorno portuario: https://secmotic.com/project/digital-port/

SEMTECH. (s.f.). *LoRa (PHY)*. Obtenido de https://www.semtech.com/lora/what-is-lora

STMicroelectronics. (s.f.). *STM32CubeIDE Product Overview*. Obtenido de https://www.st.com/en/development-tools/stm32cubeide.html

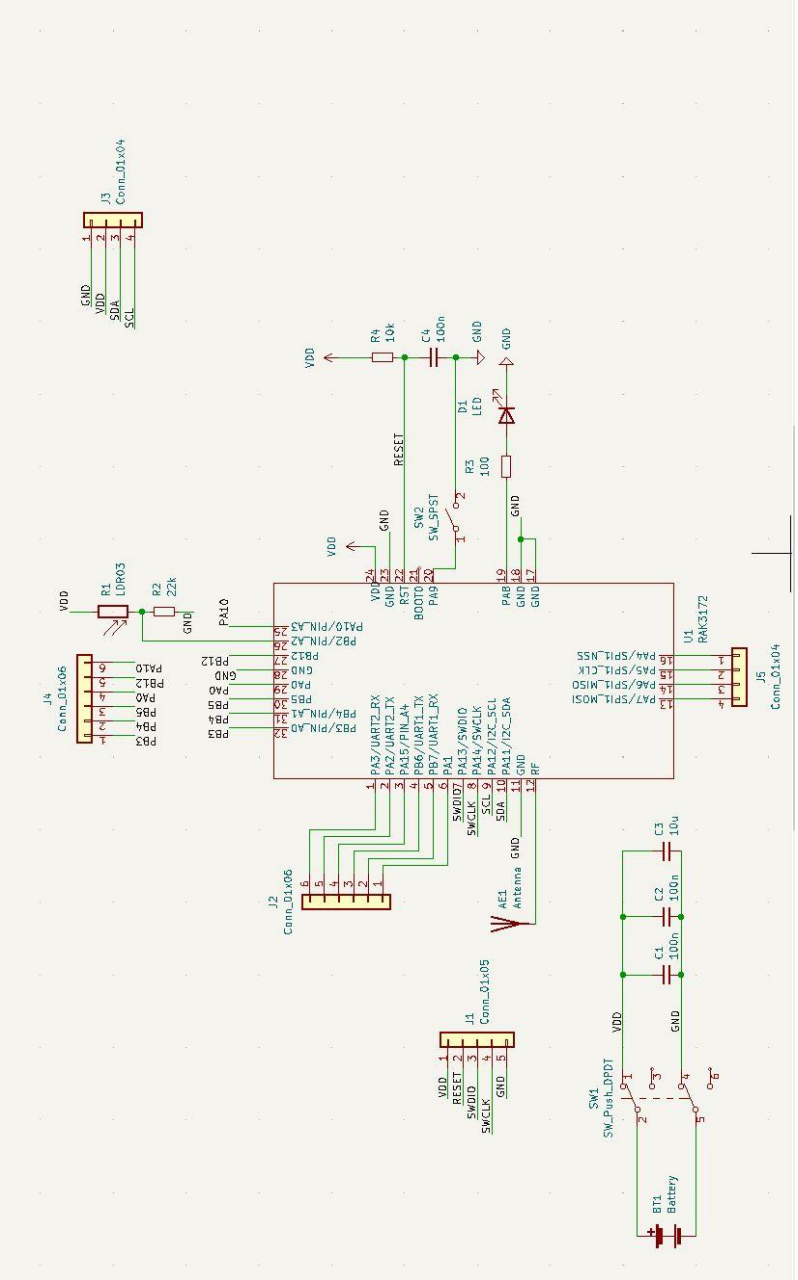
STMicroelectronics. (s.f.). *STM32CUbeMx Product Overview*. Obtenido de https://www.st.com/en/development-tools/stm32cubemx.html

STMicroelectronics STM32WLE5CCU6. (s.f.). *Sub-GHz Wireless Microcontrollers. Arm Cortex-M4 @48 MHz with 256 Kbytes of Flash memory, 64 Kbytes of SRAM. LoRa, (G)FSK, (G)MSK, BPSK modulations. AES 256-bit. Multiprotocol System-on-Chip*. Obtenido de https://www.st.com/en/microcontrollers-microprocessors/stm32wle5cc.html

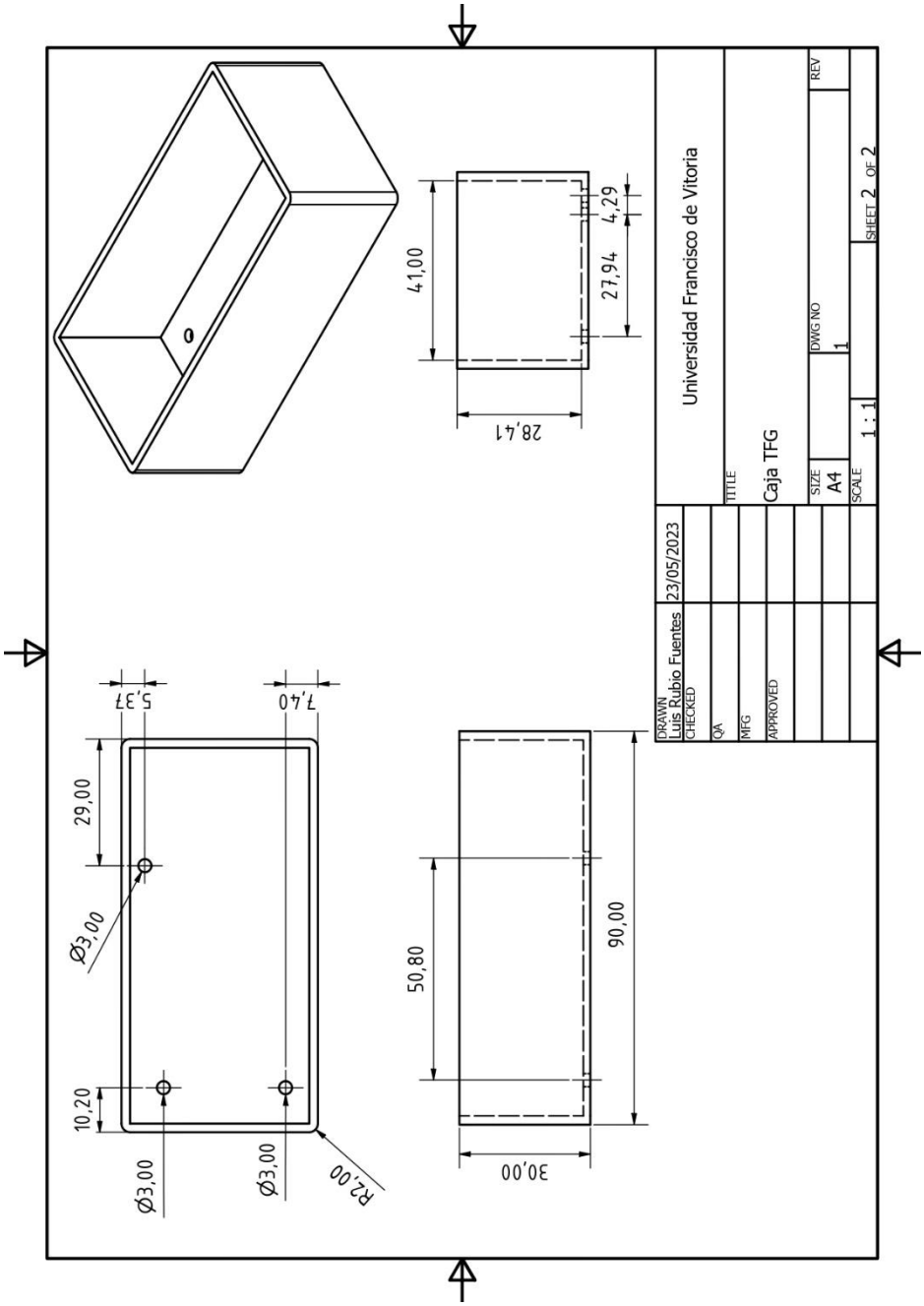
Stoyanov, Y. (s.f.). *Open Drain Output vs. Push-Pull Output*. Obtenido de Open4Tech: https://open4tech.com/open-drain-output-vs-push-pull-output/

Torres, D. H. (Marzo de 2018). *I2C – Puerto, Introducción, trama y protocolo*. Obtenido de hetpro-store: https://hetpro-store.com/TUTORIALES/i2c/

ANEXO A: ESQUEMÁTICO DEL PROTOTIPO PCB



ANEXO B: PLANO DE LA CAJA IMPRESA EN 3D



ANEXO C: CÓDIGO DEL PROGRAMA FINAL

main.c

```c
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2023 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "adc.h"
#include "dma.h"
#include "i2c.h"
#include "app_lorawan.h"
#include "rtc.h"
#include "usart.h"
#include "gpio.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include <stdlib.h>
#include <stdio.h>
//ACORDARSE el se-identity
/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */

/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/

/* USER CODE BEGIN PV */
uint8_t I2C_SLAVE_ADDR = 0x40 <<1;
extern I2C_HandleTypeDef hi2c2;
uint16_t intemp=0;
uint8_t inhum=0;


/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
```

```
uint8_t temper[25];
uint8_t humed[25];
uint8_t temper1[20];
uint8_t humed1[20];

uint8_t data[3];
uint8_t da[1] = {0xFE};
uint8_t read[2]= {0xE3,0xE5};
uint8_t data1[3];
float temp,tempp;
float hu,humid;
uint16_t sum_temp,sum_humid;

void si7021_init()
{
HAL_I2C_Master_Transmit(&hi2c2,0x80,da,1,100);
    HAL_Delay(20);
}

void get_si7021(float *temp,float *humid)
{


        HAL_I2C_Master_Transmit(&hi2c2,I2C_SLAVE_ADDR,&read[0],1,100);
        HAL_I2C_Master_Receive(&hi2c2,I2C_SLAVE_ADDR,data,3,1000);
        sum_temp = (data[0]<<8) | data[1];
        tempp   = sum_temp;
        *temp = (-46.85 + (175.72*(tempp/65536)));
        HAL_Delay(20);
        HAL_I2C_Master_Transmit(&hi2c2,I2C_SLAVE_ADDR,&read[1],1,100);
        HAL_I2C_Master_Receive(&hi2c2,I2C_SLAVE_ADDR,data1,3,1000);
        sum_humid= (data1[0]<<8) | data1[1];
        hu= sum_humid;
        *humid = (-6+(125*(hu/65536)));
        HAL_Delay(20);
    }




/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

```

```
140   /* Initialize all configured peripherals */
141   MX_GPIO_Init();
142   MX_DMA_Init();
143   MX_ADC_Init();
144   MX_RTC_Init();
145   MX_USART1_UART_Init();
146   MX_LoRaWAN_Init();
147   MX_I2C2_Init();
148   /* USER CODE BEGIN 2 */
149   si7021_init();
150   /* USER CODE END 2 */
151
152   /* Infinite loop */
153   /* USER CODE BEGIN WHILE */
154
155
156   //HAL_I2C_Master_Transmit_IT (I2C_HandleTypeDef *hi2c, uint16_t DevAddress, uint8_t *pData, uint16_t Size)
157   //HAL_I2C_Master_Receive_IT (I2C_HandleTypeDef *hi2c, uint16_t DevAddress, uint8_t *pData, uint16_t Size)
158   //HAL_I2C_Master_Transmit_DMA(I2C_HandleTypeDef *hi2c, uint16_t DevAddress, uint8_t *pData, uint16_t Size)
159
160
161   while (1)
162   {
163
164       MX_I2C2_Init();
165       get_si7021(&temp,&humid);
166
167        sprintf((char *)temper,"temperatura  %4.2f\r\n", temp);
168          HAL_UART_Transmit(&huart1,temper,22,HAL_MAX_DELAY);
169          HAL_Delay(100);
170         sprintf((char *)humed,"humedad  %4.2f\r\n", humid);
171         HAL_UART_Transmit(&huart1,humed,22,HAL_MAX_DELAY);
172         HAL_Delay(100);
173
174          intemp = temp*10.0;
175          inhum = humid*2.0; //PARA EL GRAFICO OPENTECH
176           sprintf((char *)temper1,"temp     %u\r\n", intemp);
177          HAL_UART_Transmit(&huart1,temper1,17,HAL_MAX_DELAY);
178         HAL_Delay(100);
179         sprintf((char *)humed1,"hum      %u\r\n", inhum);
180         HAL_UART_Transmit(&huart1,humed1,17,HAL_MAX_DELAY);
181         HAL_Delay(100);
182
183
184         MX_LoRaWAN_Process();
185
186
187     /* USER CODE END WHILE */
188     MX_LoRaWAN_Process();
189
190     /* USER CODE BEGIN 3 */
191   }
192   /* USER CODE END 3 */
193 }
194
195 /**
196   * @brief System Clock Configuration
197   * @retval None
198   */
199 void SystemClock_Config(void)
200 {
201   RCC_OscInitTypeDef RCC_OscInitStruct = {0};
202   RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
203
204   /** Configure LSE Drive Capability
205   */
206   HAL_PWR_EnableBkUpAccess();
207   __HAL_RCC_LSEDRIVE_CONFIG(RCC_LSEDRIVE_LOW);
208
209   /** Configure the main internal regulator output voltage
```

```
210   */
211   __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);
212
213   /** Initializes the CPU, AHB and APB buses clocks
214   */
215   RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_LSE|RCC_OSCILLATORTYPE_MSI;
216   RCC_OscInitStruct.LSEState = RCC_LSE_ON;
217   RCC_OscInitStruct.MSIState = RCC_MSI_ON;
218   RCC_OscInitStruct.MSICalibrationValue = RCC_MSICALIBRATION_DEFAULT;
219   RCC_OscInitStruct.MSIClockRange = RCC_MSIRANGE_11;
220   RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
221   if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
222   {
223     Error_Handler();
224   }
225
226   /** Configure the SYSCLKSource, HCLK, PCLK1 and PCLK2 clocks dividers
227   */
228   RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK3|RCC_CLOCKTYPE_HCLK
229                               |RCC_CLOCKTYPE_SYSCLK|RCC_CLOCKTYPE_PCLK1
230                               |RCC_CLOCKTYPE_PCLK2;
231   RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_MSI;
232   RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
233   RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
234   RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
235   RCC_ClkInitStruct.AHBCLK3Divider = RCC_SYSCLK_DIV1;
236
237   if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
238   {
239     Error_Handler();
240   }
241 }
242
243 /* USER CODE BEGIN 4 */
244
245 /* USER CODE END 4 */
246
247 /**
248   * @brief  This function is executed in case of error occurrence.
249   * @retval None
250   */
251 void Error_Handler(void)
252 {
253   /* USER CODE BEGIN Error_Handler_Debug */
254   /* User can add his own implementation to report the HAL error return state */
255   __disable_irq();
256   while (1)
257   {
258   }
259   /* USER CODE END Error_Handler_Debug */
260 }
261
262 #ifdef  USE_FULL_ASSERT
263 /**
264   * @brief  Reports the name of the source file and the source line number
265   *         where the assert_param error has occurred.
266   * @param  file: pointer to the source file name
267   * @param  line: assert_param error line source number
268   * @retval None
269   */
270 void assert_failed(uint8_t *file, uint32_t line)
271 {
272   /* USER CODE BEGIN 6 */
273   /* User can add his own implementation to report the file name and line number,
274      ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
275   /* USER CODE END 6 */
276 }
277 #endif /* USE_FULL_ASSERT */
```

```c
#ifndef __SI7021_H
#define __SI7021_H
/*
 Measure Relative Humidity, Hold Master Mode        0xE5
Measure Relative Humidity, No Hold Master Mode      0xF5
Measure Temperature, Hold Master Mode               0xE3
Measure Temperature, No Hold Master Mode            0xF3
Read Temperature Value from Previous RH Measurement 0xE0
Reset                                               0xFE
Write RH/T User Register 1                          0xE6
Read RH/T User Register 1                           0xE7
Write Heater Control Register                       0x51
Read Heater Control Register                        0x11
Read Electronic ID 1st Byte                         0xFA 0x0F
Read Electronic ID 2nd Byte                         0xFC 0xC9
Read Firmware Revision 0x84                         0xB8
 */

extern I2C_HandleTypeDef hi2c2;
uint8_t data[3]; /* Almacena en 3 elementos los datos recibidos del sensor*/
uint8_t da[1] = {0xFE}; /*Se encarga de activar el modo reset*/
uint8_t read[2]= {0xE3,0xE5};/*Se encarga de activar el modo measure temperature y humedad. Hold master mode*/
uint8_t data1[3];/* Almacena en 3 elementos los datos recibidos del sensor*/
float temp,tempp;
float hu,humid;
uint16_t sum_temp;
uint16_t sum_humid;

void si7021_init()
{
HAL_I2C_Master_Transmit(&hi2c2,0x80,da,1,100); /* Envía un byte de valor 0xFE al sensor para inicializar*/
    HAL_Delay(20);
}
void get_si7021(float *temp,float *humid)
{


        HAL_I2C_Master_Transmit(&hi2c2,0x80,&read[0],1,100);/* Envía la lectura de datos de temperatura*/
        HAL_I2C_Master_Receive(&hi2c2,0x80,data,3,1000);/* Recibe la lectura de datos de temperatura*/
        sum_temp = (data[0]<<8) | data[1]; /* ALmcenamos los dos bytes en una variable de 16 bits*/
        tempp   = sum_temp;
        *temp = (-46.85 + (175.72*(tempp/65536)));
        HAL_Delay(20);/*Parada de 20 ms para estabilizar el sensor*/
        HAL_I2C_Master_Transmit(&hi2c2,0x80,&read[1],1,100);/* Envía la lectura de datos de humedad*/
        HAL_I2C_Master_Receive(&hi2c2,0x80,data1,3,1000);/* Recibe la lectura de datos de humedad*/
        sum_humid= (data1[0]<<8) | data1[1]; /* Almacenamos los dos bytes en una variable de 16 bits*/
        hu= sum_humid;
        *humid = (-6+(125*(hu/65536)));
        HAL_Delay(20);
    }

#endif
```

