PROYECTO FIN DE GRADO

TÍTULO: Desarrollo y puesta a punto de un etiquetador para el seguimiento de las actividades diarias de enfermos de Parkinson

AUTOR/A: Adriel Ramos Ayuso

TITULACIÓN: Grado en Ingeniería Electrónica de Comunicaciones

TUTOR/A: Juan Manuel López Navarro

DEPARTAMENTO: Electrónica Física, Ingeniería Eléctrica y Física Aplicada

VºBº TUTOR/A

Miembros del Tribunal Calificador:

PRESIDENTE/A: Francisco Cano Broncano

TUTOR/A: Juan Manuel López Navarro

SECRETARIO/A: José Fernán Martínez Ortega

Fecha de lectura:

Calificación:

El Secretario/La Secretaria,

A mi madre, aunque después de cinco años sigues sin saber el nombre de lo que estudio, te quiero.

A mi padre, por enseñarme que mejor trabajar con la cabeza que con las manos, te quiero.

Agradecimientos

En primer lugar, agradecer a mis padres, soy quien soy por ellos y por todo el sacrificio, gracias por criarme tan bien como lo habéis hecho. También agradecer a mis abuelos, por darme techo, comida y amor durante estos años de estudios.

A mi tutor, Juan Manuel, por la dedicación y ayuda a este proyecto y por las oportunidades ofertadas para mi desarrollo profesional.

También a mis compañero y amigos de la universidad, sin vosotros todo sería peor. Sobre todo, a "Cachimba", por ser ese grupo que se creó en primero y ha perdurado con los años y a Guille, por ser ese desconocido que acabó siendo un hermano.

Y, sobre todo, gracias a mí. A mi yo del pasado, te dije que lo conseguirías. A mi yo del presente, enhorabuena, te lo mereces. A mi yo del futuro, apunta alto que conseguirás lo que te propongas.

" El paso más importante que puede dar alguien es siempre el siguiente" Brandon Sanderson. Juramentada.

Resumen

Este proyecto trata de desarrollar una solución para estudiar las actividades diarias de los pacientes de Parkinson de forma no invasiva y que no perjudique a su privacidad. Además, debido a que una gran cantidad de datos sin agrupar no serían útiles, debemos ser capaces de monitorizar cada una de las actividades y en qué momento se realizaron.

Para lograr esto, el proyecto se basa en el uso de dispositivos de tipo “beacon”, que emiten señales del Bluetooth Low Energy (BLE) al detectar movimiento del objeto o la superficie en la que se encuentran. Estos dispositivos registran las actividades y las transmiten a una aplicación móvil diseñada específicamente para el almacenamiento de las señales emitidas junto a un “timestamp” que indica el momento en el que se ha realizado dicha actividad.

El dispositivo está pensado para colocarse en objetos cotidianos del hogar de los pacientes (electrodomésticos, puertas, etc.) con el fin de registrar y clasificar las acciones diarias y así estudiar cómo estas afectan al desarrollo de la enfermedad de Parkinson. Además, el sistema está diseñado para ser eficiente en el consumo energético, aprovechando las características de bajo consumo de BLE y utilizando modos de ahorro de energía en los microcontroladores.

El proyecto se dividide principalmente en tres fases. En primer lugar, el desarrollo de un firmware específico para los microcontroladores de las balizas, responsable de la detección del movimiento y de la emisión de señales BLE. Este firmware está especialmente pensado para optimizar la vida útil de los dispositivos lo máximo posible. A continuación, se desarrollará una aplicación móvil, basada en el sistema operativo de Android, que se encargará de capturar las señales BLE emitidas por las balizas. Finalmente, con el objetivo de desarrollar una herramienta completa para la detección de las actividades se desarrollará un hardware específico que integrará todos los componentes necesarios para producir dispositivos propios que detecten el movimiento y emitan las señales BLE. De esta forma, se podrá integrar en la solución tanto balizas comerciales de otros fabricantes como las propias desarrolladas durante este proyecto

Abstract

This project tries to develop a solution to study the daily activities of Parkinson's patients in a non-invasive way that does not harm their privacy. In addition, because a large amount of ungrouped data would not be useful, we must be able to monitor each of the activities and at what time they were performed.

To achieve this, the project is based on the use of “beacon” type devices, which emit Bluetooth Low Energy (BLE) signals when they detect movement of the object or surface they are on. These devices record the activities and transmit them to a mobile application specifically designed to store the emitted signals together with a timestamp indicating the time at which the activity was performed.

The device is intended to be placed on everyday objects in patients' homes (household appliances, doors, etc.) in order to record and classify daily actions and thus study how these affect the development of Parkinson's disease. In addition, the system is designed to be energy efficient, taking advantage of the low power consumption features of BLE and using energy saving modes in the microcontrollers.

The project is mainly divided into three phases. First, the development of a specific firmware for the microcontrollers of the beacons, responsible for motion detection and BLE signal emission. This firmware is specially designed to optimize the lifetime of the devices as much as possible. Next, a mobile application will be developed, based on the Android operating system, which will be responsible for capturing the BLE signals emitted by the beacons. Finally, with the aim of developing a complete tool for the detection of the activities, a specific hardware will be developed that will integrate all the necessary components to produce proprietary devices that detect the movement and emit the BLE signals. In this way, it will be possible to integrate in the solution both commercial beacons from other manufacturers and the ones developed during this Project.

Contenido

Agradecimientos ..... **v**

Resumen ..... **vii**

Abstract ..... **ix**

Índice de figuras **1**

Índice de tablas **5**

Índice de ecuaciones **7**

1. Introducción **9**

2. Marco tecnológico **11**

2.1 Comunicaciones inalámbricas: BLE 12

2.1.1 Comunicaciones inalámbricas 12

2.1.2 Bluetooth: Origen e historia. 13

2.1.3 Evolución de Bluetooth 16

2.1.4 Bluetooth Low Energy 19

2.2 Beacons 28

2.2.1 Hardware 28

2.2.2 Protocolos 29

2.2.3 Aplicaciones de los Beacons 30

2.2.4 Beacons comerciales 30

3. Especificaciones y restricciones de diseño **33**

4. Descripción de la solución propuesta **35**

4.1 Fase 1: Desarrollo del Firmware de los microcontroladores 35

4.1.1 nRF Connect SDK 38

4.1.2 Instalación y configuración 39

4.1.3 Desarrollo del firmware 40

4.2 Fase 2: Desarrollo de la app de lectura para señales BLE 51

4.2.1 Android SDK 51

4.2.2 Android Studio 51

4.2.3 Desarrollo de la aplicación 52

4.3 Fase 3: Desarrollo y fabricación de PCBs 77

4.3.1 Herramientas 77

4.3.2 Proveedores 78

4.3.3 Desarrollo de las PCBs 79

5. Resultados **105**

5.1 Pruebas de fase 1: Desarrollo del firmware 105

5.2 Pruebas de fase 2: Desarrollo de app BLEScanner 107

5.2.1 Prueba de concepto 109

5.3 Pruebas de fase 3: Desarrollo de hardware 112

6. Presupuesto **113**

7. Impacto del proyecto **115**

8. Conclusiones 117
8.1 Futuros Trabajos 117
9. Referencias 119
Anexo A: Configuración de beacons comerciales 125
A.1 Tags M1 y M4 Lite 125
Anexo B: proceso de instalación de las herramienta necesarias para comenzar con el desarrollo del firmware 131
B.1 Instalación de nRF Command Line Tools 131
B.2 Instalación de Visual Studio Code 133
B.3 Instalación de la extensión nRF Connect para VSCode 135
B.4 Instalación de la Toolchain 135

Índice de figuras

FIGURA 1. NÚMERO DE ALTAS DE PARKINSON EN LA ÚLTIMA DÉCADA EN ESPAÑA 9
FIGURA 2. TIPOS DE COMUNICACIÓN INALÁMBRICA, VELOCIDAD DE COMUNICACIÓN Y ALCANCE 12
FIGURA 3. ORIGEN DEL SÍMBOLO BLUETOOTH 13
FIGURA 4. CLASIFICACIÓN POR POTENCIA DE DISPOSITIVOS BLUETOOTH 14
FIGURA 5. EVOLUCIÓN DE LOS SERVICIOS BLUETOOTH A LO LARGO DE LAS VERSIONES 17
FIGURA 6. VELOCIDADES Y RANGOS MÁXIMOS DE VERSIONES SIN BLE 17
FIGURA 7. VELOCIDADES Y RANGOS MÁXIMOS DE VERSIONES CON BLE 17
FIGURA 8. PREVISIÓN DE VENTA ANUAL DE DISPOSITIVOS CON BLUETOOTH 18
FIGURA 9. NUMERO DE VENTAS DE BT CLÁSICO VS BT LOW ENERGY VS DISPOSITIVOS DUALES 18
FIGURA 10. VENTAS ANUALES DE DISPOSITIVOS DE SEGUIMIENTO DE BLUETOOTH 19
FIGURA 11. RESUMEN DE LAS CARACTERÍSTICAS TÉCNICAS DE BLE 20
FIGURA 12. PILA DE PROTOCOLO BLE 20
FIGURA 13. TOPOLOGÍA CONECTADA DE BLE 21
FIGURA 14. EJEMPLO DE USO DE TOPOLOGÍAS MULTI-ROL 22
FIGURA 15. TOPOLOGÍA DE DIFUSIÓN DE BLE 22
FIGURA 16. ESTRUCTURA CAPA GATT 23
FIGURA 17. DISTRIBUCIÓN DE LOS CANALES DE RADIO BLE 24
FIGURA 18. ESQUEMA EJEMPLIFICATIVO DE INTERVALO Y VENTANA DE ESCANEO 24
FIGURA 19. TIPOS DE PERIFÉRICOS EN ADVERTISING BLE 25
FIGURA 20. TRAMA BLE 25
FIGURA 21. PDU Y SU CABECERA DE ADVERTISING 25
FIGURA 22. PDU Y DATOS DE ADVERTISING 26
FIGURA 23. DATOS DE ADVERTISING 26
FIGURA 24. COMPARACIÓN ENTRE BLUETOOTH Y BLE 27
FIGURA 25. HARDWARE DE BEACON DE EJEMPLO 28
FIGURA 26. PARÁMETROS CARACTERÍSTICOS DE IBEACON 29
FIGURA 27. BALIZA COMERCIAL MOKO M1 31
FIGURA 28. BALIZA COMERCIAL MOKO M4 LITE 31
FIGURA 29. MICROCONTROLADORES DE LA FAMILIA NRF51 (1) 35
FIGURA 30. MICROCONTROLADORES DE LA FAMILIA NRF51 (2) 35
FIGURA 31. NRF51 DK 36
FIGURA 32. DIAGRAMA DE BLOQUES DE NRF51 DK 36
FIGURA 33. STEVAL-MKI206V1 DE STMICROELECTRONICS 37
FIGURA 34. CONECTORES Y UBICACIÓN DE PINES DE NRF51 DK 37
FIGURA 35. FUNCIÓN DE LOS PINES DEL NRF51 DK 37
FIGURA 36. MONTAJE DE PROTOTIPADO 38
FIGURA 37. ARQUITECTURA DEL NRF CONNECT SDK[49] 39
FIGURA 38. DIAGRAMA DE COMPONENTES DE FIRMWARE1.0 40
FIGURA 39. DIAGRAMA DE PAQUETES DE FIRMWARE1.0 41
FIGURA 40. DIAGRAMA DE SECUENCIA DE FIRMWARE1.0 41
FIGURA 41. DIAGRAMA DE ACTIVIDAD DE FIRMWARE1.0 42
FIGURA 42. DIAGRAMA DE COMPONENTES DE FIRMWARE2.0 45
FIGURA 43. DIAGRAMA DE PAQUETES DE FIRMWARE2.0 45
FIGURA 44. DIAGRAMA DE SECUENCIA DE FIRMWARE2.0 46
FIGURA 45. DIAGRAMA DE ACTIVIDAD DE FIRMWARE2.0 47
FIGURA 46. DIAGRAMA DE CLASES DE BLESCANNER 53
FIGURA 47. DIAGRAMA DE COMPONENTES DE BLESCANNER 53
FIGURA 48. DIAGRAMA DE SECUENCIA DE BLESCANNER 54
FIGURA 49. DIAGRAMA DE CASOS DE USO DE BLESCANNER 54

FIGURA 50. DIAGRAMA DE ACTIVIDAD DE BLESCANNER....55
FIGURA 51. ICONO DE BLESCANNER....59
FIGURA 52. PÁGINA PRINCIPAL BLESCANNER....60
FIGURA 53. PÁGINA DE HISTORIAL BLESCANNER....61
FIGURA 54. DIAGRAMA DE CLASES DE BLESCANNERV2....63
FIGURA 55. DIAGRAMA DE COMPONENTES DE BLESCANNERV2....63
FIGURA 56. DIAGRAMA DE SECUENCIA DE BLESCANNERV2....64
FIGURA 57. DIAGRAMA DE CASOS DE USO DE BLESCANNERV2....64
FIGURA 58. DIAGRAMA DE ACTIVIDAD DE BLESCANNERV2....65
FIGURA 59. NOTIFICACIÓN QUE INDICA LA EJECUCIÓN DEL SERVICIO....69
FIGURA 60. ICONO DE BLESCANNERV2....70
FIGURA 61. PANTALLA PRINCIPAL DE BLESCANNERV2....70
FIGURA 62. HISTORIAL DE BLESCANNERV2....71
FIGURA 63. PUNTO DE ACCESO A LOS ARCHIVOS DEL DISPOSITIVO....73
FIGURA 64. CARPETA DOCUMENTS....73
FIGURA 65. CARPETA DE LA APP....74
FIGURA 66. CARPETA DE AÑOS....74
FIGURA 67. CARPETA DE MESES....74
FIGURA 68. CARPETA DE DÍAS....75
FIGURA 69. FICHERO HISTORY.TXT....75
FIGURA 70. CONTENIDO DEL FICHERO HISTORY.TXT....75
FIGURA 71. LOGO DE KICAD....77
FIGURA 72. LOGO DE EASYEDA....77
FIGURA 73. ESQUEMÁTICO COMPLETO DE BLE_PCB_V1....80
FIGURA 74. SÍMBOLO DEL ESQUEMÁTICO DEL NRF51422....80
FIGURA 75.ALIMENTACIÓN Y COMPONENTES DE DESACOPLO DE BLE_PCB_V1....81
FIGURA 76. RELOJ DE 16MHZ DE BLE_PCB_V1....81
FIGURA 77. RELOJ DE 32KHZ DE BLE_PCB_V1....82
FIGURA 78. BATERÍA DE BLE_PCB_V1....82
FIGURA 79. CONFIGURACIÓN DE CONECTOR FTSH....82
FIGURA 80. CONFIGURACIÓN DE ANTENA DE BLE_PCB_V1....83
FIGURA 81. TIRAS DE PINES PARA LA INSERCIÓN DE STEVAL-MKI206V1....83
FIGURA 82. LÍNEA DE SCL DEL MICROCONTROLADOR NRF51422....84
FIGURA 83. LÍNEA DE SCL DEL MICROCONTROLADOR NRF51422....84
FIGURA 84. DISEÑO DE PCB DE BLE_PCB_1....84
FIGURA 85. COMPONENTES DE CONFIGURACIÓN DE ANTENA Y SEÑAL BLE_PCB_V1....85
FIGURA 86. UBICACIÓN DE LOS OSCILADORES DE BLE_PCB_V1....85
FIGURA 87. FOOTPRINT DE LAS TIRAS DE 12 PINES PARA INSERCIÓN DE PLACA CON ACELERÓMETRO....86
FIGURA 88. UBICACIÓN DE CONECTOR FTSH Y PINES SWD....86
FIGURA 89. PLANO DE MASA DE CAPA SUPERIOR (UP)....87
FIGURA 90. PLANO DE MASA DE CAPA INFERIOR (DOWN)....87
FIGURA 91. SIMULACIÓN 3D DE LA PARTE SUPERIOR DE LA PCB....88
FIGURA 92. SIMULACIÓN 3D DE LA PARTE SUPERIOR DE LA PCB....88
FIGURA 93. STENCIL BLE_PCB_V1....88
FIGURA 94. PARTE SUPERIOR BLE_PCB_V1 SIN ENSAMBLAR....89
FIGURA 95. PARTE INFERIOR BLE_PCB_V1 SIN ENSAMBLAR....89
FIGURA 96. PARTE SUPERIOR BLE_PCB_V1 CON COMPONENTES SOLDADOS....90
FIGURA 97. PARTE INFERIOR BLE_PCB_V1 CON COMPONENTES SOLDADOS....90
FIGURA 98. ANÁLISIS DE ERROR DE DISEÑO EN PIN SWDCLK....91
FIGURA 99. CONEXIONADO DEL CONDENSADOR C2 EN BLE_PCB_V1....91
FIGURA 100. CONEXIONADO CORRECTO DEL CONDENSADOR DE DESACOPLO....92
FIGURA 101. ESQUEMÁTICO COMPLETO DE BLE_PCB_V2....93

FIGURA 102. SÍMBOLO DE NRF51822 93
FIGURA 103. COMPONENTES DE DESACOPLO BLE_PCB_V2 94
FIGURA 104. CORRECCIÓN DEL ERROR EN COMPONENTES DE DESACOPLO DE LA VERSIÓN 1 94
FIGURA 105. INDICADOR LED BLE_PCB_V2 94
FIGURA 106. RELOJES DE 16MHZ Y 32KHZ BLE_PCB_V2 95
FIGURA 107. BATERÍA DE BLE_PCB_V2 95
FIGURA 108. CONFIGURACIÓN DE CONECTOR FTSH 95
FIGURA 109. CONFIGURACIÓN DE ANTENA DE BLE_PCB_V2 MEDIANTE BALUN 96
FIGURA 110. ESQUEMÁTICO Y CONFIGURACIÓN HARDWARE DEL AIS2DW12 96
FIGURA 111. DISEÑO DE PCB DE BLE_PCB_V2 97
FIGURA 112. COMPONENTES DE CONFIGURACIÓN DE ANTENA Y SEÑAL BLE_PCB_V2 97
FIGURA 113. UBICACIÓN DE LOS OSCILADORES DE BLE_PCB_V2 98
FIGURA 114. ACELERÓMETRO EN BLE_PCB_V2 98
FIGURA 115. UBICACIÓN DE CONECTOR FTSH Y PINES SWD 99
FIGURA 116. PLANO DE MASA DE CAPA SUPERIOR (UP) 99
FIGURA 117. PLANO DE MASA DE CAPA INFERIOR (DOWN) 100
FIGURA 118. SIMULACIÓN 2D DE LA PARTE SUPERIOR DE LA PCB 100
FIGURA 119. SIMULACIÓN 2D DE LA PARTE INFERIOR DE LA PCB 101
FIGURA 120. MODELO 3D DE LA PARTE SUPERIOR BLE_PCB_V2 101
FIGURA 121. MODELO 3D DE LA PARTE INFERIOR BLE_PCB_V2 101
FIGURA 122. PARTE SUPERIOR BLE_PCB_V2 SIN ENSAMBLAR 102
FIGURA 123. PARTE INFERIOR BLE_PCB_V1 SIN ENSAMBLAR 102
FIGURA 124. PARTE SUPERIOR BLE_PCB_V1 CON COMPONENTES SOLDADOS 103
FIGURA 125. PARTE INFERIOR BLE_PCB_V1 CON COMPONENTES SOLDADOS 103
FIGURA 126. ANÁLISIS DE ERROR DE DISEÑO EN PIN SWDCLK 104
FIGURA 127. MEDIDAS DE ACELERÓMETRO EN REPOSO 105
FIGURA 128. MENSAJE DE ACTIVACIÓN DE SEÑAL BLE 106
FIGURA 129. APLICACIÓN SIN DETECCIÓN 106
FIGURA 130. APLICACIÓN CON DETECCIÓN 106
FIGURA 131. MENSAJE DE ACTIVACIÓN DE BLE EN FIRMWARE 2.0 107
FIGURA 132. BEACON INTEGRADO EN MICROONDAS 109
FIGURA 133. BEACON INTEGRADO EN NEVERA 109
FIGURA 134. BEACON INTEGRADO EN CAJÓN 110
FIGURA 135. BEACON INTEGRADO EN SILLA 110
FIGURA 136. BEACONS INTEGRADOS EN MESA Y MANDO DE TELEVISIÓN. 111
FIGURA 137. APLICACIÓN DE CONFIGURACIÓN PARA TAG M1 125
FIGURA 138. APLICACIÓN DE CONFIGURACIÓN PARA TAG M4LITE 125
FIGURA 139. SELECCIÓN DE FABRICANTE EN APP DE M1 126
FIGURA 140. PROCESO DE CONEXIÓN PARA LA CONFIGURACIÓN 126
FIGURA 141. CONTRASEÑA PARA LA CONEXIÓN 127
FIGURA 142. SLOTS CONFIGURABLES DEL DISPOSITIVO 127
FIGURA 143. COCONFIGURACIÓN DE SLOT1 128
FIGURA 144. CONFIGURACIÓN DEL TRIGGER 129
FIGURA 145. PÁGINA DE DESCARGA DE NRF COMMAND LINE TOOL 131
FIGURA 146. INSTALACIÓN DE NRF COMMAND LINE TOOL (1) 131
FIGURA 147.INSTALACIÓN DE NRF COMMAND LINE TOOL (2) 132
FIGURA 148. INSTALACIÓN DE NRF COMMAND LINE TOOL (3) 132
FIGURA 149. INSTALACIÓN DE NRF COMMAND LINE TOOL (4) 133
FIGURA 150. INSTALACIÓN DE NRF COMMAND LINE TOOL (5) 133
FIGURA 151. INSTALACIÓN DE NRF COMMAND LINE TOOL (6) 135
FIGURA 152. INSTALACIÓN DE LA EXTENSIÓN NRF CONNECT PARA VSCODE 135
FIGURA 153. INSTALAR TOOLCHAIN 135

Índice de tablas

TABLA 1. CÁLCULO DE RANGO DE ALCANCE 15
TABLA 2. BANDA DE FRECUENCIAS Y CANALES POR REGIONES 15
TABLA 3. CARACTERÍSTICAS TÉCNICAS BLUETOOTH 16
TABLA 4. PRINCIPALES MEJORAS ENTRE VERSIONES 16
TABLA 5. LIBRERÍAS INTEGRADAS EN EL FIRMWARE1.0 43
TABLA 6. FUNCIONES DE FIRMWARE1.0 43
TABLA 7. PERMISOS NECESARIOS PARA BLESCANNER 52
TABLA 8. ESTRUCTURA DE BLESCANNER 56
TABLA 9. LIBRERÍAS IMPLEMENTADAS EN BLESCANNER 56
TABLA 10. MÉTODOS DE MAINACTIVITY DE BLESCANNER 57
TABLA 11. MÉTODOS DE HISTORYACTIVITY DE BLESCANNER 58
TABLA 12. MÉTODOS DE HISTORYITEM DE BLESCANNER 58
TABLA 13. PERMISOS AÑADIDOS EN BLESCANNERV2 62
TABLA 14. ESTRUCTURA DE BLESCANNERV2 66
TABLA 15. MÉTODOS DE MAINACTIVITY DE BLESCANNERV2 67
TABLA 16. MÉTODOS DE HISTORYRESETRECEIVER DE BLESCANNERV2 67
TABLA 17. MÉTODOS DE BLUETOOTHFOREGROUNDSERVICE DE BLESCANNERV2 68
TABLA 18. TEST REALIZADOS A AMBAS VERSIONES 107
TABLA 19. RESULTADOS DE LAS MEDIDAS DEL PRIMER BLOQUE 109
TABLA 20. RESULTADOS DE LAS MEDIDAS DEL PRIMER BLOQUE 110
TABLA 21. RESULTADOS DE LAS MEDIDAS DEL PRIMER BLOQUE 111
TABLA 22. RESULTADOS DE PRUEBAS HARDWARE EN BLE_PCB_V1 112
TABLA 23. RESULTADOS DE PRUEBAS HARDWARE EN BLE_PCB_V2 112
TABLA 24. PRESUPUESTO DE DISPOSITIVOS ELECTRÓNICOS 113
TABLA 25. PRESUPUESTO DESARROLLO HARDWARE 113
TABLA 26. PRESUPUESTO DE MANO DE OBRA, LICENCIAS Y EQUIPAMIENTO 113

Índice de ecuaciones

Ecuación 1. Cálculo de la intensidad de una onda esférica..... 14
Ecuación 2. Cálculo de la distancia recorrida por una onda..... 14
Ecuación 3. Conversión de dBm en mW 15
Ecuación 4. Aplicación de ecuación 3..... 15

1. Introducción

Hoy en día la tecnología se ha abierto camino en todos los aspectos imaginables, desde la educación hasta el ocio, pasando por la medicina. Actualmente, médicos de todo el mundo usan la tecnología para detectar o tratar diversas enfermedades. No es ningún secreto que la tecnología ha avanzado de la mano con la medicina en estos últimos años, según Carmen Ruiz-Villar, jefa del departamento de Productos Sanitarios de la Agencia Española de Medicamentos y Productos Sanitarios (AEMPS) "desde la Agencia estamos en continuo contacto con las nuevas tecnologías. Raro es la semana que no recibimos consultas respecto al desarrollo de nuevos dispositivos."[2].

A nivel mundial, la Organización Mundial de la Salud (OMS) asegura que existen aproximadamente 8.5 millones de personas con Parkinson[3]. Además, a nivel nacional, como se puede ver en la figura 1, en 2020 se registraron en España un total de 39.384 altas de pacientes con Parkinson según el ministerio de Sanidad[4], [5].

Número de altas de personas de 15 y más años que padecen enfermedad de Parkinson, en un año

15+	2012	2013	2014	2015	2016	2017	2018	2019	2020
Hombre	19.801	20.247	20.721	21.637	19.851	21.783	22.955	23.187	21.573
Mujer	18.283	18.100	18.635	19.591	16.857	19.114	19.651	19.647	17.811
Ambos sexos	38.084	38.347	39.356	41.228	36.708	40.897	42.606	42.834	39.384

Figura 1. Número de altas de Parkinson en la última década en España [5]

En la actualidad, el Parkinson es una de las enfermedades neurodegenerativas más importantes a nivel mundial, afectando a millones de personas en todo el mundo y limitando significativamente su calidad de vida. A pesar de los avances tecnológicos en el tratamiento de la enfermedad, uno de los grandes desafíos sigue siendo conseguir una monitoreo efectivo y continuo de las actividades diarias de los pacientes, lo cual permite a los médicos comprender como progresa la enfermedad y como el tratamiento afecta a la vida de los pacientes. El monitoreo de tipo *"freeliving"* que se aplica es costoso, invasivo y difícil de integrar en la vida de los pacientes, lo que limita la eficiencia.

En el contexto de la investigación clínica y el seguimiento de pacientes con Parkinson, el término *"freeliving"* se refiere a las condiciones en las que los pacientes son monitorizados mientras llevan a cabo sus actividades diarias normales, sin estar restringidos o controlados por un entorno clínico. Es decir, se refiere al comportamiento del paciente en su vida cotidiana, fuera del entorno hospitalario o de laboratorio. Este tipo de monitoreo es fundamental para entender el estado funcional y motor del paciente de Parkinson en su entorno real, ya que los síntomas y el rendimiento motor pueden variar significativamente entre el ambiente clínico y la vida diaria. Herramientas como dispositivos de monitoreo portátiles (por ejemplo, wearables, acelerómetros o sensores de movimiento) son comúnmente utilizados para obtener datos sobre el estado motor, la movilidad y otros

parámetros en estas condiciones *"freeliving"*. Este enfoque es útil para evaluar la eficacia de los tratamientos, la progresión de la enfermedad y el impacto que tienen los síntomas del Parkinson en la calidad de vida diaria del paciente.

Este proyecto surge con la intención de ofrecer una solución no invasiva que permita realizar el seguimiento de las actividades diarias de los pacientes respetando su privacidad y sin interferir con su rutina diaria. El proyecto se centra en el desarrollo de un sistema basado en dispositivos tipo beacon, los cuales detectan movimiento y transmiten información mediante Bluetooth Low Energy. La idea es integrar estos dispositivos en objetos cotidianos de la casa, como electrodomésticos, puertas o muebles, de forma que sean capaces de detectar movimientos básicos sobre estas superficies, los cuales identifiquen actividades como comer, salir a la calle o subir escaleras, sin la necesidad de instalar cámaras, las cuales supondrían un gran cambio en el día a día de los pacientes debido a la pérdida de intimidad que esto supone. La información emitida por los beacons es recogida en una aplicación desarrollada específicamente con este motivo, donde las señales emitidas por los dispositivos quedan registradas junto a su marca temporal o timestamp para identificar de forma inequívoca cada una de las actividades y en qué momento se ha realizado.

El proyecto se ha dividido principalmente en tres fases. En la primera fase, se ha desarrollado el firmware que corre en los microcontroladores de lo beacons, que permite detectar el movimiento de las superficies donde los dispositivos están instalados y emitir señales BLE de una forma eficiente en cuanto a la gestión de la batería de los dispositivos. En la segunda fase se ha desarrollado una aplicación móvil para Android que recibe las señales emitidas por los dispositivos y las almacena junto a su timestamp. La tercera fase consiste en el desarrollo y fabricación de un hardware específico para las balizas, con el que se ha logrado reducir al máximo el tamaño de los dispositivos y optimizar el consumo de energía, lo que hace que sea discreto y facilita la instalación de los dispositivos en la casa del paciente.

Gracias al desarrollo de estas tres fases, se ha conseguido un sistema completo de monitoreo de actividades que facilita el seguimiento del paciente de forma no intrusiva y cómoda para el paciente.

En cuanto a la estructura de la memoria, el capítulo 1 introduce el contexto y la relevancia del proyecto, además de exponer los objetivos planteados. En el capítulo 2, se presenta el marco tecnológico, detallando las tecnologías empleadas, como Bluetooth Low Energy (BLE) y el diseño de sistemas embebidos. El capítulo 3 aborda las especificaciones y restricciones del diseño, mientras que el capítulo 4 describe la solución propuesta y su desarrollo en fases. Los capítulos 5 y 6 presentan los resultados obtenidos y el presupuesto, respectivamente, mientras que el capítulo 7 se enfoca en el impacto del proyecto. Finalmente, el capítulo 8 concluye con las consideraciones finales y posibles líneas de trabajo futuro.

2. Marco tecnológico

En la actualidad, el continuo desarrollo de la tecnología y su intersección con otros campos como la biomedicina están generando innovaciones importantes en la forma en la que monitoreamos, diagnosticamos y tratamos muchas enfermedades o condiciones de salud.

En concreto, el avance en tecnologías de comunicación inalámbrica y el desarrollo de plataformas móviles está permitiendo crear soluciones muy eficientes a la hora de recopilar y analizar datos biométricos en tiempo real. Estas tecnologías no solo facilitan el seguimiento de parámetros vitales y de comportamiento, si no que abren multitud de nuevas posibilidades para el cuidado personalizado y la mejora de calidad de vida de los pacientes.

El marco tecnológico en este ámbito se caracteriza por un crecimiento exponencial de nuevas tecnologías de transmisión, donde el Bluetooth clásico ha quedado apartado por su evolución Bluetooth Low Energy (BLE) debido a su alta eficiencia y bajo consumo, lo que permite crear dispositivos portátiles y sensores médicos que transmitan la información recolectada del paciente por grandes periodos de tiempo sin preocuparnos por la recarga de baterías.

Al mismo tiempo, el ecosistema de aplicaciones móviles evoluciona ofreciendo plataformas robustas y flexibles, las cuales pueden adaptarse a la necesidad de todos los usuarios. Con las herramientas que hoy en día se encuentran disponibles, pueden programarse aplicaciones con multitud de funciones y con una interfaz intuitiva, lo que permite que cualquier tipo de usuario pueda interactuar con el software.

Paralelamente, el diseño de Placas de Circuito Impreso (PCB) ha crecido considerablemente. Actualmente existen muchas herramientas software que nos facilitan el diseño de un hardware personalizado para una función concreta, naciendo así los sistemas embebidos. El desarrollo de un hardware a medida y optimizado es crucial para los dispositivos médicos, donde la miniaturización y el control del consumo de energía son puntos clave. Además, cabe destacar la existencia de empresas especializadas en la construcción de las PCB, de forma que puede obtenerse un acabado profesional de forma muy económica y olvidando las técnicas más rudimentarias para realizar circuitos impresos.

En esta sección se explorarán en profundidad las tecnologías clave para los avances comentados. Se revisará con detalle el papel del BLE en la comunicación y transmisión de datos, además de las tecnologías de soporte para la detección de movimiento o el estudio de la actividad en enfermos de Parkinson. Se analizará la importancia de Android como ecosistema para el despliegue de aplicaciones móviles. Por último, se estudiarán las técnicas de diseño de PCBs más comunes.

Estos elementos son esenciales para comprender el marco tecnológico que rodea al proyecto, así como también para comprender las bases de futuras innovaciones en el campo de la biomedicina o la salud digital.

2.1 Comunicaciones inalámbricas: BLE

2.1.1 Comunicaciones inalámbricas

Una comunicación inalámbrica se define como cualquier tipo de transmisión de información sin utilizar un medio de conexión cableado. La información se puede transmitir entre dos o más puntos que no se encuentran físicamente conectados. Una de las ventajas es que la distancia que la información recorre puede ser de pocos centímetros hasta miles de kilómetros como en sistemas GPS.

Actualmente, las comunicaciones inalámbricas se encuentran en la mayoría de los dispositivos como teléfonos móviles, televisión por satélite o auriculares inalámbricos. Todos estos dispositivos tienen un punto en común, usan una comunicación sin cable físico para compartir un tipo de información específica. A continuación, en la Figura 2, podemos ver diferentes tipos de comunicaciones inalámbricas junto a su alcance y velocidad de comunicación:

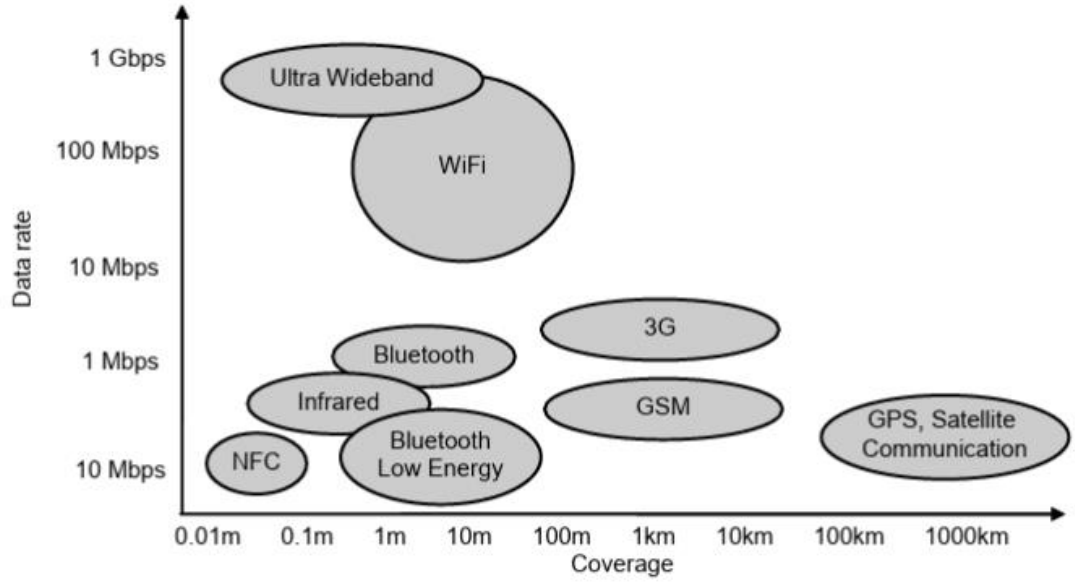


Figura 2. Tipos de comunicación inalámbrica, velocidad de comunicación y alcance [6]

2.1.1.1 Características de la comunicación inalámbrica

- **Movilidad:** Debido a que la comunicación no queda restringida por el medio físico cableado, los usuarios pueden desplazarse manteniendo la comunicación.
- **Comodidad:** Debido a la ausencia de cables, la mayoría de los dispositivos permiten un uso más cómodo y despreocupado respecto a sus modelos cableados.
- **Reducción de coste:** ahorro significativo en el despliegue de infraestructuras de comunicación debido a la ausencia de conexión física entre dispositivos.
- **Seguridad y servicios:** Las comunicaciones inalámbricas permiten que los usuarios soliciten servicios de emergencia como el 911 en cualquier parte del mundo, de forma que puedan ser localizados y asistidos en caso de necesidad.

2.1.2 Bluetooth: Origen e historia.

De forma genérica, podemos definir Bluetooth como una tecnología de comunicación inalámbrica de corto alcance (por debajo de un kilómetro)[7] y de bajo consumo que permite a dos o más dispositivos comunicarse entre sí a través de ondas de radio.

De una forma más técnica podemos considerar el Bluetooth de las siguientes dos formas:

- **Estándar:** Bluetooth es un estándar de comunicación inalámbrica y como estándar, establece las especificaciones técnicas que permiten la interoperabilidad entre dispositivos de distintos fabricantes. Es por esto por lo que Bluetooth puede ser usado por dispositivos tan diferentes entre sí.

- **Protocolo:** Bluetooth también se define como protocolo, ya que define como deben ser las reglas y procedimientos en la comunicación que los dispositivos deben seguir para establecer y mantener una comunicación

Bluetooth surgió en 1994, creado por la compañía sueca Ericson[8]. Esta tecnología recibe su nombre en honor a Harald Blåtand[9], [10], rey de Dinamarca desde 940 y rey de Noruega desde 976 hasta su muerte en 985, al cual se le atribuye la unificación de Dinamarca y Noruega, tal y como Bluetooth busca unificar la comunicación entre dispositivos. El nombre de la tecnología previene de su primer apellido, Blåtand, al inglés se traduce directamente como "Bluetooth"[11], lo cual iba a ser un nombre temporal que finalmente se hizo definitivo[9]. De igual forma, el símbolo de la tecnología se origina de Harald. Como podemos ver en la figura, surge como la combinación de dos runas del antiguo futhark nórdico, la runa «Hagall» (ᚼ) y la runa «Bjarkan» (ᛒ)[12]

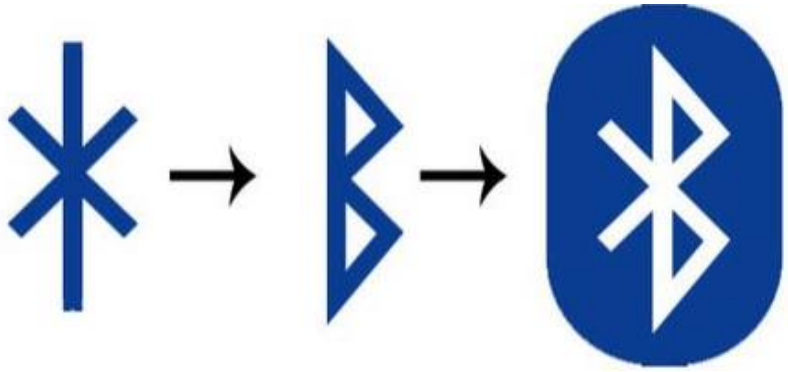


Figura 3. Origen del símbolo Bluetooth[13]

En febrero de 1998, los principales productores de telefonía entre los que se pueden destacar Ericcson, Intel, Nokia o Toshiba, fundaron *The Bluetooth Special Interest Group* (SIG) con la intención de desarrollar una especificación de conectividad inalámbrica de corto alcance[8].

En mayo de 1998 el SIG anuncia su intención, para que un año más tarde, en el verano de 1999 se publique la especificación de Bluetooth 1.0[9]

En la actualidad el SIG está formado por más de 1900 empresas, promovido principalmente por 3COM, Microsoft, Lucent y Motorola.

2.1.2.1 Principales características técnicas

- **Rango de alcance:** es la distancia a la que los dispositivos pueden comunicarse. Desde la publicación de la especificación de Bluetooth 4.0 [14] en 2010, los dispositivos bluetooth se clasifican en tres grupos según su rango. En la Figura 4, extraída de la especificación podemos ver la clasificación de los dispositivos:

Power Class	Maximum Output Power (Pmax)	Nominal Output Power	Minimum Output Power[1]	Power Control
1	100 mW (20 dBm)	N/A	1 mW (0 dBm)	Pmin<+4 dBm to Pmax Optional: Pmin[2] to Pmax
2	2.5 mW (4 dBm)	1 mW (0 dBm)	0.25 mW (-6 dBm)	Optional: Pmin[2] to Pmax
3	1 mW (0 dBm)	N/A	N/A	Optional: Pmin[2] to Pmax

Figura 4. Clasificación por potencia de dispositivos Bluetooth[14]

Estas potencias de transmisión pueden transformarse fácilmente en la distancia máxima de comunicación entre dispositivos aplicando la fórmula que calcula la intensidad de una onda esférica a partir de la potencia:

Ecuación 1. Cálculo de la intensidad de una onda esférica.

$$I = \frac{P}{4\pi r^2}$$

Despejando podemos obtener:

Ecuación 2. Cálculo de la distancia recorrida por una onda

$$r = \sqrt{\frac{P}{4\pi I}}$$

La intensidad también puede definirse como la sensibilidad que tiene el receptor para recibir una onda. Dependiendo del dispositivo y de la versión de Bluetooth con la que opere, se puede obtener una sensibilidad de entre -60dBm hasta -82dBm[15]. Por razones de la limitación de la tecnología, para realizar los cálculos tomaremos la sensibilidad mínima, obteniendo así las menores distancias para obtener un correcto funcionamiento.

