



UNIVERSITAT POLITÈCNICA DE CATALUNYA  
BARCELONATECH

Escola d'Enginyeria de Telecomunicació  
i Aeroespacial de Castelldefels

# TREBALL DE FI DE GRAU

**TÍTOL DEL TFG:** Disseny d'una xarxa sense fils de sensors amb Bluetooth Low Energy per a mesures biològiques i mediambientals

**TITULACIÓ:** Grau en Enginyeria de Sistemes de Telecomunicació

**AUTOR:** Sergi Garreta Serra

**DIRECTOR:** Josep Polo Cantero

**DATA:** 14 d'octubre de 2020



**Títol:** Disseny d'una xarxa sense fils de sensors amb Bluetooth Low Energy per a mesures biològiques i mediambientals

**Autor:** Sergi Garreta Serra

**Director:** Josep Polo Cantero

**Data:** 14 d'octubre de 2020

## Resum

Avui en dia tothom coneix la tecnologia Bluetooth però molt poca gent sap realment que Bluetooth està format per dues tecnologies diferents. En aquest treball es tractarà el Bluetooth Low Energy que s'utilitza de forma molt eficient per a dispositius mòbils.

D'aquesta manera, s'entendran les diferències més destacables de BLE respecte al Bluetooth Clàssic, l'altre versió de Bluetooth, així com les millores que aporten les versions de BLE més recents. Es veurà com ha evolucionat la tecnologia des de la seva concepció com a idea original fins a ser un dels protocols més famosos i utilitzats actualment. Es contextualitzarà Bluetooth Low Energy dins de l'entorn de tecnologies inal·làmbriques de baix consum comparant aquestes entre sí.

Convé destacar, que l'objectiu principal d'aquest treball és explicar BLE en profunditat així com detallar la informació necessària per poder desenvolupar una implementació senzilla del protocol. Tantmateix, s'explicarà com s'organitzen les dades, com es transmet la informació i el benefici d'aquesta tecnologia per a xarxes de sensors.

Posteriorment, es tractaran les eines que s'empren amb el xip CC1352R de Texas Instruments. Describint pas a pas com utilitzar l'entorn SimpleLink per desenvolupar un projecte des de zero amb el xip mencionat. Basant-se en aquestes explicacions, es poden fer implementacions de nous projectes i escenaris similars així com realitzar modificacions.

A més a més, es mesurarà quin és el consum del dispositiu i també s'analitzarà l'abast al que pot arribar aquest utilitzant BLE. De manera paral·lela, es dissenyarà un projecte real per a la comunicació de sensors que prenguin mesures mediambientals i biològiques. Finalment, es desenvoluparà una aplicació que mostrarà en pantalla les dades recollides que previament d'han transmès cap al dispositiu mòbil a través de Bluetooth Low Energy.



**Title :** Design of a Wireless Sensor Network (WSN) using Bluetooth Low Energy (BLE) for environmental and biological measurements for Internet of Things (IoT)

**Author:** Sergi Garreta Serra

**Advisor:** Josep Polo Cantero

**Date:** October 14, 2020

## Overview

Overview



Escriure aquí  
la dedicatòria





# ÍNDEX

<b>Introducció</b>	<b>1</b>
<b>CAPÍTOL 1. Orígens del Bluetooth Low Energy</b>	<b>3</b>
1.1. Ús lliure de radiofreqüència	3
1.2. Bluetooth	3
1.2.1. Història de Bluetooth Clàssic	3
1.2.2. Història de Bluetooth Low Energy	4
1.3. Bluetooth vs BLE	4
1.4. MANETS ( <i>Mobile Ad-Hoc Networks</i> )	4
1.4.1. Altres MANETS	5
1.4.2. Comparació	6
1.5. Versions de BLE	7
1.6. Pila BLE	9
1.6.1. Controller	9
1.6.2. Host	12
1.7. Anunciaments	15
1.7.1. Tipus	15
1.7.2. Paràmetres	16
1.8. Escaneig	17
1.8.1. Tipus	18
1.8.2. Paràmetres	18
1.9. Connexions	18
1.9.1. Paràmetres	19
1.10. Seguretat	20
1.11. Format	21
1.11.1. BLE 4.2	22
1.11.2. BLE 5	23
<b>CAPÍTOL 2. Desenvolupament</b>	<b>25</b>
2.1. LaunchXL CC1352R1	25