main.c | si7021.h | lora_app.c × | radio_board_if.c | radio_board_if.h

```c
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file    lora_app.c
  * @author  MCD Application Team
  * @brief   Application of the LRWAN Middleware
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2023 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */

/* Includes ------------------------------------------------------------------*/
#include "platform.h"
#include "sys_app.h"
#include "lora_app.h"
#include "stm32_seq.h"
#include "stm32_timer.h"
#include "utilities_def.h"
#include "app_version.h"
#include "lorawan_version.h"
#include "subghz_phy_version.h"
#include "lora_info.h"
#include "LmHandler.h"
#include "adc_if.h"
#include "CayenneLpp.h"
#include "sys_sensors.h"
#include "flash_if.h"

/* USER CODE BEGIN Includes */

/* USER CODE END Includes */
```

```
226   */
227 static ActivationType_t ActivationType = LORAWAN_DEFAULT_ACTIVATION_TYPE;
228
229 /**
230   * @brief LoRaWAN force rejoin even if the NVM context is restored
231   */
232 static bool ForceRejoin = LORAWAN_FORCE_REJOIN_AT_BOOT;
233
234 /**
235   * @brief LoRaWAN handler Callbacks
236   */
237 static LmHandlerCallbacks_t LmHandlerCallbacks =
238 {
239   .GetBatteryLevel =              GetBatteryLevel,
240   .GetTemperature =               GetTemperatureLevel,
241   .GetUniqueId =                  GetUniqueId,
242   .GetDevAddr =                   GetDevAddr,
243   .OnRestoreContextRequest =      OnRestoreContextRequest,
244   .OnStoreContextRequest =        OnStoreContextRequest,
245   .OnMacProcess =                 OnMacProcessNotify,
246   .OnNvmDataChange =              OnNvmDataChange,
247   .OnJoinRequest =                OnJoinRequest,
248   .OnTxData =                     OnTxData,
249   .OnRxData =                     OnRxData,
250   .OnBeaconStatusChange =         OnBeaconStatusChange,
251   .OnSysTimeUpdate =              OnSysTimeUpdate,
252   .OnClassChange =                OnClassChange,
253   .OnTxPeriodicityChanged =       OnTxPeriodicityChanged,
254   .OnTxFrameCtrlChanged =         OnTxFrameCtrlChanged,
255   .OnPingSlotPeriodicityChanged = OnPingSlotPeriodicityChanged,
256   .OnSystemReset =                OnSystemReset,
257 };
258
259 /**
260   * @brief LoRaWAN handler parameters
261   */
262 static LmHandlerParams_t LmHandlerParams =
263 {
264   .ActiveRegion =             ACTIVE_REGION,
265   .DefaultClass =             LORAWAN_DEFAULT_CLASS,
266   .AdrEnable =                LORAWAN_ADR_STATE,
267   .IsTxConfirmed =            LORAWAN_DEFAULT_CONFIRMED_MSG_STATE,
268   .TxDatarate =               LORAWAN_DEFAULT_DATA_RATE,
269   .TxPower =                  LORAWAN_DEFAULT_TX_POWER,
270   .PingSlotPeriodicity =      LORAWAN_DEFAULT_PING_SLOT_PERIODICITY,
271   .RxBCTimeout =              LORAWAN_DEFAULT_CLASS_B_C_RESP_TIMEOUT
272 };
273
274 /**
275   * @brief Type of Event to generate application Tx
276   */
277 static TxEventType_t EventType = TX_ON_TIMER;
278
279 /**
280   * @brief Timer to handle the application Tx
281   */
282 static UTIL_TIMER_Object_t TxTimer;
283
284 /**
285   * @brief Tx Timer period
286   */
287 static UTIL_TIMER_Time_t TxPeriodicity = APP_TX_DUTYCYCLE;
288
289 /**
290   * @brief Join Timer period
291   */
292 static UTIL_TIMER_Object_t StopJoinTimer;
293
294 /* USER CODE BEGIN PV */
```

```c
306void LoRaWAN_Init(void)
307{
308  /* USER CODE BEGIN LoRaWAN_Init_LV */
309  /* USER CODE END LoRaWAN_Init_LV */
310  /* USER CODE BEGIN LoRaWAN_Init_1 */
311  /* USER CODE END LoRaWAN_Init_1 */
312
313  UTIL_TIMER_Create(&StopJoinTimer, JOIN_TIME, UTIL_TIMER_ONESHOT, OnStopJoinTimerEvent, NULL);
314
315  UTIL_SEQ_RegTask((1 << CFG_SEQ_Task_LmHandlerProcess), UTIL_SEQ_RFU, LmHandlerProcess);
316
317  UTIL_SEQ_RegTask((1 << CFG_SEQ_Task_LoRaSendOnTxTimerOrButtonEvent), UTIL_SEQ_RFU, SendTxData);
318  UTIL_SEQ_RegTask((1 << CFG_SEQ_Task_LoRaStoreContextEvent), UTIL_SEQ_RFU, StoreContext);
319  UTIL_SEQ_RegTask((1 << CFG_SEQ_Task_LoRaStopJoinEvent), UTIL_SEQ_RFU, StopJoin);
320
321  /* Init Info table used by LmHandler*/
322  LoraInfo_Init();
323
324  /* Init the Lora Stack*/
325  LmHandlerInit(&LmHandlerCallbacks, APP_VERSION);
326
327  LmHandlerConfigure(&LmHandlerParams);
328
329  /* USER CODE BEGIN LoRaWAN_Init_2 */
330  /* USER CODE END LoRaWAN_Init_2 */
331
332  LmHandlerJoin(ActivationType, ForceRejoin);
333
334  if (EventType == TX_ON_TIMER)
335  {
336    /* send every time timer elapses */
337    UTIL_TIMER_Create(&TxTimer, TxPeriodicity, UTIL_TIMER_ONESHOT, OnTxTimerEvent, NULL);
338    UTIL_TIMER_Start(&TxTimer);
339  }
340  else
341  {
342    /* USER CODE BEGIN LoRaWAN_Init_3 */
343    /* USER CODE END LoRaWAN_Init_3 */
344  }
345
346  /* USER CODE BEGIN LoRaWAN_Init_Last */
347  /* USER CODE END LoRaWAN_Init_Last */
348}
349
350/* USER CODE BEGIN PB_Callbacks */
351#if 0 /* User should remove the #if 0 statement and adapt the below code according with his needs*/
352void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
353{
354  switch (GPIO_Pin)
355  {
356    case  BUT1_Pin:
357      /* Note: when "EventType == TX_ON_TIMER" this GPIO is not initialized */
358      if (EventType == TX_ON_EVENT)
359      {
360        UTIL_SEQ_SetTask((1 << CFG_SEQ_Task_LoRaSendOnTxTimerOrButtonEvent), CFG_SEQ_Prio_0);
361      }
362      break;
363    case  BUT2_Pin:
364      UTIL_SEQ_SetTask((1 << CFG_SEQ_Task_LoRaStopJoinEvent), CFG_SEQ_Prio_0);
365      break;
366    case  BUT3_Pin:
367      UTIL_SEQ_SetTask((1 << CFG_SEQ_Task_LoRaStoreContextEvent), CFG_SEQ_Prio_0);
368      break;
369    default:
370      break;
371  }
372}
373#endif
374
```

```c
static void OnRxData(LmHandlerAppData_t *appData, LmHandlerRxParams_t *params)
{
  /* USER CODE BEGIN OnRxData_1 */
    uint8_t RxPort = 0;
        if (params != NULL) {
            if (params->IsMcpsIndication) {/*Comprueba si hay información en el buffer de entrada*/
                if (appData != NULL) {
                    RxPort = appData->Port;
                    if (appData->Buffer != NULL) {
                        if (RxPort == 99) {
                            switch (appData->Buffer[0]) { /*Si hay información de entrada y el puerto es el 99 empiezo a leer los datos que han llegado*/
                            case 0x06:
                                //Encender/apagar LED
                                if (appData->Buffer[1] == 0x01) {
                                    if (appData->Buffer[2] == 0x01) {
                                        //Encendemos el LED
                                        HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin,
                                                GPIO_PIN_SET);
                                    } else if (appData->Buffer[2] == 0x00) {
                                        //Apagamos el LED
                                        HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin,
                                                GPIO_PIN_RESET);
                                    }
                                }
                                break;
                            }
                        }
                    }
                }
            }
        }
  /* USER CODE END OnRxData_1 */
}

static void SendTxData(void)
{
  /* USER CODE BEGIN SendTxData_1 */


    uint16_t batteryLevel = SYS_GetBatteryLevel(); /*Almacenamos la tensión en mV en un entero de 16 bits*/
    uint16_t LumenLevel = (float)GetLumenLevel()*1023.0/4095.0;
    //SOLO PARA EL GTAFICO OPENTECH

        LmHandlerErrorStatus_t status = LORAMAC_HANDLER_ERROR;
        UTIL_TIMER_Time_t nextTxIn = 0;
        AppData.Port = LORAWAN_USER_APP_PORT;
        AppData.Buffer[0] = (uint8_t) ((batteryLevel >> 8) & 0xFF);/*Dividimos en dos bytes para su envío*/
        AppData.Buffer[1] = (uint8_t) (batteryLevel & 0xFF);
        AppData.Buffer[2] = (uint8_t) ((intemp >> 8) & 0xFF);
        AppData.Buffer[3] = (uint8_t) (intemp & 0xFF);
        AppData.Buffer[4] = (uint8_t) ((inhum) & 0xFF);
        AppData.Buffer[5] = (uint8_t) ((512 >> 8) & 0xFF); //HUMED SUELO
        AppData.Buffer[6] = (uint8_t) (512 & 0xFF);
        AppData.Buffer[7] = (uint8_t) ((LumenLevel >> 8) & 0xFF);
        AppData.Buffer[8] = (uint8_t) (LumenLevel & 0xFF);

        AppData.BufferSize = 9; //SI NO NO RECIBE OPENTECH
        status = LmHandlerSend(&AppData, LmHandlerParams.IsTxConfirmed, false); /*Permite enviar el mensaje de forma deseada si está en true*/

        if (LORAMAC_HANDLER_SUCCESS == status) {
            APP_LOG(TS_ON, VLEVEL_L, "SEND REQUEST\r\n");
        } else if (LORAMAC_HANDLER_DUTYCYCLE_RESTRICTED == status) {
            nextTxIn = LmHandlerGetDutyCycleWaitTime(); /*Calcula cuánto tiempo tiene que esperar para enviar el mensaje*/
            if (nextTxIn > 0) {
                APP_LOG(TS_ON, VLEVEL_L, "Next Tx in  : ~%d second(s)\r\n",
                        (nextTxIn / 1000));
            }
        }

        if (EventType == TX_ON_TIMER) {
            UTIL_TIMER_Stop(&TxTimer); /*Se adapta el timer a ese tiempo de espera para volver a intentar el envío*/
            UTIL_TIMER_SetPeriod(&TxTimer, MAX(nextTxIn, TxPeriodicity));
            UTIL_TIMER_Start(&TxTimer);
        }

        HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_SET);
        HAL_Delay(100);
        HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);
  /* USER CODE END SendTxData_1 */
}

static void OnTxTimerEvent(void *context)
{
  /* USER CODE BEGIN OnTxTimerEvent_1 */

  /* USER CODE END OnTxTimerEvent_1 */
  UTIL_SEQ_SetTask((1 << CFG_SEQ_Task_LoRaSendOnTxTimerOrButtonEvent), CFG_SEQ_Prio_0);

  /*Wait for next tx slot*/
  UTIL_TIMER_Start(&TxTimer);
  /* USER CODE BEGIN OnTxTimerEvent_2 */

  /* USER CODE END OnTxTimerEvent_2 */
}

/* USER CODE BEGIN PrFD_LedEvents */

/* USER CODE END PrFD_LedEvents */
```

main.c | si7021.h | *lora_app.c | radio_board_if.c × | radio_board_if.h

```c
  /* USER CODE BEGIN RBI_Init_1 */

  /* USER CODE END RBI_Init_1 */
#if defined(USE_BSP_DRIVER)
  /* Important note: BSP code is board dependent
   * STM32WL_Nucleo code can be found
   *       either in STM32CubeWL package under Drivers/BSP/STM32WLxx_Nucleo/
   *       or at https://github.com/STMicroelectronics/STM32CubeWL/tree/main/Drivers/BSP/STM32WLxx_Nucleo/
   * 1/ For User boards, the BSP/STM32WLxx_Nucleo/ directory can be copied and replaced in the project. The copy must then be updated depending:
   *       on board RF switch configuration (pin control, number of port etc)
   *       on TCXO configuration
   *       on DC/DC configuration
   *       on maximum output power that the board can deliver*/
  return BSP_RADIO_Init();
#else
  /* 2/ Or implement RBI_Init here */
  int32_t retcode = 0;
  /* USER CODE BEGIN RBI_Init_2 */
#warning user to provide its board code or to call his board driver functions
  GPIO_InitTypeDef  gpio_init_structure = {0};
        /* Configure the Radio Switch pin */
          gpio_init_structure.Pin   = RF_SW_CTRL1_PIN;
          gpio_init_structure.Mode  = GPIO_MODE_OUTPUT_PP;
          gpio_init_structure.Pull  = GPIO_NOPULL;
          gpio_init_structure.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
          HAL_GPIO_Init(RF_SW_CTRL1_GPIO_PORT, &gpio_init_structure);
          gpio_init_structure.Pin = RF_SW_CTRL2_PIN;
          HAL_GPIO_Init(RF_SW_CTRL2_GPIO_PORT, &gpio_init_structure);
          HAL_GPIO_WritePin(RF_SW_CTRL2_GPIO_PORT, RF_SW_CTRL2_PIN, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN, GPIO_PIN_RESET);
  /* USER CODE END RBI_Init_2 */
  return retcode;
#endif  /* USE_BSP_DRIVER  */
}

int32_t RBI_DeInit(void)
{
  /* USER CODE BEGIN RBI_DeInit_1 */

  /* USER CODE END RBI_DeInit_1 */
#if defined(USE_BSP_DRIVER)
  /* Important note: BSP code is board dependent
   * STM32WL_Nucleo code can be found
   *       either in STM32CubeWL package under Drivers/BSP/STM32WLxx_Nucleo/
   *       or at https://github.com/STMicroelectronics/STM32CubeWL/tree/main/Drivers/BSP/STM32WLxx_Nucleo/
   * 1/ For User boards, the BSP/STM32WLxx_Nucleo/ directory can be copied and replaced in the project. The copy must then be updated depending:
   *       on board RF switch configuration (pin control, number of port etc)
   *       on TCXO configuration
   *       on DC/DC configuration
   *       on maximum output power that the board can deliver*/
  return BSP_RADIO_DeInit();
#else
  /* 2/ Or implement RBI_DeInit here */
  int32_t retcode = 0;
  /* USER CODE BEGIN RBI_DeInit_2 */
#warning user to provide its board code or to call his board driver functions
  HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN, GPIO_PIN_RESET);
    HAL_GPIO_WritePin(RF_SW_CTRL2_GPIO_PORT, RF_SW_CTRL2_PIN, GPIO_PIN_RESET);
    /* DeInit the Radio Switch pin */
    HAL_GPIO_DeInit(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN);
    HAL_GPIO_DeInit(RF_SW_CTRL2_GPIO_PORT, RF_SW_CTRL2_PIN);
  /* USER CODE END RBI_DeInit_2 */
  return retcode;
#endif  /* USE_BSP_DRIVER */
}
```

```c
  /* USER CODE BEGIN RBI_ConfigRFSwitch_2 */
#warning user to provide its board code or to call his board driver functions
  switch (Config)
        {
          case RBI_SWITCH_OFF:
          {
            /* Turn off switch */
            //APP_LOG(TS_OFF, VLEVEL_M, "RF SWITCH: OFF\r\n");
            HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN, GPIO_PIN_RESET);
            HAL_GPIO_WritePin(RF_SW_CTRL2_GPIO_PORT, RF_SW_CTRL2_PIN, GPIO_PIN_RESET);
            break;
          }
          case RBI_SWITCH_RX:
          {
            /*Turns On in Rx Mode the RF Switch */
            //APP_LOG(TS_OFF, VLEVEL_M, "RF SWITCH: RX\r\n");
            HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN, GPIO_PIN_SET);
            HAL_GPIO_WritePin(RF_SW_CTRL2_GPIO_PORT, RF_SW_CTRL2_PIN, GPIO_PIN_RESET);
            break;
          }
          case RBI_SWITCH_RFO_LP:
          {
            /*Turns On in Tx Low Power the RF Switch */
            //APP_LOG(TS_OFF, VLEVEL_M, "RF SWITCH: LP\r\n");
            HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN, GPIO_PIN_SET);
            HAL_GPIO_WritePin(RF_SW_CTRL2_GPIO_PORT, RF_SW_CTRL2_PIN, GPIO_PIN_SET);
            break;
          }
          case RBI_SWITCH_RFO_HP:
          {
            /*Turns On in Tx High Power the RF Switch */
            //APP_LOG(TS_OFF, VLEVEL_M, "RF SWITCH: HP\r\n");
            HAL_GPIO_WritePin(RF_SW_CTRL1_GPIO_PORT, RF_SW_CTRL1_PIN, GPIO_PIN_RESET);
            HAL_GPIO_WritePin(RF_SW_CTRL2_GPIO_PORT, RF_SW_CTRL2_PIN, GPIO_PIN_SET);
            break;
          }
          default:
            break;
        }
  /* USER CODE END RBI_ConfigRFSwitch_2 */
  return retcode;
#endif  /* USE_BSP_DRIVER */
}
```