Realizando el siguiente cálculo, podemos transformar los -60dBm en W:

Ecuación 3. Conversión de dBm en mW

$$Potencia\ (mW) = 10^{\frac{potencia\ en\ dBm}{10}}$$

Por tanto:

Ecuación 4. Aplicación de ecuación 3

$$Potencia = 10^{-\frac{60}{10}} = 10^{-6}\ mW = 10^{-9}\ W$$

Considerando que la distribución de la potencia es uniforme en todas las direcciones, podemos considerar una densidad de potencia de $10^{-9}\ W/m^2$

Finalmente, aplicando la Ecuación 2 con los valores de potencia obtenidos en la Figura 4 y con la densidad de potencia que se acaba de calcular podemos obtener los siguientes valores:

Tabla 1. Cálculo de rango de alcance

Clase	Potencia de Transmisión (mW)	Distancia calculada (m)
Clase 1	100	89.26
Clase 2	2.5	14.11
Clase 3	1	8.92

- **Espectro de frecuencia:** En la mayoría de los casos, el uso del espectro radioeléctrico tiene restricciones gubernamentales, En cambio, existen bandas reservadas internacionalmente las cuales pueden operar sin necesidad de licencia, que de forma general se usan con motivos industriales o científicos[16]. Es el caso de la tecnología Bluetooth. Esta tecnología trabaja en la ISM (Industrial Scientific and Medical) centrada en los 2.45 GHz, banda de frecuencias puede encontrarse en las denominadas bandas de ultra alta frecuencia (UHF), la cual abarca todas las frecuencias 300 MHz y 3 GHz [17]. La banda de operación es igual para todos los puntos del mundo, aunque dependiendo de la localización puede tener ligeras variaciones, tanto en frecuencia como en los canales que presenta, tal y como puede verse en la Tabla 2.

Tabla 2. Banda de frecuencias y canales por regiones[18]

Región	Rango de Frecuencias (GHz)	Número de canales
EEUU, Europa	2.4-2.4835	79
Japón	2.4-2.4835	79
España	2.445-2.475	23

- **Estructura:** Los dispositivos Bluetooth pueden tener dos formas de actuar, como maestros o como esclavos. Un maestro puede tener un máximo de siete esclavos, formando lo que se conoce como piconet. Al mismo tiempo, un master puede ser esclavo de otro master, formando así una estructura más amplia denominada scatternet.[16]

- **Otras características:** A continuación, puede observarse la Tabla 3, la cual resume las características más destacables de la tecnología.

Tabla 3. Características técnicas Bluetooth[18]

Característica	Valor de característica
Modulación	GFSK
Tasa de datos (pico)	1 Mbit/s
Ancho de banda	220 kHz / 1 MHz
Banda	Banda ISM de 2.4 GHz
Portadoras	23 (España), 79(EEUU, Europa y Japón)
Espaciamiento entre portadoras	1 MHz
Potencia de transmisión	Máximo de 20 dBm

2.1.3 Evolución de Bluetooth

Desde el nacimiento de la tecnología, Bluetooth ha ido evolucionando, mejorando e introduciendo nuevas características, las cuales se han publicado en las especificaciones[19]:

Tabla 4. Principales mejoras entre versiones[20], [21]

Especificación	Fecha de lanzamiento	Principales novedades de versión
Bluetooth 1.0	1999	Lanzamiento inicial
Bluetooth 1.1	2002	Velocidades de 721 kbps
Bluetooth 1.2	2003	Reducción de interferencias respecto al 1.1
Bluetooth 2.0 + EDR	2004	Implementación de EDR
Bluetooth 2.1	2007	Introducción de PIN para conexión.
Bluetooth 3.0 + HS	2009	Implementación de High Speed
Bluetooth 4.0	2010	Implementación de BLE
Bluetooth 4.1	2013	Encaminamiento hacia el IoT. Se permite la conexión de dispositivos pequeños a través de la nube. Coexistencia con LTE.
Bluetooth 4.2	2014	Implementación de IPv6.
Bluetooth 5.0	2016	Aumento de la capacidad de mensajes hasta 255 bytes

En la Figura 5, se muestra una tabla con la evolución de las principales características de Bluetooth:

Bluetooth Specification	V 1.1	V 2.0 + EDR	V 2.1 + EDR	V 3.0 + HS	V 4.0 + LE	V 4.1	V 4.2
Year	2002	2004	2007	2009	2010	2013	2014
Basic rate	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Enhanced Data Rate (EDR)	No	Yes	Yes	Yes	Yes	Yes	Yes
High Speed (HS)	No	No	No	Yes	Yes	Yes	Yes
Low Power (LE)	No	No	No	No	Yes	Yes	Yes

Figura 5. Evolución de los servicios Bluetooth a lo largo de las versiones[22]

A continuación, en las Figuras 6 y 7, se muestran dos tablas, una para las versiones que admiten BLE y otras para las que no. En ellas se encuentra información relevante con la velocidad máxima de transmisión y el rango máximo entre dispositivos:

Versión	Año de lanzamiento	Rango de transmisión máximo	rango máximo
Bluetooth 1.0	1999	732.2 kbit / s	10 metro (33 pie)
Bluetooth 1.1	2001	732.2 kbit / s	10 metro (33 pie)
Bluetooth 1.2	2003	1 Mbps	10 metro (33 pie)
Bluetooth 2.0	2004	2.1 Mbps	30 metro (100 pie)
Bluetooth 2.1	2007	2.1 Mbps	30 metro (100 pie)
Bluetooth 3.0	2009	24 Mbps	30 metro (100 pie)

Figura 6. Velocidades y rangos máximos de versiones sin BLE[20]

Versión	Año de lanzamiento	Velocidad máxima de transmisión	rango máximo
Bluetooth 4.0	2009	1 Mbps (EL) 3 Mbps (EDR)	60 metro (200 pie)
Bluetooth 4.1	2013	1 Mbps (EL) 3 Mbps (EDR)	60 metro (200 pie)
Bluetooth 4.2	2014	1 Mbps (EL) 3 Mbps (EDR)	60 metro (200 pie)
Bluetooth 5.0	2016	2 Mbps (EL) 50 Mbps (EDR)	240 metro (800 pie)

Figura 7. Velocidades y rangos máximos de versiones con BLE[20]

2.1.3.1 Evolución de mercado

Como se ha visto en el apartado anterior, desde 1999 hasta la actualidad, las nuevas versiones de la especificación han traído nuevas características y servicios, además de la mejora de los ya existentes. Esto ha hecho que cada vez sean más los usuarios que usan la tecnología o que adquieren productos que la integran para distintas funcionalidades.

Los analistas de las compañías que integran el SIG prevén un continuo crecimiento del número de dispositivos a nivel global, especialmente desde los años de la pandemia. Además, como se observa en la Figura 8, se calcula una venta de 7.5 billones para 2028, lo cual representa una tasa de crecimiento anual del 8%.

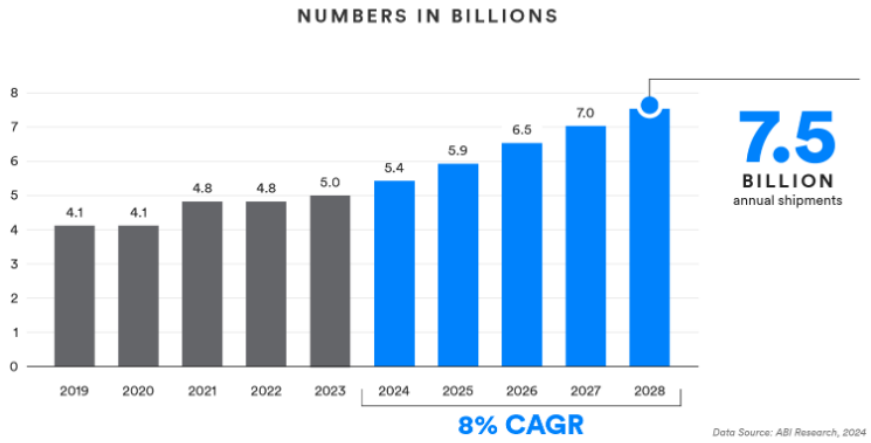


Figura 8. Previsión de venta anual de dispositivos con Bluetooth[23]

Con las nuevas versiones del estándar, cada año son más los dispositivos que se comportan de forma Dual, es decir, que admiten tanto la especificación clásica de Bluetooth como los nuevos formatos de BLE. Esto, sumado al continuo avance del IoT y la carrera tecnológica por conseguir dispositivos más eficientes, tanto en velocidades de envía como en consumos energéticos, hace que cada año sean mínimas las ventas de dispositivos que solo mantienen una especificación clásica. En cambio, como se aprecia en la Figura 9, en los últimos años, la cantidad de dispositivos que solo implementan BLE ha crecido considerablemente.

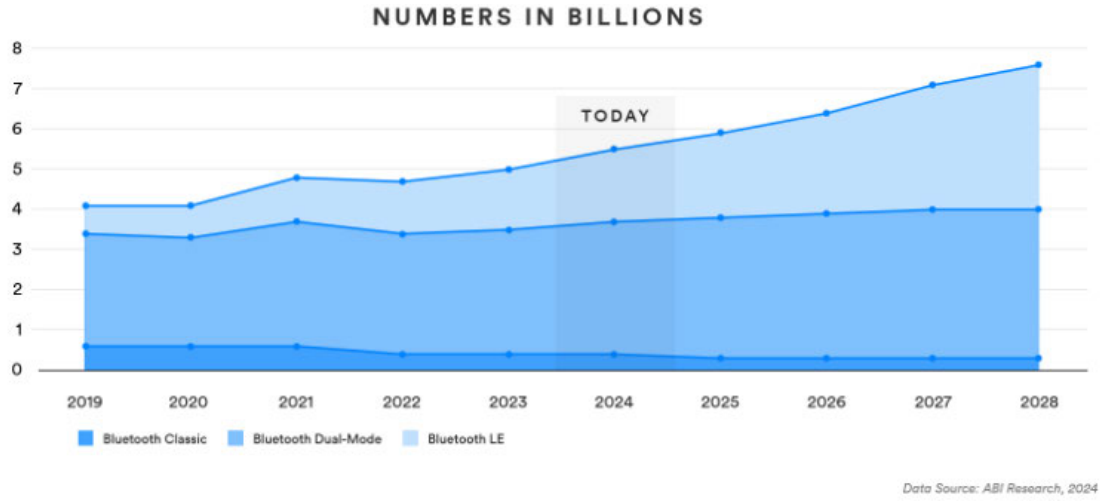


Figura 9. Numero de ventas de BT clásico Vs BT Low Energy Vs dispositivos duales[23]

Los dispositivos que integran la tecnología son de tipos muy diversos, y cada uno puedo aportar funcionalidades distintas. Para este proyecto, el tipo de dispositivo más interesante son los conocidos como “Servicios de localización”, los cuales nos permiten, como su nombre indica, conocer la localización aproximada de un objeto o cuando este tiene algún movimiento. En los últimos años, este tipo de dispositivos ha aumentado considerablemente sus ventas, tal y como se observa en la Figura 10, especialmente en el campo industrial para el control de inventario o en el de la medicina, para el control de dispositivos, medicamentos e incluso monitorización de pacientes.

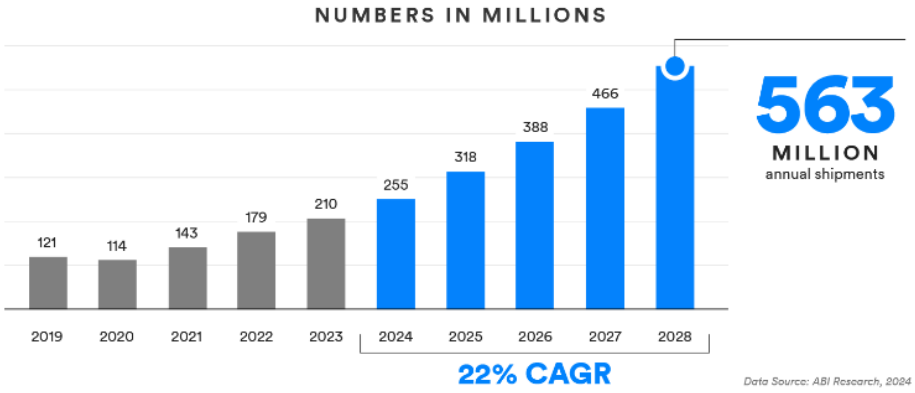


Figura 10. Ventas anuales de dispositivos de seguimiento de Bluetooth[23]

2.1.4 Bluetooth Low Energy

2.1.4.1 Introducción

Bluetooth Low Energy (BLE) nace junto a la especificación de Bluetooth 4.0, debido a algunas modificaciones que se hace en el funcionamiento, buscando un enfoque alternativo para dar soluciones a problemas nuevos que surgían con el desarrollo de la tecnología.

El punto clave de BLE es, como su nombre indica, el bajo consumo de los dispositivos, que, si de por si era mínimos con las especificaciones anteriores, a partir de 2010 se reduciría drásticamente. El uso de esta nueva versión ganó fuerza rápidamente con el desarrollo de aplicaciones de IoT o de dispositivos “wearables” donde la carga frecuente de la batería no es un punto asumible. Estas aplicaciones no requieren una velocidad de transferencia excesiva, por lo que, el SIG, decidió aplicar estas modificaciones (reducción de velocidad y consumo) para presentar su nueva especificación de Bluetooh Low Energy.

2.1.4.2 Características técnicas

Para conseguir la reducción de consumo se hicieron algunas modificaciones en las características técnicas de la especificación. Entre las más destacadas se encuentra la reducción del tamaño de los paquetes de datos, entre 27 y 251 bytes[24].

Además, los datos se envían lo mínimo posible, permitiendo así mantener encendida la transmisión del dispositivo el menor tiempo posible, lo cual reduce significativamente el consumo.

BLE fue diseñado para motivos completamente distintos a los definidos para Bluetooth clásico, por ello, trajo consigo cambios importantes tanto en topología, en la distribución de las portadoras o en la velocidad y alcance de transmisión. Estos cambios se comentarán en los siguientes apartados, no obstante, en la Figura 11 se muestran las principales características de esta nueva especificación:

Banda operativa	2400 MHz – 2483,5 MHz ~ 2,4 GHz
Ancho de banda del canal	2 MHz
Número de canales RF	40
Potencia máxima de transmisión	20 dBm 0,1 W
Máximo rendimiento de datos de la aplicación	1,4 Mbps
Alcance máximo a velocidades de datos reducidas (125 y 500 kbps)	~1000 metros

Figura 11. Resumen de las características técnicas de BLE[24]

2.1.4.3 Pila de protocolo BLE

En la pila de protocolo de esta versión hay cambios significativos, los cuales se explicarán a continuación. En le Figura 12 puede observarse un ejemplo de la propia pila:

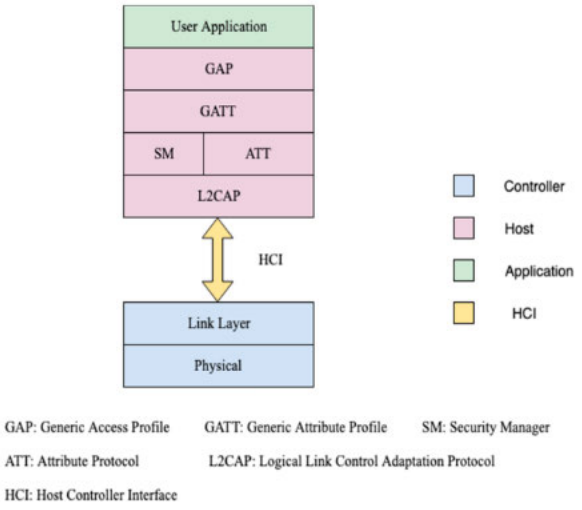


Figura 12. Pila de protocolo BLE[25]

A continuación, se explica brevemente la función que desempeña cada una de las capas[24]:

Dentro de la parte del Host:

- **L2CAP:** proporciona encapsulación a los datos de las capas superiores.
- **SMP:** es la parte encargada de la comunicación segura.
- **ATT:** Configura y define como se representan los datos a otros dispositivos.
- **GATT:** Define como se accede y usan los datos representados en ATT.
- **GAP:** Se encarga del descubrimiento y conexión de los dispositivos.

Dentro del controlador:

- **PHY:** Corresponde con la capa física y es la encargada de modular las ondas radio, su emisión y recepción.
- **LL:** Es la capa de enlace y administra los estados de radio entre "en espera", "advertising", "scanning", "inicio" o "conectado".

2.1.4.3.1 GAP: Tipos de dispositivos y topologías.

Los dispositivos involucrados en una conexión BLE pueden adquirir dos roles distintos:

- **Central (C):** El dispositivo con rol de central se encarga de escanear e iniciar las conexiones con el resto de los dispositivos. Puede asemejarse al Master de Bluetooth clásico.
- **Periférico (P):** El dispositivo con rol de periférico se encarga de aceptar las comunicaciones iniciadas por las centrales y de comunicarles la información establecida. Este rol puede asemejarse al Slave de Bluetooth clásico.

El protocolo de BLE permite dos tipos concretos de comunicación:

- **Comunicación orientada a la conexión:** existe una conexión directa entre dispositivos, lo que permite una comunicación bidireccional.
  Este tipo de comunicación se usa en las topologías conectadas, como la que se muestra en la figura 13, en las cuales cada central está conectada a un número determinado de periféricos, y solo pueden comunicarse entre los dispositivos que mantienen una conexión.

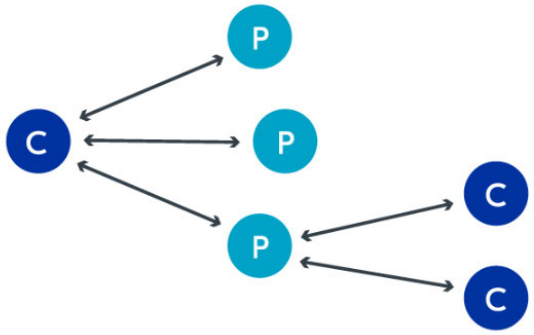


Figura 13. Topología conectada de BLE[26]

Dentro de las topologías conectadas pueden encontrarse las multi-rol, las cuales permiten que un mismo dispositivo funciones como central y como periférico. Estas topologías son esenciales en sistemas en los cuales encontramos varios sensores conectados a una pasarela que funciona como central, la cual, al mismo tiempo se conecta como periférico a dispositivos como móviles u ordenadores. A continuación, en la Figura 14 se muestra un ejemplo de esta topología:

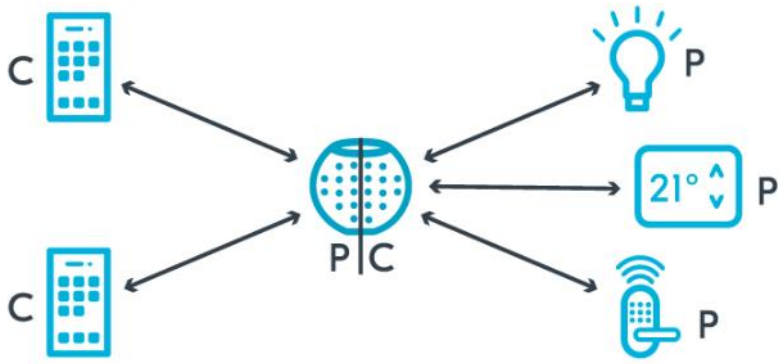


Figura 14. Ejemplo de uso de topologías multi-rol[26]

- **Comunicación por difusión (broadcast):** En esta comunicación, los dispositivos no necesitan hacer una conexión directa. En este caso, los periféricos se encargan de transmitir los datos que han recopilado de forma pública a todos los dispositivos en el rango de comunicación. Esta comunicación es propia de topologías de transmisión o difusión. La clave de estas comunicaciones son los paquetes de Advertising, los cuales ayudan a difundir la información de periféricos a centrales, como se ve en la Figura 15.

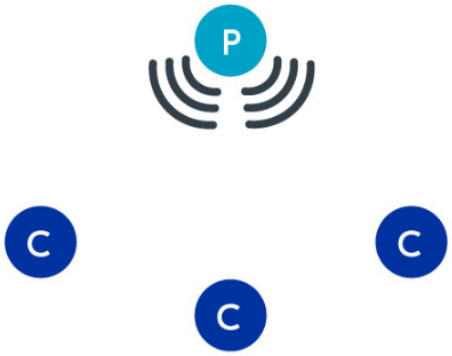


Figura 15. Topología de difusión de BLE[26]

2.1.4.3.2 ATT y GATT: Representación de datos transmitidos

Las capas de ATT y GATT se encargan de la representación de los datos que se transmiten en las conexiones de BLE, todo esto tras establecerse la conexión mediante la capa GAP que se ha visto en el punto anterior.

La capa ATT se basa en una arquitectura de cliente-servidor[27]:

- **Servidor GATT:** Almacena datos y proporciona los métodos para que el cliente pueda acceder a ellos de forma correcta.
- **Cliente GATT:** Es el dispositivo que accede al servidor en busca de los datos. Accede mediante las operaciones GATT[28].

La capa GATT se sitúa justo encima de la ATT, y se encarga de estructurar y clasificar la información de forma jerarquizada en Perfiles, Servicios, Características y Atributos, como puede observarse en la Figura 16:

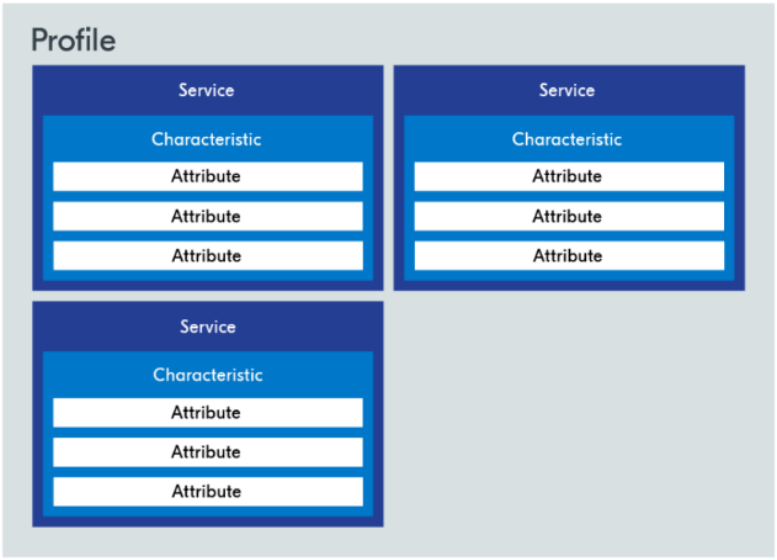


Figura 16. Estructura capa GATT[29]

- **Atributos:** Es el valor mínimo de representación. Las características están formadas por el valor, las unidades o cualquier información esencial de la magnitud que se esté almacenando.
- **Características:** Agrupan los atributos que estén relacionados entre ellos.
- **Servicio:** son las posibilidades configurables por el usuario, con ellos se puede determinar la función concreta del dispositivo.
- **Perfil:** Agrupa todos los servicios de un dispositivo para una representación ordenada. Los dispositivos pueden tener varios perfiles para intercambiar sus funciones de forma sencilla y rápida.

2.1.4.4 BLE Advertising

Como se ha visto anteriormente, una de las formas de comunicación que permite BLE es el Advertising, en el cual, no necesita mantener una conexión directa entre trasmisor y receptor. Es uno de los puntos clave en BLE, ya que es la forma de la que se comunican la mayoría de los dispositivos, especialmente en servicios de IoT

Los dispositivos BLE se comunican a través de 40 canales de frecuencia dentro de la banda definida. De estos canales, 37 están definidos para la transmisión de datos, y los 3 restantes son los usados para fines de Advertising, tal como se observa en la Figura 17:

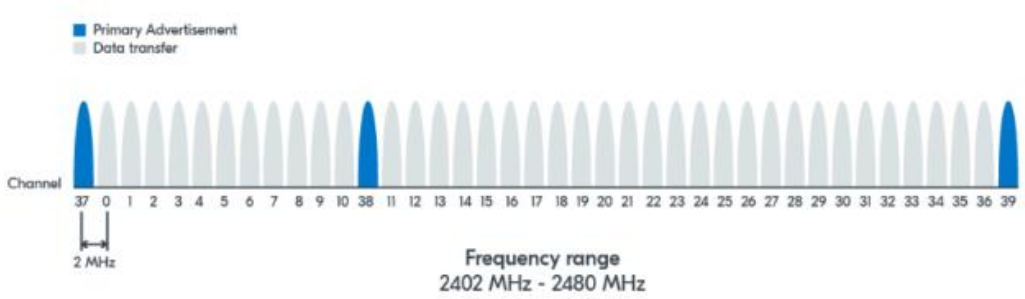


Figura 17. Distribución de los canales de radio BLE[30]

En la figura 18, se pueden observar los dos tiempos más relevantes dentro del periodo de búsqueda de paquetes de Advertising:

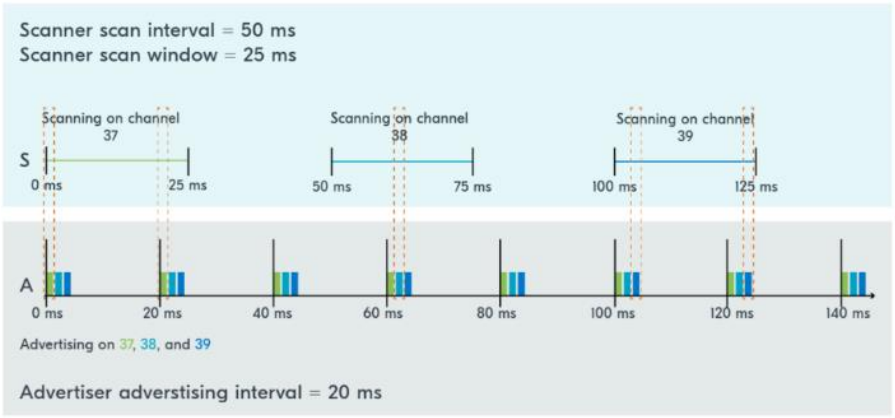


Figura 18. Esquema ejemplificativo de intervalo y ventana de escaneo[30]

- **Intervalo de escaneo:** Es el tiempo que pasa un dispositivo en el proceso de búsqueda de paquetes de Advertising.
- **Ventana de escaneo:** Es el tiempo real de escaneo de cada dispositivo.

  La configuración óptima es la de establecer periodos de escaneo relativamente largos comparados con los periodos de Advertising (tiempo que un periférico emite paquetes). De esta forma, el mayor consumo recae sobre el dispositivo de escaneo, el cual, por motivos de arquitectura, suele estar conectado a una fuente de alimentación o suele tener baterías más grandes, como es el caso de los PCs o teléfonos móviles.

Los periféricos pueden tener muchas configuraciones, dependiendo de su propósito. Los campos que pueden variar son los siguientes[31]:

- **Conectable / no conectable:** determina si una central puede o no establecer una conexión con este periférico
- **Escaneable / no escaneable:** determina si el periférico permite o no solicitudes de escaneo de un dispositivo.
- **Dirigido / no dirigido:** determina si los paquetes de Advertising están o no dirigidos a una central en concreto.

En función de cómo se configuren los campos anteriores los periféricos se pueden dividir en cuatro grupos distintos, los cuales pueden verse en la Figura 19 junto a sus características:

	Connectable	Scannable	Directed
ADV_IND	x	x	
ADV_DIRECT_IND	x		x
ADV_SCAN_IND		x	
ADV_NONCONN_IND			

Figura 19. Tipos de periféricos en Advertising BLE[31]

Durante las transmisiones de Advertising, la información viaja entre los dispositivos encapsulada en paquetes con una estructura concreta, la cual puede denominarse trama BLE. La trama BLE tiene una estructura similar a la de las tramas de otros protocolos, como IP. A continuación, desde la Figura 20 hasta la X, se presenta la estructura de estas tramas[32]:

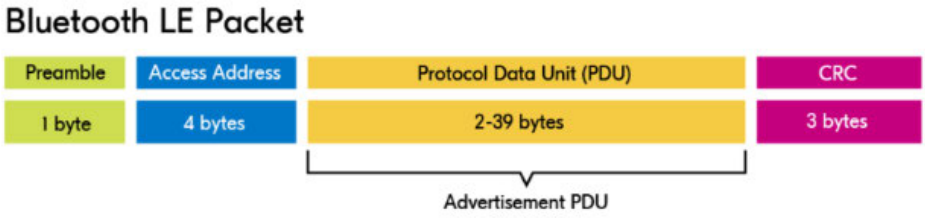


Figura 20. Trama BLE

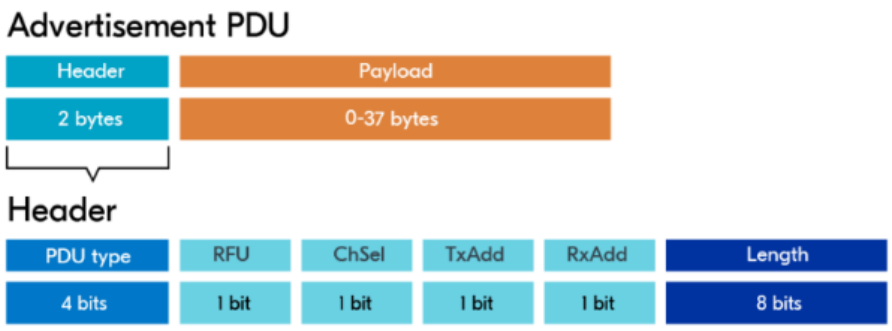


Figura 21. PDU y su cabecera de Advertising

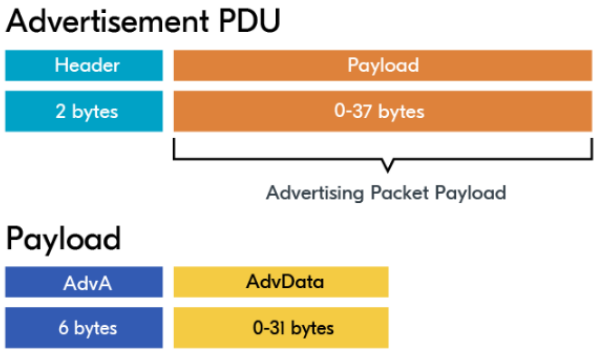


Figura 22. PDU y datos de Advertising

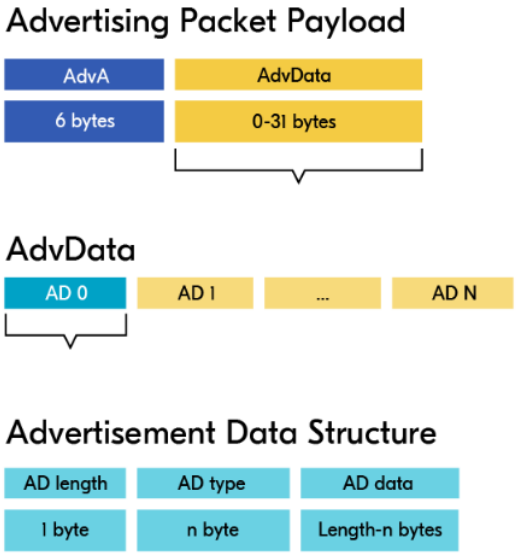


Figura 23. Datos de Advertising

2.1.4.5 Comparación entre Bluetooth y Bluetooth Low Energy

Como se ha visto en los apartados anteriores, desde el 2010, con la salida de Bluetooth 4.0 surgió un nuevo concepto, Bluetooth Low Energy, una nueva especificación de la tecnología que buscaba dar soporte a las nuevas tecnologías emergentes como el IoT. Con el paso de los años, debido a las muchas ventajas que ofrece respecto al Bluetooth clásico, ha conseguido convertirse en la especificación principal para proyectos relacionados con la medicina, como puede ser el diseño de sensores inteligentes o wearables que ayuden a monitorizar pacientes entro muchas otras posibilidades.

Algunas de estas ventajas son esenciales para las actividades de la actualidad. Entre estas ventajas, el mínimo consumo que oferta BLE le da una gran ventaja respecto a su predecesor. Incluso, a pesar de que su velocidad de transmisión es inferior, las aplicaciones para las que se escoge no necesitan grandes tasas de envío de datos. Por estos motivos, en la definición de un proyecto, si la cantidad de datos es grande, se optará por una versión clásica de la tecnología, en cambio, si con una velocidad menor es suficiente, el ahorro energético y el

coste reducido de BLE le da la victoria en la decisión de la tecnología. En la Figura 24, se observa la comparación de las características de ambas tecnologías:

	Bluetooth Classic	Bluetooth Low Energy
Frequency Band	2.4 GHZ ISM Band	2.4 GHZ ISM Band
No. of Channels	79 one MHz Channel	40 two MHz Channel
Power Consumption	Low	Less
Data Rate	1-3 Mbps	1 Mbps
Latency	Approx. 100 ms	Approx. 6 ms
Range	< 30 m	50 m (150 m in open area)
Topology	Peer-to-peer (1:1)	Peer-to-peer (1:1) Star (many:1) Broadcast (1:many) Mesh (many:many)
Device pairing	Required	Not Required
Voice capable	Yes	No
Nodes/Active Slaves	7	Unlimited
Security	64b/128 bit, user defined application layer	128 bits AES, user defined application layer
Smartphone Compatibility	100% available on smartphones	100% available on smartphones
Use Cases	Streaming applications like audio streaming, file transfer, and headsets	Location beacons, smart home applications, medical devices, industrial monitoring, fitness trackers

Figura 24. Comparación entre Bluetooth y BLE[20]

2.1.4.6 Alternativas a BLE

Existen otras tantas alternativas que pueden ser competencia directa de BLE. Entre ellas:

- **ANT y ANT+:** ANT[33] es un protocolo de comunicación de baja energía que con multitud de nodos por todo el mundo permite adaptarse a cualquier topología de forma muy flexible. Al añadir ANT+ a la base, se mejora la adquisición y análisis de los datos. Generalmente se usa en Wearables como los relojes deportivos. No se seleccionó esta tecnología ya que el campo de aplicación del proyecto era distinto al especializado de esta tecnología.
- **ZigBee**[34]**:** Es un protocolo de comunicación de baja energía donde un dispositivo coordina las comunicaciones del resto de red. Se descartó esta tecnología debido a que solo se querían dispositivos de emisión y uno de escaneo, ninguno que coordinase.
- **NFC**[35]**:** Es una tecnología de muy corto alcance que transmite información por ondas de radio cuando dos dispositivos se encuentran próximos entre sí. Se descartó el uso de esta tecnología debido a que los periféricos y la central deben estar sumamente cerca, algo que técnicamente no es viable ya que la central puede estar en cualquier punto alejado de los periféricos de control para el proyecto.

2.2 Beacons

Una baliza, también denominada beacon, es un dispositivo electrónico inalámbrico, usualmente de tamaño reducido, que emite señales de radio dependiendo de varios factores. Entre esas señales de radio podemos encontrar varios protocolos, como por ejemplo BLE.

La función principal de los beacons es transmitir mediante esas señales de radio datos recopilados por el dispositivo, acompañados de un identificador único conocido como UUID.

2.2.1 Hardware

Los beacons de distintos fabricantes pueden estar compuestos por multitud de componentes distintos, pero hay algunos importantes que se repiten en todos los dispositivos:

- **Microcontrolador:** Es la unidad principal del dispositivo, se encarga de procesar la información, realizar cálculos, configurar el resto de los dispositivos, y, en definitiva, coordinar y controlar el comportamiento del dispositivo. En la Figura 25, el microcontrolador corresponde con **el componente resaltado en azul, el NRF52810,** un microcontrolador de la familia Nordic Semiconductors [36]
- **Batería:** Al ser un dispositivo inalámbrico, es alimentado mediante baterías. Normalmente, debido al tamaño de estos dispositivo se usan pilas CR, común mente denominadas “pilas de botón”, como la que se puede observar en la figura 26.
- **Componentes de interacción:** Es normal incluir algún componente que permita interactuar físicamente con el dispositivo. Es el caso del **botón, resaltado en color violeta.** Estos componentes permiten configurar, encender, apagar o establecer algún tipo de función en el beacon.
- **Componentes informativos:** también es buena idea que contengan un componente que emita información de forma sencilla, por ejemplo, **un LED, como es el componente granate.** Con estos componentes podemos comprobar acciones como el encendido o la conexión del beacon.
- **Adquisidores de información:** dependiendo de la función del hardware puede incluirse distintos sensores. En el caso de ejemplo es un **acelerómetro (LIS2DH12),** puesto que el beacon de la figura emite señales al detectar movimiento.

Figura 25. Hardware de beacon de ejemplo

2.2.2 Protocolos

Los beacons usan sus propios protocolos para organizar la información y representarla a los usuarios y desarrolladores de una forma sencilla y práctica. Existen principalmente dos protocolos:

2.2.2.1 iBeacon: Apple

iBeacon [37] es un protocolo de comunicación diseñado para los beacons desarrollado por Apple en 2013. La tecnología inalámbrica designada es BLE. En las transmisiones de iBeacon existen 3 parámetros esenciales, los cuales se ven reflejados en la Figura 27:

Field	Size	Description
UUID	16 bytes	Application developers should define a UUID specific to their app and deployment use case.
Major	2 bytes	Further specifies a specific iBeacon and use case. For example, this could define a sub-region within a larger region defined by the UUID.
Minor	2 bytes	Allows further subdivision of region or use case, specified by the application developer.

Figura 26. Parámetros característicos de iBeacon[37]

2.2.2.2 Eddystone: Google

Eddystone es un protocolo de código abierto [38] definido por Google en 2015. Es un protocolo bastante flexible, que, a diferencia de iBeacon, que solo está enfocado al ecosistema de Apple, puede adaptarse a multitud de dispositivos. Este es el protocolo usado en las transmisiones del proyecto.

En este caso también existen 3 parámetros esenciales que definen las transmisiones:

Nombre	Descripción
Eddystone-UID	Es el Identificador único de cada interacción
Eddystone-URL	Permite a los beacons transmitir una URL, la cual puede configurarse para abrirse automáticamente cuando un dispositivo se encuentra cerca.
Eddystone-TLM	Transmite datos de telemetría, entre los que pueden representar la batería o varios parámetros medidos por sensores del hardware.

2.2.3 Aplicaciones de los Beacons

El objetivo de este proyecto encaja dentro del sector de la salud, por lo que se destacaran algunas aplicaciones de este sector:

- **Gestión de equipos médicos:** los hospitales son grandes extensiones con multitud de equipos médicos, que en ocasiones pueden encontrarse en ubicaciones desconocidas. Gracias a los beacons, cualquier trabajador del hospital puede consultar en una app la ubicación exacta del dispositivo ahorrando mucho tiempo en su búsqueda.

- **Monitoreo de pacientes:** Estos beacons se pueden usar para registrar los movimientos o localizaciones de pacientes tanto dentro como fuera del hospital. Especialmente se usan en enfermos de Alzheimer o Parkinson. Este será el área de aplicación del proyecto.

Existen además muchos otros sectores en los cuales los beacons pueden aplicarse:

- **Compras:** En este sector son ampliamente usados para mejorar la experiencia de los clientes y recomendarles distritos productos. Mediante la localización de los dispositivos móviles cercanos se pueden enviar ofertas personalizadas según el gusto de cada cliente, incluyendo información muy relevante como imágenes, descripciones e incluso información sobre una reducción en el precio.

- **Logística**[39]**:** Los beacons pueden servir para localizar el inventario en distintas empresas, además de conocer la cantidad de producto que queda. También pueden instalarse en camiones o contenedores para hacer un seguimiento en tiempo real de los pedidos de los clientes. Finalmente, también pueden instalarse en las líneas de producción para obtener métricas y optimizarlas.

- **Museos**[40]**:** existen museos los cuales usan beacons que detectan la posición de los visitantes, de forma que al acercarse a alguna de las obras expuestas pueden recibir información relevante en su dispositivo móvil, con la ventaja de poder personalizarla en el idioma deseado por el visitante.