<b>2.2. Software</b>	<b>26</b>
<b>2.3. Project 0</b>	<b>26</b>
2.3.1. Serveis de Botons i LEDs	26
2.3.2. Propietats	28
<b>2.4. Client BLE</b>	<b>29</b>
2.4.1. Btool	29
<b>2.5. Crear un Perfil propietari</b>	<b>29</b>
2.5.1. Generar Fitxers	29
2.5.2. Definir el Perfil Propi	30
 <b>CAPÍTOL 3. Projecte de sensors</b>	 <b>35</b>
<b>3.1. Experimentació</b>	<b>35</b>
3.1.1. Abast	35
3.1.2. Consum d'Energia	36
3.1.3. Simulador de sensors	38
<b>3.2. Lectura del ADC</b>	<b>39</b>
<b>3.3. Aplicació mòbil</b>	<b>40</b>
<b>3.4. Escenari</b>	<b>40</b>
<b>3.5. Continuitat del projecte</b>	<b>41</b>
 <b>Conclusions</b>	 <b>43</b>
 <b>Bibliografia</b>	 <b>45</b>
 <b>APÈNDIX A. Taula de atributs del Project 0</b>	 <b>51</b>

# ÍNDEX DE FIGURES

1.1	Logo de Wibree . . . . .	4
1.2	Pila de BLE . . . . .	9
1.3	Controller Stack . . . . .	10
1.4	Canals BLE . . . . .	10
1.5	Estats de la capa d'enllaç [9] . . . . .	11
1.6	Pila del Host . . . . .	12
1.7	Jerarquia de GATT [10] . . . . .	13
1.8	Exemple d'anunci estàt . . . . .	16
1.9	Paràmetres d'anunci [18] . . . . .	17
1.10	Paràmetres d'escaneig . . . . .	18
1.11	Establiment de connexió [19] . . . . .	20
1.12	Comparació d'esdeveniments amb latència d'esclau [20] . . . . .	21
1.13	Format del paquet en BLE 4.2 [21] . . . . .	22
1.14	Format PDU . . . . .	22
1.15	Payload de ADV_IND [21] . . . . .	23
1.16	Format de la Extended Advertising Payload [23] . . . . .	23
2.1	Placa [24] . . . . .	25
2.2	Definició dels UUIDs . . . . .	27
3.1	Abast real de BLE . . . . .	36
3.2	Punts extraïbles de la placa . . . . .	37
3.3	Captura d'Energy Trace . . . . .	37
3.4	Captura d'Energy Trace 2 . . . . .	37
3.5	Protoboard amb el circuit . . . . .	39
3.6	Ports placa . . . . .	41
3.7	Ports placa . . . . .	42



# ÍNDEX DE TAULES

1.1 Comparació entre MANETs . . . . .	6
1.2 Comparació de diferents capes físiques . . . . .	8
1.3 Exemple de possibles atributs . . . . .	14
1.4 Tipus d'anunciaments . . . . .	15
2.1 Atributs del Project 0 . . . . .	27
2.2 Valors de les propietats . . . . .	28
3.1 Tipus d'anunciaments . . . . .	40
3.2 Taula dels pins amb ADC . . . . .	41



# INTRODUCCIÓ

La arquitectura internet de les coses que s'ha popularitzat en els últims anys a comportat unes noves necessitats en molts aspectes de la tecnologia que són molt diferents als anteriors. Quan el que es vol és que tot estigui connectat entre si, queda clar que es trenquen moltes premisses amb les que s'havien dissenyat originalment les tecnologies més importants.

La transició cap a l'Internet Of Things, transforma el paradigma de les comunicacions de pocs elements a molta velocitat (xarxes centralitzades) cap a molts elements amb poca velocitat (xarxes distribuïdes). Això afecta les tecnologies sense fils que no havien estat dissenyades pel seu ús tant generalitzat.

Aquests canvis es poden veure en l'evolució de la majoria de tecnologies sense fils com WiFi, xarxes cel·lulars i Bluetooth entre d'altres. En totes aquestes tecnologies ha estat necessari fer grans canvis per acomodar a molts més usuaris dels que s'havien predit inicialment.