main.c | si7021.h | *lora_app.c | radio_board_if.c | radio_board_if.h

```c
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file    radio_board_if.h
  * @author  MCD Application Team
  * @brief   Header for Radio interface configuration
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2023 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */

/* Define to prevent recursive inclusion -------------------------------------*/
#ifndef RADIO_BOARD_IF_H
#define RADIO_BOARD_IF_H

#ifdef __cplusplus
extern "C" {
#endif

/* Includes ------------------------------------------------------------------*/
#include "platform.h"
/* USER CODE BEGIN include */

/* USER CODE END include */

/* Exported defines ----------------------------------------------------------*/
#if defined(USE_BSP_DRIVER)
/* code generated by STM32CubeMX does not support BSP */
/* In order to use BSP driver, add the correspondent files in the IDE workspace */
/* and define USE_BSP_DRIVER in the preprocessor definitions  or in platform.h */
#define RBI_CONF_RFO_LP_HP  RADIO_CONF_RFO_LP_HP
#define RBI_CONF_RFO_LP     RADIO_CONF_RFO_LP
#define RBI_CONF_RFO_HP     RADIO_CONF_RFO_HP
#else
/* USER CODE BEGIN Board Definition */

/* USER CODE END Board Definition */
#define RBI_CONF_RFO_LP_HP  0
#define RBI_CONF_RFO_LP     1
#define RBI_CONF_RFO_HP     2
/* USER CODE BEGIN Board Definition_2 */

/* USER CODE END Board Definition_2 */
#endif  /* USE_BSP_DRIVER  */

#if defined(USE_BSP_DRIVER)
/* code generated by STM32CubeMX does not support BSP */
/* In order to use BSP driver, add the correspondent files in the IDE workspace */
/* and define USE_BSP_DRIVER in the preprocessor definitions  or in platform.h */

#else
/* USER CODE BEGIN Exported Parameters */

/* USER CODE END Exported Parameters */
/* Indicates the type of switch between the ones proposed by CONFIG Constants
 */
#define RBI_CONF_RFO                        RBI_CONF_RFO_HP
```

```
/* USER CODE BEGIN Exported PinMapping */
#warning user to provide its board definitions pins

#define RF_SW_CTRL1_PIN                         GPIO_PIN_8
#define RF_SW_CTRL1_GPIO_PORT                   GPIOB
#define RF_SW_CTRL1_GPIO_CLK_ENABLE()           __HAL_RCC_GPIOB_CLK_ENABLE()
#define RF_SW_CTRL1_GPIO_CLK_DISABLE()          __HAL_RCC_GPIOB_CLK_DISABLE()

#define RF_SW_CTRL2_PIN                         GPIO_PIN_13
#define RF_SW_CTRL2_GPIO_PORT                   GPIOC
#define RF_SW_CTRL2_GPIO_CLK_ENABLE()           __HAL_RCC_GPIOC_CLK_ENABLE()
#define RF_SW_CTRL2_GPIO_CLK_DISABLE()          __HAL_RCC_GPIOC_CLK_DISABLE()
/* USER CODE END Exported PinMapping */
#endif  /* USE_BSP_DRIVER  */

/* USER CODE BEGIN ED */

/* USER CODE END ED */

/* Exported types ------------------------------------------------------------*/

#if defined(USE_BSP_DRIVER)
/* code generated by STM32CubeMX does not support BSP */
/* In order to use BSP driver, add the correspondent files in the IDE workspace */
/* and define USE_BSP_DRIVER in the preprocessor definitions  or in platform.h */
typedef enum
{
  RBI_SWITCH_OFF    = RADIO_SWITCH_OFF,
  RBI_SWITCH_RX     = RADIO_SWITCH_RX,
  RBI_SWITCH_RFO_LP = RADIO_SWITCH_RFO_LP,
  RBI_SWITCH_RFO_HP = RADIO_SWITCH_RFO_HP,
} RBI_Switch_TypeDef;

typedef enum
{
  RBI_RFO_LP_MAXPOWER = RADIO_RFO_LP_MAXPOWER,
  RBI_RFO_HP_MAXPOWER = RADIO_RFO_HP_MAXPOWER,
} RBI_RFOMaxPowerConfig_TypeDef;

#else
/* USER CODE BEGIN Exported Types */

/* USER CODE END Exported Types */
typedef enum
{
  RBI_SWITCH_OFF    = 0,
  RBI_SWITCH_RX     = 1,
  RBI_SWITCH_RFO_LP = 2,
  RBI_SWITCH_RFO_HP = 3,
} RBI_Switch_TypeDef;

typedef enum
{
  RBI_RFO_LP_MAXPOWER = 0,
  RBI_RFO_HP_MAXPOWER = 1,
} RBI_RFOMaxPowerConfig_TypeDef;
/* USER CODE BEGIN Exported Types_2 */

/* USER CODE END Exported Types_2 */
#endif  /* USE_BSP_DRIVER */
```

main.c | si7021.h | *lora_app.c | radio_board_if.c | radio_board_if.h | adc_if.c

```c
  /* USER CODE END SYS_DeInitMeasurement_1 */
}

int16_t SYS_GetTemperatureLevel(void)
{
  /* USER CODE BEGIN SYS_GetTemperatureLevel_1 */

  /* USER CODE END SYS_GetTemperatureLevel_1 */
  __IO int16_t temperatureDegreeC = 0;
  uint32_t measuredLevel = 0;
  uint16_t batteryLevelmV = SYS_GetBatteryLevel();

  measuredLevel = ADC_ReadChannels(ADC_CHANNEL_TEMPSENSOR);

  /* convert ADC level to temperature */
  /* check whether device has temperature sensor calibrated in production */
  if (((int32_t)*TEMPSENSOR_CAL2_ADDR - (int32_t)*TEMPSENSOR_CAL1_ADDR) != 0)
  {
    /* Device with temperature sensor calibrated in production:
       use device optimized parameters */
    temperatureDegreeC = __LL_ADC_CALC_TEMPERATURE(batteryLevelmV,
                                                   measuredLevel,
                                                   LL_ADC_RESOLUTION_12B);
  }
  else
  {
    /* Device with temperature sensor not calibrated in production:
       use generic parameters */
    temperatureDegreeC = __LL_ADC_CALC_TEMPERATURE_TYP_PARAMS(TEMPSENSOR_TYP_AVGSLOPE,
                                                              TEMPSENSOR_TYP_CAL1_V,
                                                              TEMPSENSOR_CAL1_TEMP,
                                                              batteryLevelmV,
                                                              measuredLevel,
                                                              LL_ADC_RESOLUTION_12B);
  }

  /* from int16 to q8.7*/
  temperatureDegreeC <<= 8;

  return (int16_t) temperatureDegreeC;
  /* USER CODE BEGIN SYS_GetTemperatureLevel_2 */

  /* USER CODE END SYS_GetTemperatureLevel_2 */
}
```

```c
uint16_t SYS_GetBatteryLevel(void)
{
  /* USER CODE BEGIN SYS_GetBatteryLevel_1 */

  /* USER CODE END SYS_GetBatteryLevel_1 */
  uint16_t batteryLevelmV = 0;
  uint32_t measuredLevel = 0;

  measuredLevel = ADC_ReadChannels(ADC_CHANNEL_VREFINT);

  if (measuredLevel == 0)
  {
    batteryLevelmV = 0;
  }
  else
  {
    if ((uint32_t)*VREFINT_CAL_ADDR != (uint32_t)0xFFFFU)
    {
      /* Device with Reference voltage calibrated in production:
         use device optimized parameters */
      batteryLevelmV = __LL_ADC_CALC_VREFANALOG_VOLTAGE(measuredLevel,
                                                        ADC_RESOLUTION_12B);
    }
    else
    {
      /* Device with Reference voltage not calibrated in production:
         use generic parameters */
      batteryLevelmV = (VREFINT_CAL_VREF * 1510) / measuredLevel;
    }
  }

  return batteryLevelmV;
  /* USER CODE BEGIN SYS_GetBatteryLevel_2 */

  /* USER CODE END SYS_GetBatteryLevel_2 */
}

/* Private Functions Definition ------------------------------------------------*/
/* USER CODE BEGIN PrFD */
```

```c
uint16_t GetLumenLevel(void)
{

    uint16_t Lumen = 0;
    uint32_t measuredLumen = 0;

    measuredLumen = ADC_ReadChannels(ADC_CHANNEL_4);
    Lumen = measuredLumen;
    return Lumen;

}



/* USER CODE END PrFD */

static uint32_t ADC_ReadChannels(uint32_t channel)
{
  /* USER CODE BEGIN ADC_ReadChannels_1 */

  /* USER CODE END ADC_ReadChannels_1 */
  uint32_t ADCxConvertedValues = 0;
  ADC_ChannelConfTypeDef sConfig = {0};

  MX_ADC_Init();

  /* Start Calibration */
  if (HAL_ADCEx_Calibration_Start(&hadc) != HAL_OK)
  {
    Error_Handler();
  }

  /* Configure Regular Channel */
  sConfig.Channel = channel;
  sConfig.Rank = ADC_REGULAR_RANK_1;
  sConfig.SamplingTime = ADC_SAMPLINGTIME_COMMON_1;
  if (HAL_ADC_ConfigChannel(&hadc, &sConfig) != HAL_OK)
  {
    Error_Handler();
  }

  if (HAL_ADC_Start(&hadc) != HAL_OK)
  {
    /* Start Error */
    Error_Handler();
  }
  /** Wait for end of conversion */
  HAL_ADC_PollForConversion(&hadc, HAL_MAX_DELAY);

  /** Wait for end of conversion */
  HAL_ADC_Stop(&hadc);   /* it calls also ADC_Disable() */

  ADCxConvertedValues = HAL_ADC_GetValue(&hadc);

  HAL_ADC_DeInit(&hadc);

  return ADCxConvertedValues;
  /* USER CODE BEGIN ADC_ReadChannels_2 */

  /* USER CODE END ADC_ReadChannels_2 */
}

```

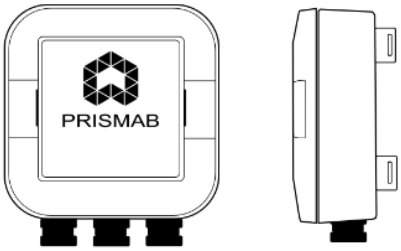
main.c | si7021.h | *lora_app.c | radio_board_if.c | radio_board_if.h | adc_if.c | sys_app.c | lora_app.h | Commissioning.h | se-identity.h

```
/* Exported constants ---------------------------------------------------------*/

/*!
 ******************************************************************************
 ********************************** WARNING ***********************************
 ******************************************************************************
  The secure-element implementation supports both 1.0.x and 1.1.x LoRaWAN
  versions of the specification.
  Thus it has been decided to use the 1.1.x keys and EUI name definitions.
  The below table shows the names equivalence between versions:
              +---------------------+-------------------------+
              |       1.0.x         |          1.1.x          |
              +=====================+=========================+
              | LORAWAN_DEVICE_EUI  | LORAWAN_DEVICE_EUI      |
              +---------------------+-------------------------+
              | LORAWAN_APP_EUI     | LORAWAN_JOIN_EUI        |
              +---------------------+-------------------------+
              | LORAWAN_GEN_APP_KEY | LORAWAN_APP_KEY         |
              +---------------------+-------------------------+
              | LORAWAN_APP_KEY     | LORAWAN_NWK_KEY         |
              +---------------------+-------------------------+
              | LORAWAN_NWK_S_KEY   | LORAWAN_F_NWK_S_INT_KEY |
              +---------------------+-------------------------+
              | LORAWAN_NWK_S_KEY   | LORAWAN_S_NWK_S_INT_KEY |
              +---------------------+-------------------------+
              | LORAWAN_NWK_S_KEY   | LORAWAN_NWK_S_ENC_KEY   |
              +---------------------+-------------------------+
              | LORAWAN_APP_S_KEY   | LORAWAN_APP_S_KEY       |
              +---------------------+-------------------------+
 ******************************************************************************
 ******************************************************************************
 ******************************************************************************
 */
/*!
 * End-device IEEE EUI (big endian)
 * When set to 00,00,00,00,00,00,00,00 DevEui is automatically set with a value provided by MCU platform
 */
#define LORAWAN_DEVICE_EUI

/*!
 * App/Join server IEEE EUI (big endian)
 */
#define LORAWAN_JOIN_EUI

/*!
 * Device address on the network (big endian)
 * When set to 00,00,00,00 DevAddr is automatically set with a value provided by MCU platform
 */
#define LORAWAN_DEVICE_ADDRESS

/*!
 * Application root key
 */
#define LORAWAN_APP_KEY

/*!
 * Network root key
 */
#define LORAWAN_NWK_KEY

/*!
 * Forwarding Network session key
 */
#define LORAWAN_NWK_S_KEY

/*!
 * Application session key
 */
#define LORAWAN_APP_S_KEY
```

[redacted]

[redacted]

- Fácil instalación
- Larga autonomía (4 años)
- Conectividad SigFox sin cuotas de comunicación o acceso a internet.
- 2 años de garantía
- Conectividad a App PRISMAB
- Protección IP67, instalable fácilmente en exterior.
- Construido en material ASA contra radiación solar
- 3 conexiones push-pull para la fácil instalación de los sensores.
- Totalmente Plug-and-Play. Reconocimiento automático de sensores sin configuración manual



Sistema de transmisión	SIGFOX
Frecuencia de radio	268.130 MHz - 869.525 MHz
Frecuencia de envío de datos	1 Hora
Potencia radiada/antena	16 dBm /w512
Frecuencia de medida del dieléctrico	70 Mhz
Autonomía media	4 años
Consumo en reposo	1,8µA
Consumo en operación / voltaje	51mA / 3,3V
Alimentación	4 x baterías AA (1,5v)
Material envolvente	ASA blanco
Tipo de protección	IP67
Rango de temperatura de trabajo	-20ºC a 54ºC
Peso	290g
Dimensiones	100 x 100 x 40 mm
Número máximo de sensores	3

RAK® Documentation Center

RAK3172 WisDuo LPWAN Module Datasheet

Overview

Description

RAK3172 is a low-power long-range transceiver module based on the STM32WLE5CC chip. It provides an easy-to-use, small-size, low-power solution for long-range wireless data applications. This module complies with Class A, B, & C of the LoRaWAN 1.0.3 specifications. It can easily connect to different LoRaWAN server platforms like TheThingsNetwork (TTN), Chirpstack, Actility, etc. It also supports LoRa Point-to-Point (P2P) communication mode, which helps you implement your own customized long-range LoRa network quickly.

You can configure the mode and operation of the module using AT commands via a UART interface. RAK3172 also offers low-power features, which are very suitable for battery-powered applications.

Features

- Based on **STM32WLE5CCU6**
- **LoRaWAN 1.0.3** specification compliant
- **Supported bands**: EU433, CN470, IN865, EU868, AU915, US915, KR920, RU864, and AS923-1/2/3/4
- LoRaWAN activation via OTAA/ABP
- LoRa Point-to-Point (P2P) communication
- Custom firmware using Arduino via RUI3 API
- Easy to use AT command set via UART interface
- Long-range - greater than 15 km with optimized antenna
- ARM Cortex-M4 32-bit
- 256 kbytes flash memory with ECC
- 64 kbytes RAM
- Ultra-low-power consumption of 1.69 µA in sleep mode
- **Supply Voltage**: 2.0 V ~ 3.6 V
- **Temperature range**: -40° C ~ 85° C

> **NOTE:**
>
> There are two variants available for the RAK3172 Module: (1) with the CE & UKCA Certification Mark and (2) with FCC, IC & RCM Certification Mark.
>
> If you need LoRa module with BLE 5.0 capability, you can check RAK11720 which is pin-to-pin compatible to RAK3172 with extra pins for additional ground and BLE RF antenna port.