2.2.4 Beacons comerciales

Para el proyecto se ha definido un requisito de uso: el proyecto debe ser aplicable en balizas comerciales de otros fabricantes.

Tras realizar una búsqueda y comparación de distintos modelos, se seleccionaron dos versiones distintas de la marca MOKO Smart[41]. Se han seleccionado estos dispositivos principalmente por dos criterios, el primero de ellos el factor económico, ya que su reducido

coste ha permitido la adquisición de varias unidades. Por otro lado, su reducido tamaño y sus buenas especificaciones han hecho que sean una opción aceptable para el proyecto.

Los dispositivos elegidos han sido los siguientes:

2.2.4.1 Moko M1[42]

Este beacon es un dispositivo del tamaño muy reducido, aproximándose al tamaño de una moneda. Esta característica lo hace un dispositivo ideal para colocarlo con el objetivo de la adquisición de datos de forma discreta.

Figura 27. Baliza comercial Moko M1[42]

2.1.1.1 Moko M4 Lite[43]

Este beacon es un dispositivo del tamaño algo superior al que se ha presentado anteriormente. Este aumento de tamaño le permite una transmisión de mayor potencia y alcance, además de ser más sensible a la detección de movimiento.

Figura 28. Baliza comercial Moko M4 Lite[43]

En el anexo A se muestra como configurar ambos dispositivos para el uso en el proyecto

3. Especificaciones y restricciones de diseño

En el desarrollo del proyecto se tendrán en cuenta las siguientes especificaciones:

- Las balizas cuentan con microcontroladores de NordicSemiconductor.
- El firmware de las balizas se construye sobre el kit de desarrollo nRF Connect SDK.
- La transmisión de información se hace empleando Bluetooth Low Energy (BLE).
- Se usará el protocolo de comunicación Eddystone para BLE.
- El tamaño de las balizas deberá ser reducido, como máximo del tamaño de una moneda de un euro.
- El proyecto implementará tanto balizas comerciales como las diseñadas en el mismo.
- La aplicación se desarrolla para un entorno Android.
- El interfaz de la aplicación desarrollada solo permitirá la visualización de información, en ningún caso podrá ser editable.
- El etiquetado de acciones aparecerá en la aplicación siguiendo el siguiente formato: [Identificador de baliza / fecha / hora]
- La fecha aparecerá con el formato: DD/MM/AAAA, siendo D (día), M (mes) y (A) Año
- La hora aparecerá con el formato: HH/MM/SS, siendo H (hora), M (minutos) y (S) segundos. La resolución mínima de la hora será de segundos.

Como requisito adicional, se debe asegurar que tanto la instalación y el uso de los dispositivos y de la aplicación sea sencillo e intuitivo, facilitando la interacción con personas de edad avanzada.

4. Descripción de la solución propuesta

El proyecto se ha separado principalmente en tres fases debido a las distintas tareas que debían realizarse.

En la primera fase se estudió el uso de una tarjeta de desarrollo proporcionada por el fabricante Nordic Semiconductor con el objetivo de comprender el uso de los microcontroladores de la familia nRF, los cuales serían usados en el desarrollo del proyecto. Una vez comprendido el funcionamiento se diseñó el Firmware que llevarían los microcontroladores cargado para la detección del movimiento y la emisión de la señal BLE.

En la segunda fase del proyecto se diseñaría una app en Android para la detección de las señales de las balizas.

Finalmente, con el proyecto en funcionamiento para balizas comerciales se decidió diseñar un hardware específico hecho a medida para diseñar balizas propias.

4.1 Fase 1: Desarrollo del Firmware de los microcontroladores

Como se ha comentado, el primer paso antes del desarrollo del firmware es conocer la arquitectura sobre la que se va a desarrollar el programa. El fabricante Nordic Semiconductor cuenta con una serie de placas de desarrollo, entre las cuales se encuentra el nRF51 DK[44], un kit de desarrollo basado en los microcontroladores de la familia nRF51.

Figura 29. Microcontroladores de la familia nRF51 (1) [44]

Figura 30. Microcontroladores de la familia nRF51 (2) [44]

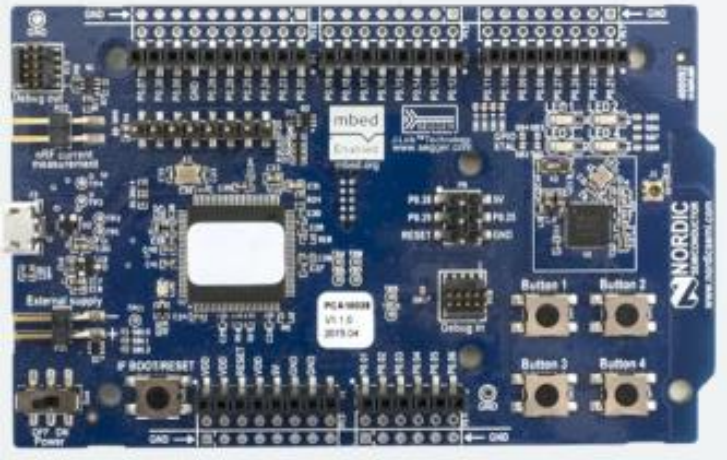


Figura 31. nRF51 DK [44]

La nRF51 DK es un kit de desarrollo de bajo coste con el que se pueden trabajar principalmente con dos protocolos inalámbricos. Esta placa soporta tanto ANT como BLE, y este último es el designado para el proyecto. La placa cuenta con algunos periféricos de interés, como 4 botones programables y 4 leds. Es posible alimentarla tanto por USB como mediante una pila CR2032. Además, los programas pueden ser cargados mediante un USB o mediante SEGGER J-Link. En la Figura 32 se puede observar el diagrama de bloques que representa todos los componentes que forman el nRF51 DK.

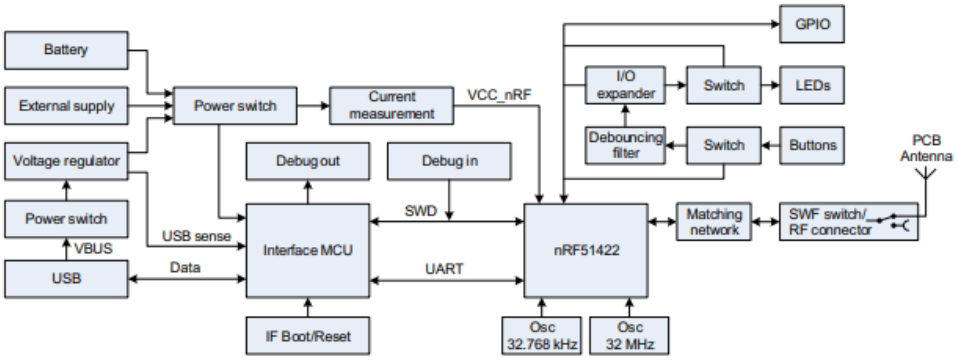


Figura 32. Diagrama de bloques de nRF51 DK [45]

El microcontrolador que está integrado en esta placa es el nRF51422 presentado en la Figura 29, y será sobre el que se desarrolle el firmware posteriormente.

Esta placa de desarrollo, a pesar de que cuenta con multitud de funcionalidades, no cuenta con un acelerómetro integrado, por lo que debe conectarse. Para el proyecto se ha seleccionado el AIS2DW12[46].

Para simplificar el desarrollo, inicialmente se hará uso de una placa de adaptación para el acelerómetro. Esta placa es STEVAL-MKI206V1, desarrollada por STMicroelectronics [47].

Figura 33. STEVAL-MKI206V1 de STMicroelectronics [47]

Esta placa de soporte será comunicada con la placa de desarrollo mediante el protocolo I2C, por lo que es necesario conectar tanto los pines SDA como SCL, además de la alimentación y la masa.

Los pines de SDA y SCL se ubican en el puerto 0, concretamente en los pines 7 y 30, ubicados en la parte superior derecha de la placa, concretamente en el conector P4:

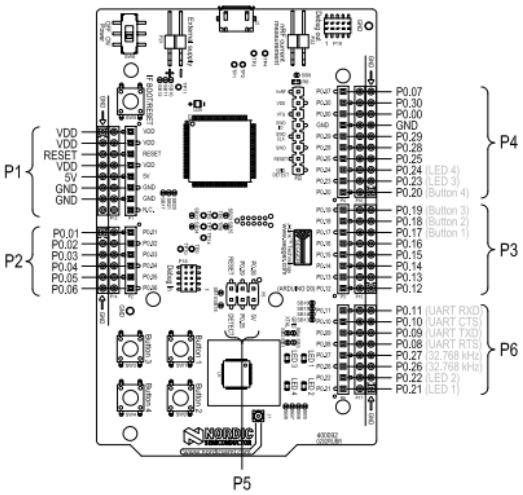


Figura 34. Conectores y ubicación de pines de nRF51 DK [48]

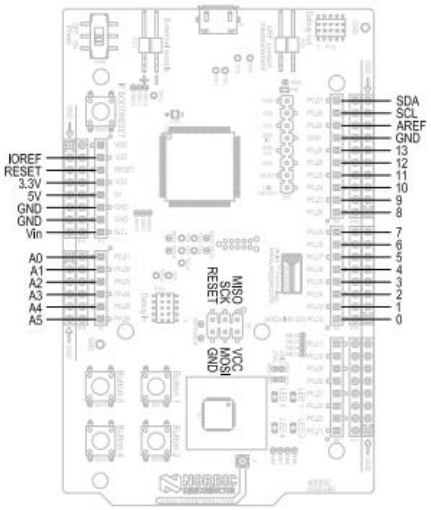


Figura 35. Función de los pines del nRF51 DK [48]

A continuación, en la figura 36, se muestra un el montaje del prototipo con el que se ha estado trabajando.

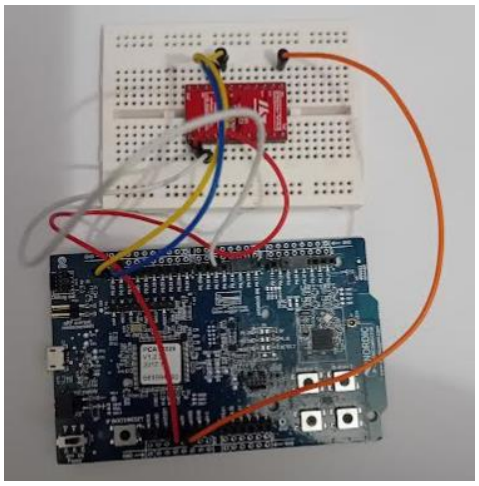


Figura 36. Montaje de prototipado

4.1.1 nRF Connect SDK

Antes de comenzar con el desarrollo es imprescindible configurar correctamente el proyecto. En este caso, el firmware se basará en el nRF Connect SDK[49].

El nRF Connect SDK ofrece una base unificada de herramientas usadas para el desarrollo de aplicaciones con los componentes y dispositivos de NordicSemiconductor. De esta forma permite a los desarrolladores un uso más cómodo de módulos o bibliotecas muy útiles, los cuales se encuentran ya integrados.

Este SDK cuenta con una integración propia para el IDE VSCode, el cual se usará para el desarrollo del firmware, pero no es imprescindible el uso de este. El código del SDK se encuentra público en GitHub[50], por lo que, como se ha comentado, puede usarse otro IDE instalando el SDK.

De forma interna, el SDK cuenta con cuatro módulos principales:

- **Nrf:** Incluye aplicaciones, ejemplos y protocolos de conectividad propios de Nordic
- **Nrfxlib:** bibliotecas y pilas de protocolos comunes usados por Nordic. En este punto entran módulos para el uso de periféricos o código para facilitar el uso de protocolos como I2C.
- **MCUBoot:** es el bootloader que usa Nordic para cerificar y cargar los firmwares creados por los usuarios.
- **Zephyr:** Configuración de la placa y sistema operativo en tiempo real (RTOS)[51]

En la Figura 37 puede verse la arquitectura del SDK, representando en azul claro los componentes propios de NordicSemiconductor, y en azul los programas de terceros que implementan dentro de su propio SDK.

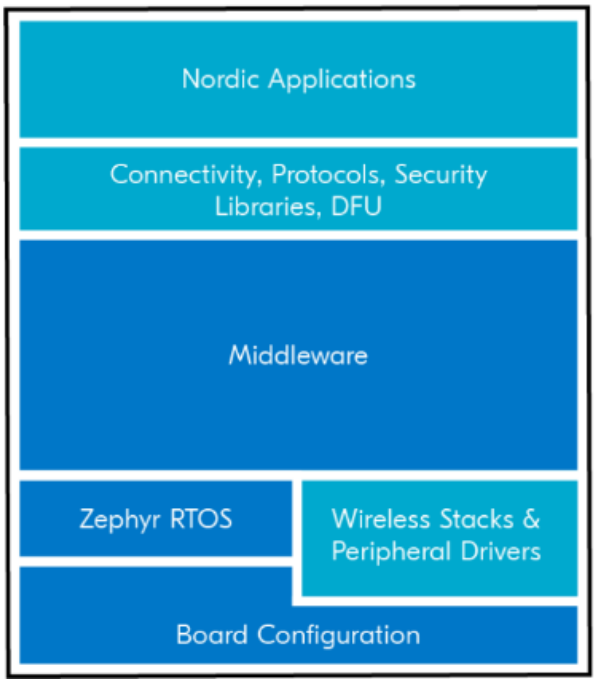


Figura 37. Arquitectura del nRF Connect SDK[49]

4.1.1.1 Zephyr

Zephyr [52] es un RTOS de código abierto [53] diseñado especialmente para dispositivos embebidos con recursos limitados como la memoria. Es un RTOS donde los desarrolladores pueden seleccionar que partes específicas necesitan para el desarrollo, permitiendo así no agregar ni compilar partes innecesarias, reduciendo así la memoria ocupada en el microcontrolador. Estas características permiten optimizar los recursos de los dispositivos.

Además, es muy interesante para el uso de sistemas embebidos debido a las características de gestión de energía, permitiendo que los dispositivos prolonguen la vida útil de su batería.

4.1.2 Instalación y configuración

En el Anexo B se muestra el proceso de instalación de las herramienta necesarias para comenzar con el desarrollo del firmware.

4.1.3 Desarrollo del firmware

4.1.3.1 Planteamiento

El firmware que se busca desarrollar debe cumplir una función muy concreta. La placa de desarrollo, y posteriormente las balizas que se diseñarán, deben ser capaces de realizar principalmente dos funciones:

- Detección de movimiento
- Gestión de la señal BLE

El planteamiento del software será el siguiente: La señal radio de BLE permanecerá en reposo con el emisor apagado por motivos de ahorro energético. Una vez que el acelerómetro seleccionado detecte movimiento, el microcontrolador activará la señal BLE por un tiempo de 4 segundos. Mientras la señal esté activa, el LED1 del nRF51 DK estará encendido. Pasado el tiempo de activación, el microcontrolador cesará la emisión y el LED se apagará.

Como es evidente, la gestión de la vida útil de la batería es algo fundamental para el proyecto, por lo que el firmware se presentará en dos versiones, desde ahora firmware1.0 y firmware2.0. En la primera versión el trabajo se centró en la configuración de los dispositivos, la lectura del acelerómetro y la gestión de la señal BLE. En el firmware2.0 se implementó la entrada a los modos de bajo consumo, y su respectiva salida mediante una interrupción generada por el movimiento del acelerómetro.

4.1.3.2 Firmware1.0

A continuación, se explicará el diseño del Firmware1.0 y se detallarán las partes más características y las decisiones tomadas a lo largo de la codificación.

4.1.3.2.1 Diagramas de diseño

En primer lugar, se muestran los diagramas de componentes, paquetes y despliegue en las figuras 38 y 39 respectivamente. Estos dos diagramas ayudan a entender la estructura y el diseño del Firmware1.0, que módulos contiene, cómo están relacionados y cómo se comunican entre ellos:

- **Diagrama de componentes:** Representa todos los módulos que gestiona el Firmware1.0 durante su ejecución.

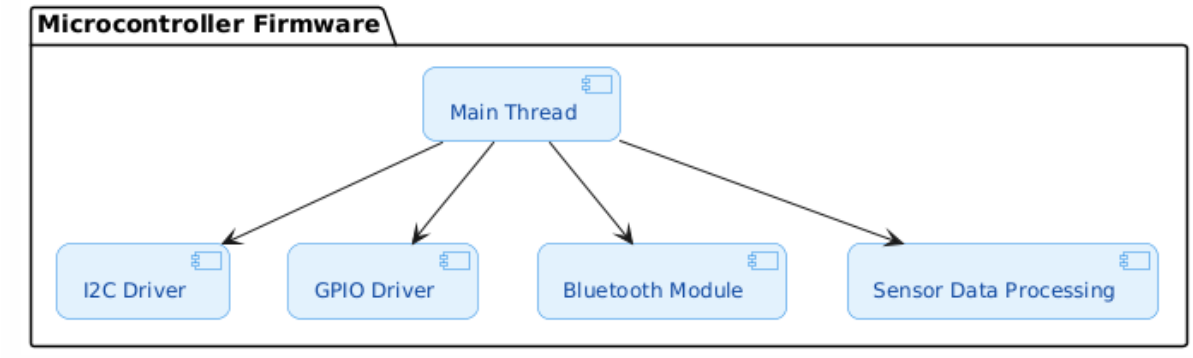


Figura 38. Diagrama de componentes de Firmware1.0

- **<u>Diagrama de paquetes:</u>** Muestra cómo se organizan de forma lógica los módulos del firmware en distintas capas de abstracción:

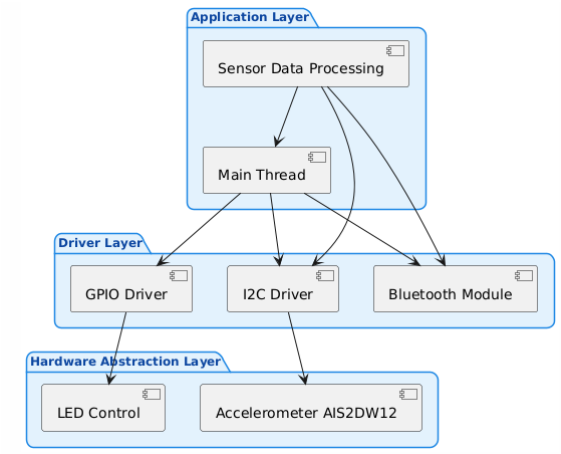


Figura 39. Diagrama de paquetes de Firmware1.0

A continuación, en la figura 40 se muestra el diagrama de secuencia del Firmware1.0, es decir, en este diagrama se va a representar el orden en el que se ejecutan las operaciones entre los distintos componentes del sistema que hemos visto anteriormente en el diagrama de paquetes:

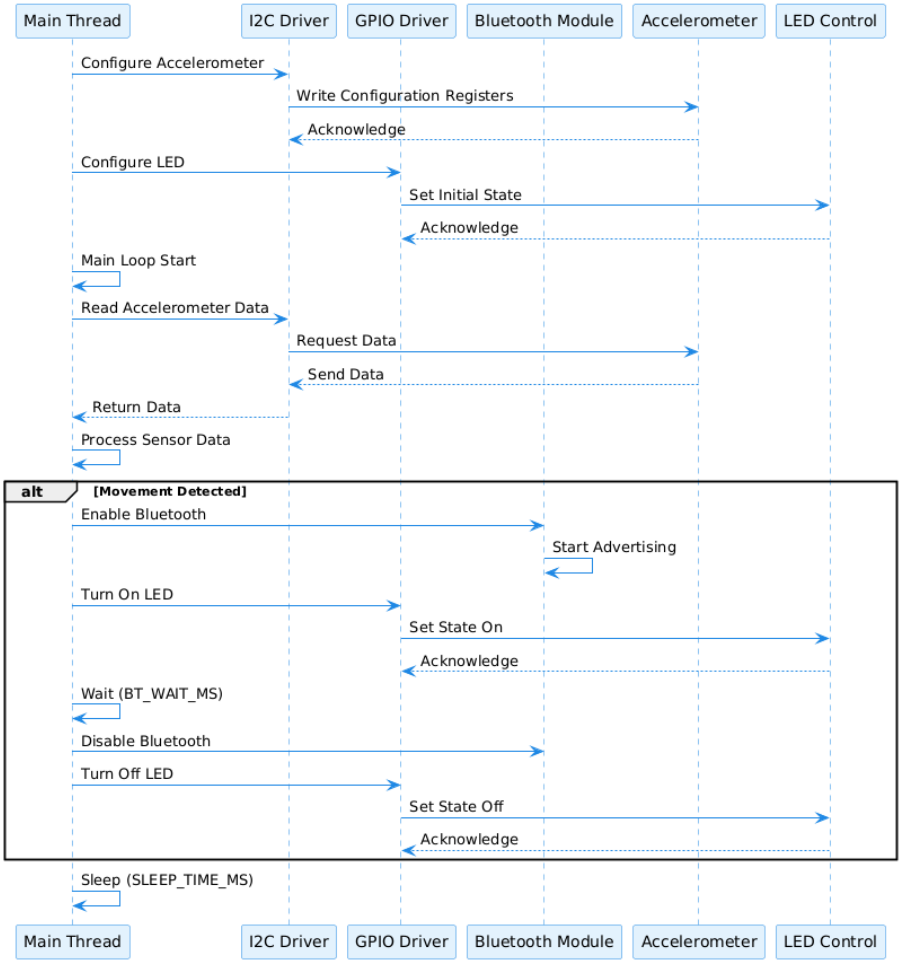


Figura 40. Diagrama de secuencia de Firmware1.0

En la figura 41 se representa el diagrama de actividad, el cual representará el flujo de trabajo y las actividades que existen dentro del Firmware1.0. Gracias a este diagrama podemos entender fácilmente la secuencia de actividades que tiene el Firmware1.0 y que decisiones toma el software en función del estado de ejecución del programa:

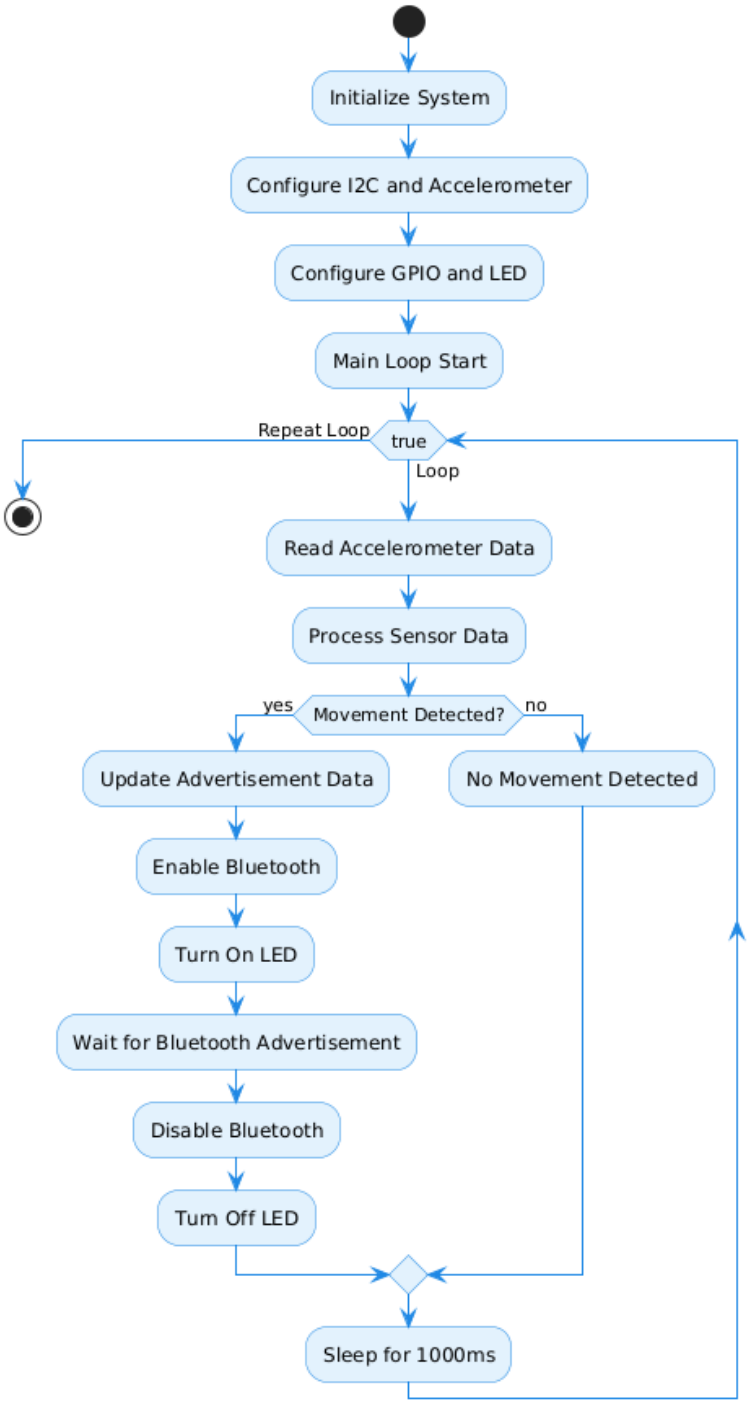


Figura 41. Diagrama de actividad de Firmware1.0

4.1.3.2.2 Librerías

En caso del programa Firmware1.0, todas las librerías implementadas son propias del nRF Connect SDK o del propio Stack de Zephyr. A continuación, se implementa una tabla con la información de todas las librerías que se han utilizado:

Tabla 5. Librerías integradas en el Firmware1.0

Nombre	Ruta	Descripción
Printk	<zephyr/sys/printk.h>	Permite realizar impresiones por la consola para verificar el proceso del firmware
Log	<zephyr/logging/log.h>	Permite imprimir por consola mensajes personalizados con diferentes niveles de severidad. Solo se aplica el de WARNING, para resaltar las conexiones de BLE
I2C	<zephyr/drivers/i2c.h>	Integra una API para interactuar con dispositivos mediante I2C. Se usa para la comunicación con el acelerómetro
GPIO	<zephyr/drivers/gpio.h>	Integra una API para interactuar con los GPIO del nRF51 DK. Esto se implementa para controlar el encendido y apagado del LED.
Bluetooth	<zephyr/bluetooth/bluetooth.h>	Integra una API para interactuar con los dispositivos bluetooth. Integra funciones como inicialización y configuración de dispositivos.
HCI	<zephyr/bluetooth/hci.h>	Integra una API que permite una conexión entre los dispositivos y la API Bluetooth a un nivel más bajo.

4.1.3.2.3 Funciones

A continuación, se incluye una tabla con las funciones principales que se definen en el código con la información detallada más relevante de las mismas:

Tabla 6. Funciones de Firmware1.0

Función	Parámetros	Salida	Descripción
config_regs	-	**int:** Devuelve 0 si la configuración es exitosa, -1 en caso de error.	Configura los registros de control del acelerómetro AIS2DW12
readyAndConfigureLEDS	-	-	Configura e inicializa el LED
bt_ready	**err (int):** Código de error que indica estado de la inicialización	-	Inicia la publicidad Bluetooth y muestra la dirección MAC del dispositivo en la consola

4.1.3.3 Firmware2.0

Como se ha comentado anteriormente, la principal diferencia en el Firmware2.0 es la gestión del bajo consumo para reducir el gasto de batería del dispositivo. Para ello se ha modificado ligeramente la estructura del software.

4.1.3.3.1 Cambios respecto a Firmware1.0

Para implementar el bajo consumo se ha usado la función k_sleep(K_FOREVER). Esta función es la única soportada por la versión de Zephyr que incorpora el nRF Connect SDK.

Esta función introduce al microcontrolador en un modo de bajo consumo excesivamente profundo. Como resultado de esto, al despertar el microcontrolador solamente ejecuta el código del handler de la interrupción que lo despierta. Cuando finaliza la ejecución, en lugar de continuar ejecutando el programa por el punto en el que había entrado en bajo consumo, el microcontrolador vuelve a entrar en bajo consumo. Esto es muy problemático, ya que, siguiendo las buenas prácticas de programación, la función que maneja la llegada de una interrupción debe ser atómica, es decir, debe durar lo menos posible. En el caso del firmware se necesita arrancar la radio, emitir la señal BLE, esperar cierto tiempo y apagar la radio. Evidentemente, esta serie de acciones no puede considerarse atómicas, por lo que se debe buscar otra forma de solucionarlo.

Para arreglar este problema, se ha decidido implementar la función k_wakeup(k_tid_t *pThread). Esta función hace que el hilo que se pasa como puntero por parámetro despierte y salga de todos los modos de bajo consumo de forma permanente, permitiendo que se continue ejecutando el código por el punto en el que se había entrado en bajo consumo.

Por este motivo la estructura del programa cambia, y se debe implementar un hilo nuevo en el que se introduce la gestión de la señal BLE del firmware. Dentro del handler de la interrupción se llamará a la función k_wakeup, despertando al hilo de la gestión BLE y permitiendo su correcta ejecución.

A continuación, se explicará el diseño del Firmware2.0:

4.1.3.3.2 Diagramas de diseño

En primer lugar, se muestran los diagramas de componentes, paquetes y despliegue en las figuras 42 y 43 respectivamente. Estos dos diagramas ayudan a entender la estructura y el diseño del Firmware2.0, que módulos contiene, cómo están relacionados y cómo se comunican entre ellos:

- **Diagrama de componentes:** Representa todos los módulos que gestiona el Firmware1.0 durante su ejecución.

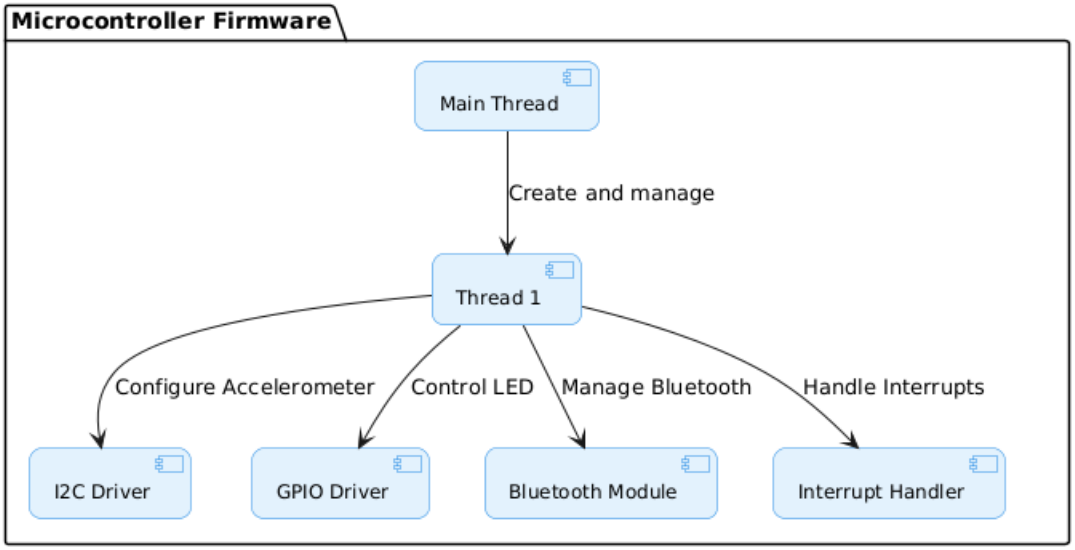


Figura 42. Diagrama de componentes de Firmware2.0

- **Diagrama de paquetes:** Muestra cómo se organizan de forma lógica los módulos del firmware en distintas capas de abstracción:

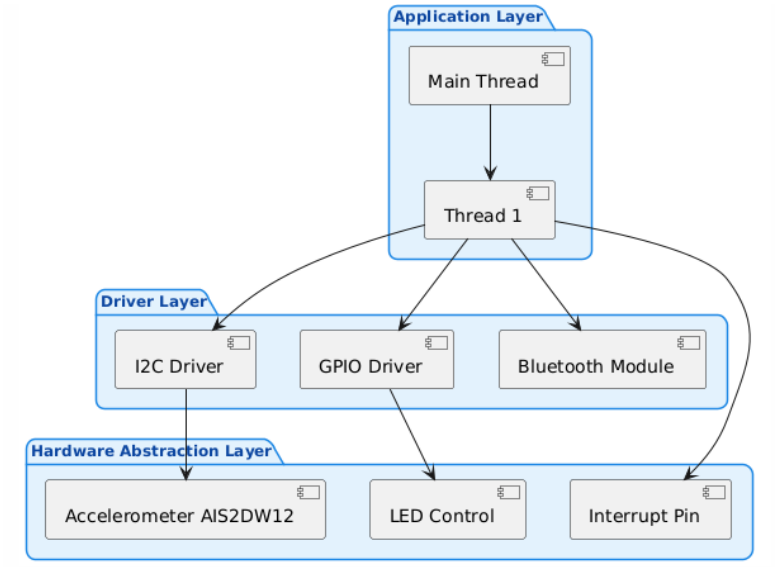


Figura 43. Diagrama de paquetes de Firmware2.0

A continuación, en la figura 44 se muestra el diagrama de secuencia del Firmware2.0, es decir, en este diagrama se va a representar el orden en el que se ejecutan las operaciones entre los distintos componentes del sistema que hemos visto anteriormente en el diagrama de paquetes:

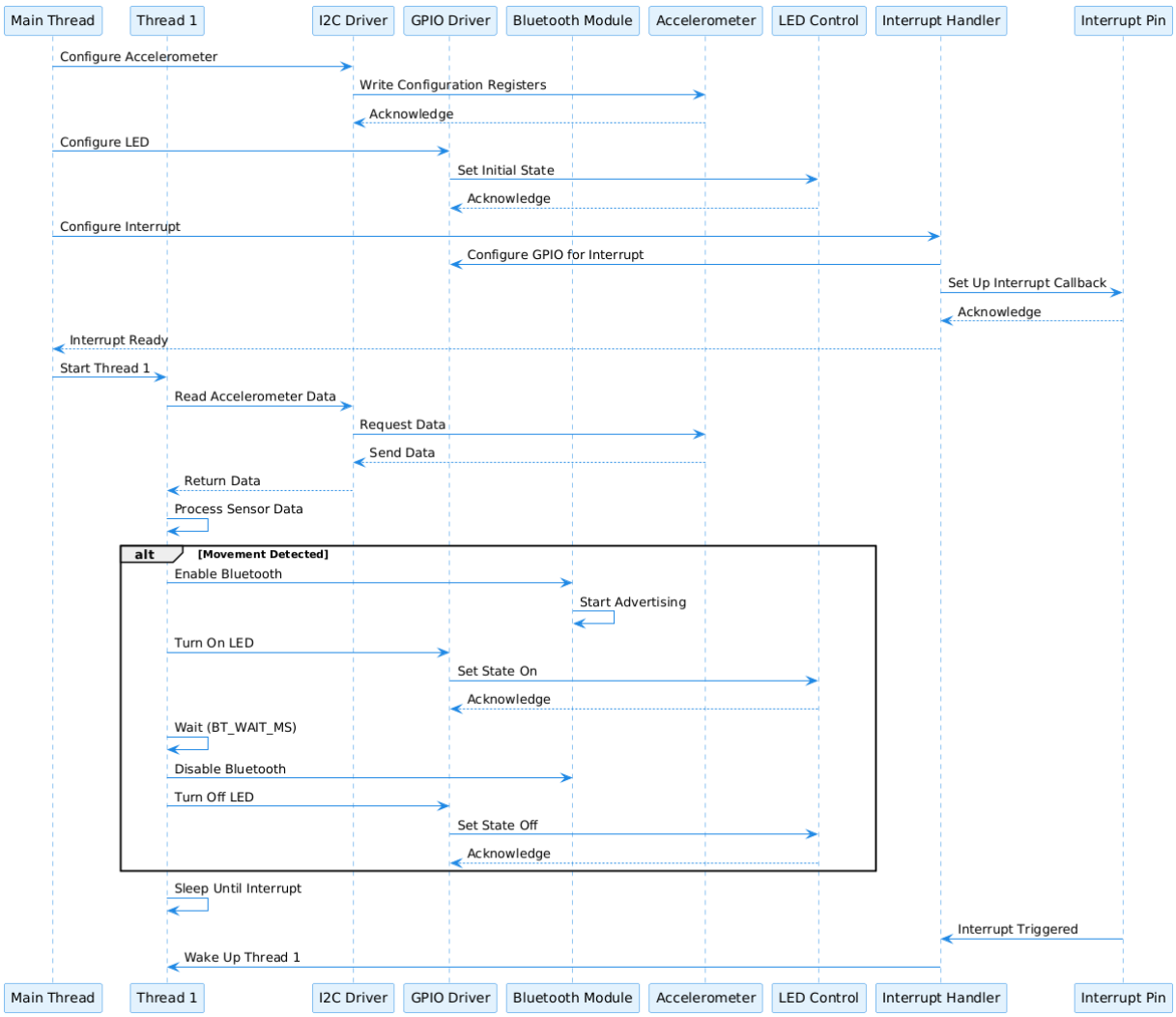


Figura 44. Diagrama de secuencia de Firmware2.0

En la figura 45 se representa el diagrama de actividad, el cual representará el flujo de trabajo y las actividades que existen dentro del Firmware2.0. Gracias a este diagrama podemos entender fácilmente la secuencia de actividades que tiene el Firmware2.0 y que decisiones toma el software en función del estado de ejecución del programa:

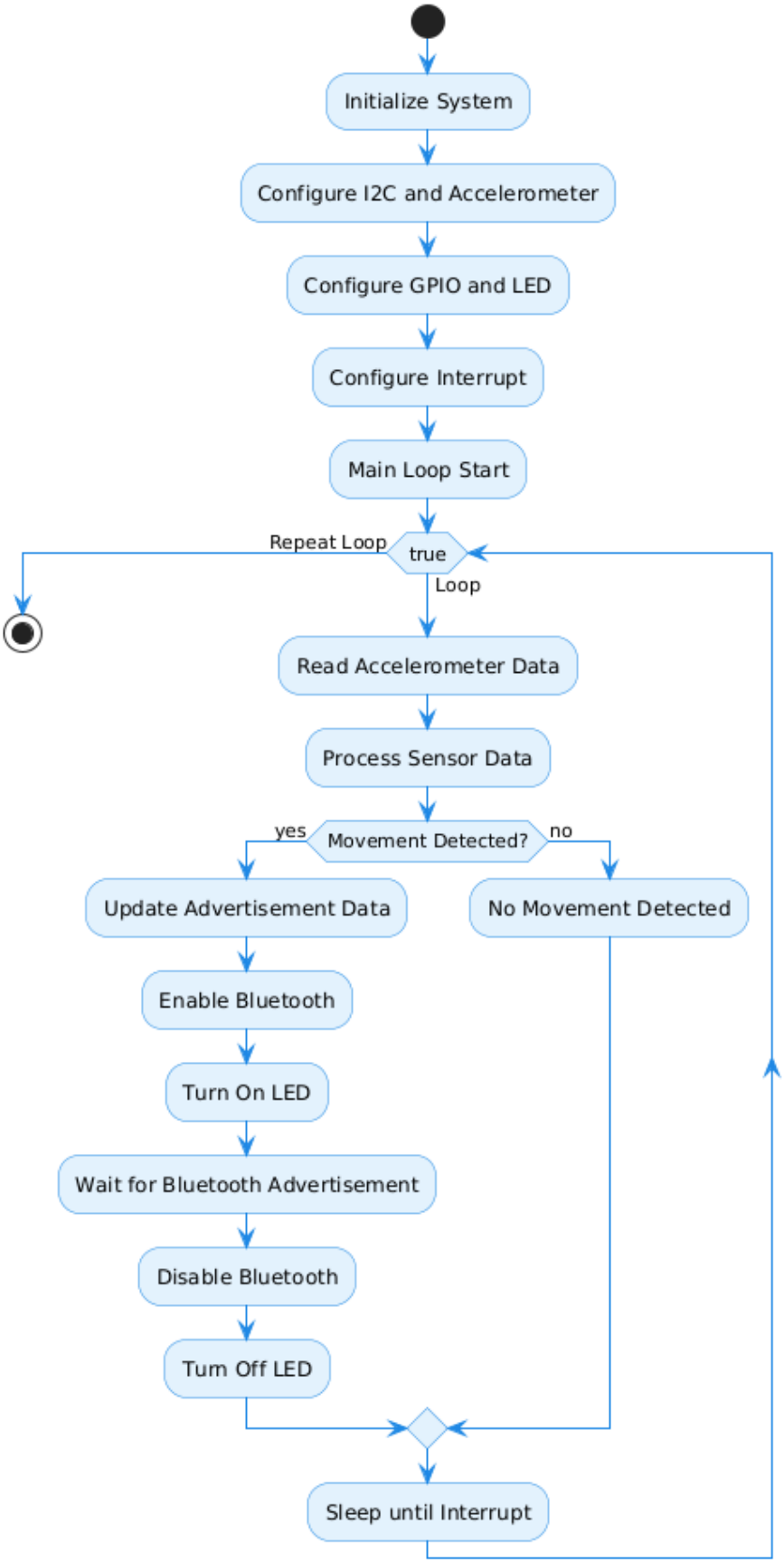


Figura 45. Diagrama de actividad de Firmware2.0

4.1.3.3.3 Librerías

Los cambios que se han implementado en la segunda versión del Firmware no afectan a las librerías importadas, por lo que en el Firmware2.0 sigue siendo válida la Tabla 5, que se presentó para Firmware1.0

4.1.3.3.4 Funciones

A continuación, en la Tabla 7, se incluye una tabla con las funciones principales que se definen nuevas en el código de Firmware2.0, puesto que las definidas en la Tabla 6, que se presentó para Firmware1.0, siguen siendo válidas para esta nueva versión:

Tabla 7. Funciones de Firmware2.0

Función	Parámetros	Salida	Descripción
interrupt_handler	const struct device *dev, struct gpio_callback *cb, uint32_t pins	-	Maneja la interrupción del GPIO, despertando el microcontrolador al activarse la interrupción.
configure_interrupt	-	-	Configura el GPIO para la interrupción, estableciendo el manejador de la misma.
thread1_func	-	-	Implementa el ciclo principal de Thread 1, que incluye la lectura de datos, procesamiento, control del LED, y operaciones de Bluetooth.