Specifications

This section covers the hardware and software specifications of RAK3172. Also, it includes the block diagram and the updated firmware link of the RAK3172 WisDuo module.

Operating Frequencies

Module	Region	Frequency
RAK3172(L)	Europe	EU433
	China	CN470
RAK3172(H)	Europe	EU868
	North America	US915
	Australia	AU915
	Korea	KR920
	Asia	AS923-1/2/3/4
	India	IN865
	Russia	RU864

Electrical Characteristics

RAK® Documentation Center

Operating Voltage

Feature	Minimum	Typical	Maximum	Unit
VCC	2.0	3.3	3.6	Volts (V)

Operating Current

Feature	Condition	Minimum	Typical	Maximum	Unit
Operating Current	TX Mode	87 (@ 20 dBm 868 Mhz)			mA
	RX Mode	5.22			mA

Sleep Current

Feature	Condition	Minimum (2.1 V)	Typical (3.3 V)	Maximum	Unit
Current Consumption	EU868	-	1.69	-	µA
	US915	-	1.69	-	µA
	CN470	-	1.69	-	µA

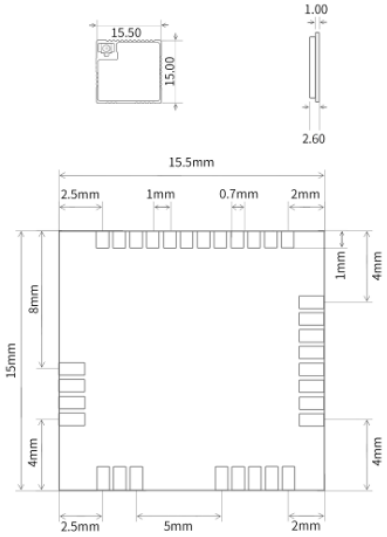


Figure 3: RAK3172 Physical Dimension

Layout Recommendation

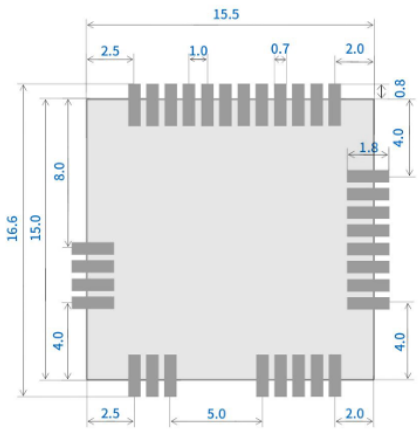


Figure 4: RAK3172 Layout

Environmental Characteristics

Operating Temperature

Feature	Minimum	Typical	Maximum	Unit
Operating Temperature	-40	25	85	°C

Storage Temperature

RAK® Documentation Center

Feature	Minimum	Typical	Maximum	Unit
Storage Temperature	-40		85	°C

Recommended Reflow Profile

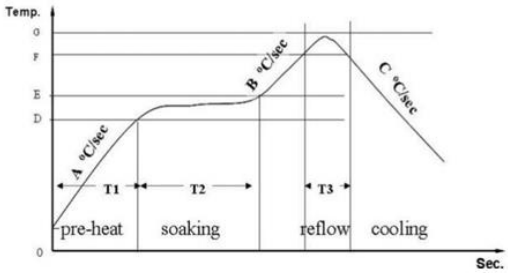


Figure 5: Reflow Profile for RAK3172

Standard conditions for reflow soldering:

- Pre-heating Ramp (A) (Initial temperature: 150° C): **1~2.5° C/sec**
- Soaking Time (T2) (150~180° C): **60~100 sec**
- Peak Temperature (G): **230~250° C**
- Reflow Time (T3) (>220° C): **30~60 sec**
- Ramp-up Rate (B): **0~2.5° C/sec**
- Ramp-down Rate (C): **1~3° C/sec**

Software

Download the latest RAK3172 WisDuo LPWAN Module firmware provided below. RAK3172(L) and RAK3172(H) use the same firmware and it will automatically detect the variant of the module being used.

The **bin file** contains the application code only and you need the RAK DFU Tool to upload this file to the module.

The **hex file** contains both the bootloader and the application code. You need to use STM32CubeProgrammer to upload this.

> **WARNING**
>
> Uploading the `.hex` file via STM32CubeProgrammer will erase all configured data on the device.

RAK3172 uses UART2 serial pins to upload the latest firmware.

> **NOTE:**
>
> RAK3172 should automatically go to BOOT mode when the firmware is uploaded via RAK DFU Tool or WisToolBox.
>
> If BOOT mode is not initiated, you can manually send `AT+BOOT` command to start bootloader mode.

SILONEX® Inc

TO-8 Photocells

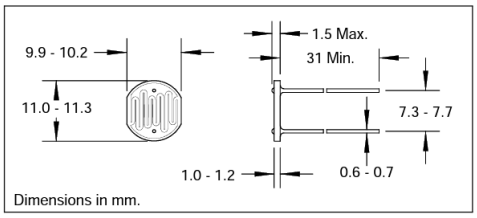
Ceramic Package

Features

- Three photoconductive materials
- Tolerance: ±40% @1 ftc, ±33% @ 2 ftc

Description

The Silonex TO-8 series of photoconductive cells provide up to eight standard resistance ranges in CdS or CdSe materials. Most of these devices are also available in a hermetic package.



Absolute Maximum Ratings

Operating & Storage Temperature	-60°C to +75°C
Power Dissipation @ 25°C (1)	250 mW

Electrical Characteristics (T_A=25°C, source at 2854 K)

Silonex Part No	Sensitive Material	Spectral Peak (nm)	R @ 1 ftc (Kohms) (3)	R @ 2 ftc (Kohms) (3)	R_{TYP} @ 100 ftc (ohms)	Min. R_{DARK} (Kohms) (2)	V_{MAX}(Volts) (peak AC or DC)
NSL-4912	Type 4 CdS	515	9.3		150	560	80
NSL-4922			4.0		80	250	80
NSL-4932			1.7		50	100	80
NSL-4942			0.725		30	50	80
NSL-4952			40		860	2400	320
NSL-4962			17		500	1000	320
NSL-4972			7.25		260	400	320
NSL-4982			114		2500	6800	320
NSL-5912	Type 5 CdS	550		1.5	150	100	170
NSL-5922				3.0	300	200	170
NSL-5932				6.0	500	400	250
NSL-5942				9.0	700	600	250
NSL-6922	Type 6 CdSe	690		0.510	50	300	170
NSL-6952				2.7	200	720	170

Specifications subject to change without notice. 103269 REV 3

Notes: (1) derate linearly to zero at 75°C.
(2) 5 sec. after removal of test light.
(3) cells light adapted at 30 to 50 ftc for 16 hrs minimum prior to electrical tests.

5200 St. Patrick St., Montreal
Que., H4E 4N9, Canada
Tel: 514-768-8000
Fax: 514-768-8889

The Old Railway, Princes Street
Ulverston, Cumbria, LA12 7NQ, UK
Tel: 01 229 581 551
Fax: 01 229 581 554

life.augmented

STM32WLE5xx STM32WLE4xx

Multiprotocol LPWAN 32-bit Arm® Cortex®-M4 MCUs, LoRa®, (G)FSK, (G)MSK, BPSK, up to 256KB flash, 64KB SRAM

Datasheet - production data

Features

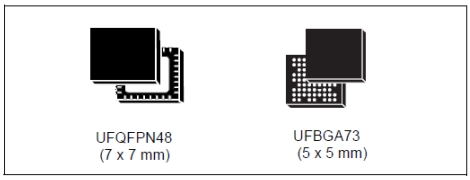
Includes ST state-of-the-art patented technology

Radio

- Frequency range: 150 MHz to 960 MHz
- Modulation: LoRa®, (G)FSK, (G)MSK and BPSK
- RX sensitivity: –123 dBm for 2-FSK (at 1.2 Kbit/s), –148 dBm for LoRa® (at 10.4 kHz, spreading factor 12)
- Transmitter high output power, programmable up to +22 dBm
- Transmitter low output power, programmable up to +15 dBm
- Available integrated passive device (IPD) companion chips for optimized matching, filtering and balun, all in one very compact solution covering each package and each main use cases (22 dBm @ 915 MHz, 14 dBm @ 868 MHz, 17 dBm @ 490 MHz)
- Compliant with the following radio frequency regulations such as ETSI EN 300 220, EN 300 113, EN 301 166, FCC CFR 47 Part 15, 24, 90, and the Japanese ARIB STD-T30, T-67, T-108
- Compatible with standardized or proprietary protocols such as LoRaWAN®, Sigfox™, W-MBus and more (fully open wireless system-on-chip)

Ultra-low-power platform

- 1.8 V to 3.6 V power supply
- –40 °C to +105 °C temperature range
- Shutdown mode: 31 nA (V_{DD} = 3 V)
- Standby (+ RTC) mode: 360 nA (V_{DD} = 3 V)
- Stop2 (+ RTC) mode: 1.07 µA (V_{DD} = 3 V)



- Active-mode MCU: < 72 µA/MHz (CoreMark®)
- Active-mode RX: 4.82 mA
- Active-mode TX: 15 mA at 10 dBm and 87 mA at 20 dBm (LoRa® 125 kHz)

Core

- 32-bit Arm® Cortex®-M4 CPU
  - Adaptive real-time accelerator (ART Accelerator) allowing 0-wait-state execution from flash memory, frequency up to 48 MHz, MPU and DSP instructions
  - 1.25 DMIPS/MHz (Dhrystone 2.1)

Security and identification

- Hardware encryption AES 256-bit
- True random number generator (RNG)
- Sector protection against read/write operations (PCROP, RDP, WRP)
- CRC calculation unit
- Unique device identifier (64-bit UID compliant with IEEE 802-2001 standard)
- 96-bit unique die identifier
- Hardware public key accelerator (PKA)

Supply and reset management

- High-efficiency embedded SMPS step-down converter
- SMPS to LDO smart switch
- Ultra-safe, low-power BOR (brownout reset) with 5 selectable thresholds
- Ultra-low-power POR/PDR

December 2022 DS13105 Rev 12 1/149

This is information on a product in full production. www.st.com

Table 2. Main features and peripheral counts

Feature		STM32WLE5Cx STM32WLE4Cx	STM32WLE5Jx STM32WLE4Jx
CPU		Arm Cortex-M4	
Maximum CPU frequency (MHz)		48	
Flash memory density (Kbytes)		256/128/64	
SRAM density (Kbytes)	SRAM1	32/16/0	
	SRAM2	32/32/20	
Radio	LoRa	Available on STM32WLE5xx devices. Not available on STM32WLE4xx devices	
	(G)FSK	Yes	
	(G)MSK Tx		
	BPSK Tx		
Radio power amplifier	Low output power (up to 15 dBm)	Yes	
	High output power (up to 22 dBm)		
Timer	General purpose	4	
	Low-power	3	
	SysTick	1	
Communication interface	SPI/I^2S	2 (1 supporting I^2S)	
	I^2C	3	
	USART	2	
	LPUART	1	
Watchdog	Independent	1	
	Window	1	
RTC (with wakeup counter)		1	
DMA (7 channels)		2	
Semaphores		1	

Table 2. Main features and peripheral counts (continued)

Feature		STM32WLE5Cx STM32WLE4Cx	STM32WLE5Jx STM32WLE4Jx
Security	AES 256 bits	1	
	RNG	1	
	PKA	1	
	PCROP, RDP, WRP	1	
	CRC	1	
	64-bit UID compliant with IEEE 802-2001 standard	1	
	96-bit die ID	1	
Tamper pins		3	3
Wakeup pins		3	3
GPIOs		29	43
ADC (number of channels, ext + int)		1 (9 + 4)	1 (12 + 4)
DAC (number of channels)		1 (1)	
Internal VREFBUF		No	Yes
Analog comparator		2	
Operating voltage		1.8 to 3.6 V	
Ambient operating temperature		–40 °C to +105 °C / –40 °C to +85 °C (with radio)[1]	
Junction temperature		–40 °C to +125 °C / –40 °C to +105 °C (with radio)	
Package		UFQFPN48 (7 x 7 mm)	UFBGA73 (5 x 5 mm)

1. Devices with suffix 6 operate up to 85 °C. Devices with suffix 7 can operate up to 105 °C except radio.

SILICON LABS

Si7021-A20

I²C HUMIDITY AND TEMPERATURE SENSOR

Features

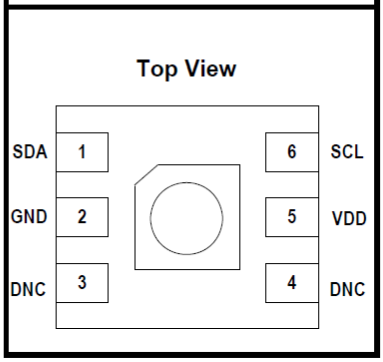
- Precision Relative Humidity Sensor
  - ± 3% RH (max), 0–80% RH
- High Accuracy Temperature Sensor
  - ±0.4 °C (max), –10 to 85 °C
- 0 to 100% RH operating range
- Up to –40 to +125 °C operating range
- Wide operating voltage (1.9 to 3.6 V)
- Low Power Consumption
  - 150 µA active current
  - 60 nA standby current
- Factory-calibrated
- I²C Interface
- Integrated on-chip heater
- 3x3 mm DFN Package
- Excellent long term stability
- Optional factory-installed cover
  - Low-profile
  - Protection during reflow
  - Excludes liquids and particulates

Ordering Information:
See page 29.

Applications

- HVAC/R
- Thermostats/humidistats
- Respiratory therapy
- White goods
- Indoor weather stations
- Micro-environments/data centers
- Automotive climate control and defogging
- Asset and goods tracking
- Mobile phones and tablets

Pin Assignments



Patent Protected. Patents pending

Description

The Si7021 I²C Humidity and Temperature Sensor is a monolithic CMOS IC integrating humidity and temperature sensor elements, an analog-to-digital converter, signal processing, calibration data, and an I²C Interface. The patented use of industry-standard, low-K polymeric dielectrics for sensing humidity enables the construction of low-power, monolithic CMOS Sensor ICs with low drift and hysteresis, and excellent long term stability.

The humidity and temperature sensors are factory-calibrated and the calibration data is stored in the on-chip non-volatile memory. This ensures that the sensors are fully interchangeable, with no recalibration or software changes required.

The Si7021 is available in a 3x3 mm DFN package and is reflow solderable. It can be used as a hardware- and software-compatible drop-in upgrade for existing RH/temperature sensors in 3x3 mm DFN-6 packages, featuring precision sensing over a wider range and lower power consumption. The optional factory-installed cover offers a low profile, convenient means of protecting the sensor during assembly (e.g., reflow soldering) and throughout the life of the product, excluding liquids (hydrophobic/oleophobic) and particulates.

The Si7021 offers an accurate, low-power, factory-calibrated digital solution ideal for measuring humidity, dew-point, and temperature, in applications ranging from HVAC/R and asset tracking to industrial and consumer platforms.

1. Electrical Specifications

Unless otherwise specified, all min/max specifications apply over the recommended operating conditions.

Table 1. Recommended Operating Conditions

Parameter	Symbol	Test Condition	Min	Typ	Max	Unit
Power Supply	V_{DD}		1.9	—	3.6	V
Operating Temperature	T_A	I and Y grade	–40	—	+125	°C
Operating Temperature	T_A	G grade	–40	—	+85	°C

Table 2. General Specifications

1.9 ≤ V_{DD} ≤ 3.6 V; T_A = –40 to 85 °C (G grade) or –40 to 125 °C (I/Y grade); default conversion time unless otherwise noted.

Parameter	Symbol	Test Condition	Min	Typ	Max	Unit
Input Voltage High	V_{IH}	SCL, SDA pins	0.7xV_{DD}	—	—	V
Input Voltage Low	V_{IL}	SCL, SDA pins	—	—	0.3xV_{DD}	V
Input Voltage Range	V_{IN}	SCL, SDA pins with respect to GND	0.0	—	V_{DD}	V
Input Leakage	I_{IL}	SCL, SDA pins	—	—	1	µA
Output Voltage Low	V_{OL}	SDA pin; I_{OL} = 2.5 mA; V_{DD} = 3.3 V	—	—	0.6	V
		SDA pin; I_{OL} = 1.2 mA; V_{DD} = 1.9 V	—	—	0.4	V
Current Consumption	I_{DD}	RH conversion in progress	—	150	180	µA
		Temperature conversion in progress	—	90	120	µA
		Standby, –40 to +85 °C[2]	—	0.06	0.62	µA
		Standby, –40 to +125 °C[2]	—	0.06	3.8	µA
		Peak I_{DD} during powerup[3]	—	3.5	4.0	mA
		Peak I_{DD} during I^2C operations[4]	—	3.5	4.0	mA
Heater Current[5]	I_{HEAT}		—	3.1 to 94.2	—	mA

Notes:
1. Initiating a RH measurement will also automatically initiate a temperature measurement. The total conversion time will be $t_{CONV}(RH) + t_{CONV}(T)$.
2. No conversion or I^2C transaction in progress. Typical values measured at 25 °C.
3. Occurs once during powerup. Duration is <5 msec.
4. Occurs during I^2C commands for Reset, Read/Write User Registers, Read EID, and Read Firmware Version. Duration is <100 µs when I^2C clock speed is >100 kHz (>200 kHz for 2-byte commands).
5. Additional current consumption when HTRE bit enabled. See Section "5.5. Heater" for more information.

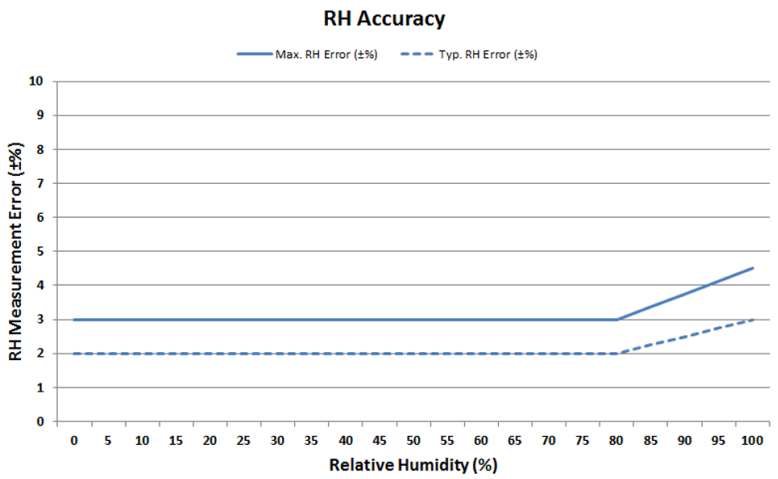


Figure 2. RH Accuracy at 30 °C

Table 5. Temperature Sensor

1.9 ≤ V_{DD} ≤ 3.6 V; T_A = –40 to +85 °C (G grade) or –40 to +125 °C (I/Y grade) default conversion time, unless otherwise noted.

Parameter	Symbol	Test Condition	Min	Typ	Max	Unit
Operating Range		I and Y Grade	–40	—	+125	°C
		G Grade	–40	—	+85	°C
Accuracy[1]		–10 °C≤ t_A ≤ 85 °C	—	±0.3	±0.4	°C
		–40 ≤ t_A ≤ 125 °C		Figure 3		
Repeatability/Noise		14-bit resolution	—	0.01	—	°C RMS
		13-bit resolution	—	0.02	—	
		12-bit resolution	—	0.04	—	
		11-bit resolution	—	0.08	—	
Response Time[2]	$\tau_{63\%}$	Unmounted device	—	0.7	—	s
		Si7021-EB board	—	5.1	—	
Long Term Stability			—	≤ 0.01	—	°C/Yr

Notes:
1. 14b measurement resolution (default).
2. Time to reach 63% of final value in response to a step change in temperature. Actual response time will vary dependent on system thermal mass and air-flow.

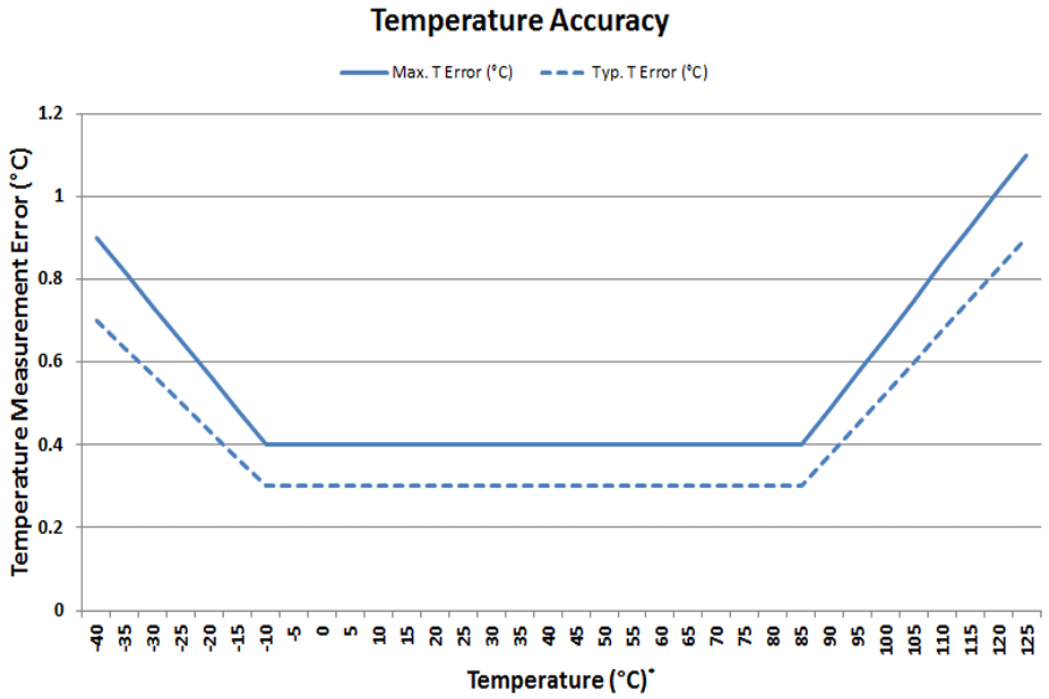
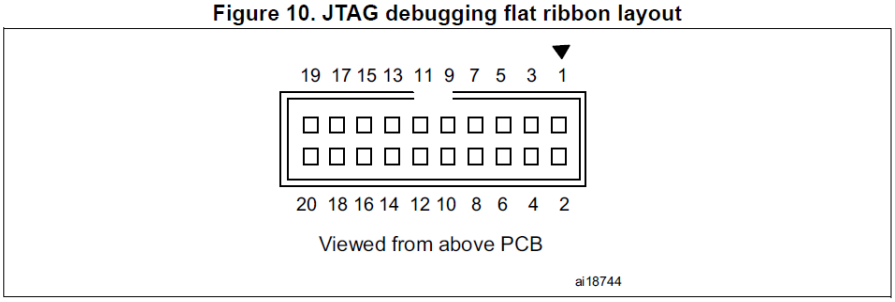


Figure 3. Temperature Accuracy*

***Note:** Applies only to I and Y grade devices beyond +85 °C.

Figure 10. JTAG debugging flat ribbon layout



4.2 Connection with STM32

For development of applications based on STM32 microcontrollers, the ST-LINK/V2 must be connected to the application using the standard 20-pin JTAG flat ribbon provided.

Table 4 summarizes the signals names, functions, and target connection signals of the standard 20-pin JTAG flat ribbon.

Table 4. JTAG/SWD cable connections

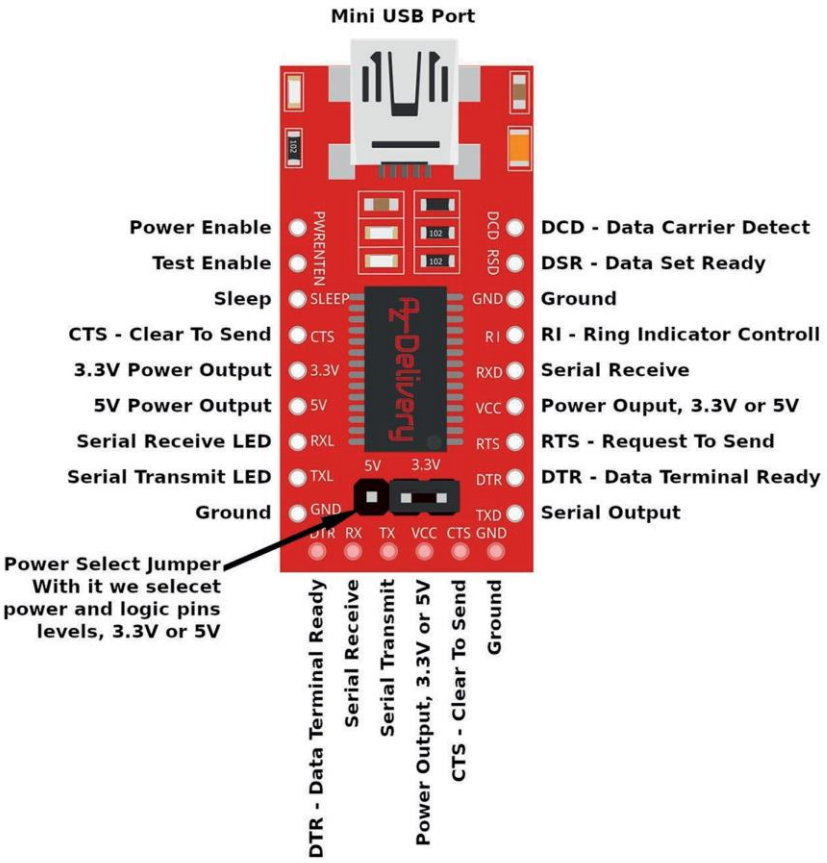
Pin no.	ST-LINK/V2 connector (CN3)	ST-LINK/V2 function	Target connection (JTAG)	Target connection (SWD)
1	VAPP	Target VCC	MCU VDD(1)	MCU VDD(1)
2				
3	TRST	JTAG TRST	JNTRST	GND(2)
4	GND(3)	GND(3)	GND(3)(4)	GND(3)(4)
5	TDI	JTAG TDO	JTDI	GND(2)
6	GND(3)	GND(3)	GND(3)(4)	GND(3)(4)
7	TMS_SWDIO	JTAG TMS, SW IO	JTMS	SWDIO
8	GND(3)	GND(3)	GND(3)(4)	GND(3)(4)
9	TCK_SWCLK	JTAG TCK, SW CLK	JTCK	SWCLK
10	GND(5)	GND(5)	GND(4)(5)	GND(4)(5)
11	Not connected	Not connected	Not connected	Not connected
12	GND	GND	GND(4)	GND(4)
13	TDO_SWO	JTAG TDI, SWO	JTDO	TRACESWO(6)
14	GND(5)	GND(5)	GND(4)(5)	GND(4)(5)
15	NRST	NRST	NRST	NRST
16	GND(3)	GND(3)	GND(3)(4)	GND(3)(4)
17	Not connected	Not connected	Not connected	Not connected
18	GND	GND	GND(4)	GND(4)
19	VDD(3)	VDD (3.3 V)(3)	Not connected	Not connected
20	GND	GND	GND(4)	GND(4)

1. The power supply from the application board is connected to the ST-LINK/V2 debugging and programming board to ensure signal compatibility between the boards.
2. Connect to GND for noise reduction on the ribbon.
3. Available on ST-LINK/V2 only, not connected on ST-LINK/V2-ISOL.
4. At least one of these pin must be connected to the ground for correct behavior, it is recommended to connecting all of them.
5. GND on ST-LINK/V2, used by SWIM on ST-LINK/V2-ISOL (see *Table 3*).
6. Optional: for Serial Wire Viewer (SWV) trace.

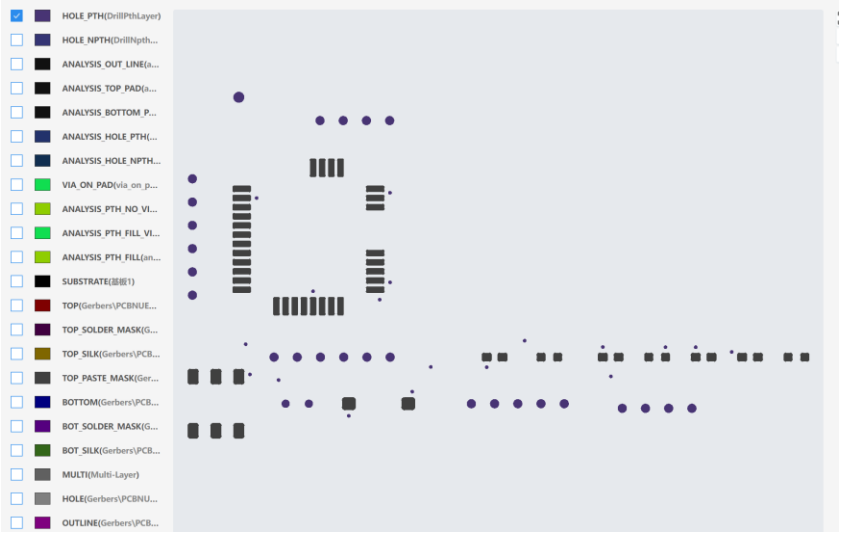
ANEXO J: DATASHEET TTL AZDELIVERY FT232RL

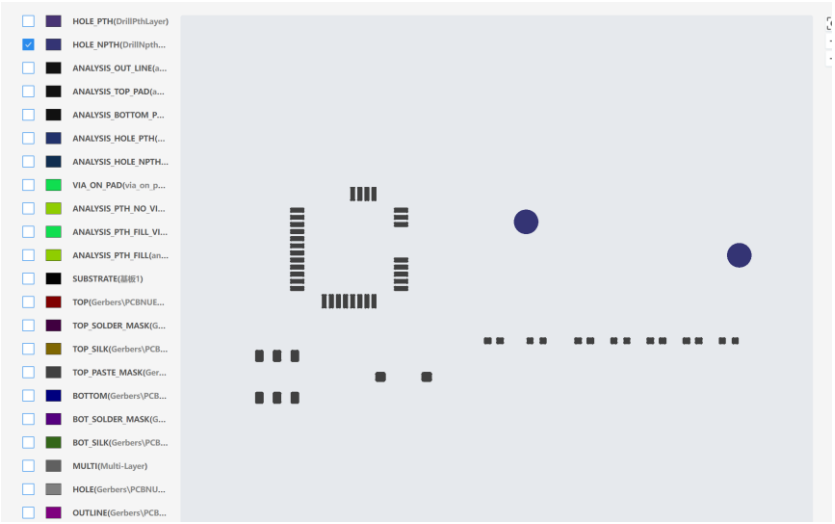
2. Absolute Maximum Ratings

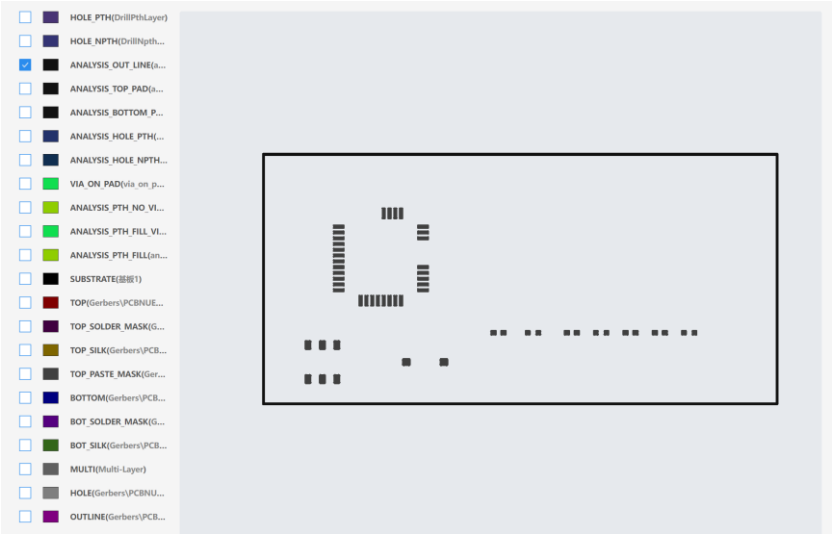
Parameter	Value	Units
Storage Temperature	-65 to 150	°C
Floor Life (Out of Bag) At Factory Ambient (30°C / 60% Relative Humidity)	168 (IPC/JEDEC J-STD-033A MSL Level 3 Compliant)*	Hours
Ambient Temperature (Power Applied)	-40 to 85	°C
MTTF	11162037	hours
VCC Supply Voltage	-0.5 to +6.00	V
DC Input Voltage – USBDP and USBDM	-0.5 to +3.8	V
DC Input Voltage – High Impedance Bidirectional	-0.5 to + (VCC +0.5)	V
DC Input Voltage – All Other Inputs	-0.5 to + (VCC +0.5)	V
DC Output Current – Outputs	24	mA
DC Output Current – Low Impedance Bidirectional	24	mA
Power Dissipation (VCC = 5.25V)	500	mW