El cambio principal respecto a la primera versión es la introducción del thread que permite la correcta ejecución del código que maneja la señal BLE. Es importante entender el ciclo de manejo de la interrupción y la ejecución del thread.

Al iniciar el firmware se crea un thread paralelo al principal, denominado thread1, el cual contiene una única función, thread1_func. Tal y como se ha explicado en la tabla 7, esta función se encarga de manejar las operaciones para las señales BLE, además de la notificación mediante el uso del LED.

Al inicio, Thread1 se encuentra en bajo consumo, por lo que no se ejecuta nada. Cuando el acelerómetro detecta movimiento, genera una interrupción, la cual es dirigida al pin 17 del microcontrolador para su lectura. Al despertar el microcontrolador, se ejecuta la función interrupt_handler definida también en la tabla 7. En el handler de la interrupción se invoca la función k_wakeup, la cual permite despertar el thread1, que ejecutará una vez su código y volverá al modo de bajo consumo. Este ciclo se repetirá de forma continua cada vez que se detecte la interrupción del acelerómetro.

4.2 Fase 2: Desarrollo de la app de lectura para señales BLE

El objetivo de esta fase es desarrollar una app para dispositivos Android la cual permita leer las distintas señales BLE generadas tanto por las balizas comerciales como por las diseñadas en la fase 3 de este proyecto.

Además, algunos requisitos adicionales de la aplicación son:

- Posibilidad de almacenar los datos obtenidos
- Ejecución de la aplicación en segundo plano
- Uso sencillo e intuitivo

Estos objetivos se han conseguido a lo largo de distintas iteraciones. Como resultado de estas iteraciones se han desarrollado dos versiones diferenciadas, las cuales, al igual que en el apartado anterior se denominarán como BLEScanner y BLEScannerV2desde ahora.

La primera versión se centra en la detección de las señales BLE, análisis de la información, representación en pantalla y guardado en el historial. El layout usado no es el definitivo, al igual que el diseño gráfico. El salvado de datos en el almacenamiento interno del dispositivo solo puede realizarse de forma manual mediante la pulsación del botón habilitado en el historial.

En la segunda versión se presenta el diseño gráfico definitivo, con una nueva paleta de colores y detalles acabados. Además, se le implementa un logo a la aplicación. En cuanto a funcionalidad se integra la ejecución en segundo plano incluso con la pantalla apagada. También se integra el guardado del historial en un fichero txt de forma automatizada en una estructura de ficheros que se mostrará posteriormente tras el análisis de ambas versiones.

4.2.1 Android SDK

La aplicación hace uso del propio SDK del entorno Android con el cual se ha diseñado desarrollado y depurado la aplicación. Se hace uso de la versión 34 del SDK, la cual corresponde con la versión de Android 14. Además, se ha definido como versión mínima de SDK la 21, correspondiente con Android 5.0 Lollipop. De esta forma la app estará pensada y diseñada para Android 14, pero será compatible desde Android 5.0

4.2.2 Android Studio

La instalación del SDK viene incluida con el entorno Android Studio[54], el IDLE oficial para el desarrollo de aplicaciones Android.

4.2.3 Desarrollo de la aplicación

En la introducción a la fase 2 del proyecto se ha explicado el objetivo de la aplicación, su funcionalidad básica y el motivo por el que existen dos versiones diferenciadas. A continuación, se explicará en detalle el desarrollo de cada una de ellas:

4.2.3.1 BLEScanner

La app BLEScanner basa su funcionalidad en tres ficheros y dos más para el diseño gráfico de la app, que se explicarán más tarde.

En cuanto a la funcionalidad, podemos separar la estructura del proyecto en:

- **MainActivity:** Es la actividad principal de la app. En ella se maneja el análisis de dispositivos Bluetooth con funciones como iniciar y detener el escaneo, obtener la información de cada dispositivo o actualizar la lista de detecciones.
- **HistoryActivity:** esta actividad gestiona la visualización del historial de detecciones.
- **HistoryItem:** Esta clase define la forma en la que se representan los dispositivos dentro del propio historial.

4.2.3.1.1 Permisos

En la app se incluyen algunas funcionalidades que necesitan la concesión explícita de permisos. Por ello es necesario definir qué serie de permisos se van a solicitar junto a la aplicación. Estos permisos se definen dentro del fichero AndrodiManifest.xml dentro de la carpeta "manifest" de la raíz de la app. Es estrictamente necesario que la app cuente con la concesión de los permisos definidos para el correcto funcionamiento de la esta.

A continuación, se muestra una tabla con los permisos necesarios:

Tabla 7. Permisos necesarios para BLEScanner

LOCALIZACIÓN	
ACCESS_FINE_LOCATION	Permite que la app acceda a la ubicación precisa. Necesario en las versiones más recientes de Android.
ACCESS_COARSE_LOCATION	Permite que la app acceda a la ubicación aproximada. Necesario en las versiones más recientes de Android.
BLUETOOTH	
BLUETOOTH	Permite el uso de funcionalidades Bluetooth.
BLUETOOTH_ADMIN	Permite la administración del Bluetooth del dispositivo.
BLUETOOTH_SCAN	Permite las funciones de escaneo de dispositivos bluetooth cercanos. Necesario a partir de Android 12.
BLUETOOTH_CONNECT	Permite las funciones de conexión con dispositivos bluetooth cercanos. Necesario a partir de Android 12.
android.hardware.bluetooth_le	Con esta directiva se asegura que solo dispositivos con compatibilidad para BLE puedan correr la app
ALMACENAMIENTO	
READ_EXTERNAL_STORAGE	Permite leer el almacenamiento externo del dispositivo.
WRITE_EXTERNAL_STORAGE	Permite escribir en el almacenamiento externo.

4.2.3.1.2 Diagramas de diseño

En primer lugar, en la figura 46, se presenta el diagrama del clases. Este diagrama permite visualizar las clases que integran la app, como se relacionan y los de cada una.

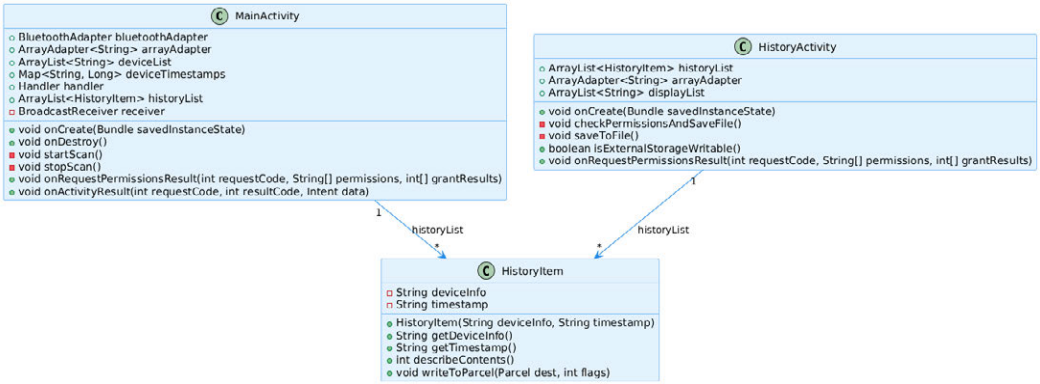


Figura 46. Diagrama de clases de BLEScanner

Los métodos y atributos se representan de la siguiente forma:

Los atributos se representan en la primera mitad de la clase, justo encima de la línea divisoria.

- **Atributos públicos:** Se representan con un círculo verde sin relleno.
- **Atributos privados:** Se representan con un cuadrado rojo sin relleno.

Los métodos se representan en la segunda mitad de la clase, justo debajo de la línea divisoria.

- **Métodos públicos:** Se representan con un círculo verde con relleno.
- **Métodos privados:** Se representan con un cuadrado rojo con relleno.

El diagrama de componentes de la figura 47 proporciona una visión global de todos los módulos que ejecutan distintas acciones dentro de las clases definidas por el diagrama anterior.

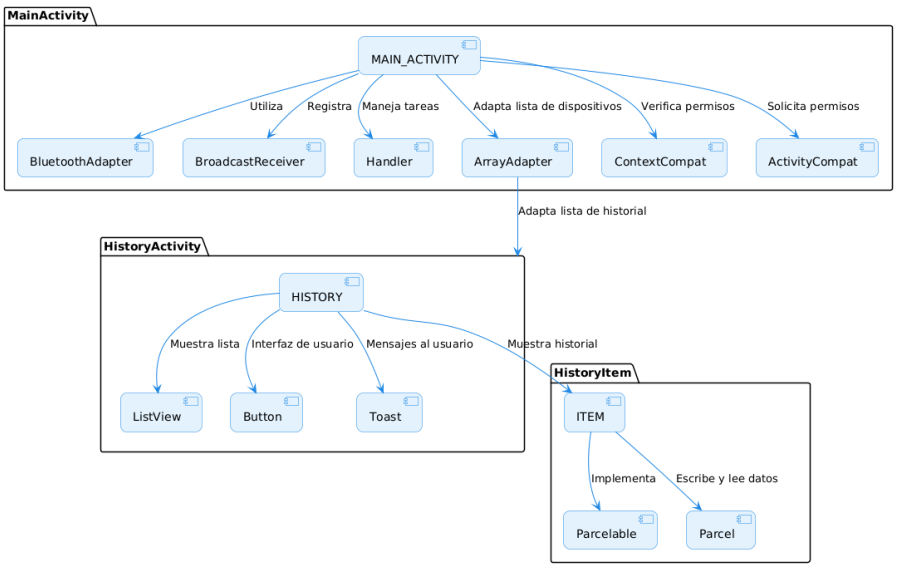


Figura 47. Diagrama de componentes de BLEScanner

También es importante explicar el diagrama de secuencia, el cual puede observarse en la figura 48. Este diagrama permite ver cómo evoluciona la app y que acciones se ejecutan entre los distintos módulos para comprender el flujo de información que se maneja de forma interna.

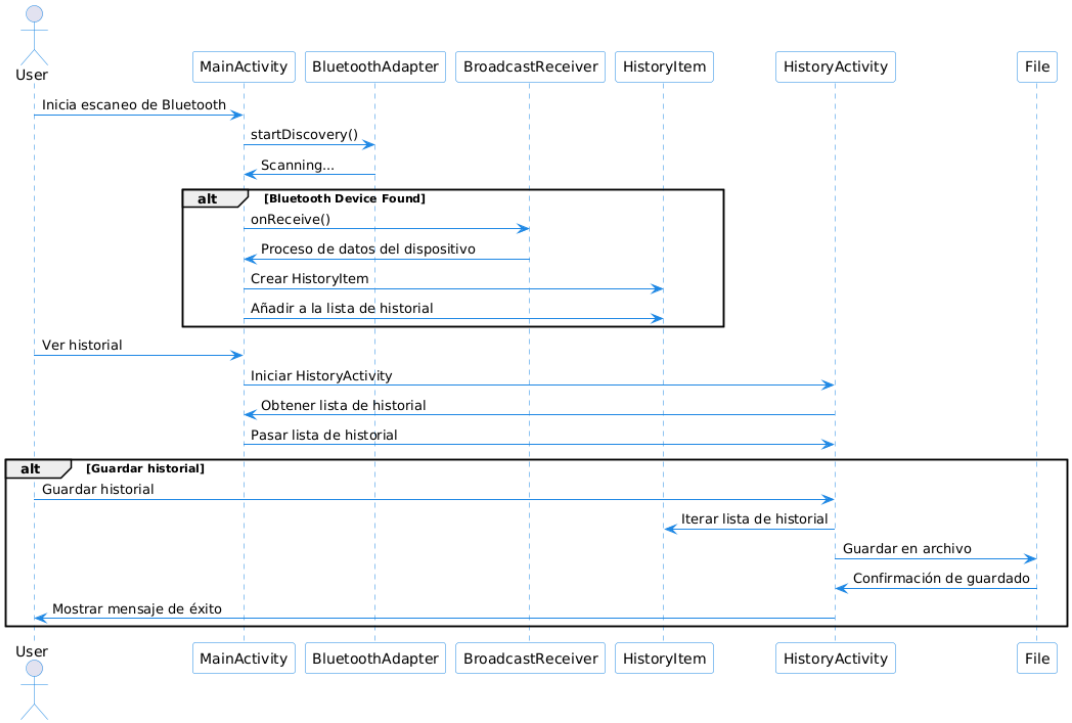


Figura 48. Diagrama de secuencia de BLEScanner

A diferencia que, con el firmware, en el scanner el usuario marca un papel fundamental para el funcionamiento, ya que puede ejecutar varias acciones. En la figura 49 se presenta el diagrama de casos de uso, el cual ayuda a tener una visión clara de esas acciones y los resultados que se obtienen.

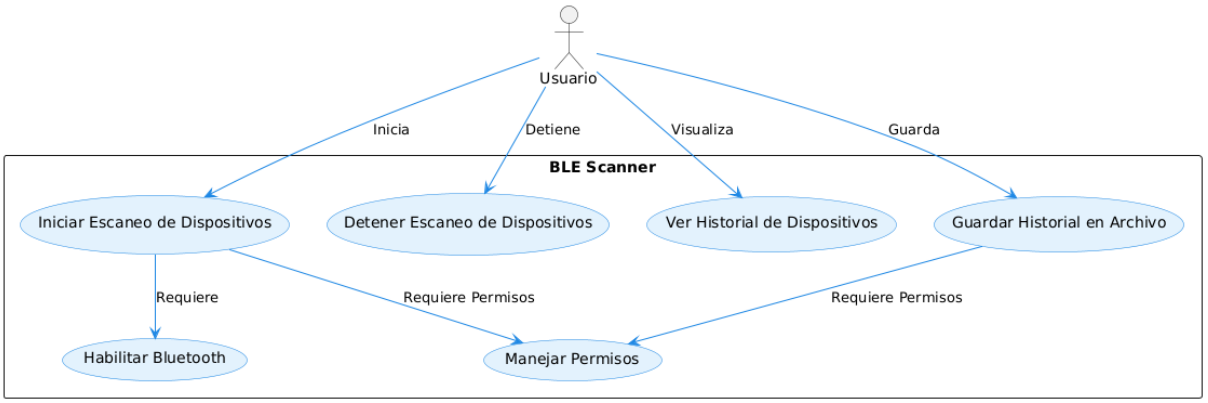


Figura 49. Diagrama de casos de uso de BLEScanner

Debido a la cantidad de permisos que el usuario puede aceptar o denegar y las acciones que hemos visto en el diagrama anterior, existe una gran cantidad de posibilidades en el desarrollo cotidiano de la app. En la figura 50 el diagrama de actividad muestra los puntos críticos de la aplicación y que actividades se ejecutan en función de las decisiones tomadas.

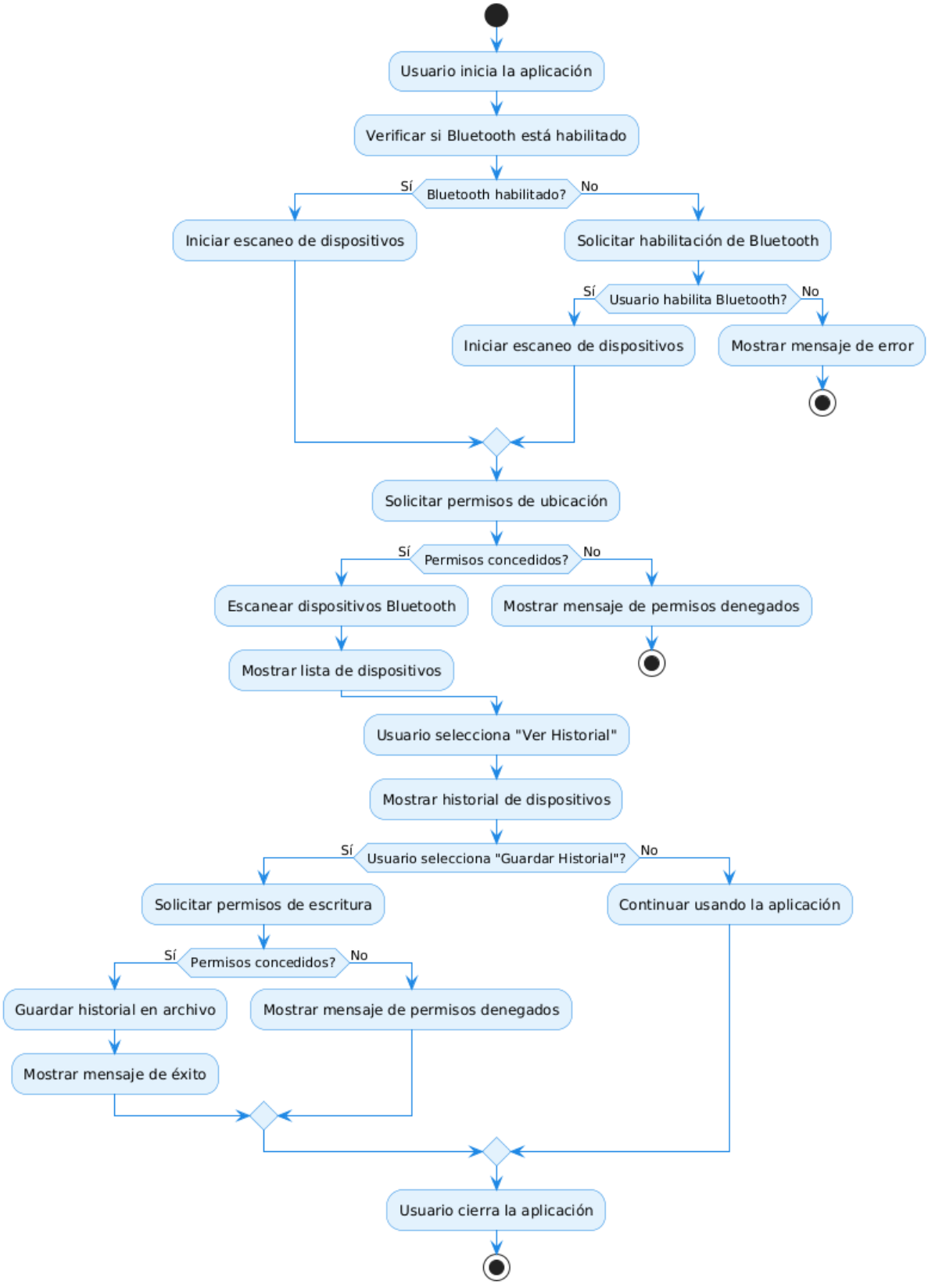


Figura 50. Diagrama de actividad de BLEScanner

4.2.3.1.3 Estructura del proyecto

La estructura del proyecto sigue el esquema estándar de las aplicaciones Android. En la siguiente tabla se muestra la información relevante de la estructura:

Tabla 8. Estructura de BLEScanner

Nombre	Ruta	Contenido	Descripción
Blescanner	src/main/java/ com/adriel/blescanner	MainActivity.java HistoryActivity.java HistoryItem.java	Contiene el código fuente de la aplicación
Layout	res/layout	Activity_main.xml Activity_history.xml	Contiene el código de la parte gráfica
values	res/values	Strings.xml	Contiene las cadenas de texto implementadas
Manifests	Manifests	AndroidManifest.xml	Declara permisos e información fundamental de la aplicación

4.2.3.1.4 Librerías

Las librerías que se han usado están incluidas en el SDK de Android. A continuación, se explican las principales librerías usadas para desarrollar la aplicación:

Tabla 9. Librerías implementadas en BLEScanner

Nombre	Descripción
AndoridX Core y AndroidX APPCompat	Permite implementar las actividades de forma que se asegure la compatibilidad con versiones anteriores
Android Bluetooth API	Es la API estándar de Android para el manejo de operaciones y dispositivos bluetooth. Incorpora clases como BluetoothAdapter, que permite la gestión de la señal o BluetoothDevice, que permite la representación de un dispositivo con todos sus parámetros
Andorid Handler	API que permite enviar y manejar mensajes entre tareas o hilos. Su uso es esencial en la gestión de los dispositivos escaneados por la app.
Android BroadcastReceiver	API para la recepción y gestión de Intents externos al sistema. Esencial para la detección de dispositivos.
Android Intent	API para la comunicación entre componentes de la app
Android Toast	API para mostrar mensajes de feedback al usuario de forma temporal
Android ListView	Permite el uso de listas desplazables para información
Android ArrayAdapter	API que facilita la integración de arrays en ListView
Java Util	Conjunto de utilidades propias de Java

4.2.3.1.5 Métodos

A continuación, se presenta una descripción de los métodos implementados en la aplicación separados por los ficheros en los que se encuentran:

Tabla 10. Métodos de MainActivity de BLEScanner

Método	Parámetros	Salida	Descripción
onCreate	**Bundle savedInstanceState :** Estado anterior de la actividad si está disponible	-	Inicializa la actividad Configura la interfaz de usuario Registra el BroadcastReceiver para eventos de Bluetooth.
onDestroy	-	-	Cancela el escaneo de dispositivos Bluetooth y elimina el BroadcastReceiver para evitar fugas de memoria.
onRequestPermissionsResult	**int requestCode:** Código de solicitud de permiso **String[] permissions:** Permisos solicitados **int[] grantResults:** Resultados de la solicitud	-	Maneja la respuesta a la solicitud de los permisos presentados al usuario
onActivityResult	**int requestCode:** Código de solicitud **int resultCode:** Resultado de la operación **Intent data**: Datos adicionales	-	Maneja el resultado de la solicitud de habilitación de Bluetooth, iniciando el escaneo si Bluetooth fue habilitado correctamente.
startScan	-	-	Inicia el escaneo de dispositivos Bluetooth
stopScan	-	-	Detiene el escaneo de dispositivos Bluetooth
deviceReceiver.onReceive	**Context context:** Contexto de la aplicación **Intent intent:** Intento que contiene los datos	-	Maneja los eventos de detección de dispositivos Bluetooth. Actualiza la lista de dispositivos Añade elementos al historial si corresponde.

Tabla 11. Métodos de HistoryActivity de BLEScanner

Método	Parámetros	Salida	Descripción
onCreate	**Bundle savedInstanceState:** Estado anterior de la actividad si está disponible	-	Inicializa la actividad y carga los dispositivos guardados en el historial
onRequestPermissionsResult	**int requestCode, String[] permissions, int[] grantResults**	-	Maneja la respuesta del usuario a las solicitudes de permisos, concretamente a los permisos de escritura
saveToFile	-	-	Guarda el historial en el almacenamiento del dispositivo
isExternalStorageWritable	-	Retorna **true** si está disponible, **false** en caso contrario.	Verifica si el almacenamiento está disponible para ecritura

Tabla 12. Métodos de HistoryItem de BLEScanner

Método	Parámetros	Salida	Descripción
HistoryItem	**String deviceInfo:** Nombre del dispositivo **String timestamp**: fecha y hora de la detección	-	Constructor del objeto HistoryItem
getDeviceInfo	-	String: Devuelve la información del dispositivo	Obtiene la información del dispositivo
getTimestamp	-	String: Devuelve el Timestamp de detección	Obtiene la marca el Timestamp en el que se detectó el dispositivo
describeContents	-	Int: devuelve 0	Método requerido para la interfaz “Parcelable”
writeToParcel	**Parcel dest:** destino de la información **Int flags:** flags que se envían	-	Escribe los datos de HistoryItem a un Parcel

4.2.3.1.6 Integración con servicios Android

Para la comunicación de las distintas clases (Activity) que se han desarrollado se usan servicios proporcionados por el propio SDK de Android.

- **Intents:** Los Intents[55] son uno de los principales mecanismos implementados por Android para la navegación entre diferentes componentes. En BLEScanner los Intents se usan para los siguientes casos:
  - Inicializar HistoryActivity desde el MainActivity cuando se solicita ver el historial.
  - Pasar datos entre actividades: HistoryActivity manda la lista de dispositivos hasta el MainAcitvity mediante un Intent. Para facilitar el uso primero se encapsulan dentro de un Parcelable[56].
  - Solicitudes: Si necesita solicitarse un permiso, la aplicación usa un Intent, el cual desencadena un diálogo para que el usuario acepte o rechace la concesión del permiso.
  - IntentFilters: son usados para filtrar el tipo de Intent que se recibe. Este componente es clave para la gestión de los dispositivos bluetooth en el BroadcastReceiver.

- **BroadcastReceiver:** es un componente que permite recibir Intents transmitidos por el sistema o por otras aplicaciones externas, como por ejemplo las balizas BLE. En BLEScanner el BroadcastReceiver se registra dinámicamente en el MainActivity para recibir notificaciones cuando se detecta un dispositivo Bluetooth.

4.2.3.1.7 Layout

A continuación, se mostrará la representación gráfica de todos los elementos de la aplicación. En primer lugar, al instalar BLEScanner en el dispositivo se creará un enlace directo como el de la figura 51:

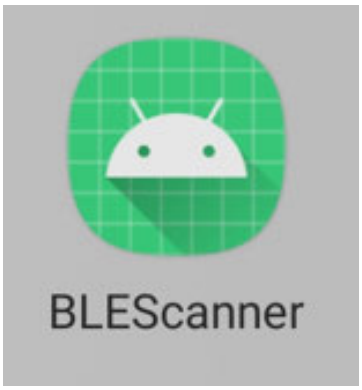


Figura 51. Icono de BLEScanner

El Layout de BLEScanner se basa en 2 ficheros xml:

- **Activity main.xml:** Este fichero define la estructura donde se implementa MainActivity. En las siguientes figuras se observa la representación gráfica del fichero:

  En la figura 52 se muestra la pantalla principal de la app según inicia. En la parte superior se encuentra el título de la aplicación. Justo debajo, en la parte central de la pantalla se encuentra la lista en la que se representan todos los dispositivos Bluetooth detectados por el móvil. Finalmente, en el parte inferior encontramos tres botones, los cuales, de izquierda a derecha sirven para: activar escaneo, parar escaneo y mostrar el historial de dispositivos. Cuando se pulsa el botón de *START* o *STOP*, el usuario es notificado mediante el uso de un toast[57], con los mensajes de "Scanning started" o "Scanning Stoped" respectivamente.

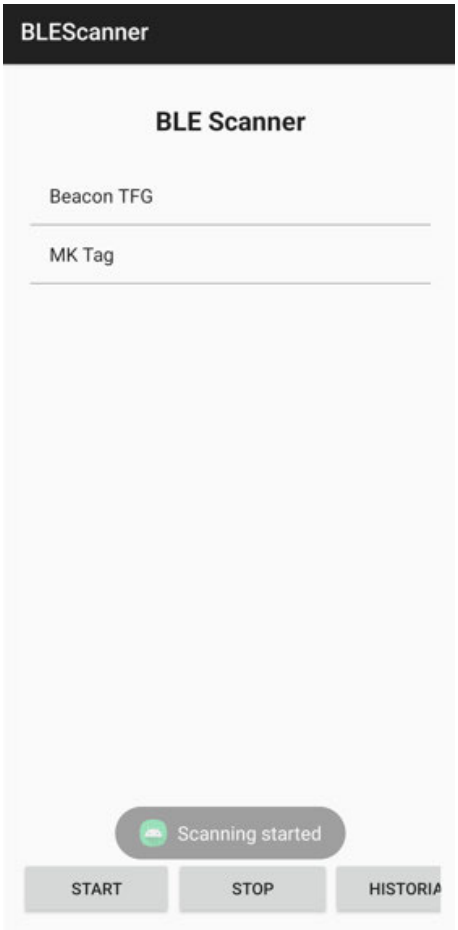


Figura 52. Página principal BLEScanner

Si se presiona sobre el botón de *HISTORIAL,* se cambia la vista de la aplicación y se ejecuta el HistoryActivity, como puede verse en la figura 53.

- **History_main.xml:** Este fichero define la estructura donde se implementa HistoryActivity. En las siguientes figuras se observa la representación gráfica del fichero: En esta vista se encuentran dos componentes:

  En la parte central se encuentra una lista de todos los dispositivos detectados, guardados con nombre y timestamp en el que fueron detectados. Se encuentran ordenados de forma descendente por el tiempo de detección.

  En la parte inferior existe el botón de *SAVE TO FILE,* que como su nombre indica, permite almacenar el historial en un fichero .txt en el almacenamiento interno del dispositivo.

Figura 53. Página de historial BLEScanner

4.2.3.2 BLEScannerV2

Tal y como se ha explicado anteriormente, la segunda versión de la aplicación se centra en tres objetivos:

- Conseguir la detección de dispositivos en segundo plano sin necesidad de tener la aplicación abierta de forma continua.
- Mejorar el sistema de guardado de los dispositivos registrados.
- Mejorar el apartado visual de la aplicación.

A continuación, se desglosarán los cambios de esta nueva versión:

4.2.3.2.1 Permisos

Debido a las nuevas funcionalidades, deben establecerse permisos nuevos para la aplicación. Los permisos definidos para BLEScanner siguen aplicándose en esta versión.

A continuación, en la tabla 13 se muestran los permisos de esta versión:

Tabla 13. Permisos añadidos en BLEScannerV2

LOCALIZACIÓN	
ACCESS_FINE_LOCATION	Permite que la app acceda a la ubicación precisa. Necesario en las versiones más recientes de Android.
ACCESS_COARSE_LOCATION	Permite que la app acceda a la ubicación aproximada. Necesario en las versiones más recientes de Android.
BLUETOOTH	
BLUETOOTH	Permite el uso de funcionalidades Bluetooth.
BLUETOOTH_ADMIN	Permite la administración del Bluetooth del dispositivo.
BLUETOOTH_SCAN	Permite las funciones de escaneo de dispositivos bluetooth cercanos. Necesario desde Android12.
BLUETOOTH_CONNECT	Permite las funciones de conexión con dispositivos bluetooth cercanos. Necesario desde Android12.
android.hardware.bluetooth_le	Con esta directiva se asegura que solo dispositivos con compatibilidad para BLE puedan correr la app
ALMACENAMIENTO	
READ_EXTERNAL_STORAGE	Permite leer el almacenamiento externo.
WRITE_EXTERNAL_STORAGE	Permite escribir en el almacenamiento externo.
SEGUNDO PLANO	
FOREGROUND_SERVICE	Permite que la app se mantenga activa en primer plano, aunque no esté abierta.
FOREGROUND_SERVICE_LOCATION	Permite que la app acceda a la ubicación en un servicio de primer plano, aunque no esté abierta.
WAKE_LOCK	Permite a la aplicación adquirir una forma de evitar que la app entre en modo suspensión al apagar la pantalla.

4.2.3.2.2 Diagramas de diseño

En primer lugar, en la figura 54, se presenta el diagrama del clases. Este diagrama permite visualizar las clases que integran la app, como se relacionan y los de cada una.

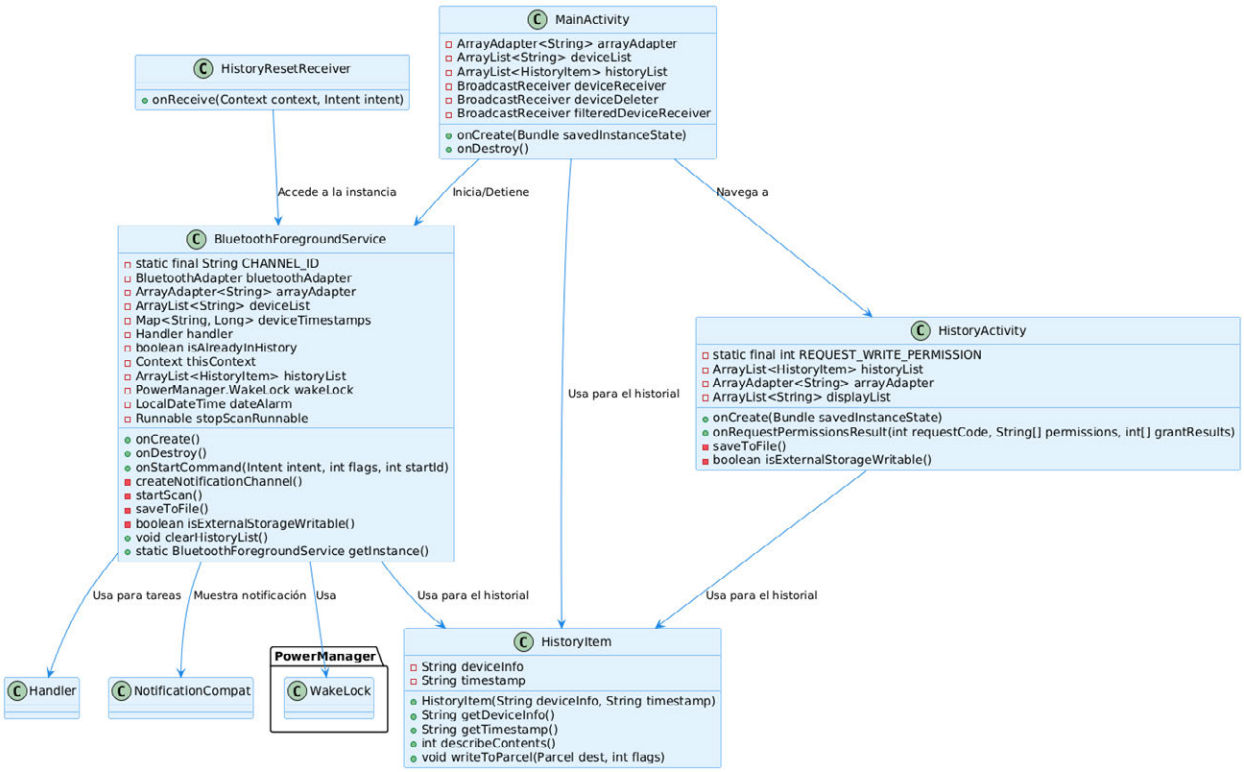


Figura 54. Diagrama de clases de BLEScannerV2

El diagrama de componentes de la figura 55 proporciona una visión global de todos los módulos que ejecutan distintas acciones dentro de las clases definidas por el diagrama anterior.

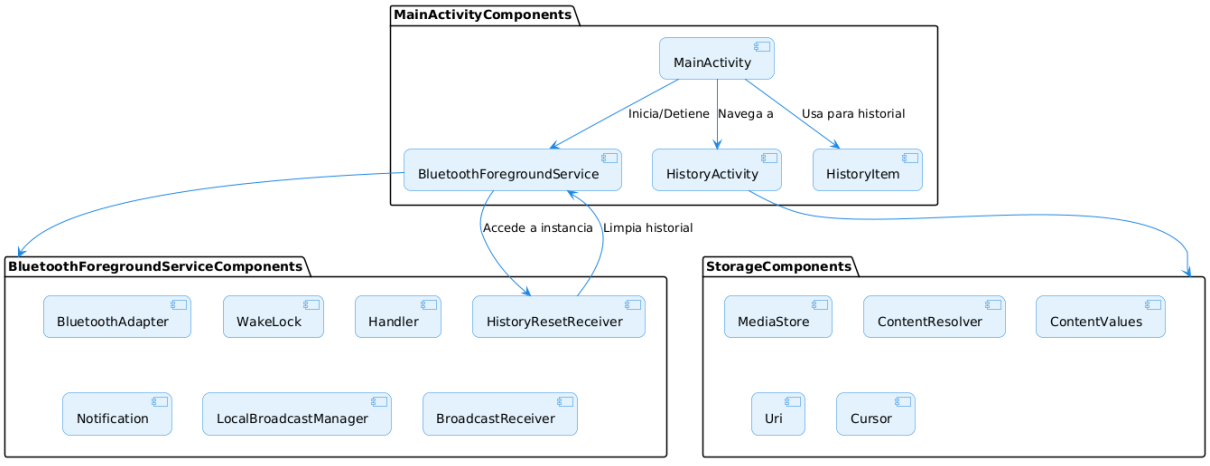


Figura 55. Diagrama de componentes de BLEScannerV2

También es importante explicar el diagrama de secuencia, el cual puede observarse en la figura 56. Este diagrama permite ver cómo evoluciona la app y que acciones se ejecutan entre los distintos módulos para comprender el flujo de información que se maneja de forma interna.

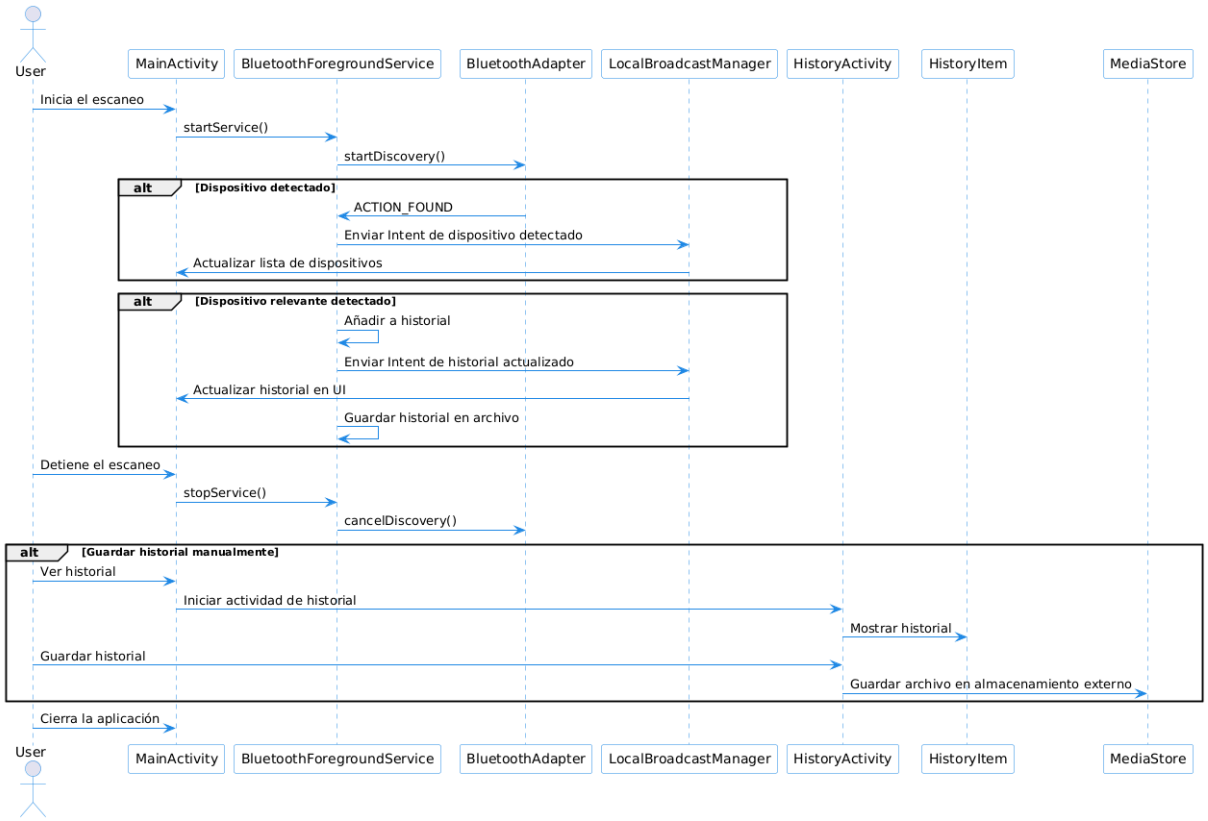


Figura 56. Diagrama de secuencia de BLEScannerV2

A diferencia que, con el firmware, en el scanner el usuario marca un papel fundamental para el funcionamiento, ya que puede ejecutar varias acciones. En la figura 57 se presenta el diagrama de casos de uso, el cual ayuda a tener una visión clara de esas acciones y los resultados que se obtienen.