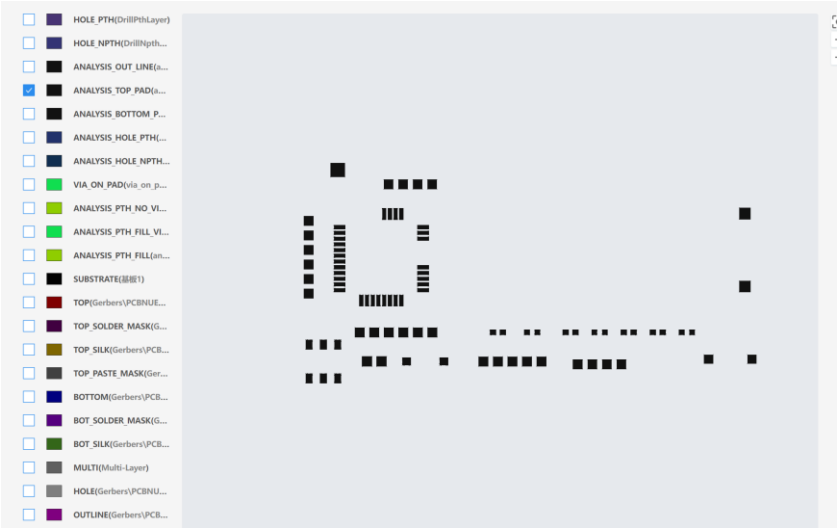


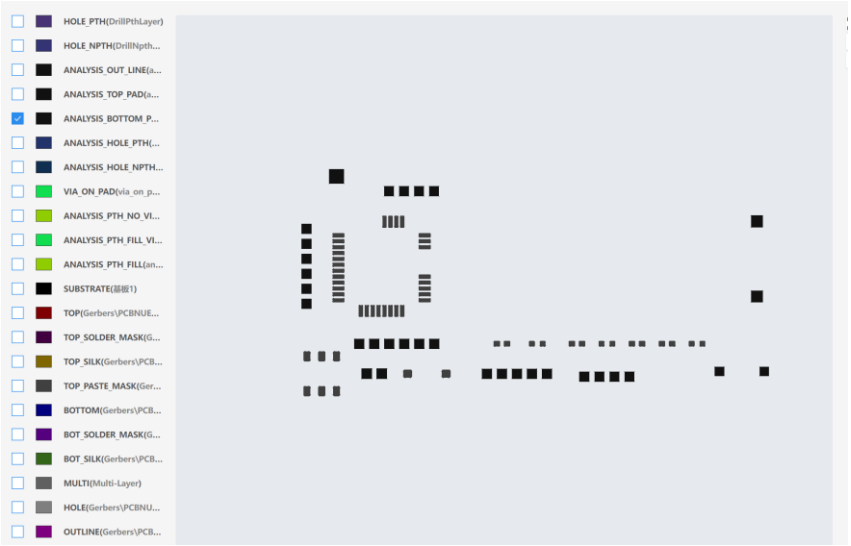
ANEXO K: GERBERS DEL PROTOTIPO

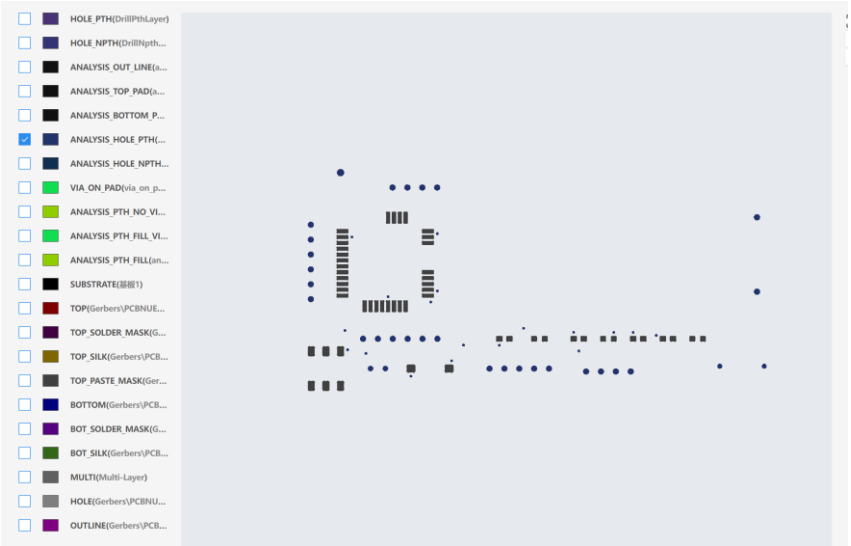


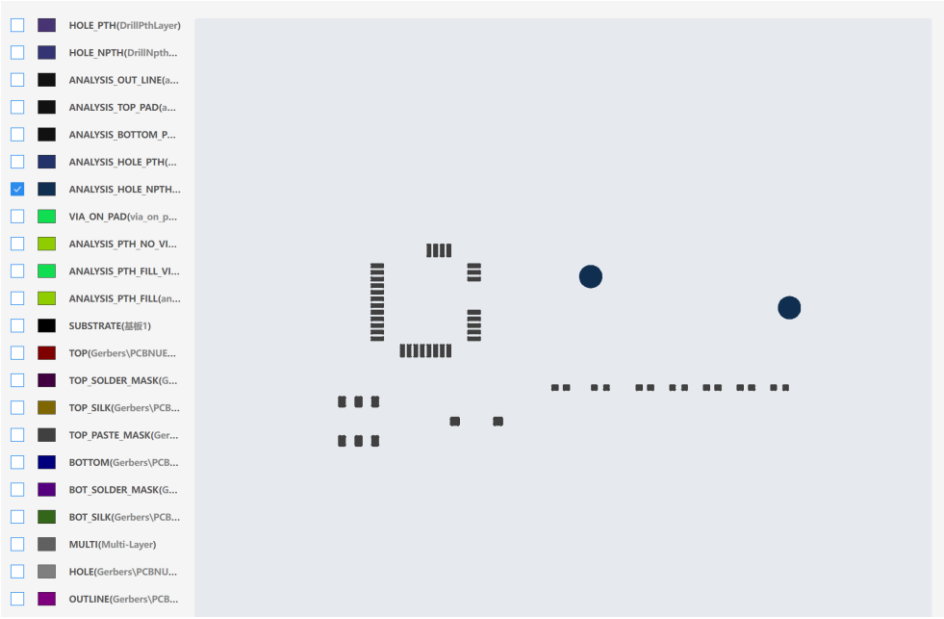


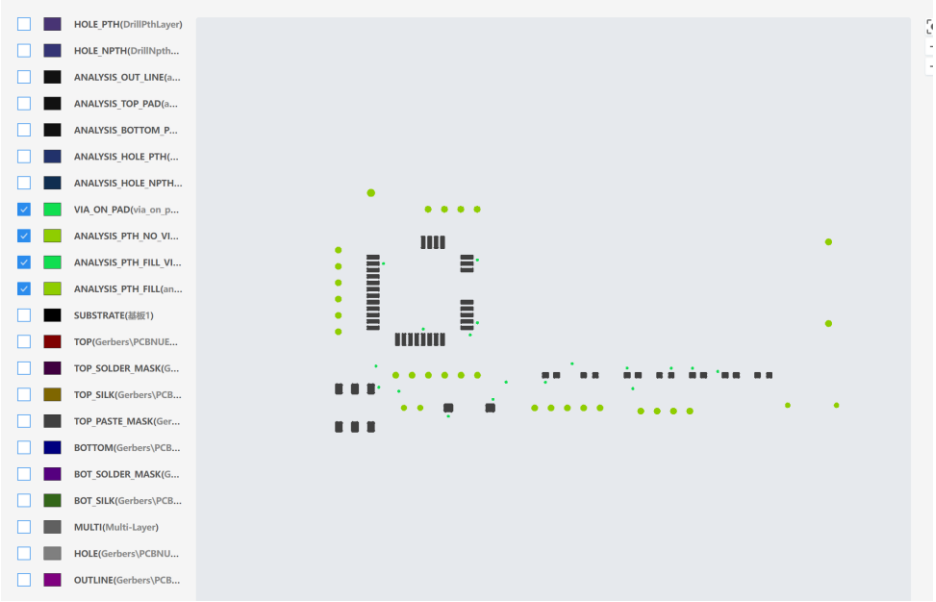


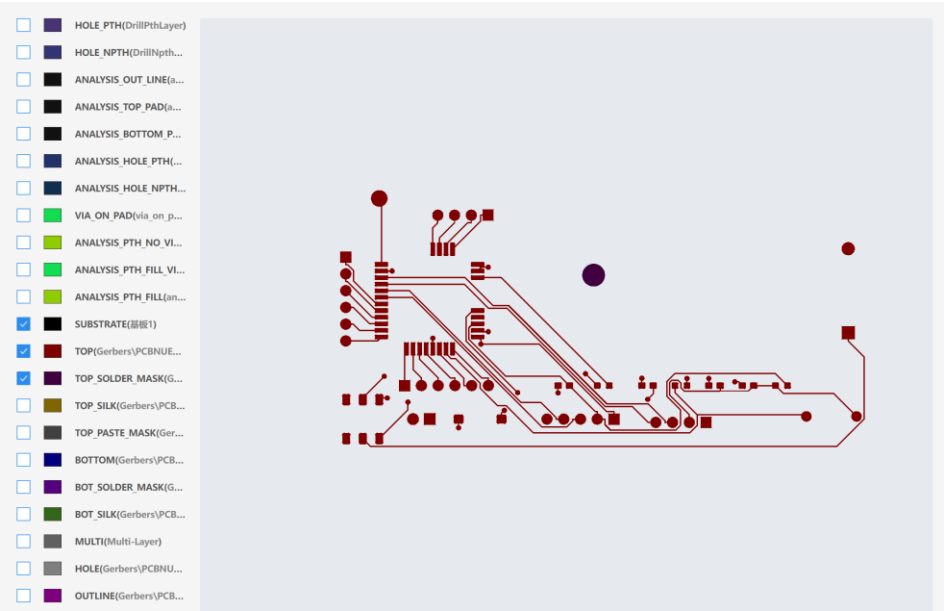

HOLE_PTH(DrillPthLayer)
HOLE_NPTH(DrillNpth...
ANALYSIS_OUT_LINE(a...
ANALYSIS_TOP_PAD(a...
ANALYSIS_BOTTOM_P...
ANALYSIS_HOLE_PTH(...
ANALYSIS_HOLE_NPTH...
VIA_ON_PAD(via_on_p...
ANALYSIS_PTH_NO_VI...
ANALYSIS_PTH_FILL_VI...
ANALYSIS_PTH_FILL(an...
SUBSTRATE(基板1)
TOP(Gerbers\PCBNUE...
TOP_SOLDER_MASK(G...
TOP_SILK(Gerbers\PCB...
TOP_PASTE_MASK(Ger...
BOTTOM(Gerbers\PCB...
BOT_SOLDER_MASK(G...
BOT_SILK(Gerbers\PCB...
MULTI(Multi-Layer)
HOLE(Gerbers\PCBNU...
OUTLINE(Gerbers\PCB...

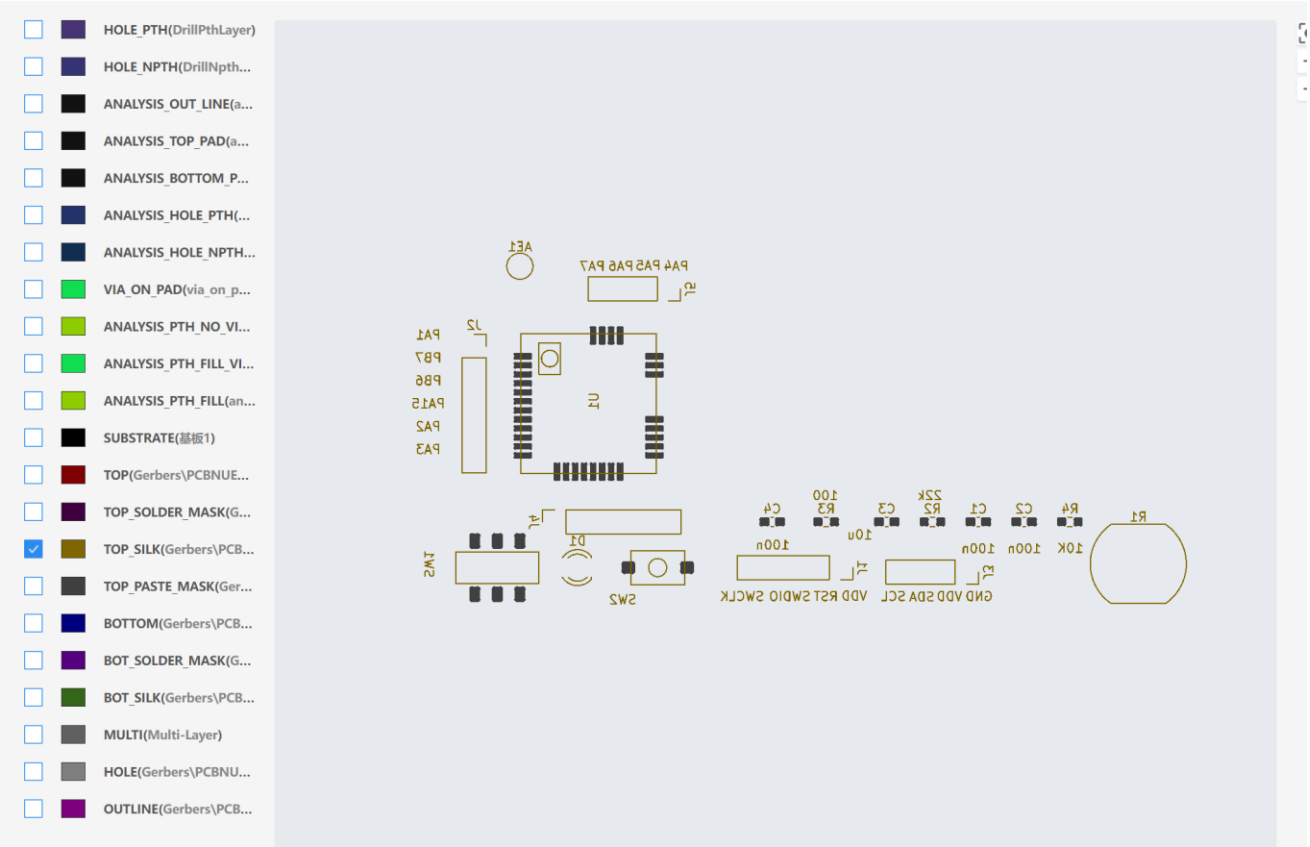

HOLE_PTH(DrillPthLayer)
HOLE_NPTH(DrillNpth...
ANALYSIS_OUT_LINE(a...
ANALYSIS_TOP_PAD(a...
ANALYSIS_BOTTOM_P...
ANALYSIS_HOLE_PTH(...
ANALYSIS_HOLE_NPTH...
VIA_ON_PAD(via_on_p...
ANALYSIS_PTH_NO_VI...
ANALYSIS_PTH_FILL_VI...
ANALYSIS_PTH_FILL(an...
SUBSTRATE(基板1)
TOP(Gerbers\PCBNUE...
TOP_SOLDER_MASK(G...
TOP_SILK(Gerbers\PCB...
TOP_PASTE_MASK(Ger...
BOTTOM(Gerbers\PCB...
BOT_SOLDER_MASK(G...
BOT_SILK(Gerbers\PCB...
MULTI(Multi-Layer)
HOLE(Gerbers\PCBNU...
OUTLINE(Gerbers\PCB...

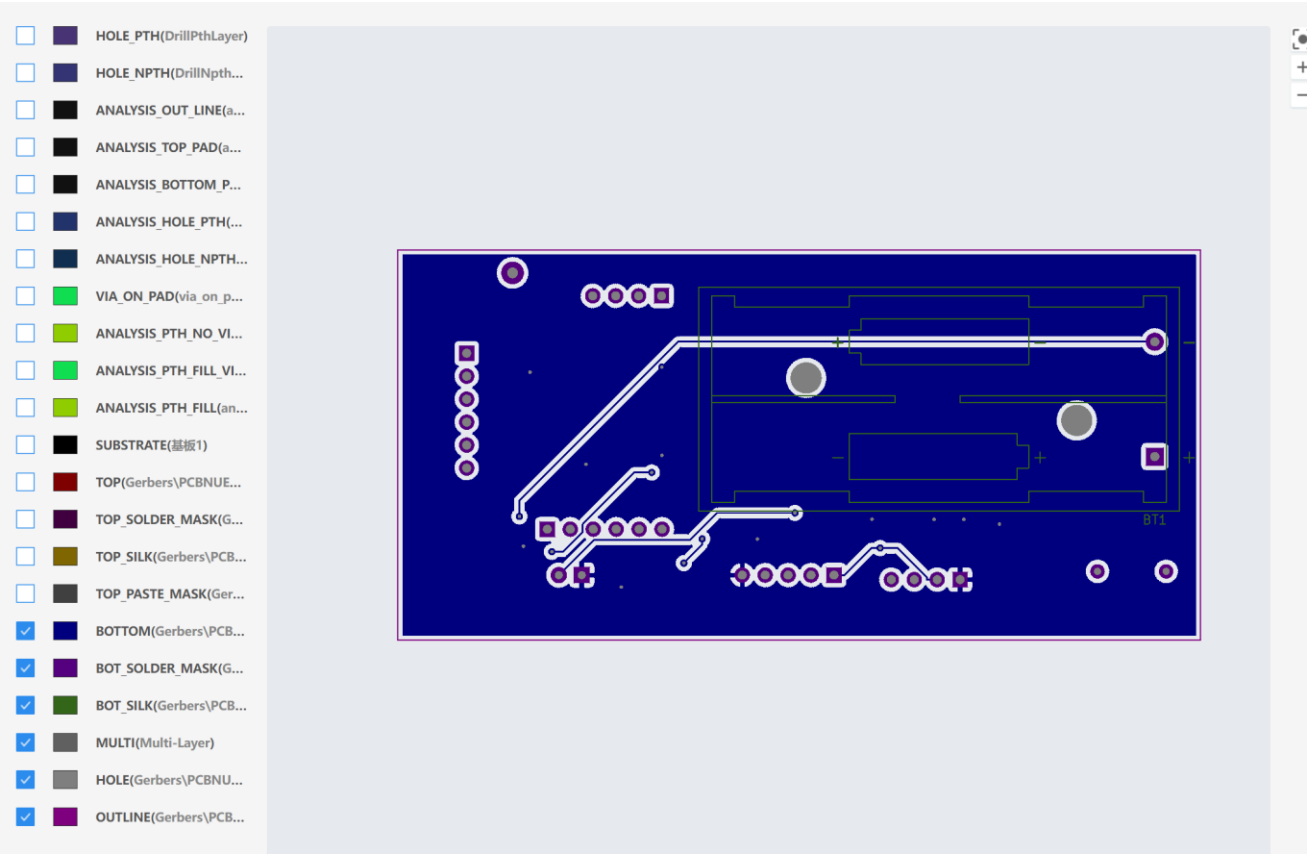

HOLE_PTH(DrillPthLayer)
HOLE_NPTH(DrillNpth...
ANALYSIS_OUT_LINE(a...
ANALYSIS_TOP_PAD(a...
ANALYSIS_BOTTOM_P...
ANALYSIS_HOLE_PTH(...
ANALYSIS_HOLE_NPTH...
VIA_ON_PAD(via_on_p...
ANALYSIS_PTH_NO_VI...
ANALYSIS_PTH_FILL_VI...
ANALYSIS_PTH_FILL(an...
SUBSTRATE(基板1)
TOP(Gerbers\PCBNUE...
TOP_SOLDER_MASK(G...
TOP_SILK(Gerbers\PCB...
TOP_PASTE_MASK(Ger...
BOTTOM(Gerbers\PCB...
BOT_SOLDER_MASK(G...
BOT_SILK(Gerbers\PCB...
MULTI(Multi-Layer)
HOLE(Gerbers\PCBNU...
OUTLINE(Gerbers\PCB...