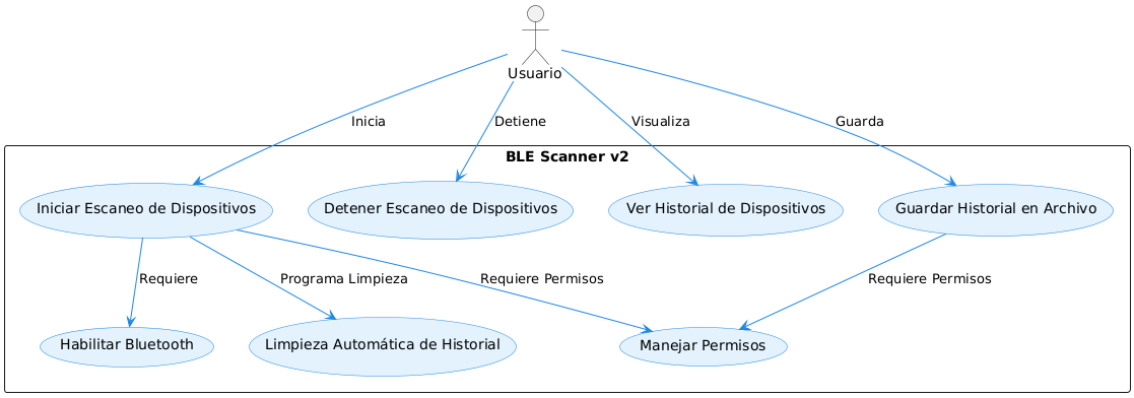


Figura 57. Diagrama de casos de uso de BLEScannerV2

Debido a la cantidad de permisos que el usuario puede aceptar o denegar y las acciones que hemos visto en el diagrama anterior, existe una gran cantidad de posibilidades en el desarrollo cotidiano de la app. En la figura 58 el diagrama de actividad muestra los puntos críticos de la aplicación y que actividades se ejecutan en función de las decisiones tomadas.

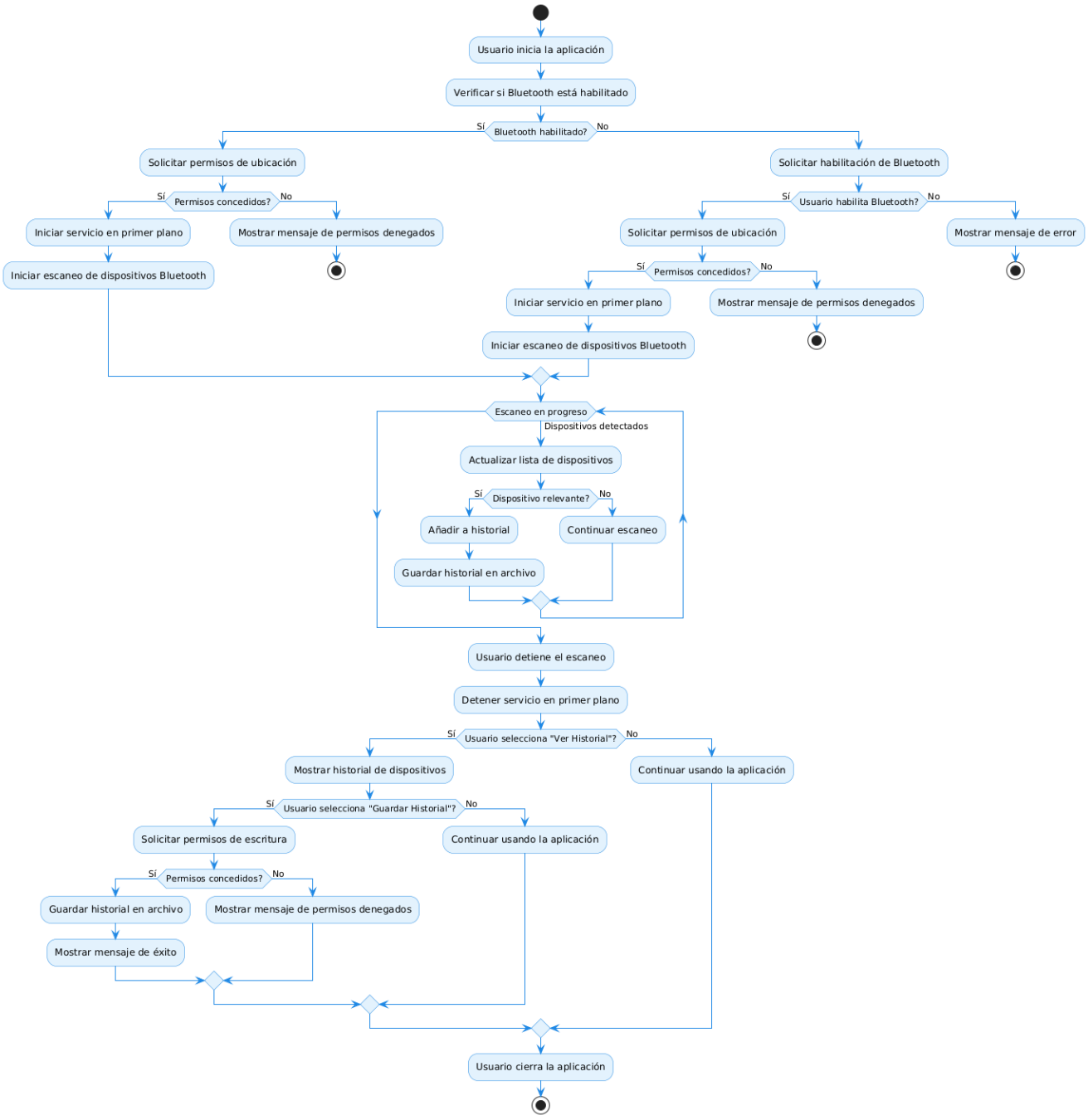


Figura 58. Diagrama de actividad de BLEScannerV2

4.2.3.2.3 Estructura del proyecto

La estructura del proyecto sigue el esquema estándar de las aplicaciones Android. En la siguiente tabla se muestra la información relevante de la estructura:

Tabla 14. Estructura de BLEScannerV2

Nombre	Ruta	Contenido	Descripción
Blescanner	src/main/java/ com/adriel/blescannerv2	MainActivity.java HistoryActivity.java HistoryItem.java BluetoothForegrounService.java HistoryResetReceiver.java	Contiene el código fuente de la aplicación
Layout	res/layout	Activity_main.xml Activity_history.xml	Contiene el código de la parte gráfica
values	res/values	Strings.xml Colors.xml Styles.xml Themes.xml	Contiene definiciones necesarias en el proyecto
Manifests	Manifests	AndroidManifest.xml	Declara permisos e información fundamental de la aplicación

4.2.3.2.4 Librerías

Las librerías que se han usado están incluidas en el SDK de Android. A continuación, se explican las principales librerías implementadas en esta versión. Las librerías de BLSEScanner, expuestas en la tabla 9, siguen siendo aplicables para esta versión.

Nombre	Descripción
LocalBroadcastManager	Similar a BroadcastManager, pero en vez de gestionar Intents externos, se encarga de gestionar los internos que comunican los elementos de la app.
Android PowerManager	Gestiona el estado de energía en el que se encuentra la app. Implementa la posibilidad de configurar el Wake Log que permite ejecutar la app con la pantalla apagada
Android MediaStore	API para facilitar el acceso y gestión de los archivos del almacenamiento externo del dispositivo.
Android Notification	Permite mostrar notificaciones en la barra de estado o indicadores de que la app está corriendo en segundo plano

4.2.3.2.5 Métodos

A continuación, se presenta una descripción de los nuevos métodos implementados en la nueva versión. Los métodos presentados en el apartado correspondiente a la versión anterior siguen siendo aplicables en este caso.

Tabla 15. Métodos de MainActivity de BLEScannerV2

Método	Parámetros	Salida	Descripción
deviceDeleter. onReceive	**Context context:** contexto propio de la app **Intent intent:** comunicador entre elementos	-	Limpia la lista de dispositivos detectados cuando recibe la señal de actualización de la lista
filteredDeviceRevceiver. onreceive	**Context context:** contexto propio de la app **Intent intent:** comunicador entre elementos	-	Actualiza la lista de dispositivos filtrados para guardarlos en el historial
deviceReceiver. onReceive	**Context context:** contexto propio de la app **Intent intent:** comunicador entre elementos	-	Maneja la recepción de los dispositivos bluetooth

Estos tres métodos permiten comunicarse con la clase BluetoothForegroundService, que en esta versión es la que se encarga de la gestión de la lectura de los dispositivos Bluetooth, ya que estas acciones en la versión anterior se realizaban directamente en la MainActivity

Los métodos definidos en las clases de HistoryActivity e HistoryItem son exactamente los mismos en ambas versiones, por lo que no se mostrarán de nuevo en este apartado.

La clase HistoryResetReceiver solo implementa un método, que se encarga de limpiar la lista del historial a partir de las 23.59 para resetear el historial.

Tabla 16. Métodos de HistoryResetReceiver de BLEScannerV2

Método	Parámetros	Salida	Descripción
onReceive	**Context context:** contexto propio de la app **Intent intent:** comunicador entre elementos	-	Maneja la señal que se envía para la limpieza del historial una vez al día

La clase de BluetoothForegroundService implementa toda la funcionalidad relacionada con la detección de dispositivos Bluetooth. En la versión anterior esto se implementaba en la actividad principal, pero en esta nueva versión es necesaria la implementación de este servicio para poder ejecutarlo en segundo plano, permitiendo que los dispositivos sigan siendo detectados con la aplicación cerrada, e incluso con la pantalla apagada.

Tabla 17. Métodos de BluetoothForegroundService de BLEScannerV2

Método	Parámetros	Salida	Descripción
onCreate	-	-	Inicializa el servicio. Inicializa el WakeLock
onDestroy	-	-	Detiene el escaneo de dispositivos y libera el WakeLock
onStartCommand	Intent Intent Int flags Int startId	**Int**: Devuelve el modo en el que se puede reiniciar el servicio	Inicia el servicio en primer plano y comienza el escaneo
createNotificationChannel	-	-	Crea el canal de notificación para las notificaciones persistentes del servicio en ejecución
startScan	-	-	Inicia el escaneo de dispositivos Blutooth
saveToFile	-	-	Guarda el historial en el almacenamiento del dispositivo
isExternalStorageWritable	-	**true** si está disponible, **false** en caso contrario.	Verifica si el almacenamiento externo está disponible
clearHistoryList	-	-	Limpia la lista del historial
getInstance	-	**Bluetooth Foreground Service:** Retorna la instancia del servicio.	Obtiene la instancia del servicio

4.2.3.2.6 Integración con servicios Android

En la versión anterior se usaban Intents y BroadcastReceiver. Estos servicios también son implementados en esta nueva versión, por lo que no se vuelven a explicar en este apartado.

En BLEScannerV2 se implementa un servicio en primer plano (BluetoothForegroundService). Este servicio permite que la aplicación se siga ejecutando en primer plano, aunque se encuentre en segundo plano. Por ello, aunque la app esté minimizada o con la pantalla apagada.

Este servicio se encuentra vinculado a una notificación persistente en el menú de aplicaciones, tal y como se puede ver en la figura 58, la cual asegura que el usuario esté al tanto de que el servicio se está ejecutando.

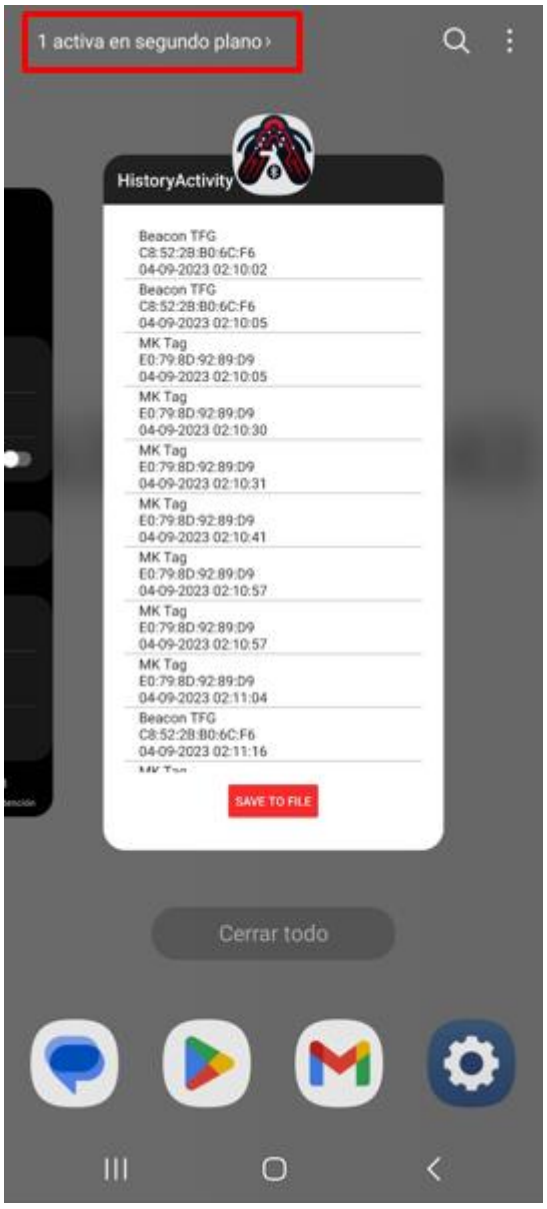


Figura 59. Notificación que indica la ejecución del servicio

4.2.3.2.7 Layout

En primer lugar, al instalar BLEScannerV2 en el dispositivo se creará un enlace directo como el de la figura 60. En esta segunda versión se ha implementado un Icono propio para la aplicación, en vez del uso de uno genérico de Android.

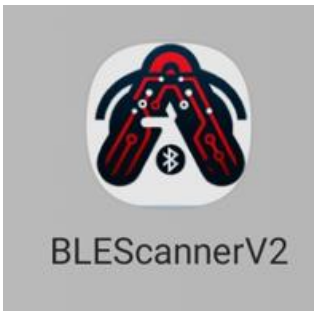


Figura 60. Icono de BLEScannerV2

El Layout de BLEScannerV2 se basa en 2 ficheros xml:

- **Activity main.xml:** Este fichero define la estructura donde se implementa MainActivity. En las siguientes figuras se observa la representación gráfica del fichero:

Figura 61. Pantalla principal de BLEScannerV2

En la figura 61 podemos ver la pantalla principal. Comparada con la versión anterior, el layout es muy similar, pero se ha implementado una paleta de colores más atractiva para mejorar la interfaz gráfica. Se puede ver los siguientes elementos:

- **Título de la aplicación:** En la parte superior se muestra el título de BLEScannerV2.
- **Icono:** justo debajo del título se muestra el icono que se ha diseñado para la aplicación.
- **Lista de dispositivos:** En la parte intermedia de la pantalla se muestran los dispositivos detectados. Respecto a la versión anterior presenta una diferencia, se muestra tanto el nombre como la dirección MAC, permitiendo así una mejor identificación de los dispositivos detectados.
- **Feedback:** en esta nueva versión se hace un guardado del historial en el almacenamiento interno cada vez que se detecta uno de los dispositivos de interés. Cuando se realiza el guardado se muestra el directorio en un toast como confirmación del salvado.
- **Botones:** En la parte inferior se muestran los mismos botones que en la versión anterior.

- **Activity history.xml:** Este fichero define la estructura donde se implementa HistoryActivity. En las siguientes figuras se observa la representación gráfica del fichero:

Figura 62. Historial de BLEScannerV2

En la figura 62 se muestra la página de historial de BLEScannerV2. Al igual que la pantalla principal, el layout es idéntico al de la primera versión. Se ha implementado la misma paleta de color que en la pantalla principal.

El cambio más significativo es la representación de la dirección MAC junto al nombre y el timestamp de cada dispositivo.

4.2.3.3 Salvado de historial

Tal y como se ha explicado en los apartados anteriores, una de las funcionalidades más destacables de ambas versiones es la posibilidad de salvar el historial de las balizas detectadas a lo largo del periodo de escaneo.

Existe una diferencia principal en las formas de guardado entre las dos versiones:

- BLEScanner: En la primera versión solamente se puede guardar el historial en el fichero .txt cuando el usuario acciona el botón *SAVE TO FILE* en la pantalla del historial.
- BLEScannerV2: En la segunda versión se implementa un total de 3 formas para guardar el fichero en el almacenamiento del dispositivos:
    - Se mantiene el salvado manual que se desarrolló en la primera versión.
    - De forma automática el historial se guarda en el fichero .txt cada vez que se detecta la señal de una de las balizas de interés
    - De forma automática se guarda la última copia del historial en el fichero .txt a las 23.59 de cada día. Posteriormente al guardado se elimina el historial de ese día y se crea el directorio para el siguiente. Esto asegura que se genere un fichero .txt nuevo para cada día, permitiendo así una correcta identificación temporal de cada detección.

Para mantener un orden adecuado de los ficheros guardados se ha creado una estructura de carpetas ordenada por fecha. Dentro de la carpeta documentos del dispositivo se encuentra un directorio bajo el nombre de la aplicación.

Dentro de esta carpeta ya encontramos la organización por fecha. Como se observa en la figura 66 encontramos una carpeta por cada año, si se entra en ella puede observarse una carpeta por mes, como en la figura 67, y finalmente, en cada una de ellas, una carpeta por día, como en la figura 68.

Dentro de estas carpetas se guarda el fichero .txt que contiene la información del historial, pero con la particularidad de que se han evitado los permisos de escritura, evitando así posibles errores (por la edición del fichero) en el monitoreo de la actividad del paciente.

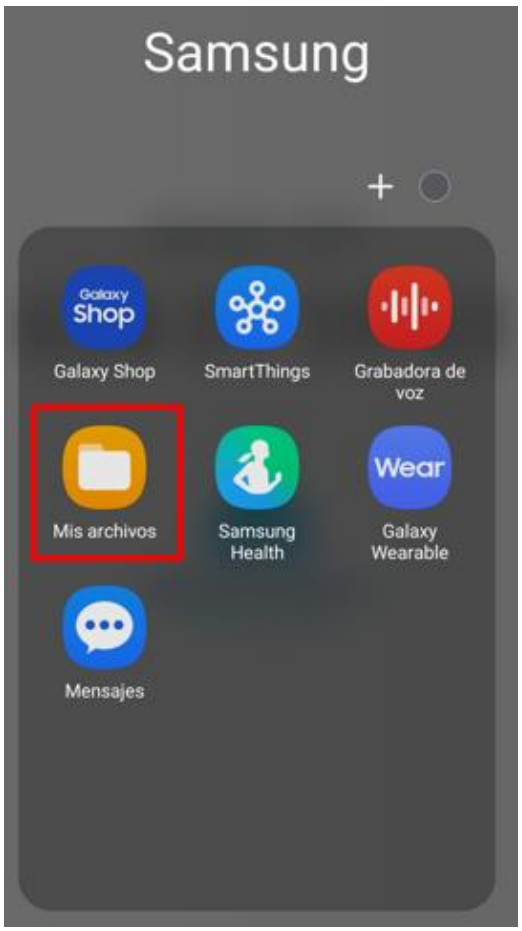


Figura 63. Punto de acceso a los archivos del dispositivo

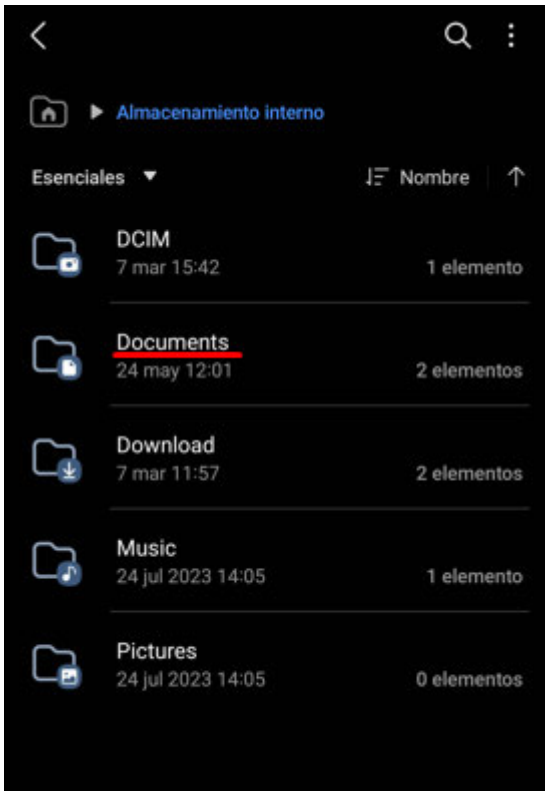


Figura 64. Carpeta Documents

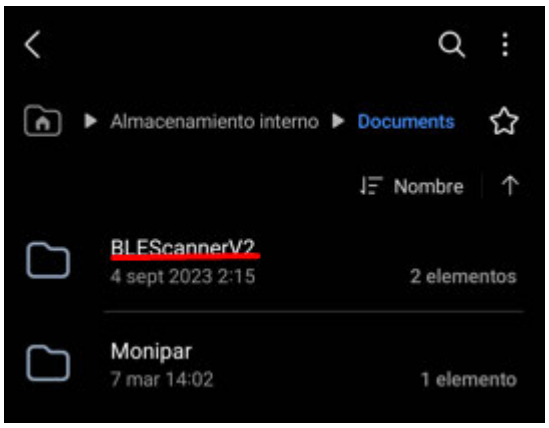


Figura 65. Carpeta de la app

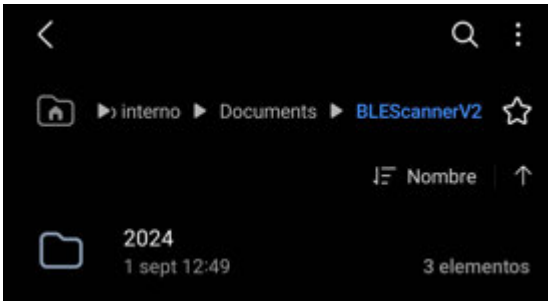


Figura 66. Carpeta de años

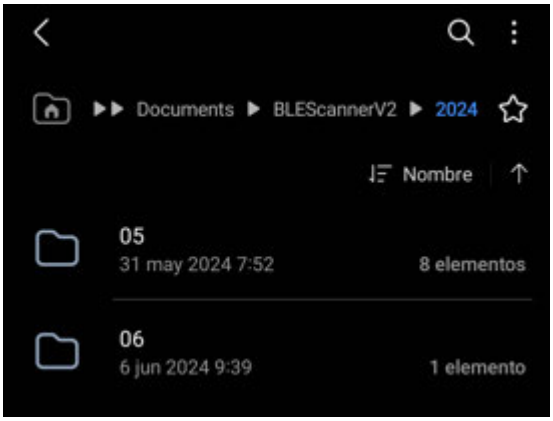


Figura 67. Carpeta de meses

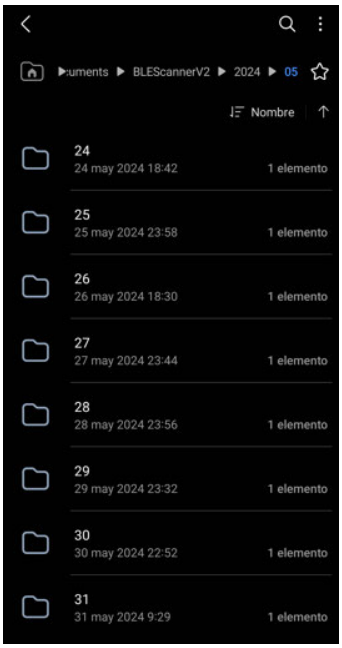


Figura 68. Carpeta de días

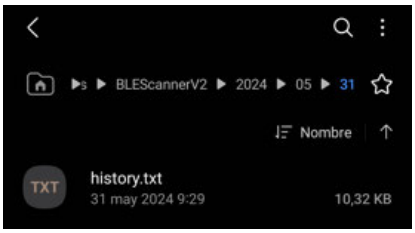


Figura 69. Fichero history.txt

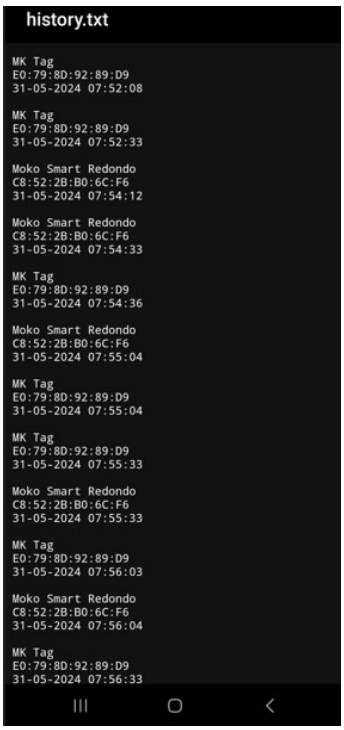


Figura 70. Contenido del fichero history.txt

4.3 Fase 3: Desarrollo y fabricación de PCBs

En esta tercera fase, se comenzó a realizar el diseño de un hardware específico para el objetivo central del proyecto.

Para ello se estudió la disposición y desarrollo de otros dispositivos como las balizas comerciales que usamos y que se expusieron en el apartado 2.2.4.

De igual forma que en las tres fases anteriores, se han obtenido dos versiones de desarrollo.

4.3.1 Herramientas

Para el diseño del hardware se han usado dos herramientas, aunque en principio solo iba a usarse una de ellas. A continuación, se detallan los motivos y características de cada una:

4.3.1.1 Kicad 8.0

Kicad[58] es una herramienta CAD que integra todo lo necesario para el desarrollo de una PCB. La característica más destacable de Kicad es su sencillez. Este programa permite al usuario aprender el manejo básico en un corto periodo de tiempo, y pone a su disposición herramientas suficientes como para conseguir un acabado profesional en el proyecto.

Figura 71. Logo de Kicad

En este software se realizó la primera versión del hardware, el cual se denominará BLE_PCB_V1. Además, también se planificó el desarrollo de la segunda versión, BLE_PCB_V2, aunque finalmente se decidió cambiar de herramienta. El motivo se explicará en los siguientes apartados.

4.3.1.2 EasyEDA

EasyEDA[59] es una herramienta CAD de diseño hardware que puede ejecutarse de forma local o en la nube. Está desarrollada por la compañía JLCPCB[60]. Cuenta con características similares a las de KiCAD.

Figura 72. Logo de EasyEDA

4.3.1.3 Motivo de cambio de herramienta

EL motivo del cambio de herramienta para el diseño fue por causa de una mala planificación temporal. El primer prototipo de BLE_PCB_V1 se recibió a finales de julio. Esto permitió que se tuvieran pocos días para el ensamblado de componentes y pruebas de funcionamiento. A pesar de ello, pudo realizarse en la propia escuela.

La segunda versión debía pedirse en el mes de agosto, y debido a la falta de herramientas para un ensamblado profesional, puesto que la escuela se encontraba cerrada, se decidió el siguiente cambio:

El proveedor JLCPCB, además de fabricar la PCB, también la ensamblaría. Como se ha visto anteriormente, EasyEDA es propiedad de JLCPCB, por lo que la integración entre la herramienta y el pedido del producto facilita enormemente la tarea.

Desde EasyEDA pueden seleccionarse los componentes que se desean ensamblar, la posición y la orientación y se general ficheros tres tipos de ficheros:

- **Gerber:** Estos ficheros permiten al fabricante conocer el diseño de la PCB.
- **BOM:** Este fichero permite conocer al fabricante los componentes seleccionados con todo detalle.
- **PickAndPlace:** Este fichero permite conocer al fabricante la ubicación y orientación exacta de los componentes.

Estos tres ficheros pueden generarse con cualquier herramienta, incluida Kicad, pero deben darse con un formato muy concreto, el cual ya está implementado en los ficheros de EasyEDA.

Por este motivo, para ahorrar tiempo en el formateo de los ficheros y por eliminar el proceso de ensamblado manual, se tomó la decisión de exportar el proyecto de BLE_PCB_V2 a EasyEDA.

4.3.2 Proveedores

Una vez desarrollada una versión, existen dos tipos de proveedores:

- **Proveedores de PCB:** Se encargan de fabricar la placa de circuito impreso diseñada en la herramienta seleccionada. En el caso de este proyecto, como ya se ha comentado, se ha elegido JLCPCB.

- **Proveedores de componentes:** En caso de realizar el ensamblado de los componentes en la PCB, estos deben comprarse por separado. Para este proyecto los componentes se adquirieron en el proveedor online MouserElectronics[61].

4.3.3 Desarrollo de las PCBs

El desarrollo de las PCBs puede separarse principalmente en dos partes:

Primero debe **diseñarse el esquemático**. El esquemático define el diseño eléctrico de un circuito, es decir, representa visualmente como es el conexionado entre los componentes electrónicos que forman un circuito.

Una vez que se ha obtenido un esquemático completo y compilado sin errores se debe **diseñar la PCB**. Este proceso consiste en el diseño del layout físico de la placa, donde es necesario abordar dos hitos importantes:

- En primer lugar, se deben **colocar los componentes** dentro del perímetro de la PCB. Es importante colocar de forma adecuada los componentes agrupándolos en bloques funcionales y que representen un orden lógico para la lectura de la PCB. Además, es recomendable situar los bloques funcionales de forma que se facilite el siguiente paso.

- Una vez colocados los componentes es hora de **trazar las pistas** que van a conectar físicamente los pines de cada dispositivo. Estas pistas son las encargadas de llevar las señales eléctricas, por lo que es importante trazarlas de forma que se adapten al tipo de señal o a la cantidad de corriente que van a transportar. Esto se traduce en que las pistas con mayor corriente, como puede ser la alimentación, siempre deben tener un mayor grosor. En algunas ocasiones pueden usarse plugins de autoruter, los cuales trazan de forma automática las pistas, aunque no suelen seguir de forma rigurosa las reglas de diseño, por lo que no es recomendable abusar de ellos.

Las herramientas de diseño cuentan con una amplia gama de librerías que implementan componentes, pero puede darse el caso de haber seleccionado un componente que no se encuentra importado. Para ello se usarán páginas de librerías. En este proyecto se ha usado UltraLibrarian[62]. De este tipo de páginas puedes descargar tanto el símbolo (representación de componente en el esquemático), como el footprint (representación del componente en la PCB).

Al igual que puede pasar con las herramientas, cabe la posibilidad de que este tipo de páginas no contengan un determinado componente. En este caso, se puede diseñar tanto el símbolo como el footprint con las propias funcionalidades de las herramientas CAD. Para ello debe consultarse el datasheet del componente para asegurarse de que tanto las dimensiones como el conexionado es el adecuado.

A continuación, se presentan las dos versiones de desarrollo que se han realizado en este proyecto:

4.3.3.1 BLE_PCB_V1

Esta versión es un prototipado inicial del hardware que se busca diseñar. El objetivo de esta versión es construir el circuito que se va a reproducir en la segunda versión, pero en un tamaño mayor. Esto facilita el diseño de la PCB, el ensamblado y la detección de posibles problemas.

4.3.3.1.1 Esquemático

A continuación, se va a mostrar el diseño del esquemático de BLE_PCB_V1:

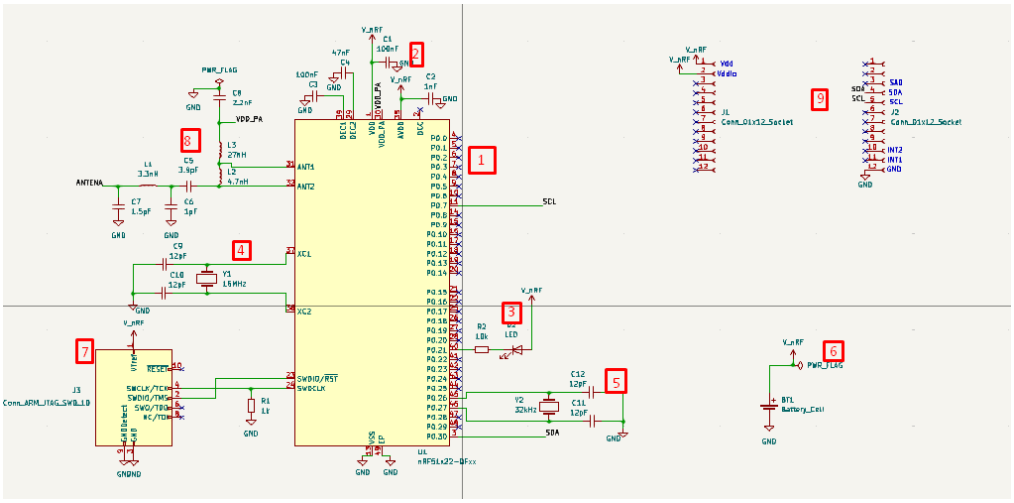


Figura 73. Esquemático completo de BLE_PCB_V1

Como puede observarse en la figura 73, el esquemático puede dividirse en varias partes, la cuales se explican a continuación:

1) Microcontrolador

El microcontrolador es el "cerebro" del circuito. Para esta versión se usará el nRF51422 , proporcionado por NordicSemiconductor. Este componente es el mismo microcontrolador que estaba en la placa de desarrollo nRF51 DK en la que se realizó el firmware del proyecto. En la figura 74 puede observarse el símbolo del microcontrolador:

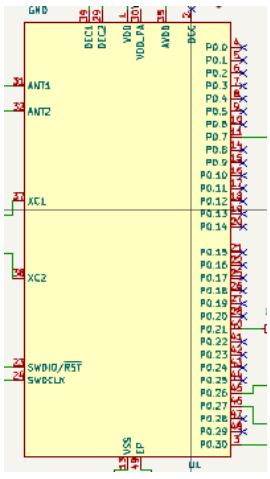


Figura 74. Símbolo del esquemático del nRF51422

2) Componentes de desacoplo y alimentación

Los pines de alimentación son aquellos que reciben las señales de potencia para distribuirla internamente por todos los pines del microcontrolador. Los componentes de desacoplo se encargan de mantener el nivel de esta alimentación completamente estable. En la figura 75 se pueden observar las alimentaciones marcadas en rojo y los condensadores de desacoplo en azul. En el pin 35 y el condensador C2 existe un **error de diseño** que se explicará más tarde

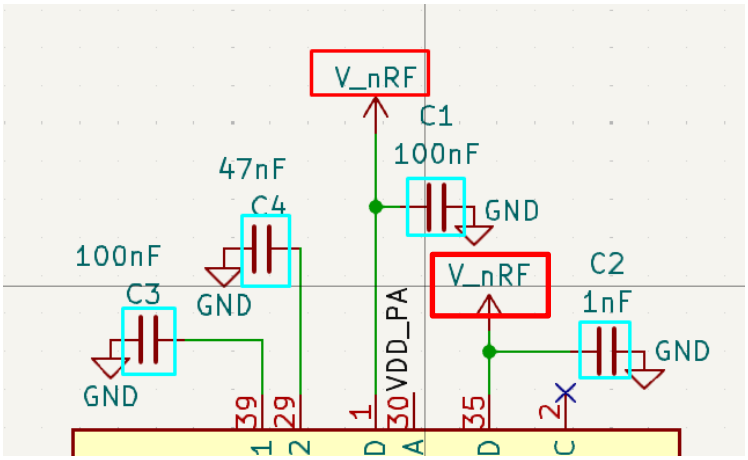
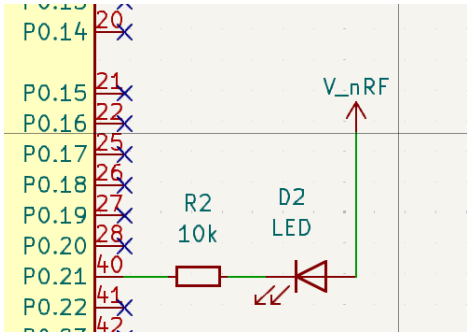


Figura 75.Alimentación y componentes de desacoplo de BLE_PCB_V1

3) Led

En el firmware se definió un led para mostrar de forma visual información sin necesidad de una consola de depuración. En la figura 76 puede observarse la configuración de pull-up de este led:



4) Reloj de 16MHz

Este componente es el principal encargado de generar una señal que oscila a 16MHz y que sirve como reloj del sistema para generar toda la base de tiempos. Puede verse el conexionado en la figura 77:

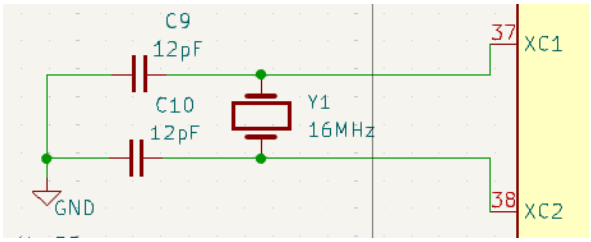


Figura 76. Reloj de 16MHz de BLE_PCB_V1

5) Reloj de 32kHz

Este componente es opcional. Permite tener una segunda fuente de reloj de menor valor que se usará principalmente cuando la frecuencia del reloj caiga en exceso, por ejemplo, en los modos de bajo consumo más extremos del microcontrolador.

El pin SDA que se muestra en la figura **NO pertenece** a la configuración del reloj. Este pin se explicará en los siguientes apartados.

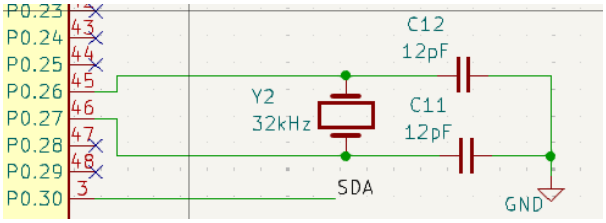


Figura 77. Reloj de 32kHz de BLE_PCB_V1

6) Batería

La alimentación de este dispositivo va a ser por baterías. Concretamente pilas de tipo CR, las cuales permiten un voltaje aproximado de 3.3V en un reducido tamaño. En esta versión se ha usado una pila CR2032 En la figura 78 puede verse el símbolo:

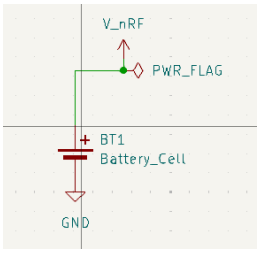


Figura 78. Batería de BLE_PCB_V1

7) Conector

Se ha elegido un conector FTSH de 6 pines con paso 1.27mm en cada pin. Este tipo de conectores pueden usar emuladores Segger para cargar el firmware a través de la interfaz SDW, la cual solo necesita dos pines, uno de reloj (SWDCLK) y uno de datos (SWDIO). En la figura 79 se puede ver la configuración. Cabe destacar, que la resistencia R1 que configura el SWDCLK como pull-down es un **error de diseño** que se comentará más tarde.

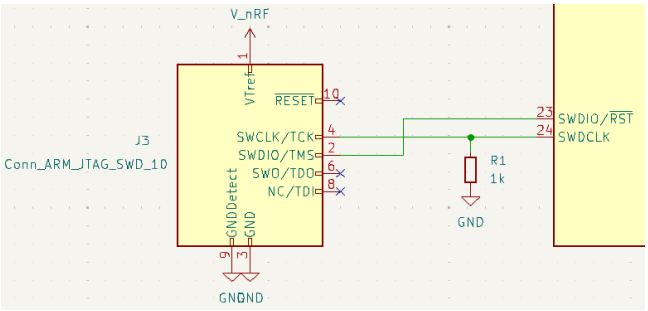


Figura 79. Configuración de conector FTSH

8) Configuración de antena

Los componentes que se muestran en la figura 80 ayudan a tratar la señal de Bluetooth que va a ser transmitida por el microcontrolador.

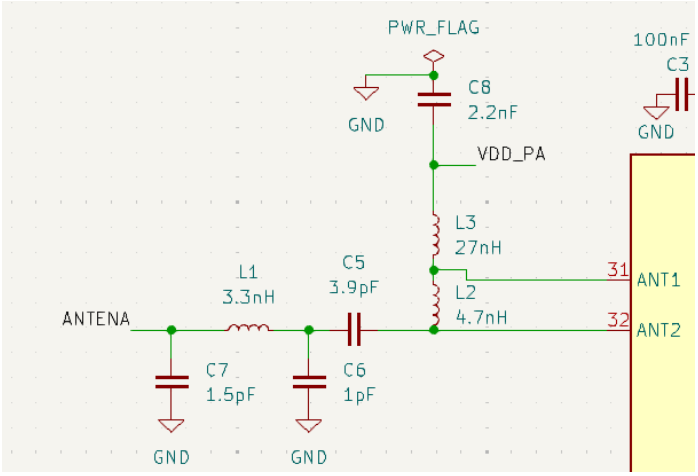


Figura 80. Configuración de antena de BLE_PCB_V1

9) Conexión con acelerómetro

En esta primera versión del hardware se va a implementar la placa STEVAL-MKI206V1, la cual facilita la integración del acelerómetro. Debido a que no se ha encontrado un símbolo adecuado, se han usado dos tiras de 12 pines para la inserción de la palca.