HOLE_PTH(DrillPthLayer)
HOLE_NPTH(DrillNpth...
ANALYSIS_OUT_LINE(a...
ANALYSIS_TOP_PAD(a...
ANALYSIS_BOTTOM_P...
ANALYSIS_HOLE_PTH(...
ANALYSIS_HOLE_NPTH...
VIA_ON_PAD(via_on_p...
ANALYSIS_PTH_NO_VI...
ANALYSIS_PTH_FILL_VI...
ANALYSIS_PTH_FILL(an...
SUBSTRATE(基板1)
TOP(Gerbers\PCBNUE...
TOP_SOLDER_MASK(G...
TOP_SILK(Gerbers\PCB...
TOP_PASTE_MASK(Ger...
BOTTOM(Gerbers\PCB...
BOT_SOLDER_MASK(G...
BOT_SILK(Gerbers\PCB...
MULTI(Multi-Layer)
HOLE(Gerbers\PCBNU...
OUTLINE(Gerbers\PCB...


HOLE_PTH(DrillPthLayer)
HOLE_NPTH(DrillNpth...
ANALYSIS_OUT_LINE(a...
ANALYSIS_TOP_PAD(a...
ANALYSIS_BOTTOM_P...
ANALYSIS_HOLE_PTH(...
ANALYSIS_HOLE_NPTH...
VIA_ON_PAD(via_on_p...
ANALYSIS_PTH_NO_VI...
ANALYSIS_PTH_FILL_VI...
ANALYSIS_PTH_FILL(an...
SUBSTRATE(基板1)
TOP(Gerbers\PCBNUE...
TOP_SOLDER_MASK(G...
TOP_SILK(Gerbers\PCB...
TOP_PASTE_MASK(Ger...
BOTTOM(Gerbers\PCB...
BOT_SOLDER_MASK(G...
BOT_SILK(Gerbers\PCB...
MULTI(Multi-Layer)
HOLE(Gerbers\PCBNU...
OUTLINE(Gerbers\PCB...


HOLE_PTH(DrillPthLayer)
HOLE_NPTH(DrillNpth...
ANALYSIS_OUT_LINE(a...
ANALYSIS_TOP_PAD(a...
ANALYSIS_BOTTOM_P...
ANALYSIS_HOLE_PTH(...
ANALYSIS_HOLE_NPTH...
VIA_ON_PAD(via_on_p...
ANALYSIS_PTH_NO_VI...
ANALYSIS_PTH_FILL_VI...
ANALYSIS_PTH_FILL(an...
SUBSTRATE(基板1)
TOP(Gerbers\PCBNUE...
TOP_SOLDER_MASK(G...
TOP_SILK(Gerbers\PCB...
TOP_PASTE_MASK(Ger...
BOTTOM(Gerbers\PCB...
BOT_SOLDER_MASK(G...
BOT_SILK(Gerbers\PCB...
MULTI(Multi-Layer)
HOLE(Gerbers\PCBNU...
OUTLINE(Gerbers\PCB...


HOLE_PTH(DrillPthLayer)
HOLE_NPTH(DrillNpth...
ANALYSIS_OUT_LINE(a...
ANALYSIS_TOP_PAD(a...
ANALYSIS_BOTTOM_P...
ANALYSIS_HOLE_PTH(...
ANALYSIS_HOLE_NPTH...
VIA_ON_PAD(via_on_p...
ANALYSIS_PTH_NO_VI...
ANALYSIS_PTH_FILL_VI...
ANALYSIS_PTH_FILL(an...
SUBSTRATE(基板1)
TOP(Gerbers\PCBNUE...
TOP_SOLDER_MASK(G...
TOP_SILK(Gerbers\PCB...
TOP_PASTE_MASK(Ger...
BOTTOM(Gerbers\PCB...
BOT_SOLDER_MASK(G...
BOT_SILK(Gerbers\PCB...
MULTI(Multi-Layer)
HOLE(Gerbers\PCBNU...
OUTLINE(Gerbers\PCB...


HOLE_PTH(DrillPthLayer)
HOLE_NPTH(DrillNpth...
ANALYSIS_OUT_LINE(a...
ANALYSIS_TOP_PAD(a...
ANALYSIS_BOTTOM_P...
ANALYSIS_HOLE_PTH(...
ANALYSIS_HOLE_NPTH...
VIA_ON_PAD(via_on_p...
ANALYSIS_PTH_NO_VI...
ANALYSIS_PTH_FILL_VI...
ANALYSIS_PTH_FILL(an...
SUBSTRATE(基板1)
TOP(Gerbers\PCBNUE...
TOP_SOLDER_MASK(G...
TOP_SILK(Gerbers\PCB...
TOP_PASTE_MASK(Ger...
BOTTOM(Gerbers\PCB...
BOT_SOLDER_MASK(G...
BOT_SILK(Gerbers\PCB...
MULTI(Multi-Layer)
HOLE(Gerbers\PCBNU...
OUTLINE(Gerbers\PCB...
BT1

ANEXO L: FACTURAS DE COMPONENTES

CONTRATO DE COMPRA-VENTA

Estos términos y condiciones, junto con su confirmación de pedido, constituyen un CONTRATO entre nuestra empresa y usted.
A continuación, detallamos su pedido.

Código del pedido	W71053AM14288a9c0f
Fecha del pedido	27-02-2023

FORMA DE PAGO	Tarjeta **Documento de confirmación de compra. El pago de su pedido a Electrónica Embajadores se ha realizado correctamente.**

	Datos de facturación	**Datos de envío**
Empresa	Luis Rubio Fuentes	
Cif/Nif (opcional) #	ES000000000	
Nombre con apellidos	Luis Rubio Fuentes	Luis Rubio Fuentes
Email	[redacted]	[redacted]
Dirección - Línea 1	[redacted]	[redacted]
Dirección - Línea 2	2º2	2º2
Dirección - Línea 3		
Código Postal	[redacted]	[redacted]
Localidad	Madrid	Madrid
Provincia/Estado	Madrid	Madrid
Pais	España	España
Teléfono	[redacted]	[redacted]
Móvil	[redacted]	[redacted]

DATOS DEL PEDIDO

Descripción del producto	**Código**	**Cantidad**	**PRECIO**
Juego 6 Pinzas Antiestáticas	HRG1305	1	8.61€
JBC FL 9582 - Flux - 15 ml - 0046565	HRJ1010	1	6.91€
MBO - Soldadura de Estaño 60%Sn 38%Pb 2%Cu, 0,5 mm 100 g - 121402	HRJ1220	1	7.92€
MBO - Jeringuilla de Estaño en Crema Sn62pb36ag2 30 g	HRJ1401	1	16.33€

			39.77€
Gastos de envío			4.50€

IVA	9.30€
TOTAL (Incluye todos los gastos e impuestos vigentes en el momento de contratación)	**53.57€**

JLCPCB

JiaLiChuang (HongKong) Co., Limited

Unit 21, 28/F, Metropole Square
No.2 On Yiu Street, Shatin, New Territories
HONG KONG, China
support@jlcpcb.com
+86 755 23919769
JLCPCB.COM

Invoice No: 551570[redacted]
Invoice Date: 12/02/2023
Batch No.: W202302122205618
Ship Via: FedEx International Priority

Ship To:

Luis Rubio Fuentes
[redacted]
Madrid Madrid [redacted]
SPAIN
Email: [redacted]
Tel: [redacted]
VAT No:

Billing To:

Luis Rubio Fuentes
[redacted]
Madrid Mad[redacted]
SPAIN
Email [redacted]
Tel: [redacted]
VAT No:

Product	Order Number	QTY	Unit Price	Total
Solder Paste Stencil	SO12302126196	1	USD $7.0000	USD $7.00
Printed Circuit Board	Y1	5	USD $0.6200	USD $3.10

Shipping:	USD $17.15
Subtotal:	USD $10.10
Customs duties & taxes:	USD $4.68
Discount:	-USD $5.00
Grand Total:	USD $26.93

Factura
Farnell Components S.L.

Farnell
AN AVNET COMPANY
es.farnell.com

Parque Empresarial Cityparc
Edificio Londres, 2ª Planta
Ctra. de Hospitalet, 147-149
08940 Cornellá (Barcelona)

Tel: +93 475 88 01
e-mail: contabilidad@farnell.com
Por favor envie comprobante a:
cashteamspain@farnell.com

N° Factura	34[redacted]
Fecha Factura	13 FEB 2023
Fecha Pedido	13 FEB 2023
Nº Cuenta Cliente	8[redacted]
N° Albarán	6563085
Página	1

LUIS RUBIO FUENTES
[redacted]MADRID
MADRID Spain
SPAIN

LUIS RUBIO FUENTES
[redacted]MADRID
MADRID Spain

Dirección de Envío

C.I.F.: 45134735H

Nº Pedido 12/02/23 15.46 PM	Nuestra Ref: 9312[redacted]

N Linea	Cod Farnell / Descripcion	U.E.	Cantidad Enviada	Unitario	Precio Neto	IVA	Importe
1	3168359 NSL 4962. LDR, 1MOHM, 250MW, NSL SERIES Tariff Code: 85334010	UD	4	2.1100	2.1100	21.00	8.44
	PORTES						14.00
	PEDIDO REALIZADO POR LUIS RUBIO FUENTES *** PAGADO POR TARJETA DE CRÉDITO *** Authorisation[redacted] RII AEE : nÂº 2340 REI RPA : nÂº 980						

Conforme a lo establecido en la Ley Orgánica 15/1999, de Protección de Datos de Carácter Personal y su normativa de desarrollo, le informamos que los datos personales proporcionados son confidenciales y forman parte de los ficheros titularidad de FARNELL COMPONENTS, S.L. con la finalidad de gestionar la relación comercial y contractual, para la que son necesarios, así como proporcionarle información referente a nuestros productos y servicios, vinculados directamente con la relación comercial y/o contractual que nos une, ya sea por correo electrónico, postal o fax.
En cualquier caso podrá ejercer los derechos de acceso, rectificación, cancelación y oposición previstos en la ley mediante escrito dirigido a FARNELL COMPONENTS, S.L. Parque Empresarial Cityparc, Edificio Londres 2º, Crta. Hospitales, 147-149 08940 Cornellá Barcelona o mediante correo electrónico a la dirección spmarketing@farnell.com junto con su identificación a través del DNI.

Aviso Importante
La no entrega o cualquier discrepancia en la misma debe notificarse, por escrito a Farnell Components S.L. en un plazo máximo de 3 días desde la recepción del material, en otro caso ninguna reclamación será aceptada. El propietario de la mercancía es Farnell Components S.L. hasta la recepción del pago total de la misma. Condiciones de venta indicadas en el catálogo actual.

IVA %	Base Imponible	IVA
21.00	22.44	4.71

Subtotal	22.44
IVA	4.71
Total Factura	EUR 27.15

Fecha de vencimiento:	Forma de pago:
13 FEB 2023	Payment by credit card

Reg. Merc. Madrid:
Tomo 13.951, Libro 0, Folio 143, Sección 8, hoja M.-228575,
Insc. 1ª - C.I.F.: B82229907

Bank of America N.A. Sucursal en Espana
[redacted]
BIC: BOFAES2X

NUESTRA CUENTA BANCARIA			
BANCO	OFICINA	D.C.	CUENTA
1485	0001	01	0035674015

Polígono Romica - Avenida A, 34
02080 - Albacete
✆ 967 500 426
dieltron@dieltron.com
B02190825

LUIS RUBIO FUENTES
[redacted]

Envio: TIPSA
LUIS RUBIO FUENTES .
[redacted]
CP: [redacted] - Madrid (Madrid)

PS#NMMHITRPR

Factura FAC2[redacted]

Fecha: 13-02-2023

Referencia	Descripción	Cant.	Precio	IMPORTE
Albarán ALB2023A2312 13-02-2023 Su pedido PS#NMMHITRPR				
SW074-DTSH31NB	Micro Pulsador 2 pines 4.3mm SW074	4	0,348	1,39 €
ENVIO	Envio a la peninsula - 211	1	10,000	10,00 €

BASE	IMPUESTOS	TOTAL
11,39 €	IVA 21%: 2,39 €	**13,78 €**

FORMA DE PAGO	RECIBO	IMPORTE	VENCIMIENTO
Redsys	-	13,78 €	Pagado

Dieltron Electrónica, S.L. Inscrita en el Registro Mercantil de Albacete, el 30 de Septiembre de 2014. Hoja AB-4320 Folio 113 Tomo 887 Libro 651 Inscripción 5 CIF B02190825 nº RII-AEE 5207

Commercial Invoice # RAKSZ[redacted]

[redacted]

Page: 1/ 1

Ship To

Luis Rubio Fuentes, Luis Rubio Fuentes

Luis Rubio Fuentes, Luis Rubio Fuentes

[redacted]

[redacted] Madrid

Spain

[redacted]

Ship By

RAKwireless

深圳市南山区桃源街道平山一路新视艺创客公园B栋506室/Room 506, Building B, New Compark, Pingshan First Road, Taoyuan Street, Nanshan District, Shenzhen, Guangdong, China

Ken Zhong +86 755 8610 8311

Bill To

Luis Rubio Fuentes

Luis Rubio Fuentes

[redacted]

[redacted] Madrid

Spain

[redacted]

Notify To

Luis Rubio Fuentes

Luis Rubio Fuentes

[redacted]

[redacted] Madrid

Spain

[redacted]

Reference	Country of Origin	Order Date	Payment Term	Currency	Incoterm	Courier	Reason For Export
[redacted]	China	2023-02-13	Immediate Payment	USD	DAP	-	Permanent / Sold

#	PID	HS CODE	Description	QTY	Unit Price	Amount
1	305045	8542399000	Processing module of Reception Apparatus//RAK3172	2.00	5.99	11.98
2	920201	8544200000	Component of Reception Apparatus//RF cable	1.00	5.00	5.00
3	925036	8517710000	Component of Reception Apparatus//ANT	1.00	7.00	7.00
			Total Quantity:	4.0	Subtotal:	23.98
					TAX/VAT:	reverse charged
					Shipping:	25.59
					Total:	49.57

RAK® | RAKwireless Technology Co. Ltd

inquiry@RAKwireless.com
www.RAKwireless.com
store.RAKwireless.com

Referencia pedido cliente	Pedido RS	Cliente	Código cliente
173824885	173824885	Sr. Luis Rubio Fuentes [redacted]	913[redacted]

Código destino
17091606

DIRECCION ENTREGA PEDIDO
LUIS RUBIO FUENTES
Luis Rubio Fuentes
[redacted]
[redacted]Madrid-MADRID

Albarán **Fecha de envío**
1210[redacted] **13.02.2023**

Artículo	Código RS	Descripción	Cantida d	Unidad de venta	Suministrado en múltiplos de	Precio unitario	Importe línea	IVA
20	8048921	0603 Resistor, 0.1W, 1%,10K	5.000	C/U	5.000	0,001	6,95	21,00%
		Código de exportación HS 85332100 País de origen Taiwan						
50	2644416	Cond cerámico SMD 0805 X7R,0.1uF 50V	25	C/U	25	0,158	3,96	21,00%
		Código de exportación HS 85322400 País de origen China						
		*** SERVICIO ESTANDAR ***						

Albarán **Fecha de envío**
1216423669 **13.02.2023**

Artículo	Código RS	Descripción	Cantida d	Unidad de venta	Suministrado en múltiplos de	Precio unitario	Importe línea	IVA
60	1854604	AA x 2 with lead wire 150mm	2	C/U		0,86	1,72	21,00%
		Código de exportación HS 85366990 País de origen China Código de importación HS 8536699099						
70	8048952	0603 Resistor, 0.1W, 1%,22K	5.000	C/U	5.000	0,002	7,63	21,00%
		Código de exportación HS 85332100 País de origen Taiwan Código de importación HS 8533210000						
80	8048760	0603 Resistor, 0.1W, 1%,100R	5.000	C/U	5.000	0,002	9,92	21,00%
		Código de exportación HS 85332100 País de origen Taiwan Código de importación HS 8533210000						
90	8410815	Capacitor MLCC 10uF 50V X7R 1210 10%	5	C/U	5	1,066	5,33	21,00%
		Código de exportación HS 85322400 País de origen Varios Código de importación HS 8532240000						

*** SIGUE ***

Su pedido queda aceptado bajo las condiciones de ventas publicadas en nuestro catálogo. En caso de de que tenga que realizar una reclamación sobre el contenido de esta factura, por favor contacte con nosotros dentro de los 10 días siguientes a la fecha de emisión de este documento

Pago. Los cheques deben ser enviados a la dirección que aparece reflejada en el principio de este documento, y nominativo a AMIDATA S.A.U. En el caso de que deseen pagar por transferencia, le remitimos nuestros datos bancarios: BNP Paribas, Ribera del Loira, nº28, 28042, Madrid IBAN= ES1301490101110060198001 - SWIFT CODE= BNPAESMSXXX. Les rogamos que indiquen como referencia el número de factura/as que abonen.