Tal y como se ha visto en la fase 1 con el desarrollo del firmware, el acelerómetro va a ser comunicado con el microcontrolador mediante I2C. Para ello necesitamos conectar los pines de SDA y SCL del microcontrolador con los del acelerómetro. BLE_PCB_V1 está pensado para correr con la primera versión del firmware (firmware1.0), por tanto, no se implementa una linea de interrupciones para el microcontrolador por el movimiento del acelerómetro.

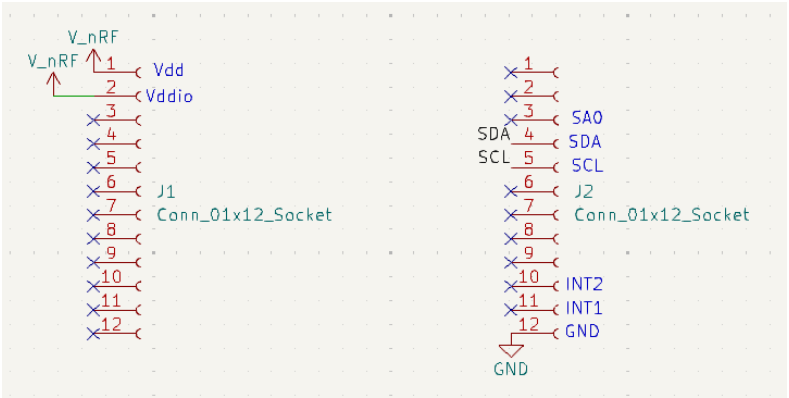


Figura 81. Tiras de pines para la inserción de STEVAL-MKI206V1

Figura 82. Línea de SCL del microcontrolador nRF51422

Figura 83. Línea de SCL del microcontrolador nRF51422

4.3.3.1.2 Desarrollo de PCB

Se ha diseñado una PCB de forma rectangular de 63mm de largo por 39.65mm de ancho. Puede observarse el resultado final en la figura 84, la cual se desglosará a continuación:

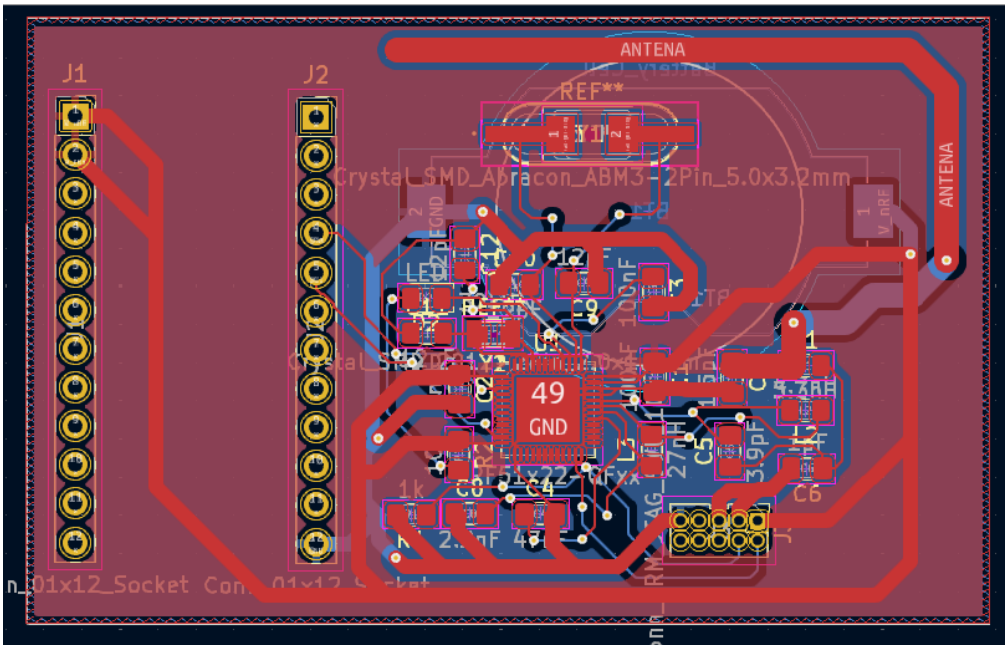


Figura 84. Diseño de PCB de BLE_PCB_1

Tal y como se ve en la figura 84, el microcontrolador se ha situado en el centro de la PCB. Esta decisión ayuda a una mejor distribución de los bloques funcionales alrededor del micro, de forma que las conexiones se faciliten.

A continuación, se indicará la posición de los bloques funcionales más representativos y el porqué de esta ubicación. Para facilitar la visión de los componentes se eliminarán las pistas y planos de masa de la imagen:

En primer lugar, como podemos ver en la figura 85, los componentes que configuran la señal para mejorar su transmisión se encuentran justo a la derecha del microcontrolador, rodeados por las líneas verdes. Se les ha designado esta ubicación debido a que los pines de ANT1 y ANT2 del microcontrolador, los encargados de la emisión de la señal se encuentran en el lateral derecho, marcados por las flechas amarillas.

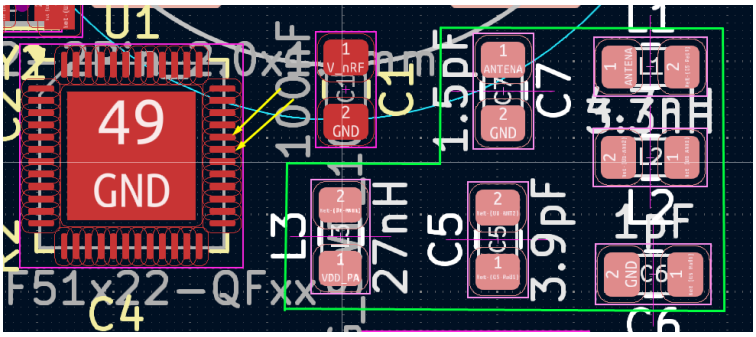


Figura 85. Componentes de configuración de antena y señal BLE_PCB_V1

En la figura 86 se puede ver la ubicación de los osciladores, tanto de 16MHz (arriba del todo), como el de 32kHz (justo encima del microcontrolador). Ambos relojes cuentan con los condensadores necesarios para su funcionamiento, los cuales se sitúan alrededor y están marcados de un color más claro. Se han colocado en esta ubicación ya que los pines de reloj están situados en el lateral superior, marcados en amarillo los del oscilador de 16MHz y los principales de la PCB, y en naranja los de 32kHz. Ambos osciladores se han diseñado con una doble huella, es decir, admiten dos footprints por cada uno de los componentes. Esto se hace con intención de poder ensamblar uno de mayor tamaño para la depuración de la placa o uno de menor tamaño para una presentación más profesional.

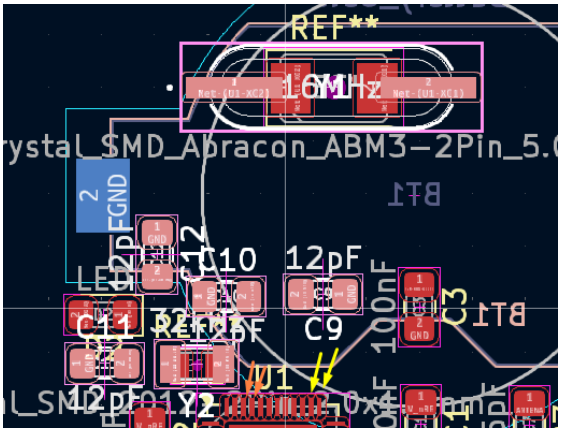


Figura 86. Ubicación de los osciladores de BLE_PCB_V1

En la parte izquierda se han situado las tiras de pines que servirán para insertar la placa de desarrollo que incluye el acelerómetro. Se ha designado esta ubicación puesto que los pines de SDA y de SCL del microcontrolador se encuentran en la parte izquierda del mismo:

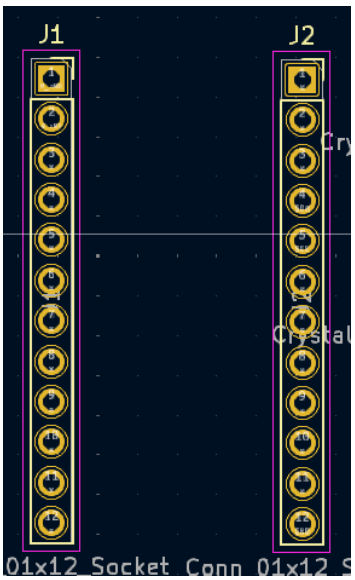


Figura 87. Footprint de las tiras de 12 pines para inserción de placa con acelerómetro

El conector FTSH se encuentra en la parte inferior, cerca de los pines de SWDIO y SWCLK, ambos marcados con las flechas amarillas en la figura 88:

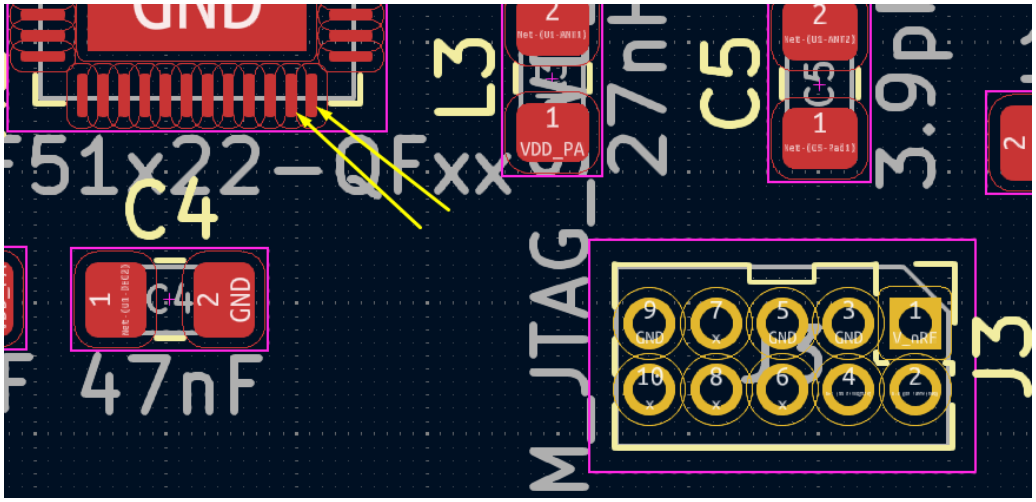


Figura 88. Ubicación de conector FTSH y pines SWD

El resto de los componentes que se han presentado en el esquemático no necesitan una ubicación precisa, aunque de todas formas se han situado de forma que las conexiones se produzcan de la forma más sencilla. Todos los componentes quedan ensamblados en la parte superior, a excepción del zócalo de la batería, el cual se sitúa en la inferior para ahorrar espacio.

Tal y como se ha comentado anteriormente, se han situado dos planos de masa, uno en cada una de las capas, tal y como puede verse en las figuras 89 y 90.

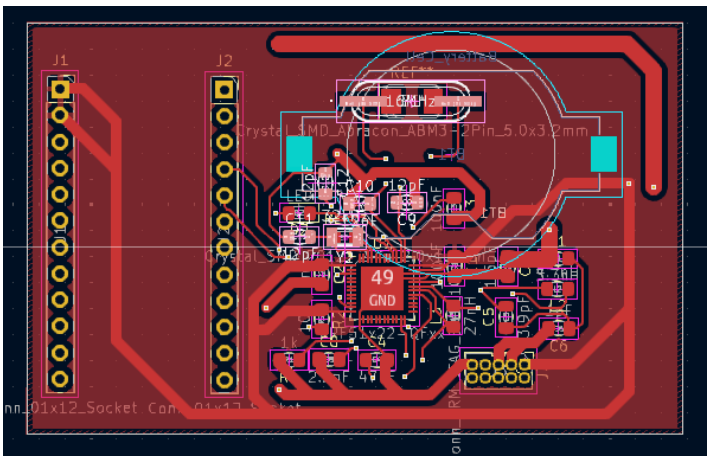


Figura 89. Plano de masa de capa superior (UP)

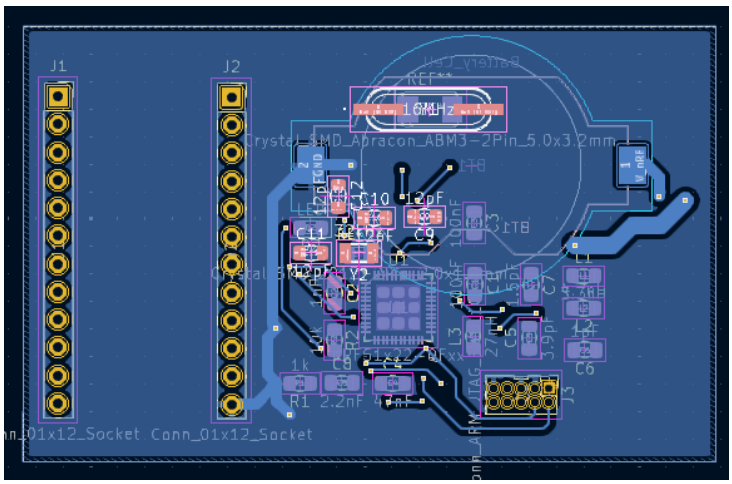


Figura 90. Plano de masa de capa inferior (DOWN)

Estos planos, a pesar de no ser necesarios puesto que se han implementado pistas con conexión a masa, son beneficiosos a la hora de la reducción de ruidos e interferencias electromagnéticas, las cuales pueden influir de forma negativa en la emisión de la señal BLE.

4.3.3.1.3 Presentación de resultados de fabricación

Una vez que se ha finalizado el diseño de la PCB se obtienen los ficheros gerber, los cuales aportan toda la información necesaria al fabricante para la producción.

A continuación, se presentan los resultados de la primera versión del hardware, tanto los simulados con la herramienta Kicad como los fabricados:

- Visor 3D: Gracias a algunas de las herramientas que incluye la herramienta de desarrollo, puede observarse una simulación en 3D del resultado final del diseño. En las figuras 91 y 92 se presenta la simulación de la parte superior e inferior de la PCB:

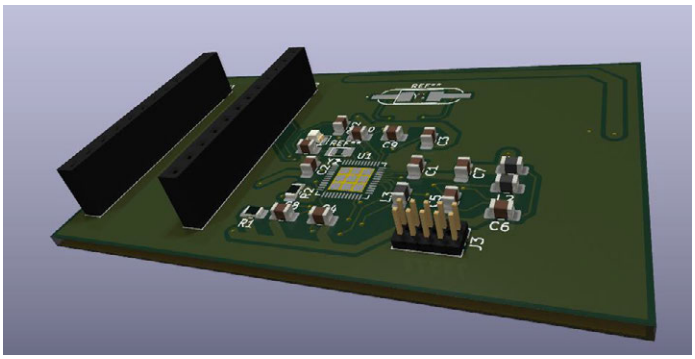


Figura 91. Simulación 3D de la parte superior de la PCB

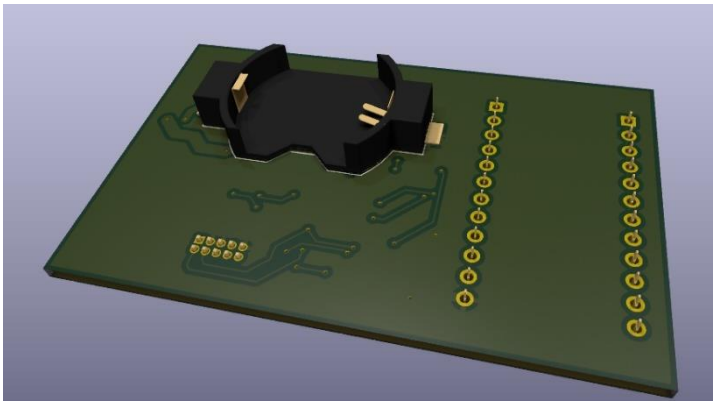


Figura 92. Simulación 3D de la parte superior de la PCB

Una vez recibido el pedido nos encontramos con seis elementos en su interior:

- Una plancha de stencil.
- Cinco copias de la PCB.

La plancha de stencil es una fina lámina, generalmente hecha con acero inoxidable o aluminio, la cual contiene huecos con la forma de los pads de todos los componentes. Esta herramienta facilita enormemente la aplicación de la pasta de soldar en la PCB.

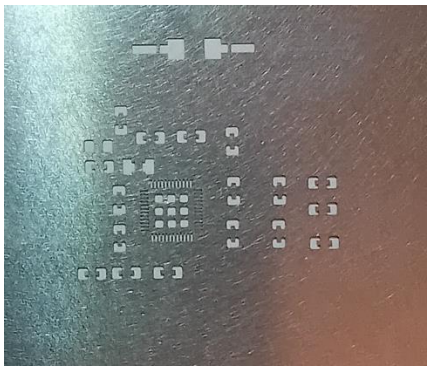


Figura 93. Stencil BLE_PCB_V1

En la figura 94 y 95 se muestra una de las PCBs recibidas, aun sin ensamblar:

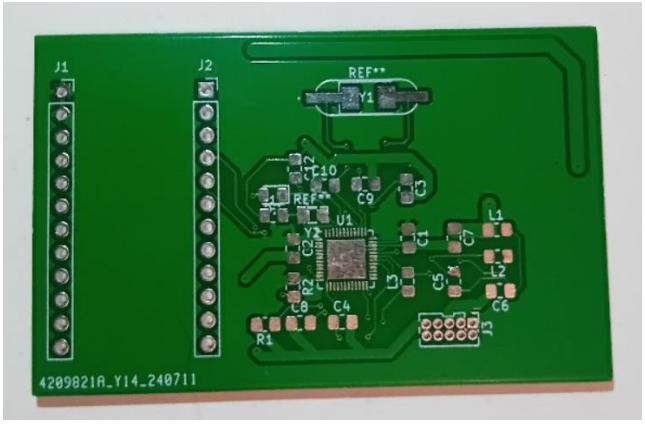


Figura 94. Parte superior BLE_PCB_V1 sin ensamblar

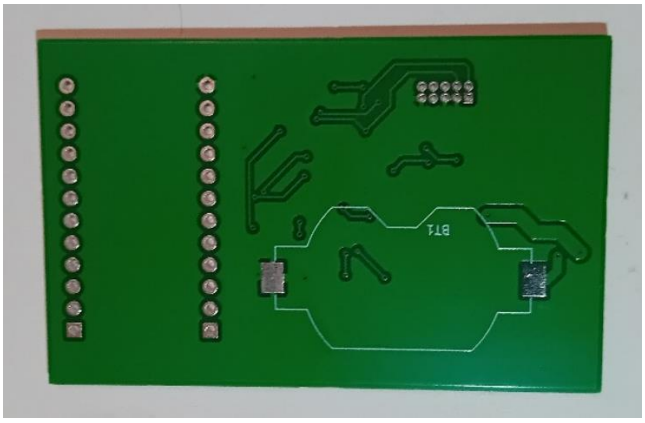


Figura 95. Parte inferior BLE_PCB_V1 sin ensamblar

Una vez recibidas las placas se procede al proceso de ensamblado. El primer paso es aplicar la pasta de soldar, que con ayuda del stencil se convierte en un proceso rápido, aunque delicado.

Se aplicará pasta de soldar en todos los componentes de montaje SMD de la parte superior, por tanto, tanto las tiras de pines como el conector quedan excluidos de este proceso. Solamente puede soldarse una de las capas, por lo que el zócalo de la batería será soldado a mano posteriormente.

Una vez aplicada la pasta se colocan los componentes de forma superficial, y se introduce la PCB en la bandeja de horneado. Se utilizará una máquina de soldado por infrarrojo, las cuales calientan la pasta mediante la radiación infrarroja (IR) haciendo que el componente se adhiera correctamente a la PCB.

Una vez finalizado el proceso y con la placa fría se pasa a la soldadura manual. Con ayuda de hilo de estaño y un soldador de mano se adhieren los componentes faltantes.

A continuación, se presentan las figuras 96 y 97, donde pueden observarse ambas caras de la PCB con los componentes ya soldados:

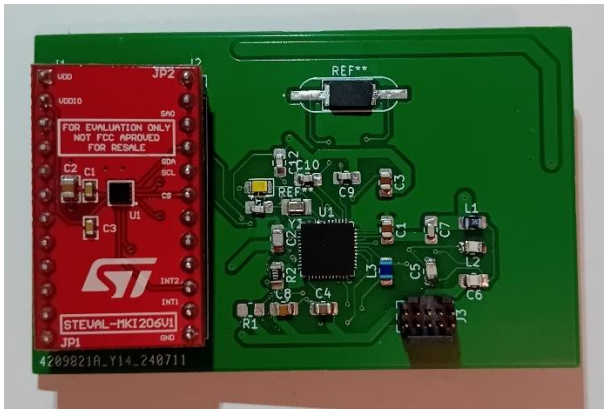


Figura 96. Parte superior BLE_PCB_V1 con componentes soldados

Figura 97. Parte inferior BLE_PCB_V1 con componentes soldados

Como puede observarse en la figura 96, la resistencia R1 no se encuentra soldada. En el siguiente apartado se explica el motivo.

4.3.3.1.4 Detección de errores y solución

Una vez soldados todos los componentes en la PCB debían realizarse pruebas de funcionamiento. En primer lugar, se conectó la pila al zócalo, y con ayuda de un multímetro se comprobó que todos los puntos de alimentación recibían un valor estable y de buen nivel.

En el siguiente paso se conectó el programador y se cargó el firmware1.0, el cual se guardó correctamente en el mapa de memoria del microcontrolador.

A pesar de ello, el dispositivo no realizaba ninguna de las acciones configuradas para el programa. Por tanto, tras repasar los circuitos de los esquemáticos, las configuraciones del hardware y el conexionado, se encontraros los siguientes dos errores:

Error de diseño en la configuración del pin SWCDLK

Tal y como se comentó en el apartado 4.3.3.1.1, existe un error de diseño relacionado con la resistencia R1, la cual se encarga de configurar como pull-down la línea de SWDCLK. En la figura 98 se muestra la configuración y la solución al problema:

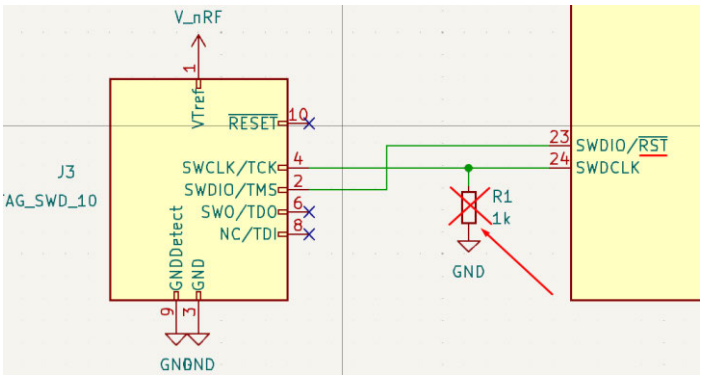


Figura 98. Análisis de error de diseño en pin SWDCLK

El pin de reset del microcontrolador es el 23, el cual está lógicamente asociado con el pin SWDIO. Este pin debe ponerse con un valor lógico bajo, es decir '0', para producir un reset en el microcontrolador. La consexión de R1 pretendía configurar este pin en pull-down, pero debido a un error de conexión, se configuró el pin SWDCLK.

Este error no es la causa principal de que el dispositivo no arranque y ejecute el firmware, pero puede causar problemas de estabilidad en la carga del software, por lo que debe arreglarse.

Una solución provisional consiste en desoldar la resistencia R1, tal y como se ha comentado en el apartado anterior, eliminando la configuración de pull-down al pin y dejándolo en estado "flotante".

Error de componentes de desacoplo

Al igual que en el error anterior, este problema se comentó en el apartado 4.3.3.1.1 cuando se presentaron los componentes de desacoplo de la alimentación. En este caso se produce por la falta de conexionado de un pin. En la figura 99 se muestra la configuración actual del hardware.

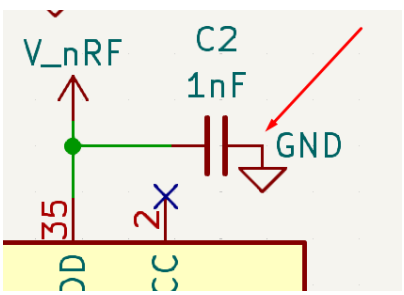


Figura 99. Conexionado del condensador C2 en BLE_PCB_V1

Como puede observarse en la figura, el condensador se encuentra conectado por un pin a alimentación y al pin 35 del microcontrolador, mientras que por el otro se encuentra conectado a GND. Este conexionado es correcto, pero como se puede apreciar en la figura 100, extraída del esquemático del nRF51 DK que se ha usado en el desarrollo del firmware, el pin 2 del condensador debe ir conectado tanto a GND como al pin VSS del microcontrolador.

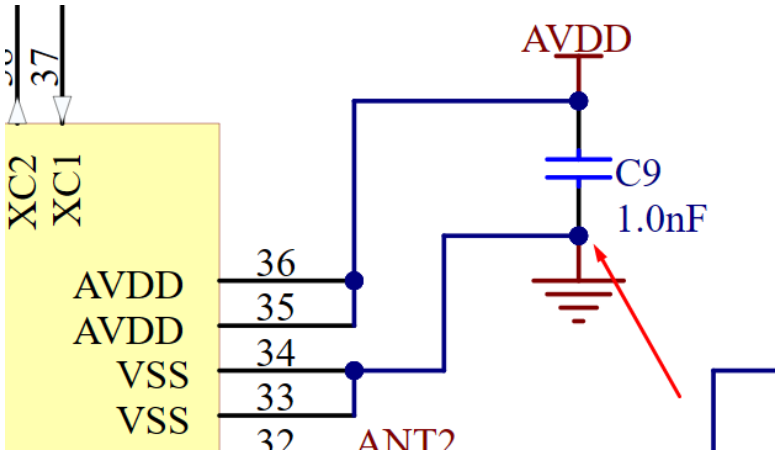


Figura 100. Conexionado correcto del condensador de desacoplo

Este componente actúa como desacoplo de la señal analógica, la cual se usa para partes del microcontrolador como los ADC o la señal de radio frecuencia (RF). Debido al mal conexionado, además de afectar a la calidad de la señal BLE, hace que la alimentación analógica no sea estable, lo cual produce fallos en el arranque y hace que el microcontrolador se reinicie, no pudiendo ejecutar el firmware.

Debido al reducido tamaño de los componentes y los pines del microcontrolador, sería realmente complicado integrar una solución correcta. Por este motivo, se ha decidido no arreglar este error y solucionarlo para la versión de BLE_PCB_V2

4.3.3.2 BLE_PCB_V2

Esta versión es una mejora del hardware inicial descrito en los apartados anteriores. El objetivo de esta versión es solucionar los errores detectados en la versión anterior y generar un hardware de menor tamaño.

4.3.3.2.1 Esquemático

A continuación, se va a mostrar el diseño del esquemático de BLE_PCB_V2, destacando principalmente los puntos que han cambiado respecto a la versión anterior:

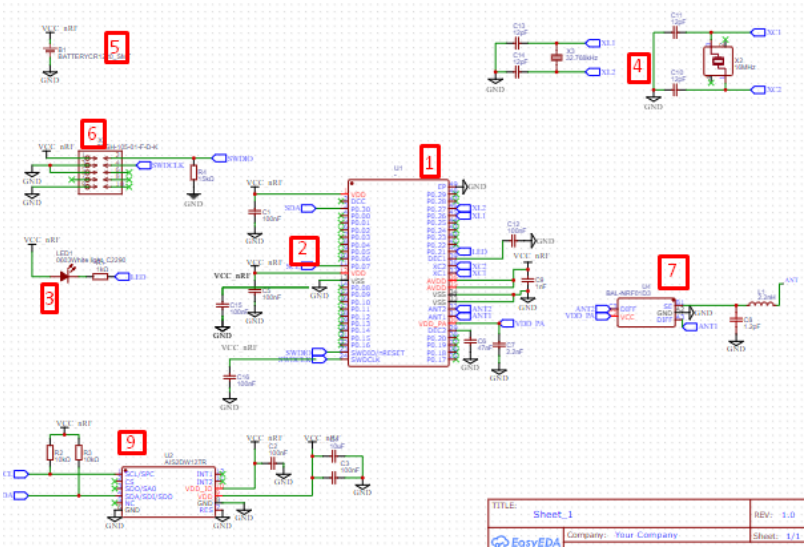


Figura 101. Esquemático completo de BLE_PCB_V2

Como puede observarse en la figura 101, el esquemático puede dividirse en varias partes:

1) Microcontrolador

Como se ha comentado anteriormente, se ha diseñado la segunda versión en EasyEDA para facilitar el ensamblado. Por este mismo motivo algunos de los componentes han tenido que variar ligeramente. Es el caso del microcontrolador. Debido a la disponibilidad de compra, JLCPCB no ensambla el nRF51422, ya que la familia nRF51 está descatalogada, y algunos de los chips son más difíciles de encontrar. El microcontrolador se ha cambiado por el nRF51822, el cual es idéntico al nRF51422, pero con la siguiente diferencia:

- nRF51422 soporta dos protocolos inalámbricos, BLE y ANT/ANT+.
- nRF51822 solo soporta BLE.

Debido a que el protocolo principal del proyecto es el BLE, el cambio no afecta en absoluto, y ni si quiera debe modificarse los firmwares creados, puesto que ambos microcontroladores son compatibles entre sí. En la figura 102 se muestra el símbolo del microcontrolador:

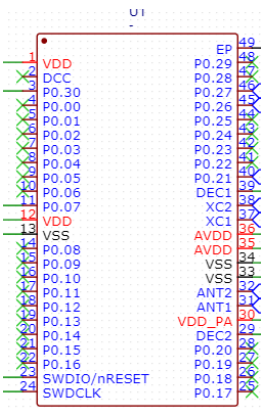


Figura 102. Símbolo de nRF51822

2) Componentes de desacoplo y alimentación

No se han realizado cambios respecto a la versión anterior, salvo la corrección del error comentado en el apartado 4.3.3.1.4.

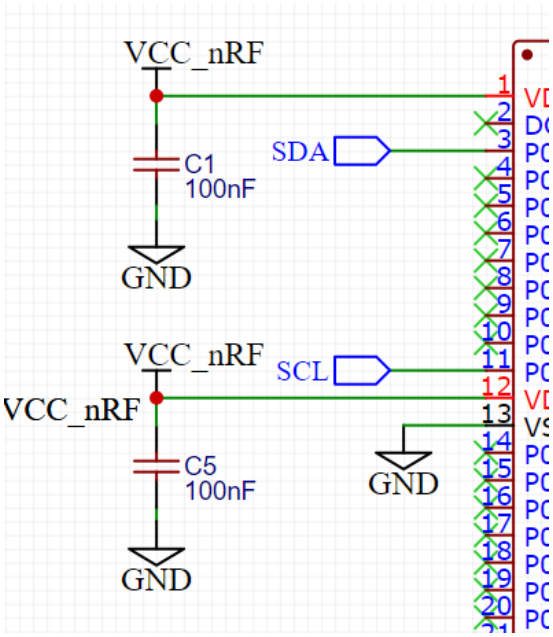


Figura 103. Componentes de desacoplo BLE_PCB_V2

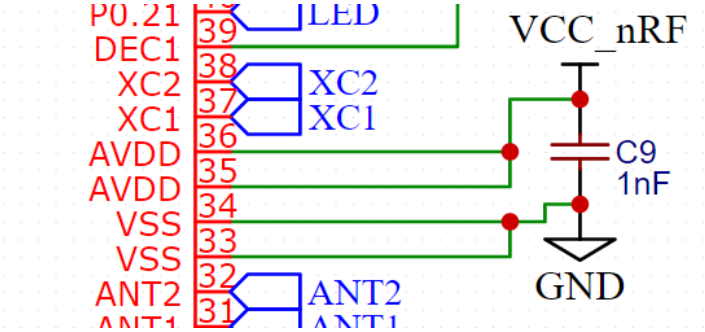


Figura 104. Corrección del error en componentes de desacoplo de la versión 1

3) Led

No se han producido cambios en este componente.

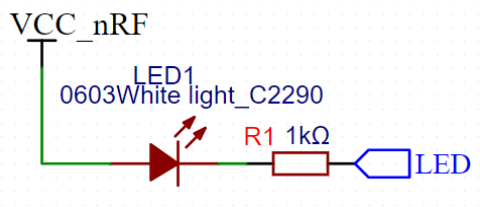


Figura 105. Indicador LED BLE_PCB_V2

4) Relojes de 16MHz y 32kHz

El componente se ha modificado, pero las conexiones y funcionamiento siguen siendo iguales que en la versión anterior:

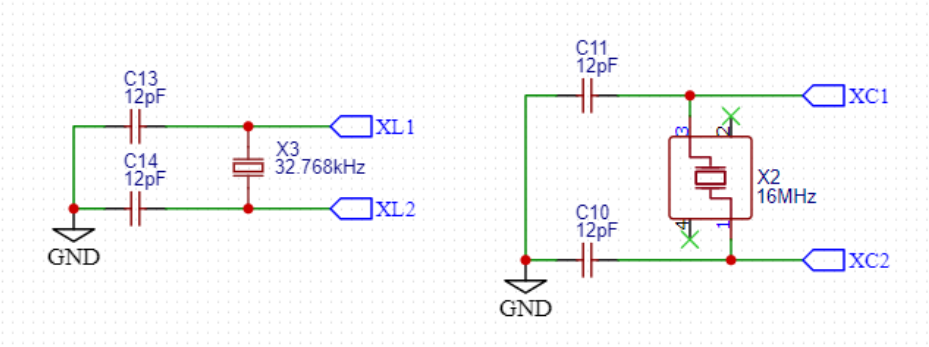


Figura 106. Relojes de 16MHz y 32kHz BLE_PCB_V2

5) Batería

Se ha reducido el tamaño a una pila CR1220

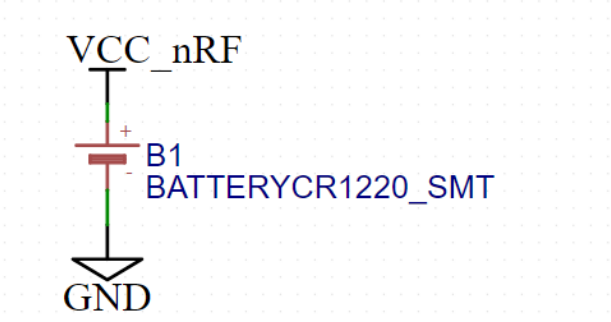


Figura 107. Batería de BLE_PCB_V2

6) Conector

Se mantiene el conector FTSH. Se ha eliminado la configuración de pull-down del SWDCLK corrigiendo el error de la versión anterior. A pesar de ello, **se ha cometido un nuevo error de diseño**, el cual se comentará en el apartado correspondiente.

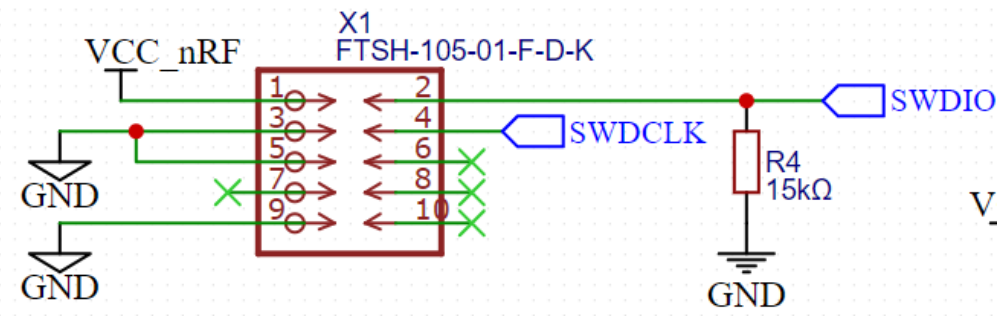


Figura 108. Configuración de conector FTSH

7) Configuración de antena

Con el objetivo de reducir el tamaño de la placa se ha modificado la configuración de la antena mediante componentes pasivos por el uso de un componente que los tiene internamente integrados. Se ha seleccionado el BAL-NRF01D3[63]. Los balun son componentes integrados que permiten convertir una señal diferencial en una señal balanceada de un solo extremo.

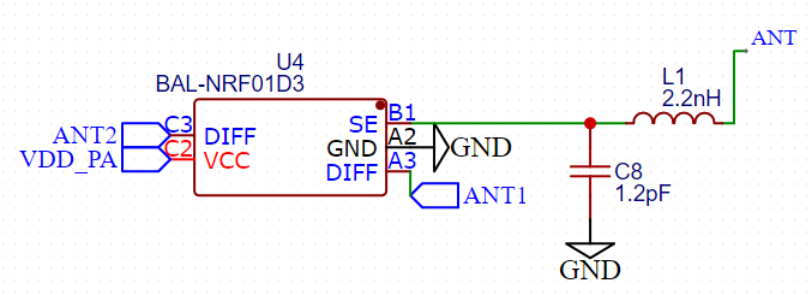


Figura 109. Configuración de antena de BLE_PCB_V2 mediante balun

8) Conexión con acelerómetro

En esta versión eliminamos la placa de desarrollo para integrar de forma individual el AIS2DW12, tal y como se observa en la figura 110, permitiendo así la reducción del tamaño de la PCB. En esta versión se debe añadir los componentes de desacoplo y las líneas pull-up de SCL y SDA.

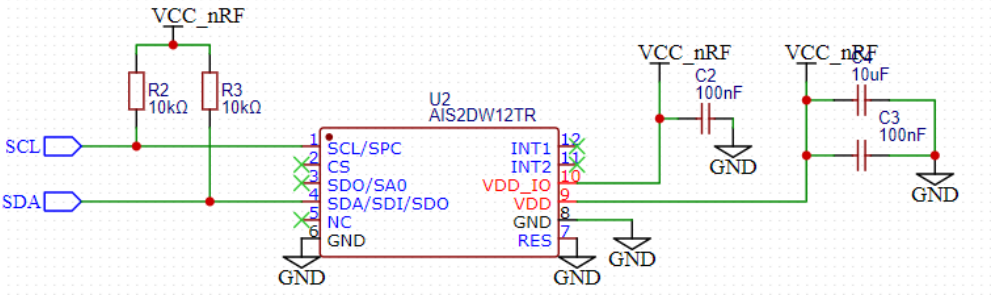


Figura 110. Esquemático y configuración hardware del AIS2DW12

4.3.3.2.2 Desarrollo de PCB

Se ha diseñado una PCB de forma circular de 11.9mm de radio. Puede observarse el resultado final en la figura 111, la cual se desglosará a continuación:

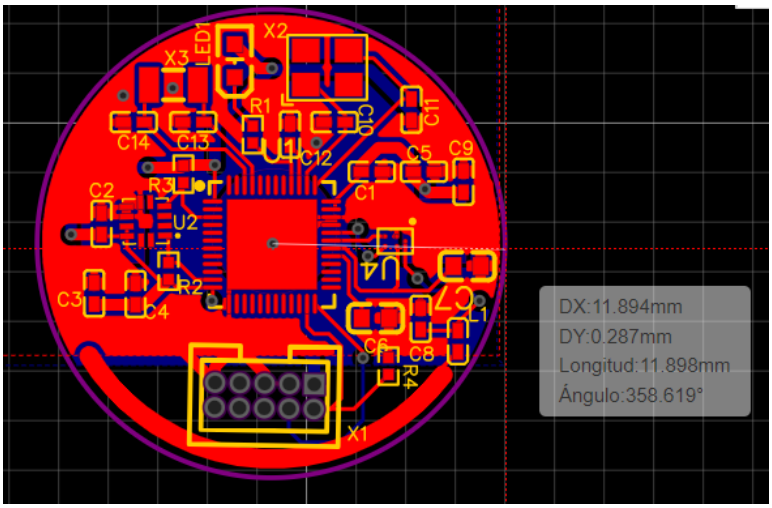


Figura 111. Diseño de PCB de BLE_PCB_V2

Tal y como se ve en la figura, el microcontrolador se ha situado en el centro de la PCB. Esta decisión ayuda a una mejor distribución de los bloques funcionales alrededor del micro, de forma que las conexiones se faciliten.