Inscrito en el Registro Mercantil de Madrid, tomo 73, libro 67, sección 3ª, folio 24, Hoja 61026-1, inscripción 1ª, con CIF A78913993 No. R.I. AEE Productor 3457

Amidata S.A.U.
Avenida de Bruselas, 6 Planta 3ª
28108 Alcobendas
Madrid

DUPLICADO DE FACTURA

Fecha de Factura	13.02.2023
Pagador nº	[redacted]
CIF	ES451[redacted]
Factura Nº	630[redacted]
Página	2 de 2

DIRECCION ENVIO FACTURA
LUIS RUBIO FUENTES
[redacted]
[redacted]Madrid-MADRID

*** SERVICIO ESTANDAR ***

Portes	7,00
Base Imponible	**42,51**
IVA	8,93
Total - EUR	**51,44**

Valor neto Total	Portes	Base Imponible	%IVA	IVA (EUR)
35,51	7,00	42,51	21,00%	8,93

Pagado con tarjeta de pago

****

NOTA IMPORTANTE : Les informamos que nuestro sistema informático no tiene la flexibilidad de retrasar las fechas de pago por motivo vacacional.Les rogamos que efectúen los pagos sin demoras, a fin de evitar la suspensión de envío de material.

****

MOUSER ELECTRONICS.

1000 North Main Street, Mansfield, TX 76063
Representante de Atención al Cliente: Internet Customer Service
Atención al Cliente: +34 936455263
Crédito: +49 (0)89 520 462 115
CIF# FR 27512466913

Factura - Recibo de tarjeta de crédito
Invoice - Credit Card Receipt

N.º de factura Invoice No.	Fecha de factura Invoice Date	N.º de página Page No.
7[redacted]	03-MAR-23	1 of 1

N.º de la orden de com Purchase Order No.	N.º de seguimiento Master Tracker No.
[redacted]	63[redacted]

INCOTERMS: DDP

Nombre del contacto Contact Name	Envío mediante Ship Via	Número de cliente Customer Num	Términos Terms	Fecha del pedido Order Date
LUIS RUBIO FUENTES	FEDEX INT'L PRIORITY	[redacted]	[redacted]	03-MAR-23

Facturar [redacted] Bill To
RUBIO FUENTES, LUIS
[redacted]
[redacted]
[redacted]
[redacted]MADRID
MADRID
SPAIN

Enviar [redacted] Ship To
LUIS RUBIO FUENTES
[redacted]
[redacted]
[redacted]
MADRID
SPAIN

N.º de línea Line No.	Referencia de Mouser / N.º ref. de cliente/fabricante / Descripción	Mouser Part Number / Customer/MFG Part No. / Description	Cantidad pedida Quantity Ordered	Cantidad enviada Quantity Shipped	Cantidad pendiente Quantity Pending	Precio unitario Unit Price (EUR)	Precio total Extended Price (EUR)
1	474-SEN-13763 Ref. fabricante: SEN-13763 SparkFun Humidity and Temp Se / Herramientas de desarrollo del sensor de temperatura TARIC:8534001900 ECCN:EAR99 COO:US		1	1	0	10.460	10.46
2	538-211140-0100 Ref. fabricante: 211140-0100 Molex ISM 868/915MHz 100mm / Antenas TARIC:8529106900 ECCN:EAR99 COO:CN		3	3	0	2.480	7.44

Mercancía Merchandise	Manipulación Handling	Transporte Freight	21% IVA 21% VAT	Pagado con tarjeta de crédito Paid by credit card	EUR 45.86
17.90	0.00	20.00	7.96		

Información del envío / Shipping Information

Estos artículos son controlados por el Gobierno de los Estados Unidos y cuentan con la autorización pertinente para su exportación solo al país de destino final para su uso por parte de los destinatarios o usuarios finales identificados en el presente documento. No podrán ser revendidos, transferidos ni cedidos a ningún otro país ni a ninguna otra persona que no sea el destinatario final o el usuario o usuarios finales autorizados, ya sea en su forma original o después de haber sido incorporados a otros artículos, sin obtener previamente la aprobación del Gobierno de los Estados Unidos, o bien tal como lo autoricen las leyes y reglamentos de los Estados Unidos.
Fecha de envío: Mar 03, 2023
These items are controlled by the U.S. Government and authorized for export only to the country of ultimate destination for use by the ultimate consignee or end-user(s) herein identified. They may not be resold, transferred, or otherwise disposed of, to any other country or to any person other than the authorized ultimate consignee or end-user(s), either in their original form or after being incorporated into other items, without first obtaining approval from the U.S. government or as otherwise authorized by U.S. law and regulations.

Número de seguimiento y peso facturado	Tracking Number(s) and Billed Weights	
637586069646 0.73 lb		

Representante fiscal en Francia: Fiscal Solutions S.A.R.L, 23 Rue du Clos d'Orléans, Fontenay-sous-Bois, 94 120, FR
Todos los pedidos están sujetos a todos los términos y condiciones mencionados en: http://www.mouser.es/saleterms/
This order is subject to all terms and conditions displayed at: http://www.mouser.es/saleterms/

AZ-Delivery Vertriebs GmbH - Lärchenstraße 10 - 94469 Deggendorf

Luis Rubio Fuentes
[redacted]
Spain

Contact us

Email	info@az-delivery.com
Phone	+49 991/99927827
Taxnumber	**108 /** [redacted]
Vat ID	**ESN2764556C**
Date	**2023/01/31**
Customer	[redacted]
Invoice	**20** [redacted]

Your order no. 404-1446728-8544327 dated 31.01.2023

Invoice 20233347820

Deliverydate 2023/01/31

Pos	Item-no.	Description	Quantity	Unit price	Amount
1	IM-XL1J-5B9V	AZDelivery 3X Adaptador Compatible con FT232RL ,USB a TTL Serial Adaptador Convertidor, Módulo convertidor 3.3 V y 5V Compatible con Arduino con E-Bo	1	12,39	12,39 €

Net amount	12,39 €
VAT 21%	2,60 €
Total amount	**14,99 €**

Thank you for your order on Amazon.
Your order has been paid on 31.01.2023 and has been shipped on 31.01.2023.

RII-AEE register no. 7878; RII-PyA register no. 2363

Geschäftsführer: Malte Horeyseck
AZ-Delivery Vertriebs GmbH,
Lärchenstraße 10, 94469
Deggendorf Deutschland

Tel.: +49 991 99927827
Mail: info@az-delivery.com
Web: https://www.az-delivery.de
Amtsgericht Deggendorf: HRB 4592

USt-IdNr.: DE296200645
Deutsche Postbank AG
IBAN: DE60760100850096194850
BIC: PBNKDEFF

Seite 1/2

MOUSER ELECTRONICS.

Factura - Recibo de tarjeta de crédito
Invoice - Credit Card Receipt

1000 North Main Street, Mansfield, TX 76063
Representante de Atención al Cliente: Internet Customer Service
Atención al Cliente: +34 936455263
Crédito: +49 (0)89 520 462 115
CIF# FR 27512466913

INCOTERMS: DDP

N.º de factura Invoice No.	Fecha de factura Invoice Date	N.º de página Page No.
72[redacted]	13-FEB-23	1 of 1

N.º de la orden de com Purchase Order No.	N.º de seguimiento Master Tracker No.
28[redacted]	63[redacted]

Nombre del contacto Contact Name	Envío mediante Ship Via	Número de cliente Customer Num	Términos Terms	Fecha del pedido Order Date
LUIS RUBIO FUENTES	FEDEX INT'L PRIORITY	[redacted]	V[redacted]	12-FEB-23

Facturar
[redacted]
Bill To

RUBIO FUENTES, LUIS
ATTN:[redacted]
[redacted]
[redacted]
[redacted]
MADRID
SPAIN

Enviar
[redacted]
Ship To

LUIS RUBIO FUENTES
ATTN[redacted]
JO[redacted]
[redacted]
[redacted]MADRID
MADRID
SPAIN

N.º de línea Line No.	Referencia de Mouser / N.º ref. de cliente/fabricante / Descripción	Mouser Part Number / Customer/MFG Part No. / Description	Cantidad pedida Quantity Ordered	Cantidad enviada Quantity Shipped	Cantidad pendiente Quantity Pending	Precio unitario Unit Price (EUR)	Precio total Extended Price (EUR)
1	474-SEN-13763 Ref. fabricante: SEN-13763 SparkFun Humidity and Temp Se / Herramientas de desarrollo del sensor de temperatura TARIC:8534001900 ECCN:EAR99 COO:US		1	1	0	10.460	10.46
2	710-452403012014 Ref. fabricante: 452403012014 Wurth Elektronik WS-SLSV Slide Switch / Conmutadores deslizantes TARIC:8536699099 ECCN:EAR99 COO:CN		3	3	0	2.510	7.53

Mercancía Merchandise	Manipulación Handling	Transporte Freight	21% IVA 21% VAT	Pagado con tarjeta de crédito Paid by credit card	EUR 45.97
17.99	0.00	20.00	7.98		

Información del envío / Shipping Information

Estos artículos son controlados por el Gobierno de los Estados Unidos y cuentan con la autorización pertinente para su exportación solo al país de destino final para su uso por parte de los destinatarios o usuarios finales identificados en el presente documento. No podrán ser revendidos, transferidos ni cedidos a ningún otro país ni a ninguna otra persona que no sea el destinatario final o el usuario o usuarios finales autorizados, ya sea en su forma original o después de haber sido incorporados a otros artículos, sin obtener previamente la aprobación del Gobierno de los Estados Unidos, o bien tal como lo autoricen las leyes y reglamentos de los Estados Unidos.
Fecha de envío: Feb 13, 2023
These items are controlled by the U.S. Government and authorized for export only to the country of ultimate destination for use by the ultimate consignee or end-user(s) herein identified. They may not be resold, transferred, or otherwise disposed of, to any other country or to any person other than the authorized ultimate consignee or end-user(s), either in their original form or after being incorporated into other items, without first obtaining approval from the U.S. government or as otherwise authorized by U.S. law and regulations.

Número de seguimiento y peso facturado	Tracking Number(s) and Billed Weights	
632958010334 0.30 lb		

Representante fiscal en Francia: Fiscal Solutions S.A.R.L, 23 Rue du Clos d'Orléans, Fontenay-sous-Bois, 94 120, FR
Todos los pedidos están sujetos a todos los términos y condiciones mencionados en: http://www.mouser.es/saleterms/
This order is subject to all terms and conditions displayed at: http://www.mouser.es/saleterms/

Factura

Pagado

Nº de referencia de pago YW[redacted]

Vendido por Guangzhouxinqinwangluokejiyouxiangongsi

Fecha de la factura/Fecha de la entrega	02.02.2023
Número de la factura	DS-ASE[redacted]96
Total pendiente	11,99 €

LUIS RUBIO FUENTES
[redacted]
ES

El IVA ha sido declarado por	Amazon Services Europe S.a.r.L.
IVA	LU19647148

Si tienes preguntas sobre tus pedidos, visita https://www.amazon.es/contacto

Dirección de facturación

Luis Rubio Fuentes
[redacted]
Madrid, Madrid[redacted]
ES

Dirección de envío

Luis Rubio Fuentes
[redacted]
Madrid, Madrid,[redacted]
ES

Vendido por

Guangzhouxinqinwangluokejiyouxiangongsi
Tianhequhuanxinanbianjiezhongxingli11hao201fang
Guangzhoushi, Guangdongsheng, 510000
CN

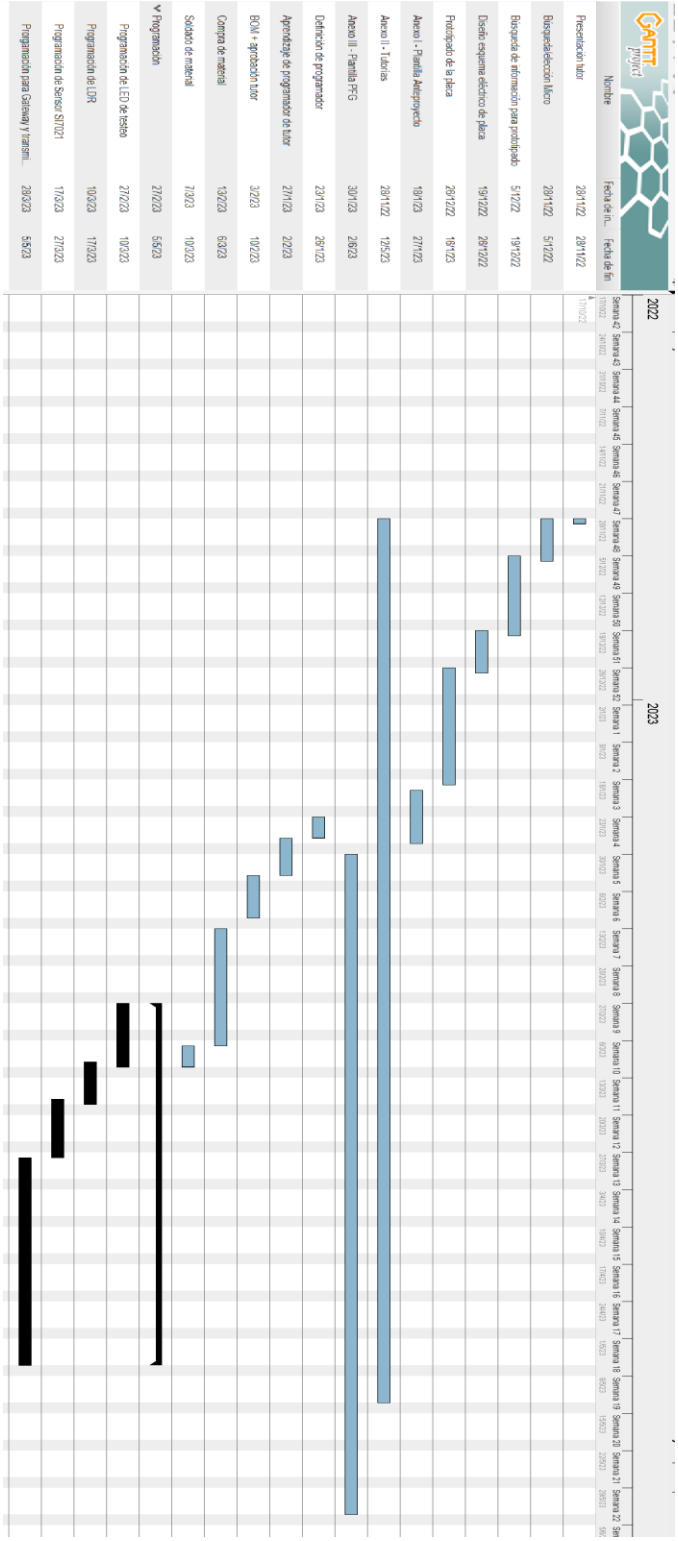
Información del pedido

Fecha del pedido	30.01.2023
Número del pedido	405[redacted]

Detalles de la factura

Descripción	Cant.	P. Unitario (IVA excluido)	IVA %	P. Unitario (IVA incluido)	Precio total (IVA incluido)
ARCELI ST-Link / V2 (Versión CN) ST MCU Microcontrolador STM8 STM32 JTAG SWD Swim Emulador en Circuito del Programador Emulador | B07RKVM7H8 ASIN: B07RKVM7H8	1	9,91 €	21%	11,99 €	11,99 €
Envío		0,00 €		0,00 €	0,00 €

Total 11,99 €

	IVA %	Precio total (IVA excluido)	IVA
	21%	9,91 €	2,08 €
Total		9,91 €	2,08 €

ANEXO M: DIAGRAMA DE GANTT