A continuación, se indicará la posición de los bloques funcionales más representativos y el porqué de esta ubicación. Para facilitar la visión de los componentes se eliminarán las pistas y planos de masa de la imagen:

En primer lugar, y al igual que en su versión anterior, los componentes relacionados con la generación de la señal BLE se han situado a la derecha del microcontrolador, tal y como se aprecia en la figura 112:

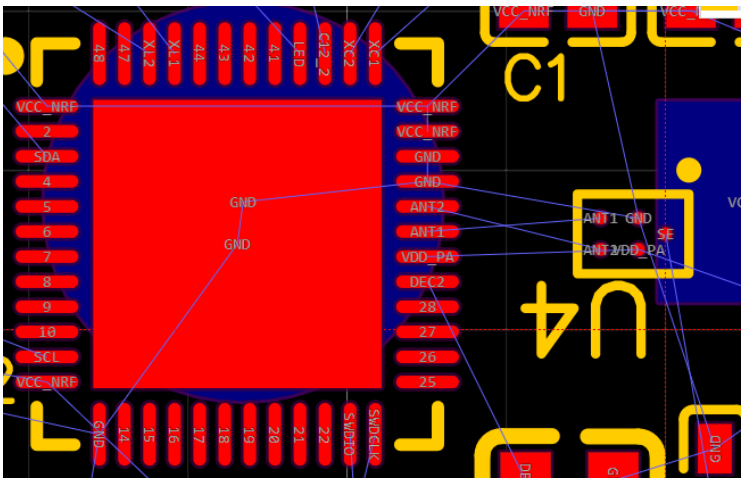


Figura 112. Componentes de configuración de antena y señal BLE_PCB_V2

En la figura 113 se puede ver la ubicación de los osciladores, tanto de 16MHz (X2), como el de 32kHz (X3). Ambos relojes cuentan con los condensadores necesarios para su funcionamiento, los cuales se sitúan alrededor (C10, C12, C13, C14). Se han colocado en esta ubicación ya que los pines de reloj están situados en el lateral superior, marcados en verde, a la derecha los del oscilador de 16MHz, y a la izquierda los de 32kHz.

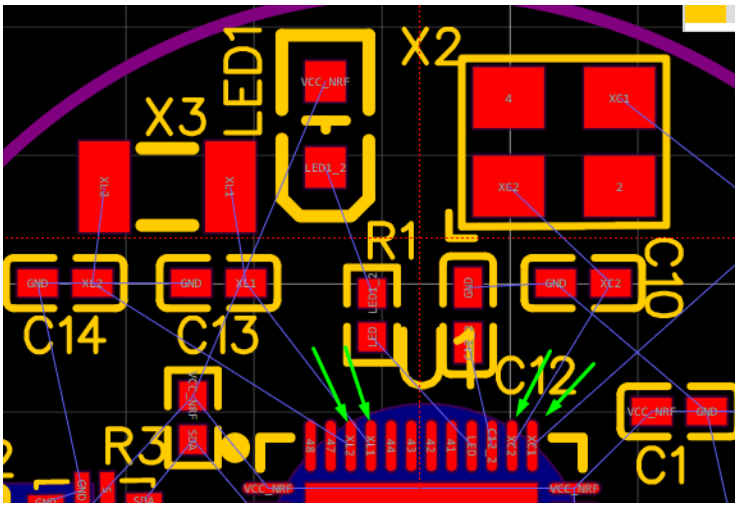


Figura 113. Ubicación de los osciladores de BLE_PCB_V2

En la parte izquierda se han situado el acelerómetro (U2). Se ha designado esta ubicación puesto que los pines de SDA y de SCL del microcontrolador se encuentran en la parte izquierda del mismo, tal y como se observa en la figura 114:

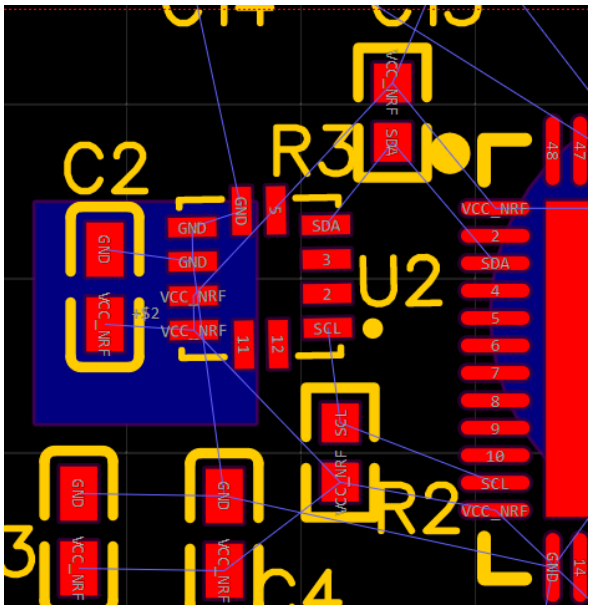


Figura 114. Acelerómetro en BLE_PCB_V2

El conector FTSH se encuentra en la parte inferior (X1), cerca de los pines de SWDIO y SWCLK, ambos marcados con las flechas verdes en la figura 115:

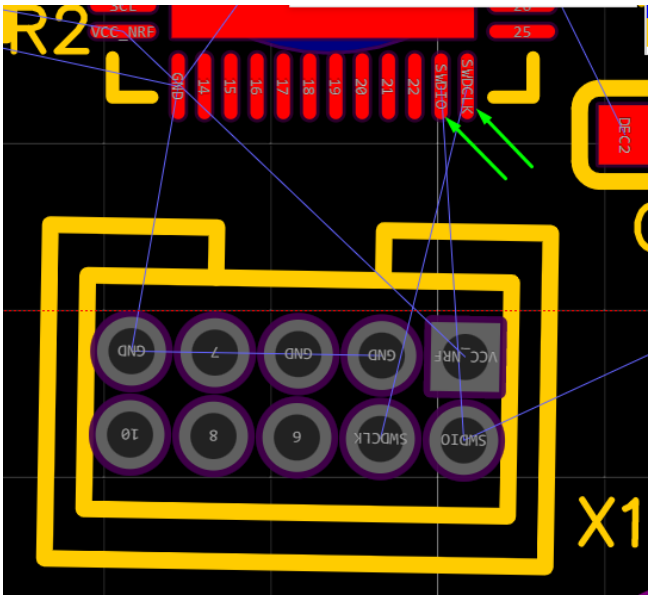


Figura 115. Ubicación de conector FTSH y pines SWD

El resto de los componentes que se han presentado en el esquemático no necesitan una ubicación precisa, aunque de todas formas se han situado de forma que las conexiones se produzcan de la forma más sencilla. Todos los componentes quedan ensamblados en la parte superior, a excepción del zócalo de la batería, el cual se sitúa en la inferior para ahorrar espacio.

Tal y como se ha comentado anteriormente, se han situado dos planos de masa, uno en cada una de las capas, tal y como puede verse en las figuras 116 y 117.

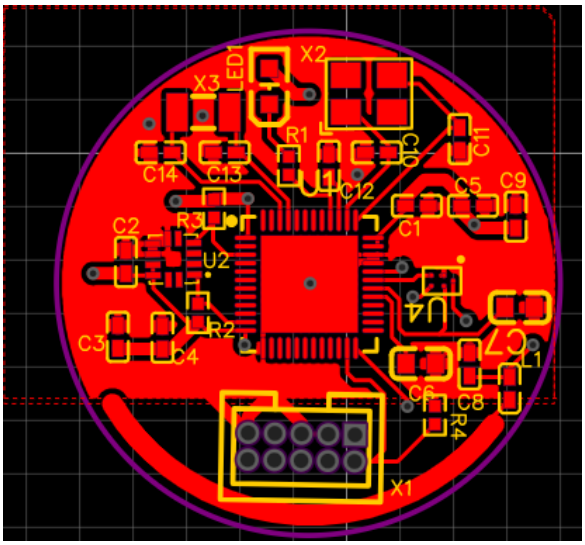


Figura 116. Plano de masa de capa superior (UP)

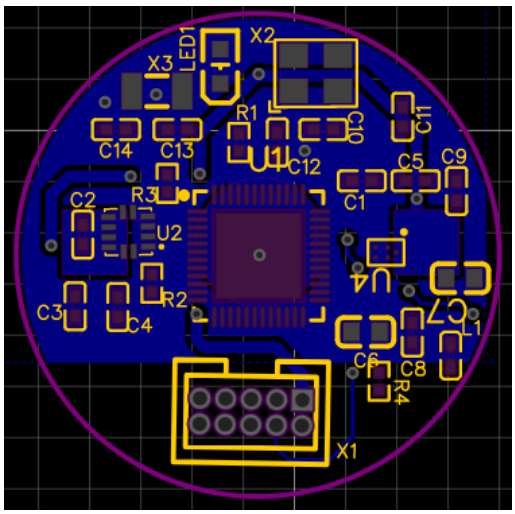


Figura 117. Plano de masa de capa inferior (DOWN)

Es importante destacar que los planos de masa no llegan a la parte inferior. Esto se debe a que es la zona por la que pasa la pista que hace de antena, y esto podría apantallar la señal y reducir tanto su potencia como su alcance.

4.3.3.2.3 Presentación de resultados de fabricación

Una vez que se ha finalizado el diseño de la PCB se obtienen los ficheros gerber, los cuales aportan toda la información necesaria al fabricante para la producción.

A continuación, se presentan los resultados de la segunda versión del hardware, tanto los simulados con la herramienta EasyEDA como los fabricados:

- Visor 2D: EasyEDA incluye un visor en 2D que puede ayudar a observar el resultado del layout final de la PCB sin componentes antes de fabricarla.

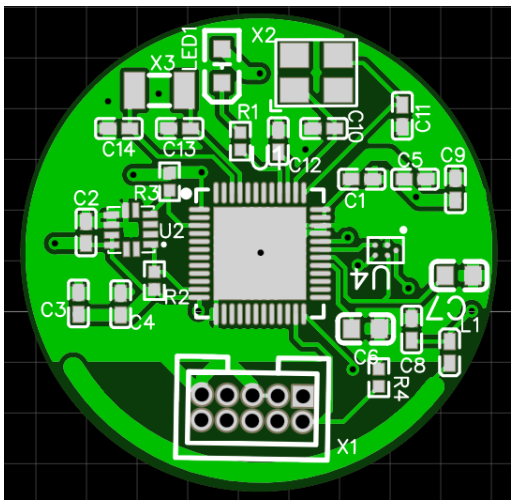


Figura 118. Simulación 2D de la parte superior de la PCB

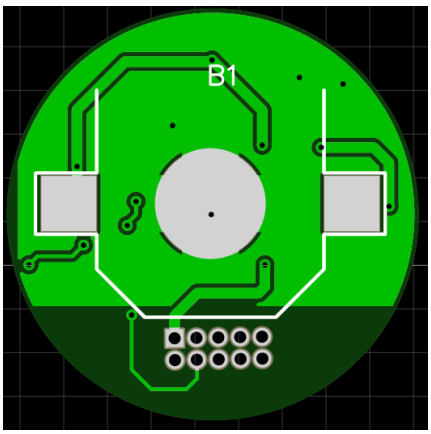


Figura 119. Simulación 2D de la parte inferior de la PCB

Esta herramienta también incluye un visor 3D, el cual permite una visión más realista de la PCB junto al modelo los componentes ensamblados. En las figuras 120 y 121 podemos ver ambas partes:

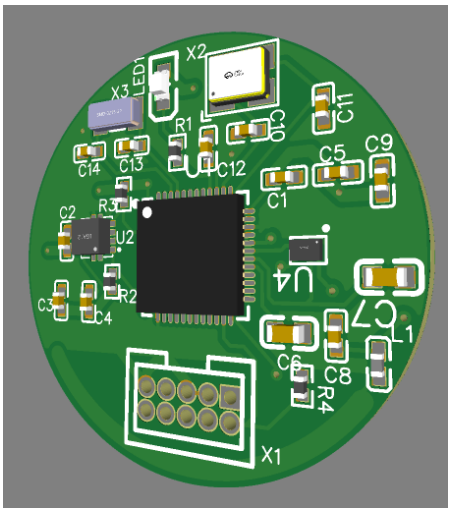


Figura 120. Modelo 3D de la parte superior BLE_PCB_V2

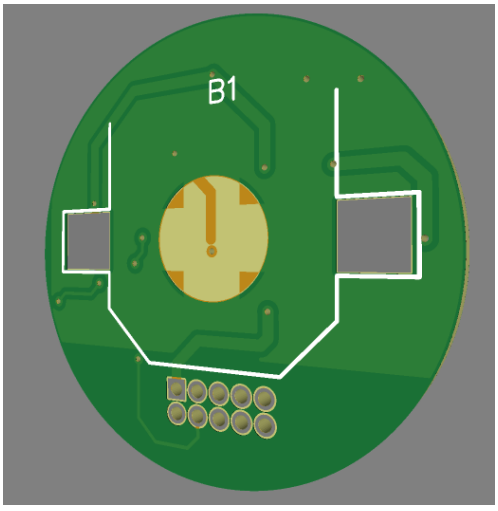


Figura 121. Modelo 3D de la parte inferior BLE_PCB_V2

Una vez recibido el pedido nos encontramos con cinco elementos en su interior. En este caso, al solicitar el ensamblaje no se recibe el stencil, puesto que no será necesario.

En la configuración del pedido se ha solicitado que solo dos de las cinco PCBs recibidas vengan ensambladas.

Al igual que pasaba con la versión anterior, tan solo una de las caras puede ser ensamblada con los componentes SMD de forma automática, por lo que, evidentemente, se ha solicitado el ensamblado de la parte superior (TOP), en la cual se concentran prácticamente todos los componentes.

Ya que solo se ensamblan componentes SMD, una vez recibido el producto ha sido necesario soldar de forma manual el conector del programador, el cual es de tipo THT. También se ha tenido que soldar manualmente el zócalo de la batería en la capa posterior.

En la figura 123 y 124 se muestra una de las PCBs recibidas, sin ensamblar:

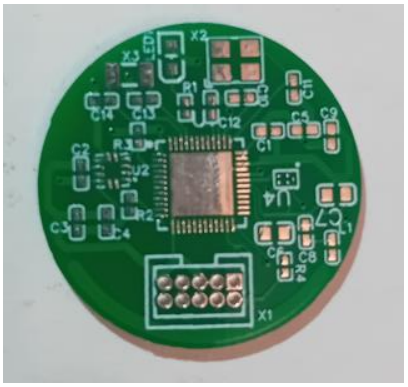


Figura 122. Parte superior BLE_PCB_V2 sin ensamblar

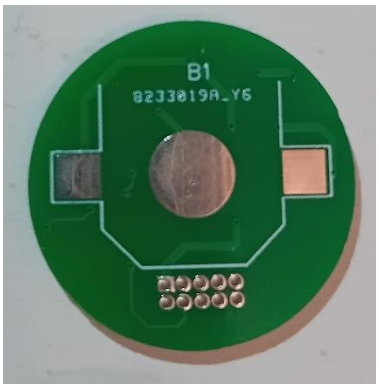


Figura 123. Parte inferior BLE_PCB_V1 sin ensamblar

A continuación, se presentan las figuras 125 y 126, donde pueden observarse ambas caras de la PCB con los componentes ya soldados:

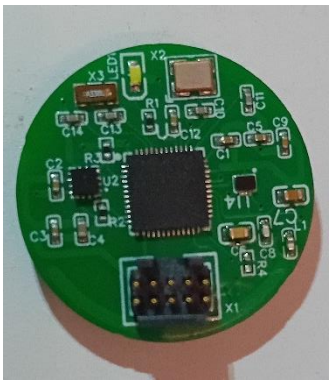


Figura 124. Parte superior BLE_PCB_V1 con componentes soldados

Figura 125. Parte inferior BLE_PCB_V1 con componentes soldados

Como puede observarse en la figura 126, la resistencia R4 no se encuentra soldada. En el siguiente apartado se explica el motivo.

4.3.3.2.4 Detección de errores y solución

Una vez soldados todos los componentes en la PCB debían realizarse pruebas de funcionamiento. En primer lugar, se conectó la pila al zócalo, y con ayuda de un multímetro se comprobó que todos los puntos de alimentación recibían un valor estable y de buen nivel.

En el siguiente paso se conectó el programador y se cargó el firmware2.0, el cual se guardó correctamente en el mapa de memoria del microcontrolador.

A pesar de ello, el dispositivo no realizaba ninguna de las acciones configuradas para el programa. Por tanto, tras repasar los circuitos de los esquemáticos, las configuraciones del hardware y el conexionado, se encontraros los siguientes dos errores:

Error de diseño en la configuración del pin SWCDLK

Tal y como se comentó en el apartado 4.3.3.2.1, existe un error de diseño relacionado con la resistencia R4, la cual se encarga de configurar como pull-down la línea de SWDCLK. En la figura 127 se muestra la configuración y la solución al problema:

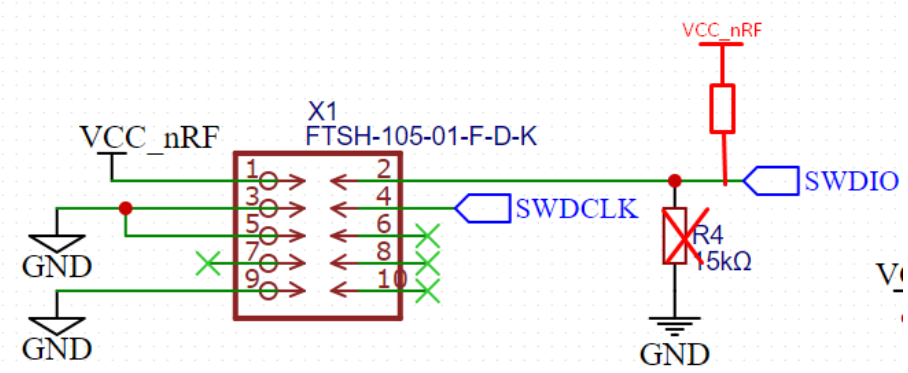


Figura 126. Análisis de error de diseño en pin SWDCLK

El pin de reset del microcontrolador es el 23, el cual está lógicamente asociado con el pin SWDIO. Este pin debe ponerse con un valor lógico bajo, es decir '0', para producir un reset en el microcontrolador. La conexión de R4 pretendía configurar este pin en pull-up, pero debido a un error de conexión, se configuró el pin como pull-down.

Este error causa que el pin de reset se configure de forma permanente con un nivel bajo, lo que provoca que el microcontrolador esté de forma continua reseteándose, sin permitir que se ejecute el firmware.

Una solución provisional consiste en desoldar la resistencia R4, tal y como se ha comentado en el apartado anterior, eliminando la configuración de pull-down al pin y dejándolo en estado "flotante".

Como solución oficial debería diseñarse una tercera versión que fijase la configuración como pull-up, ya que, si se deja el pin en flotante, de forma ocasional pueden surgir resets aleatorios en el dispositivo.

5. Resultados

En este capítulo se presentarán los resultados obtenidos de las diferentes pruebas realizadas a lo largo del ciclo de vida del proyecto, las cuales han permitido verificar el correcto funcionamiento de cada una de las fases. Se presentarán las pruebas realizadas divididas en tres bloques, ya que cada uno de los desarrollos de las distintas fases se probó de forma individual.

5.1 Pruebas de fase 1: Desarrollo del firmware

Una vez finalizado el desarrollo del firmware1.0 se realizaros varias pruebas con la tarjeta de desarrollo nRF51 DK.

Para comprobar el correcto funcionamiento del software se usaron principalmente dos herramientas, además de la propia depuración del código:

- **Terminal de VsCode:** En ella se ha utilizado diversas funciones para imprimir por pantalla información relevante que ayude a verificar el correcto funcionamiento del firmware. Principalmente se pueden observar dos mensajes:

    - Mensaje en reposo: Como se ha explicado en los apartados anteriores, en la primera versión del firmware, el acelerómetro toma una medida cada segundo. Los valores obtenidos son mostrados por pantalla, como se puede observar en la figura 127:

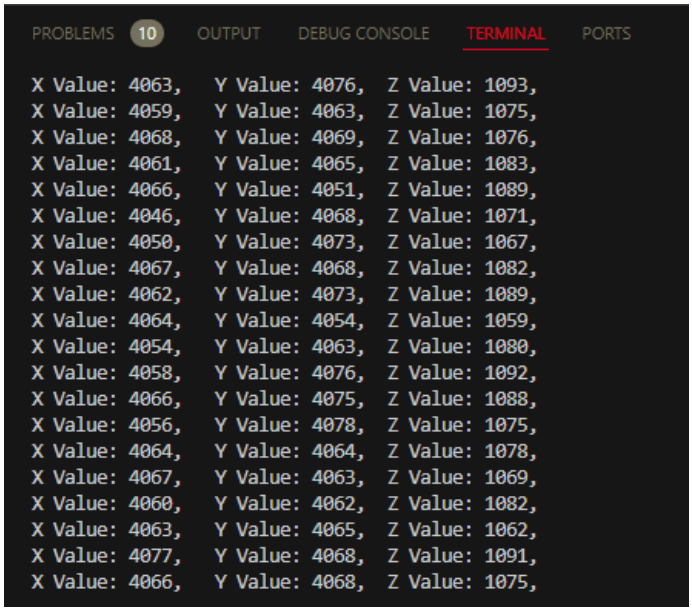


Figura 127. Medidas de acelerómetro en reposo

- Mensaje de activación: Una vez que el acelerómetro detecta un cambio significativo en los valores medidos, se activa la señal BLE desde el microcontrolador. En la terminal se muestra información relevante sobre la señal, la cual puede verse en la figura 128, además de dos avisos resaltados en color amarillo, que indican el momento concreto de activación y desactivación de la señal BLE:

```
[00:09:23.735,656] <inf> bt_hci_core: HW Platform: Nordic Semiconductor (0x0002)
[00:09:23.735,778] <inf> bt_hci_core: HW Variant: nRF51x (0x0001)
[00:09:23.735,931] <inf> bt_hci_core: Firmware: Standard Bluetooth controller (0x00) Version 3.4 Build 99
[00:09:23.737,182] <inf> bt_hci_core: Identity: FE:BC:15:F7:DC:13 (random)
[00:09:23.737,365] <inf> bt_hci_core: HCI: version 5.4 (0x0d) revision 0x0000, manufacturer 0x05f1
[00:09:23.737,548] <inf> bt_hci_core: LMP: version 5.4 (0x0d) subver 0xffff
Bluetooth inicializado
Beacon started, Direccion MAC: FE:BC:15:F7:DC:13 (random)
[00:09:23.741,302] <wrn> BEACON: [BT] CAMBIO: BT ACTIVADO

[00:09:27.742,279] <wrn> BEACON: [BT] CAMBIO: BT DESACTIVADO
```

Figura 128. Mensaje de activación de señal BLE

- **Aplicación de lectura de nRF Connect:** Esta aplicación, que puede obtenerse de forma gratuita desde el app store, permite hacer una lectura de todos los dispositivos bluetooth que están emitiendo en cada momento. En las figuras 129 y 130 se observa la diferencia entre el momento de reposo y el de emisión del firmware:

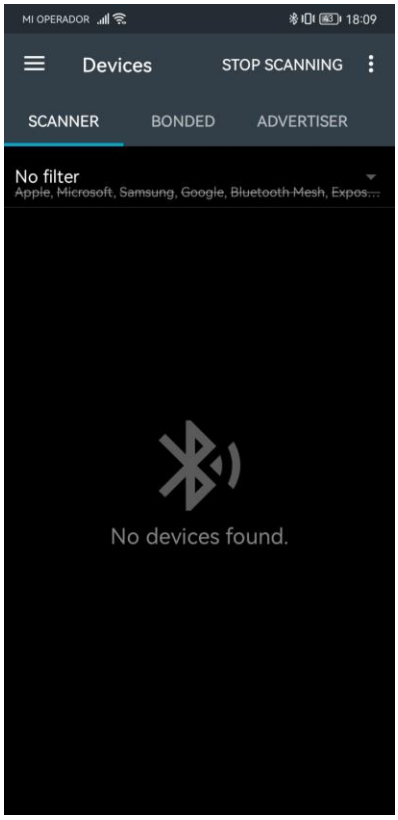


Figura 129. Aplicación sin detección

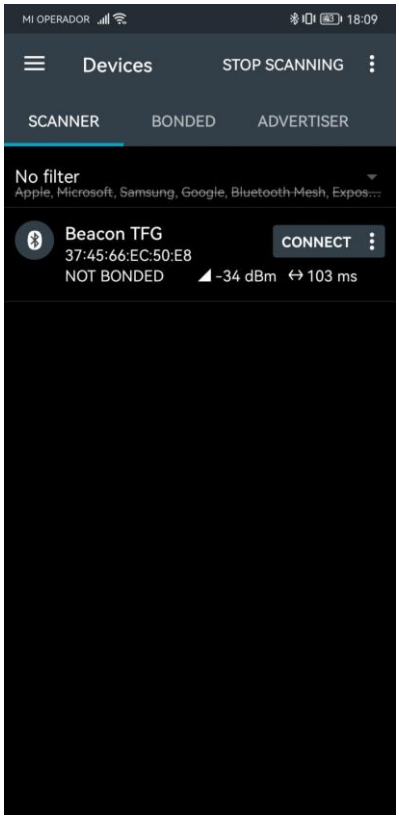


Figura 130. Aplicación con detección

Para el firmware2.0 se ha seguido el mismo proceso. Como se ha explicado en apartados anteriores, en esta segunda versión el microcontrolador permanece en estado de bajo consumo hasta que recibe una interrupción generada por el acelerómetro. Debido a esto, no se ha establecido mensajes para el reposo, tan solo una notificación de recepción de la interrupción, marcada en rojo en la siguiente figura, la cual muestra el mensajes de activación impreso por la terminal del IDE.

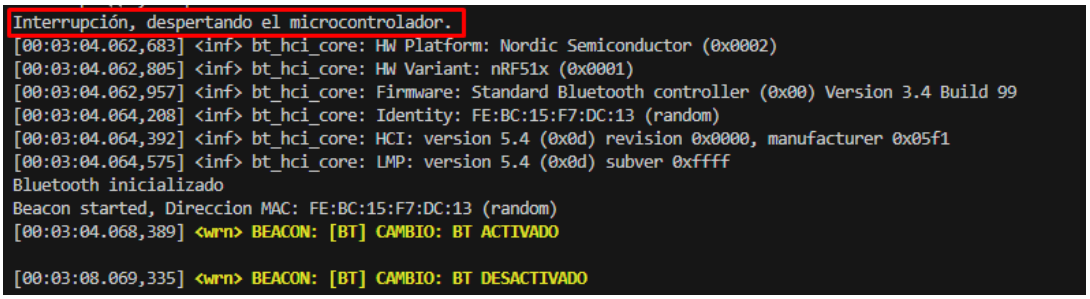


Figura 131. Mensaje de activación de BLE en firmware 2.0

5.2 Pruebas de fase 2: Desarrollo de app BLEScanner

En esta fase se ha realizado dos tipos de prueba. En primer lugar, pequeños test aislados para comprobar las funcionalidades de la aplicación. En segundo lugar, una prueba de concepto, la cual solo se realizó con la versión definitiva de la aplicación, para ver el comportamiento en un contexto real. Para evitar la dependencia con otras fases se realizaron las pruebas con los beacons comerciales de MokoSmart.

En los test aislados se probaron cada una de las funcionalidades de ambas versiones. A continuación, se muestra una tabla para cada una de las versiones con las pruebas realizadas:

Tabla 18. Test realizados a ambas versiones

BLEScannerV1			
Test	Descripción	Resultado	Observaciones
Detección de señal	Se comprueba si la aplicación es capaz de leer dispositivos.	**Test superado**	No Aplica
Señales mostradas en pantalla principal	Se comprueba si las señales se muestran correctamente junto a la información	**Test deficiente**	A pesar de que se muestran siempre, algunas veces lo hacen con retraso
Señales guardadas en historial	Se comprueba si las señales detectadas se guardan en el historial	**Test superado**	Se debe tener en cuenta que el cerrar la aplicación reiniciará el historial.
Guardado en fichero txt	Se comprueba que el guardado se hace de forma correcta	**Test deficiente**	Se debe dar permiso cada vez que se inicia la aplicación

BLEScannerV2			
Test	Descripción	Resultado	Observaciones
Detección de señal	Se comprueba si la aplicación es capaz de leer dispositivos.	**Test superado**	No Aplica
Señales mostradas en pantalla principal	Se comprueba si las señales se muestran correctamente junto a la información	**Test superado**	Se ha corregido el retraso en la escritura reduciendo el tiempo de actualización del listview.
Señales guardadas en historial	Se comprueba si las señales detectadas se guardan en el historial	**Test superado**	Debido a la ejecución en segundo plano, el historial se conserva al cerrar la app siempre y cuando el servicio siga corriendo.
Guardado en fichero txt	Se comprueba que el guardado se hace de forma correcta	**Test superado**	Se ha corregido para solicitar permisos solo la primera vez que se inicia la app.
Detección de señal con aplicación minimizada	Se comprueba que la aplicación sigue leyendo las señales a pesar de que se encuentre minimizada	**Test superado**	Este experimento se realiza con la pantalla encendida
Detección de señal con pantalla apagada	Se comprueba que la aplicación sigue leyendo las señales a pesar de que la pantalla se encuentre apagada.	**Test superado**	Se ha probado tanto con la app abierta como minimizada
Guardado automático de señales en txt	Se comprueba que se guarden las señales detectadas de forma automática, tanto cuando se detecta una de interés como cuando son las 23.59	**Test superado**	No aplica
Limpieza del historial	Se comprueba que se limpia el historial a las 23.59	**Test superado**	No aplica

5.2.1 Prueba de concepto

Una vez comprobado el correcto funcionamiento de la segunda versión de la aplicación decidió realizar una prueba de concepto. Se midieron un total de seis movimientos separados en tres grupos de dos movimientos cada una. Se probaron cuatro umbrales de movimiento en cada grupo. Esta prueba de concepto tenía principalmente dos objetivos:

- Comprobar el comportamiento en una situación real
- Comprobar el umbral de movimiento más adecuado para configurar los beacons.

5.2.1.1 Primer grupo: Nevera y Microondas

Se colocaron dos beacons, uno es la nevera y otro en el microondas simulando una cocina, de forma que se pudiese monitorizar actividades como cocinar o comer.

Como puede obervarse en las figuras 132 y 133, los beacons se han colocado en las puertas, cerca del agarre para detectar de forma correcta las aperturas del electrodomestico. Es importante colocarlos alejados de elementos que puedan producir falsos movimeintos, como el motor de la nevera o la parte trasera del microondas.

Figura 132. Beacon integrado en microondas

Figura 133. Beacon integrado en nevera

Las medidas fueron las siguientes:

Tabla 19. Resultados de las medidas del primer bloque

	Umbrales de movimiento			
Dispositivo	0.1 g	0.3 g	0.7 g	1 g
Nevera	Detectado*	Detectado	Detectado	No detectado
Microondas	Detectado	Detectado	No detectado	No detectado

En este grupo de medidas se detectaron algunos falsos positivos en la nevera, que podrían darse a causa de las vibraciones del motor. En el microondas no se detectaron falsos positivos, aunque no se puso en marcha para realizar las medidas. Todos los movimientos se hicieron con un esfuerzo medio, idéntico al que se realizaría en una tarea cotidiana.

5.2.1.2 Segundo grupo: Cajón y silla

Se colocaron dos beacons, uno es la silla y otro en el cajón simulando una oficina o escritorio, de forma que se pudiese monitorizar escribir o sentarse a trabajar. Los beacons se han colocado de esta forma para detectar basicamente dos movimeintos:

- **Moviemiento de la silla:** El primer y último movimeintos detectados significan el inicio y el fin de una acción realizada en el escritorio. Todos los moviemientos intermedios son confirmaciones de que la actividad sigue en curso, por lo que puede estudiarse su duración con mayor precisión.
- **Apertura/Cierre del cajón:** Teniendo en cuenta los elementos guardados en el cajón puede identificarse una actividad más concreta. Por ejemplo, si se guardase un ordenador portatil, la apertura y posteriormente el cierrre del cajón podrían identificar el tiempo de uso del ordenador.

Como puede obervarse en las figuras 134 y 135, los beacons se han colocado en el respaldo de la silla y en la parte exterior del cajón. Es importante tener en cuenta colocarlos alejados de zonas en las que puedan recibir golpes, ya que esto podría producir falsos moviemientos.

Figura 134. Beacon integrado en cajón

Figura 135. Beacon integrado en silla

Las medidas fueron las siguientes:

Tabla 20. Resultados de las medidas del primer bloque

	Umbrales de movimiento			
Dispositivo	0.1 g	0.3 g	0.7 g	1 g
Cajón	Detectado	Detectado	No Detectado	No detectado
Silla	Detectado	Detectado	No detectado	No detectado

En este grupo de medidas no se observaron falsos positivos en ninguno de los casos.

Un resultado destacable es que en este grupo el umbral de 0.7g no ha detectado movimiento. Esto podría ser porque tanto la silla como el cajón cuentan con ruedas que, al contrario que la apertura de la nevera o microondas, suavizan el movimiento. Todos los movimientos se hicieron con un esfuerzo medio, idéntico al que se realizaría en una tarea cotidiana.

5.2.1.1 Tercer grupo: mando y mesa

Se colocaron dos beacons, uno es el mando de la televisión y otro en el apoyado en la mesa tal y como se aprecia en la figura 136, simulando un lugar de descanso. Todos los movimientos se hicieron con un esfuerzo medio, idéntico al que se realizaría en una tarea cotidiana.

Figura 136. Beacons integrados en mesa y mando de televisión.

Las medidas fueron las siguientes:

Tabla 21. Resultados de las medidas del primer bloque

	Umbrales de movimiento			
Dispositivo	0.1 g	0.3 g	0.7 g	1 g
Mesa	Detectado	No detectado	No detectado	No detectado
Mando	Detectado	Detectado	Detectado	Detectado

En este grupo de medidas se observaron dos comportamientos importantes. En la mesa la detección fue muy complicada, por lo que se obtuvo como conclusión que los beacons detectan de forma sencilla variaciones en la aceleración, es decir, movimientos, pero tienen mayor dificultad a la hora de detectar pequeñas vibraciones producidas por golpes sobre superficies.

En el caso del mando se observa como los movimientos producidos al agarrar y desplazar un objeto son los más fáciles de detectar, llegando a obtener algunos resultados en el umbral más alto, aunque no todos los intentos fueron detectados.

Con estas medidas se puede sacar en conclusión que el mejor umbral para configurar los beacons es en torno a 0.3g, de forma que facilite la detección, pero no sea extremadamente sensible a pequeñas vibraciones.

5.3 Pruebas de fase 3: Desarrollo de hardware

En estas pruebas se ha verificado el correcto funcionamiento de los dispositivos fabricados. Estas pruebas resultaban imposibles de aislar, ya que es necesario el uso de ambas versiones de los firmware desarrollados. Una vez obtenido el hardware completamente ensamblado debía probarse. Se siguieron los pasos reflejados en la siguiente tabla:

Tabla 22. Resultados de pruebas hardware en BLE_PCB_V1

Prueba	Descripción	Resultado
Carga de firmware	Se prueba la carga del firmware desarrollado en el dispositivo	Prueba satisfactoria
Funcionamiento completo	Se comprueba si el dispositivo cumple con su funcionamiento	Prueba no satisfactoria. El dispositivo no reacciona frente a ningún estímulo
Medidas de alimentación	Se comprueba que el circuito distribuya correctamente la alimentación en todos los puntos	Prueba satisfactoria.
Solución error: pin SWCLK configurado en pulldown	Se elimina la resistencia de configuración pulldown del pin SWCLK	Prueba no satisfactoria. El dispositivo sigue sin reaccionar
Solución error: falta de condensador de alimentación analógica	Entre los pines AVDD y VSDD debe haber un condensador, el cual no se diseñó correctamente	Prueba no satisfactoria. Debido al reducido tamaño se decide solucionarlo en la segunda versión.

Una vez obtenido el hardware de la segunda versión, que implementaría mejoras para solucionar los errores de la primera versión, se realizaron las siguientes pruebas:

Tabla 23. Resultados de pruebas hardware en BLE_PCB_V2

Prueba	Descripción	Resultado
Carga de firmware	Se prueba la carga del firmware desarrollado en el dispositivo	Prueba satisfactoria.
Funcionamiento completo	Se comprueba si el dispositivo cumple con su funcionamiento	Prueba no satisfactoria. El dispositivo no reacciona frente a ningún estímulo.
Medidas de alimentación	Se comprueba que el circuito distribuya correctamente la alimentación en todos los puntos	Prueba satisfactoria.
Solución error: pin SWIO configurado en pulldown	Se elimina la resistencia de configuración pulldown del pin SWIO	Prueba satisfactoria. El dispositivo funciona y cumple todas las funciones definidas.

6. Presupuesto

En este apartado se desglosará el presupuesto del proyecto. A continuación, se presentan distintas tablas que recogen los costes del proyecto por categorías:

Tabla 24. Presupuesto de dispositivos electrónicos

Título	Unidades	Precio(€)/unidad	Total(€)	Observaciones
Beacon M1	3	17.5	52.5	N/A
Beacon M4Lite	3	17.5	52.5	N/A
nRF51DK	1	51.97	51.97	N/A
STEVAL-MKI206V1	1	19.01	19.01	Ya estaba adquirido
Subtotal			***175.98***	

Tabla 25. Presupuesto desarrollo hardware

Título	Unidades	Precio(€)/unidad	Total(€)	Observaciones
Pedido PCB primera versión	5	4	20	Incluye el transporte
Componentes PCB primera versión	N/A	N/A	34.24	Componentes comprados en Mouser Electronics
Pedido PCB segunda versión + ensamblado + componentes	10	47.64	95.28	Incluye el transporte. Se realizaron dos pedidos de 5 PCBs cada uno, debido a problemas en aduanas
Subtotal			***149.52***	

Tabla 26. Presupuesto de mano de obra, licencias y equipamiento

Título	Unidades	Precio(€)/unidad	Total(€)	Observaciones
Mano de obra	330 (horas)	14.48	4778.4	Sueldo neto ingeniero junior
Licencias	N/A	0	0	Todas las herramientas usadas son gratuitas
Equipo de desarrollo (PC)	1	756.97	756.97	Ya estaba adquirido
Subtotal			***5535.37***	

Realizando la suma de todas las categorías se obtiene un **presupuesto total de 5860.87€**

7. Impacto del proyecto

En este apartado se analizará el impacto que tiene este proyecto en la sociedad, especialmente enfocándose en los aspectos éticos, ambientales y económicos, destacando también la contribución a los objetivos de desarrollo sostenible (ODS).

Este proyecto tiene como principal objetivo desarrollar un dispositivo capaz de detectar movimiento de objetos cotidianos para ayudar en el monitoreo de pacientes de Parkinson. Desde el **punto de vista de la sociedad**, esto tiene un gran impacto, puesto que la implementación de este proyecto en el día a día de los pacientes puede significar un aumento considerable en la calidad de vida. Además, como beneficio colateral, el uso de esta herramienta reduciría la carga de trabajo en el personal sanitario, que podría destinar mayor parte de su jornada al cuidado del resto de enfermos.

A **nivel ético**, es fundamental asegurar que los datos recogidos por el dispositivo se manejen con estrictas medidas de confidencialidad y cumplan con normativas como el RGPD[63] y la Ley Orgánica de Protección de Datos . Esto garantizará que la privacidad y los derechos de los pacientes estén siempre protegidos

El **impacto ambiental** del proyecto se relaciona principalmente con el consumo de energía y los materiales utilizados para el desarrollo del dispositivo. Se debe considerar el ciclo de vida del dispositivo, desde la obtención de materias primas hasta su eventual reciclaje o desecho, para minimizar la huella de carbono y otros posibles efectos ambientales negativos. El uso eficiente de la energía y la implementación de medidas para prolongar la vida útil del dispositivo serán aspectos clave para mitigar su impacto ambiental. Además, se recomienda realizar un análisis de los materiales y recursos utilizados, priorizando aquellos que sean reciclables o tengan un bajo impacto ambiental.

El desarrollo de este proyecto tiene un **impacto económico positivo**, tanto en términos de reducción de costos en el ámbito sanitario como en la posibilidad de crear nuevas oportunidades de negocio en el sector tecnológico. La comercialización de un dispositivo que mejore la vida de pacientes con enfermedades crónicas como el Parkinson puede generar un nicho de mercado importante, lo que conlleva beneficios financieros a largo plazo.

Este proyecto contribuye directamente a varios **Objetivos de Desarrollo Sostenible (ODS),** especialmente el ODS 3 (Salud y Bienestar) al mejorar el monitoreo de enfermedades crónicas como el Parkinson. También se alinea con el ODS 9 (Industria, Innovación e Infraestructura), ya que promueve el desarrollo de tecnologías avanzadas para el bienestar de la sociedad.

En resumen, el proyecto tiene un impacto positivo en varios frentes, siempre y cuando se tomen las medidas adecuadas para garantizar la sostenibilidad y el respeto por los derechos de los usuarios y el medio ambiente.

8. Conclusiones

El desarrollo de este proyecto ha logrado cumplir con los objetivos principales establecidos, diseñando un dispositivo eficiente para el monitoreo de pacientes con Parkinson, basado en la detección de movimientos. A lo largo del proceso, se han implementado tecnologías avanzadas que no solo permiten una detección precisa, sino que también facilitan el análisis de los datos en tiempo real, lo que se traduce en una herramienta de gran utilidad tanto para los pacientes como para los profesionales de la salud.

El impacto social y médico es notable, ya que el dispositivo promete mejorar la calidad de vida de los pacientes, brindándoles mayor independencia y posibilitando una atención más personalizada y eficiente. Además, se han tomado en cuenta consideraciones ambientales durante el desarrollo, empleando materiales que reducen el impacto ecológico. El proyecto también ha demostrado ser económicamente viable, abriendo la puerta a nuevas oportunidades en el mercado tecnológico de la salud.

8.1 Futuros Trabajos

- **Optimización del dispositivo para la integración en sistemas de atención médica**: Aunque el dispositivo ha demostrado ser eficaz en el monitoreo de movimientos, una futura línea de trabajo sería su integración con plataformas de salud más amplias, permitiendo la interoperabilidad con historiales médicos electrónicos y otros sistemas de monitoreo de salud a distancia.
- **Desarrollo de algoritmos más avanzados**: La implementación de algoritmos basados en inteligencia artificial podría mejorar la precisión y personalización del análisis de movimientos. Esto permitiría identificar patrones más complejos en los datos recogidos, que podrían ser utilizados para predecir crisis o eventos importantes en la progresión de la enfermedad.
- **Ampliación del uso del dispositivo a otras enfermedades neurodegenerativas**: Aunque el dispositivo se ha centrado en pacientes con Parkinson, la tecnología podría adaptarse para el monitoreo de otras enfermedades neurodegenerativas como el Alzheimer o la esclerosis múltiple. Un futuro trabajo debería investigar estas posibilidades.
- **Ampliación del estudio con un mayor número de usuarios**: Para obtener resultados más robustos, sería interesante realizar un estudio más amplio que incluya una mayor cantidad de pacientes y condiciones de prueba variadas, lo que podría proporcionar datos más sólidos sobre la efectividad del dispositivo en distintos entornos.

En conclusión, el proyecto ofrece una solución innovadora y práctica para el monitoreo de movimientos en pacientes con Parkinson, y deja la puerta abierta a importantes mejoras y desarrollos futuros que podrían ampliar su impacto tanto en el ámbito médico como en el tecnológico.

9. Referencias

[1] "Las nuevas tecnologías facilitan la relación con... | Roche España." Accessed: Aug. 01, 2024. [Online]. Available: https://www.roche.es/actualidad/notas-prensa/2020/julio/nuevas-tecnologias-facilitan-relacion-paciente

[2] "OMS: Parkinson." Accessed: Feb. 29, 2024. [Online]. Available: https://www.who.int/es/news-room/fact-sheets/detail/parkinson-disease

[3] "Ministerio de Sanidad - Prensa e comunicación - Noticias." Accessed: Aug. 01, 2024. [Online]. Available: https://www.sanidad.gob.es/gl/gabinete/notasPrensa.do?id=6072

[4] "Evaluación de la Estrategia en enfermedades Neurodegenerativas del Sistema Nacional de Salud", Accessed: Aug. 01, 2024. [Online]. Available: https://www.sanidad.gob.es/areas/calidadAsistencial/estrategias/enfermedadesNeurodegenerativas/docs/Informe_resultados._ACCESIBLE.pdf

[5] Naresh Gupta, "Inside Bluetooth Low Energy," First Edition., Artech House, 2013, p. 3.

[6] Naresh Gupta, "Inside Bluetooth Low Energy," First Edition., Artech House, 2013, p. 6.

[7] K. V. S. S. S. S. Sairam, N. Gunasekaran, and S. Rama Reddy, "Bluetooth in wireless communication," *IEEE Communications Magazine*, vol. 40, no. 6, pp. 90–96, Jun. 2002, doi: 10.1109/MCOM.2002.1007414.

[8] C. Bisdikian, "An overview of the Bluetooth wireless technology," *IEEE Communications Magazine*, vol. 39, no. 12, pp. 86–94, Dec. 2001, doi: 10.1109/35.968817.

[9] "Bluetooth." Accessed: Aug. 16, 2024. [Online]. Available: https://www.etsist.upm.es/estaticos/ingeniatic/index.php/tecnologias/item/385-bluetooth.html

[10] "Harald Blåtand - EcuRed." Accessed: Aug. 16, 2024. [Online]. Available: https://www.ecured.cu/Harald_Bl%C3%A5tand

[11] "El enigma del símbolo de Bluetooth." Accessed: Aug. 16, 2024. [Online]. Available: https://brandemia.org/origen-simbolo-de-bluetooth

[12] "Bluetooth: de dónde viene y qué significa su controversial logo - El Cronista." Accessed: Aug. 16, 2024. [Online]. Available: https://www.cronista.com/infotechnology/online/Bluetooth-de-donde-viene-y-que-significa-su-controversial-logo-20161013-0002.html

[13] "Core Specification 4.0 | Bluetooth® Technology Website." Accessed: Aug. 17, 2024. [Online]. Available: https://www.bluetooth.com/specifications/specs/core-specification-4-0/

[14] "3 Key Factors That Determine the Range of Bluetooth | Bluetooth® Technology Website." Accessed: Aug. 22, 2024. [Online]. Available: https://www.bluetooth.com/blog/3-key-factors-that-determinethe-range-of-bluetooth/

[15] Jesús Alonso-Zárate, "Sistemas de comunicación en la banda ISM."

[16] "Radiocom Frequency Plan – ICTA Website." Accessed: Aug. 24, 2024. [Online]. Available: https://www.icta.mu/radiocom-frequency-plan/

[17] Benny Bing, *Wireless local area networks*. John Wiley & sons, 2002.

[18] "Specifications | Bluetooth® Technology Website." Accessed: Aug. 24, 2024. [Online]. Available: https://www.bluetooth.com/specifications/specs/

[19] "Guía sobre diferentes versiones de Bluetooth: De 1.0 a 5.4 y más allá." Accessed: Aug. 24, 2024. [Online]. Available: https://www.mokosmart.com/es/guide-on-different-bluetooth-versions/

[20] "Bluetooth Technology: What Has Changed Over The Years | by Jaycon | Jaycon | Medium." Accessed: Aug. 24, 2024. [Online]. Available: https://medium.com/jaycon-systems/bluetooth-technology-what-has-changed-over-the-years-385da7ec7154

[21] "Evolution of Bluetooth services", Accessed: Aug. 24, 2024. [Online]. Available: https://gtrusted.com/product/121524#

[22] "2024 Bluetooth® Market Update | Bluetooth® Technology Website." Accessed: Aug. 15, 2024. [Online]. Available: https://www.bluetooth.com/2024-market-update/

[23] "What is Bluetooth LE? - Nordic Developer Academy." Accessed: Aug. 25, 2024. [Online]. Available: https://academy.nordicsemi.com/courses/bluetooth-low-energy-fundamentals/lessons/lesson-1-bluetooth-low-energy-introduction/topic/what-is-bluetooth-le/

[24] F. J. Dian, A. Yousefi, and S. Lim, "A practical study on Bluetooth Low Energy (BLE) throughput," *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON 2018*, pp. 768–771, Jul. 2018, doi: 10.1109/IEMCON.2018.8614763.

[25] "GAP: Device roles and topologies - Nordic Developer Academy." Accessed: Aug. 25, 2024. [Online]. Available: https://academy.nordicsemi.com/courses/bluetooth-low-energy-fundamentals/lessons/lesson-1-bluetooth-low-energy-introduction/topic/gap-device-roles-and-topologies/

[26] S. Kumar, "A REVIEW ON CLIENT-SERVER BASED APPLICATIONS AND RESEARCH OPPORTUNITY," *Article in International Journal of Scientific Research*, 2019, doi: 10.24327/ijrsr.2019.1007.3768.

[27] "GATT operations - Nordic Developer Academy." Accessed: Aug. 25, 2024. [Online]. Available: https://academy.nordicsemi.com/courses/bluetooth-low-energy-fundamentals/lessons/lesson-4-bluetooth-le-data-exchange/topic/gatt-operations/

[28] "ATT & GATT: Data representation and exchange - Nordic Developer Academy." Accessed: Aug. 25, 2024. [Online]. Available: https://academy.nordicsemi.com/courses/bluetooth-low-energy-fundamentals/lessons/lesson-1-bluetooth-low-energy-introduction/topic/att-gatt-data-representation-and-exchange/

[29] "Advertising process - Nordic Developer Academy." Accessed: Aug. 25, 2024. [Online]. Available: https://academy.nordicsemi.com/courses/bluetooth-low-energy-fundamentals/lessons/lesson-2-bluetooth-le-advertising/topic/advertising-process/

[30] "Advertising types - Nordic Developer Academy." Accessed: Aug. 25, 2024. [Online]. Available: https://academy.nordicsemi.com/courses/bluetooth-low-energy-fundamentals/lessons/lesson-2-bluetooth-le-advertising/topic/advertising-types/

[31] "Advertisement packet - Nordic Developer Academy." Accessed: Aug. 25, 2024. [Online]. Available: https://academy.nordicsemi.com/courses/bluetooth-low-energy-fundamentals/lessons/lesson-2-bluetooth-le-advertising/topic/advertisement-packet/

[32] "The Wireless Sensor Network Solution - THIS IS ANT." Accessed: Aug. 25, 2024. [Online]. Available: https://www.thisisant.com/

[33] "Zigbee | Complete IOT Solution - CSA-IOT." Accessed: Aug. 25, 2024. [Online]. Available: https://csa-iot.org/all-solutions/zigbee/

[34] "NFC Forum." Accessed: Aug. 25, 2024. [Online]. Available: https://nfc-forum.org/

[35] "Nordic Semiconductor | Empowering Wireless Innovation - nordicsemi.com." Accessed: Mar. 05, 2024. [Online]. Available: https://www.nordicsemi.com/

[36] "Getting Started with iBeacon," 2014, Accessed: Aug. 25, 2024. [Online]. Available: https://developer.apple.com/ibeacon

[37] "google/eddystone: Specification for Eddystone, an open beacon format from Google." Accessed: Aug. 25, 2024. [Online]. Available: https://github.com/google/eddystone

[38] B. Q. Tan, F. Wang, J. Liu, K. Kang, and F. Costa, "A Blockchain-Based Framework for Green Logistics in Supply Chains," *Sustainability 2020, Vol. 12, Page 4656*, vol. 12, no. 11, p. 4656, Jun. 2020, doi: 10.3390/SU12114656.

[39] P. Spachos and K. N. Plataniotis, "BLE Beacons for Indoor Positioning at an Interactive IoT-Based Smart Museum," *IEEE Syst J*, vol. 14, no. 3, pp. 3483–3493, Sep. 2020, doi: 10.1109/JSYST.2020.2969088.

[40] “Sus dispositivos IoT ODM & Socio JDM - MOKOSmart #1 Solución de dispositivo inteligente en China.” Accessed: Sep. 01, 2024. [Online]. Available: https://www.mokosmart.com/es/

[41] “Etiqueta de baliza ultradelgada M1 | MOKOSmart.” Accessed: Sep. 01, 2024. [Online]. Available: https://www.mokosmart.com/es/beacon-tag/

[42] “Etiqueta M4 Lite - MOKOSmart #1 Solución de dispositivo inteligente en China.” Accessed: Sep. 01, 2024. [Online]. Available: https://www.mokosmart.com/es/m4-lite-tag/

[43] “nRF51 development kit for Bluetooth Low Energy - nordicsemi.com.” Accessed: Aug. 26, 2024. [Online]. Available: https://www.nordicsemi.com/Products/Development-hardware/nRF51-DK

[44] “Developing with the MDK-ARM Microcontroller Development Kit User Guide v1.2,” 2017.

[45] “AIS2DW12 pdf, AIS2DW12 Description, AIS2DW12 Datasheet, AIS2DW12 view ::: ALLDATASHEET :::” Accessed: Aug. 26, 2024. [Online]. Available: https://www.alldatasheet.com/datasheet-pdf/view/1441572/STMICROELECTRONICS/AIS2DW12.html

[46] “STEVAL-MKI206V1 - AIS2DW12 adapter board for a standard DIL 24 socket - STMicroelectronics.” Accessed: Aug. 26, 2024. [Online]. Available: https://www.st.com/en/evaluation-tools/steval-mki206v1.html

[47] “Developing with the MDK-ARM Microcontroller Development Kit User Guide v1.2,” 2017.

[48] “Get Started with nRF Connect SDK - nordicsemi.com.” Accessed: Aug. 26, 2024. [Online]. Available: https://www.nordicsemi.com/Products/Development-software/nRF-Connect-SDK/GetStarted#infotabs

[49] “nrfconnect/sdk-nrf: nRF Connect SDK main repository.” Accessed: Aug. 26, 2024. [Online]. Available: https://github.com/nrfconnect/sdk-nrf

[50] P. Hambarde, R. Varma, and S. Jha, “The survey of real time operating system: RTOS,” *Proceedings - International Conference on Electronic Systems, Signal Processing, and Computing Technologies, ICESC 2014*, pp. 34–39, 2014, doi: 10.1109/ICESC.2014.15.

[51] “Zephyr Project – A proven RTOS ecosystem, by developers, for developers.” Accessed: Aug. 26, 2024. [Online]. Available: https://zephyrproject.org/

[52] “zephyrproject-rtos/zephyr: Primary Git Repository for the Zephyr Project. Zephyr is a new generation, scalable, optimized, secure RTOS for multiple hardware architectures.” Accessed: Aug. 26, 2024. [Online]. Available: https://github.com/zephyrproject-rtos/zephyr

[53] "Cómo descargar Android Studio y App Tools - Android Developers." Accessed: Sep. 01, 2024. [Online]. Available: https://developer.android.com/studio?hl=es-419

[54] "Intents y filtros de intents | Android Developers." Accessed: Sep. 01, 2024. [Online]. Available: https://developer.android.com/guide/components/intents-filters?hl=es-419

[55] "Parcelable | Android Developers." Accessed: Sep. 01, 2024. [Online]. Available: https://developer.android.com/reference/android/os/Parcelable

[56] "Descripción general de los avisos | Android Developers." Accessed: Sep. 01, 2024. [Online]. Available: https://developer.android.com/guide/topics/ui/notifiers/toasts?hl=es-419

[57] "KiCad EDA - Schematic Capture & PCB Design Software." Accessed: Sep. 03, 2024. [Online]. Available: https://www.kicad.org/

[58] "EasyEDA – Simulador de circuitos y diseño de circuitos impresos online." Accessed: Sep. 03, 2024. [Online]. Available: https://easyeda.com/es

[59] "PCB Manufacturing & Assembly Capabilities - JLCPCB." Accessed: Sep. 03, 2024. [Online]. Available: https://jlcpcb.com/capabilities/pcb-capabilities

[60] "Electronic Components Distributor - Mouser Electronics Spain." Accessed: Sep. 03, 2024. [Online]. Available: https://www.mouser.es/?utm_id=93823169&gad_source=1&gclid=CjwKCAjw59q2BhBOEiwAKc0ijQdRvWMuvxZb7hAWjMCcUHZfRaabWSPEAbPR0XI6uo-XVp06ibJYYBoCTCgQAvD_BwE

[61] "Free Online PCB CAD Library | Ultra Librarian." Accessed: Sep. 03, 2024. [Online]. Available: https://www.ultralibrarian.com/?utm_term=ultra%20librarian&utm_campaign=SEM_Branded&utm_source=google&utm_medium=cpc&hsa_acc=2461698748&hsa_cam=1489056918&hsa_grp=57116751323&hsa_ad=661390980134&hsa_src=g&hsa_tgt=kwd-308115260373&hsa_kw=ultra%20librarian&hsa_mt=e&hsa_net=adwords&hsa_ver=3&gad_source=1&gclid=CjwKCAjw59q2BhBOEiwAKc0ijbxc89CDbjzlrOSOWWaRqrko6LPtvlsQLMo5Jfk7EXvjDcx3DRPZnBoCOBgQAvD_BwE

[62] "BAL-NRF01D3," 2017, Accessed: Sep. 04, 2024. [Online]. Available: www.st.com

[63] "REGLAMENTO (UE) 2016/ 679 DEL PARLAMENTO EUROPEO Y DEL CONSEJO - de 27 de abril de 2016 - relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos dat os y por el que se deroga la Directiva 95/ 46/ CE (Reglamento general de protección de datos)".

[64] “nRF Command Line Tools - Downloads - nordicsemi.com.” Accessed: Sep. 09, 2024. [Online]. Available: https://www.nordicsemi.com/Products/Development-tools/nRF-Command-Line-Tools/Download?lang=en#infotabs

Anexo A: Configuración de beacons comerciales

En el siguiente anexo se describirá el proceso necesario para configurar los beacons comerciales la empresa Moko Smart para el uso que se va a hacer dentro de este proyecto

A.1 Tags M1 y M4 Lite

Para la configuración del Tag M1, es necesario descargar una aplicación específica. Esta aplicación se encuentra en el play store de Android. Se muestra la aplicación en la siguiente figura:

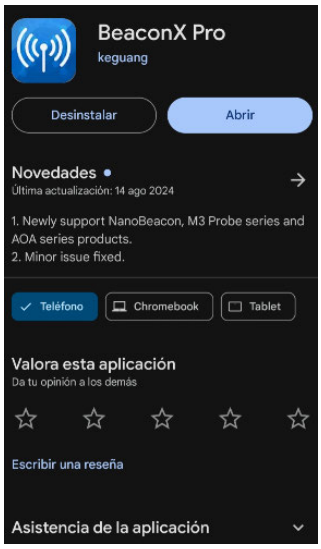


Figura 137. Aplicación de configuración para tag M1

El Tag M4 Lite puede ser configurado desde otra aplicación específica. De la misma forma que en el caso anterior, esta app se puede encontrar de forma gratuita en el app store. Se muestra la app en la siguiente figura:

Figura 138. Aplicación de configuración para tag M4Lite

Para el caso de la app para la configuración del Tag M1, al iniciar la app se mostrará un selector para varios tipos de dispositivos. Estos beacons funcionan con microcontroladores de NordicSemicondictor, por lo que se selecciona la primera opción, tal y como se observa en la siguiente figura:

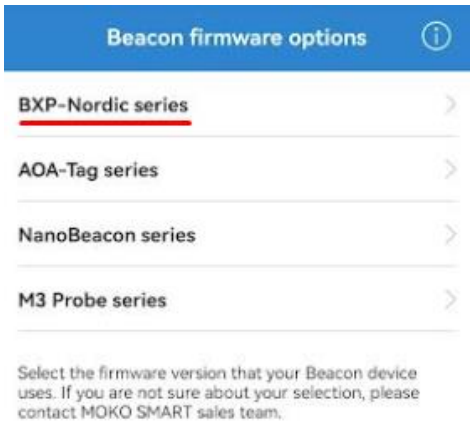


Figura 139. Selección de fabricante en app de M1

Desde este punto, los pasos a replicar son idénticos para ambos tags, ya que las apps son muy similares. Se mostrará el proceso con pantallazos de configuración del tag M1, pero el proceso es replicable para el M4 Lite con su correspondiente aplicación.

Una vez descargada y abierta se mostrará una pantalla en blanco, la cual estará buscando beacons que configurar. Para que aparezca el dispositivo es necesario que esté emitiendo señal BLE, por lo que se recomienda mantenerlo en movimiento hasta finalizar la conexión. En la figura 140 se muestra el resultado de la detección del dispositivo. En ella podemos observar como se muestra información relevante sobre el tag. Se puede observar el nombre, la dirección MAC o la batería entre otros atributos. Para continuar, se pulsa sobre el botón "CONNECT", señalado por la flecha.

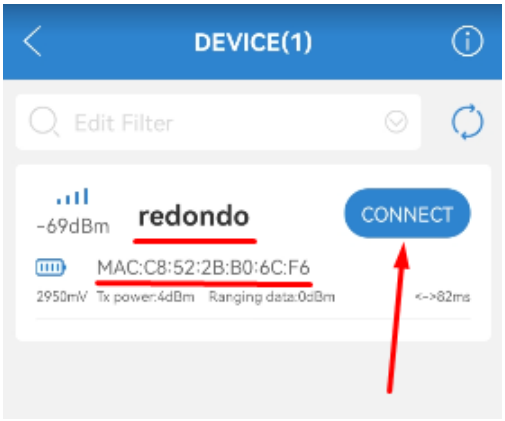


Figura 140. Proceso de conexión para la configuración

Una vez seleccionado el dispositivo se pedirá una contraseña, tal y como se muestra en la figura 141. Por defecto es Moko4321, aunque puede ser modificada desde la pestaña "Settings" de cada dispositivo, aunque es algo que no se cubre en este anexo.

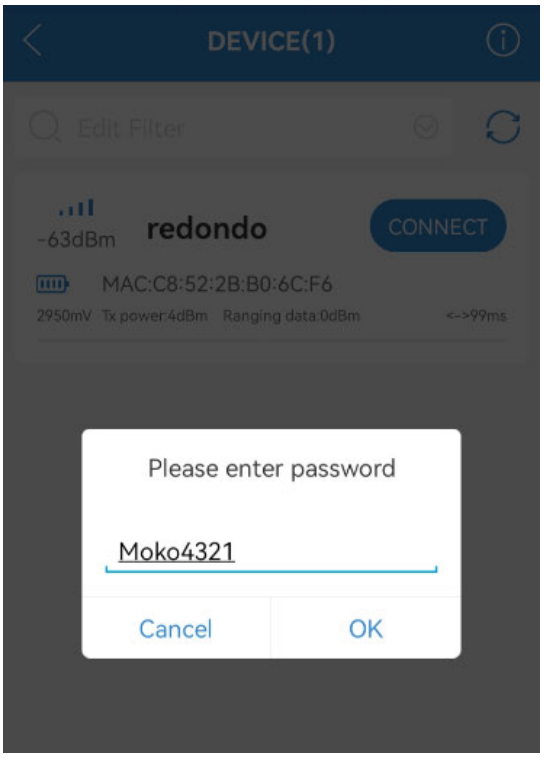


Figura 141. Contraseña para la conexión

Cuando la conexión queda establecida con el dispositivo, se muestra un conjunto de SLOTS, los cuales pueden modificarse para mostrar distinta información del tag. Todos los SLOTS pueden configurarse de igual manera. Para mostrar como configurar el tag seleccionaremos el SLOT1, ya que este muestra de forma predeterminada la información del dispositivo.

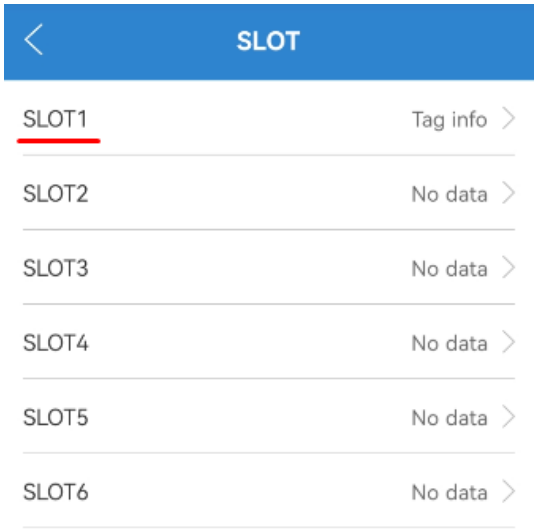


Figura 142. Slots configurables del dispositivo

Dentro de la pestaña de cada SLOT tenemos varios puntos importantes, todos ellos marcados en la figura 143.

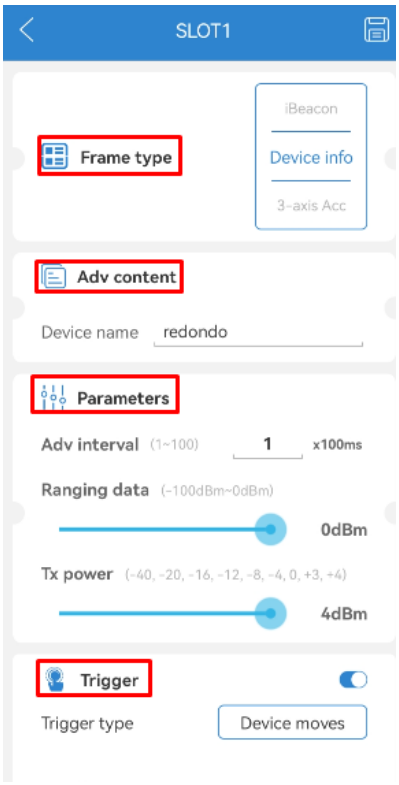


Figura 143. COnfiguración de SLOT1

- **Frame type:** indica el tipo de información que se va a mostrar en el SLOT, Pueden seleccionarse varias opciones, entre ellas la configuración del sensor de movimiento, la personalización del protocolo iBeacon, la personalización para Eddystone o la información del dispositivo.

- **Adv content:** En este parámetro se modificará el nombre visible del tag.

- **Parameters:** En este bloque se puede personalizar los parámetros relacionados con el periodo y la potencia de transmisión de la señal del beacon.

- **Trigger:** Esta opción aparece deshabilitada de forma predeterminada. Este bloque permite que el beacon no emita de forma continua, si no que pueda configurarse una señal habilite la emisión de la señal BLE del beacon. Este bloque es el que se debe configurar para cambiar el funcionamiento del beacon, tal y como se busca en el proyecto. Para ello debe configurarse como en la figura 144. Con esta configuración, el tag comenzará a emitir cuando el sensor de movimiento detecte un cambio significativo en la aceleración, es decir, cuando se produzca un movimiento. Con la configuración establecida, el beacon dejará de emitir cuando hayan pasado 10 segundos desde la última interrupción de movimiento.

Figura 144. Configuración del trigger

Con esta configuración, el beacon queda listo para el uso que se le dará en el proyecto, emitiendo señal BLE tan solo por 10 segundos desde que se detecta el movimiento.

Anexo B: proceso de instalación de las herramienta necesarias para comenzar con el desarrollo del firmware

A continuación, se describen la serie de pasos que deben seguirse para completar la instalación de todas las herramientas necesarias para el desarrollo del firmware del proyecto.

B.1 Instalación de nRF Command Line Tools

nRF COmmand Line Tools es la herramienta base para el desarrollo y compilación de los fimrwares. Puede descargarse desde el enlace de la siguiente referencia [63]. Este enlace muestra una página como la de la siguiente figura:

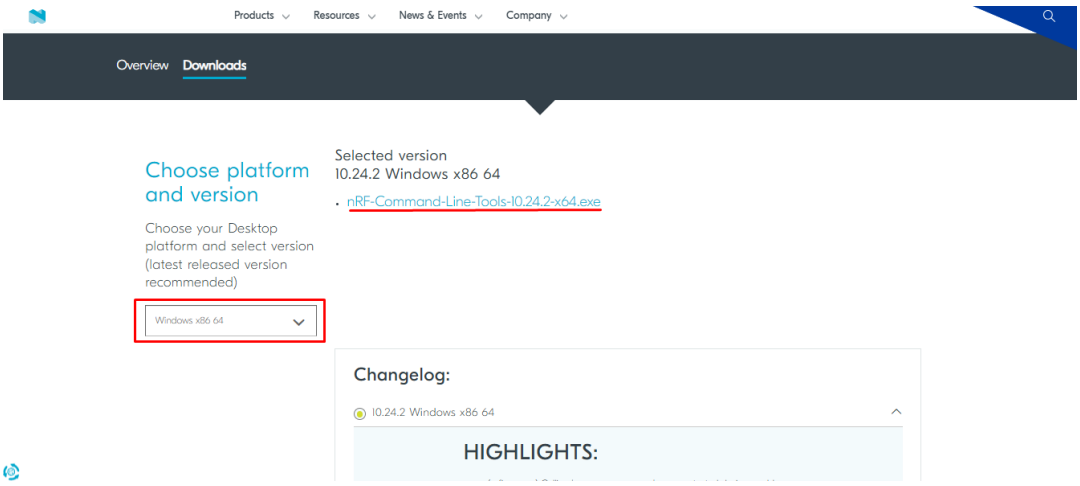


Figura 145. Página de descarga de nRF Command Line Tool

Es importante descargar la última versión. Para ello, se pincha sobre el recuadro, se elige el sistema operativo en el que quieres descargarlo, y posteriormente, se pincha en el enlace que está subrayado. Con estos pasos se descargará un ejecutable, el cual debe abrirse.

Una vez ejecutado el fichero seguimos los siguientes pasos:

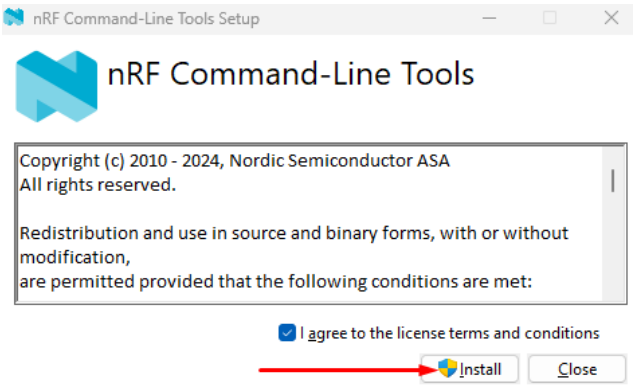


Figura 146. Instalación de nRF Command Line Tool (1)

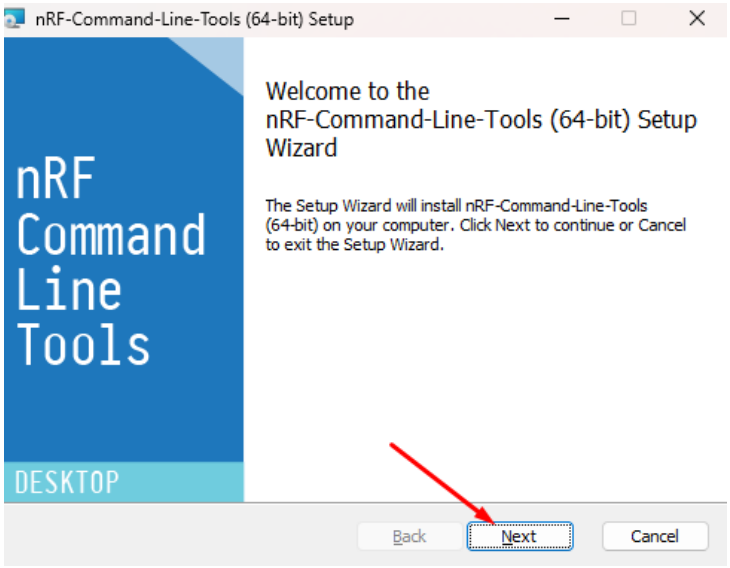


Figura 147.Instalación de nRF Command Line Tool (2)

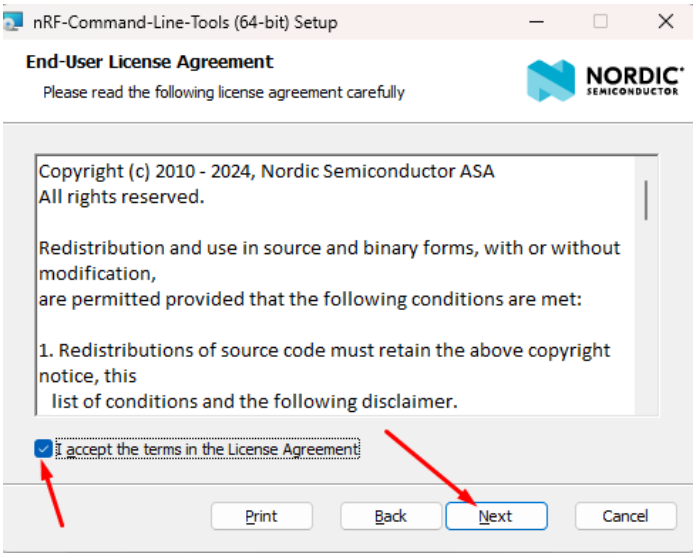


Figura 148. Instalación de nRF Command Line Tool (3)

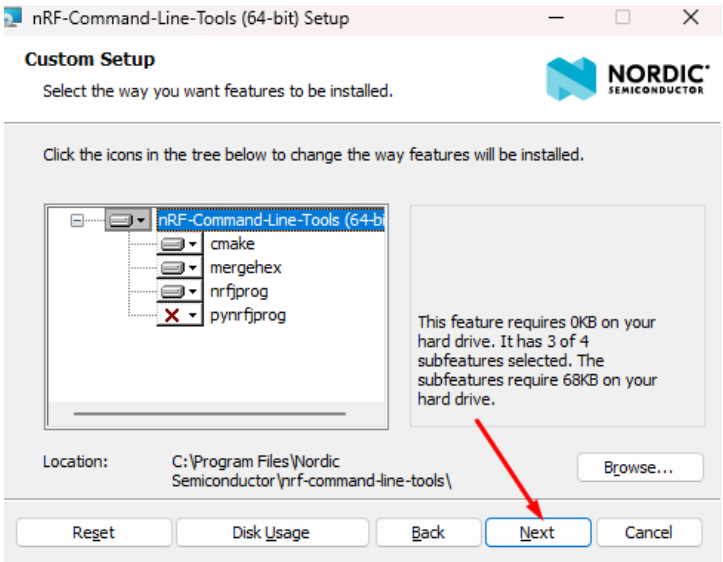


Figura 149. Instalación de nRF Command Line Tool (4)

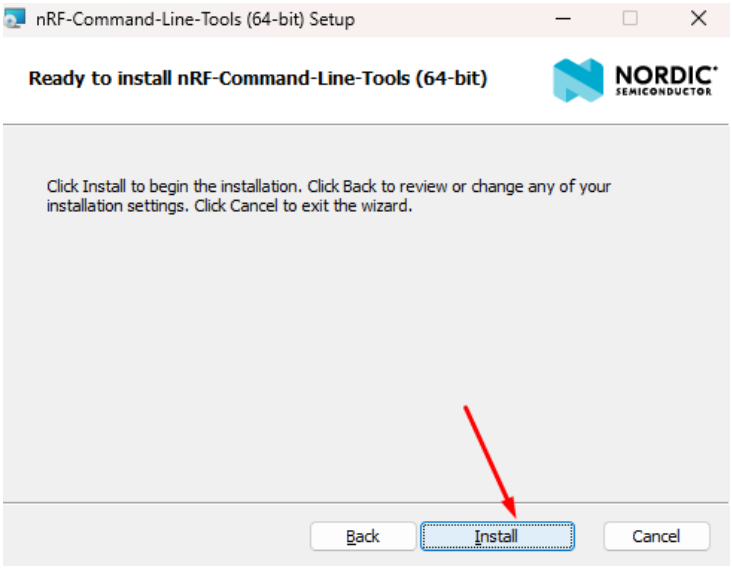


Figura 150. Instalación de nRF Command Line Tool (5)

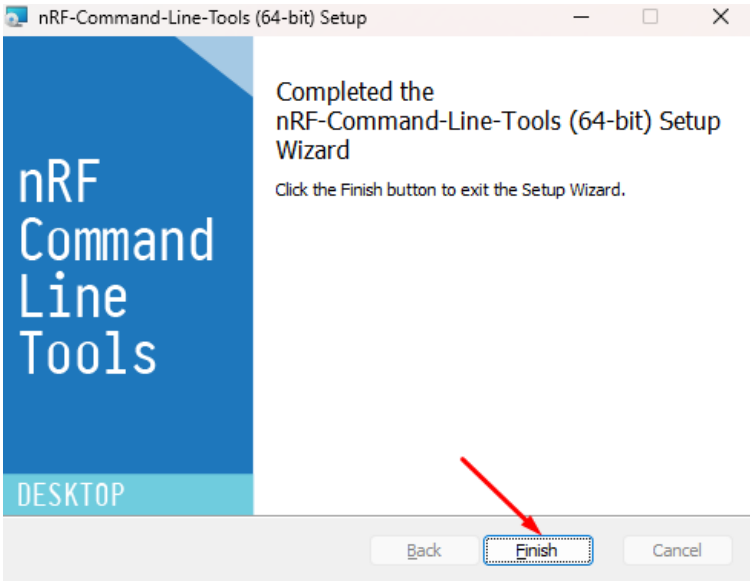


Figura 151. Instalación de nRF Command Line Tool (6)

B.2 Instalación de Visual Studio Code

Visual Studio Code es el IDE con el que se ha desarrollado el firmware. No es obligatorio realizarlo con este, pero si recomendable, ya que las extensiones que se mostrarán a continuación son de gran utilidad para el desarrollador. La instalación de VSCode puede seguirse en un tutorial externo, ya que no es objetivo de este anexo.

B.3 Instalación de la extensión nRF Connect para VSCode

En VSCode seleccionar la pestaña de extensiones y usar el buscador para encontrar nRF Connect. Deben instalarse todas las extensiones relacionadas. En la siguiente figura se muestran las extensiones, aunque puede ahorrarse tiempo si se selecciona el "Extensión Pack", el cual incluye todo lo necesario.

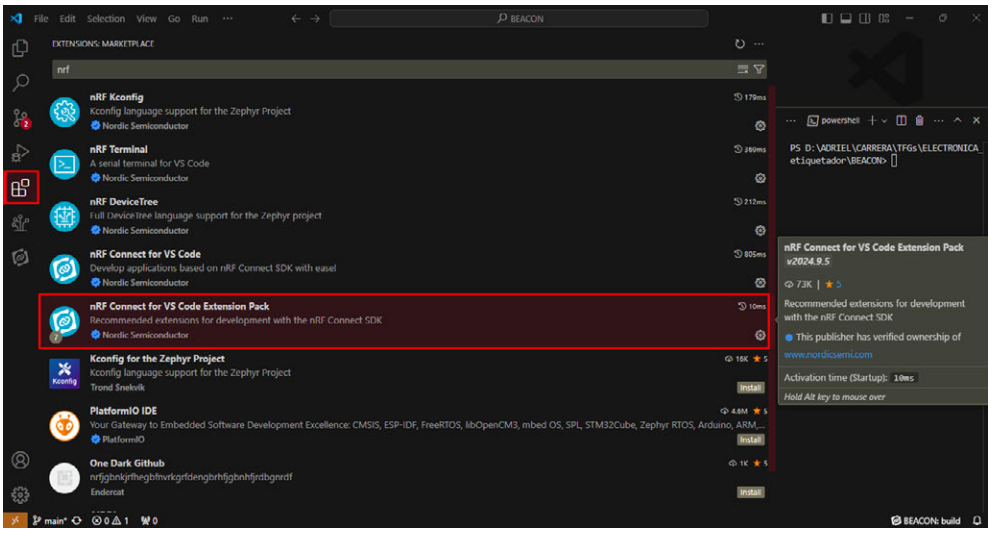


Figura 152. Instalación de la extensión nRF Connect para VSCode

B.4 Instalación de la Toolchain

Finalmente debe instalarse la toolchain. Para ello se debe entrar en el icono de la extensión que se acaba de instalar y marcar la opción de "install toolchain". Se recomienda elegir la versión más reciente.

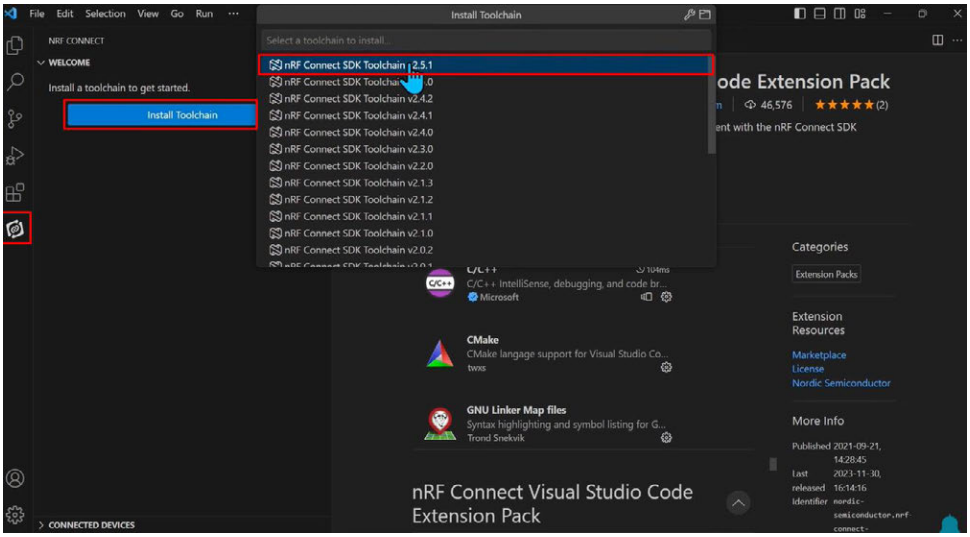


Figura 153. Instalar toolchain