El Bluetooth clàssic és un exemple de una tecnologia que tenia limitacions a l'hora de utilitzar-se en certs escenaris. No permetia comunicació entre més d'un dispositiu alhora, tenia un abast limitat, poques mesures per contrarestar les interferències... Al llarg de la seva història es van definir noves versions que anaven millorant les mancances del protocol. Tot i així, el cost que suposa haver d'establir una connexió per transmetre dades, encara que siguin molt poques, segueix sent molt alt.

És per això que Bluetooth va incorporar la extensió Low Energy de forma opcional. En aquesta extensió està definit el protocol anomenat Bluetooth Low Energy que soluciona el problema de Bluetooth Clàssic ja que permet molt fàcilment i utilitzant molt poca energia transmetre dades (que inclou a múltiples dispositius) sense haver d'establir una connexió.

Però Bluetooth Low Energy també va anar millorant amb les noves versions, millorant el rendiment en escenaris concrets. Aquestes millores inclouen augmentar el màxim de dades que es poden transmetre sense connexió o augmentar la taxa de dades de transmissió per poder estalviar energia. En aquest treball es veurà com funciona el BLE, com ha evolucionat al llarg del temps i es farà una implementació real del protocol.





# CAPÍTOL 1. ORÍGENS DEL BLUETOOTH LOW ENERGY

## 1.1. Ús lliure de radiofreqüència

En tot l'espai radioelèctric hi ha bandes que estan assignades per a un ús privat però lliure. Per utilitzar-les es necessari respectar la normativa nacional que normalment assimila la normativa comuna segons la ITU. Aquestes bandes es designen ISM, acrònim de *Industrial, Scientific i Medical*, que són els usos pel que s'havia pensat que serviria originalment, avui en dia els usos són més diversos.

Com que les bandes ISM estan suficientment establertes arreu del món es crea la possibilitat d'utilitzar la comunicació inal·làmbrica amb molta facilitat pel públic general. Aquesta situació proporciona el disseny de productes per al consumidor habitual que ràpidament veu els avantatges de la comunicació inal·làmbrica entre dispositius. En aquest entorn sorgeix la necessitat d'establir estàndards entre les companyies principals dels sectors corresponents. Per definir els estàndards s'agrupen les companyies i formen grups com la WI-FI Alliance o el NFC Forum.

## 1.2. Bluetooth

Bluetooth és un estàndard per l'intercanvi de dades de curt abast que utilitza la banda ISM de 2.4 GHz. És un dels protocols més coneguts i utilitzats, sobretot a través dels dispositius mòbils. A continuació, s'explicarà l'origen d'aquesta tecnologia.

### 1.2.1. Història de Bluetooth Clàssic

El desenvolupament de la tecnologia per a connexions de curt abast que acabarien esdevenint el que avui es coneix com a Bluetooth Clàssic va començar l'any 1989 per part d'Ericsson. Inicialment l'objectiu d'aquesta tecnologia era poder connectar auriculars i ordinadors sense necessitat d'una connexió per cables. D'altra banda a IBM es volia integrar la connectivitat de la xarxa de telefonia als ordinadors portàtils per poder realitzar i rebre trucades des d'aquests mateixos. El problema era que això suposava un consum d'energia significatiu en els ordinadors portàtils de l'època.

IBM va veure que la tecnologia que estava desenvolupant Ericsson podria servir per tenir connectivitat de telefonia en els portàtils si es connectava el telefon mòbil amb l'ordinador. Les dues empreses van acordar utilitzar aquesta tecnologia de curt abast en els seus productes corresponents. El resultat va ser que els mòbils Ericsson i ordinadors ThinkPad es podien comunicar entre si i així des del portàtil seria possible fer trucades. Com que ni Ericsson ni IBM tenien majoria en la quota de mercat dels respectius productes van decidir que la tecnologia fos oberta. D'aquesta manera es buscava integrar a més participants en aquesta tecnologia per tal d'estendre-la a la majoria de dispositius possibles. El 1998 es van unir al grup Intel, Nokia i Toshiba i totes 5 companyies van fundar el Bluetooth

SIG (*Bluetooth Special Interest group*, d'ara endavant SIG). Finalment, al 2001 van sortir a la venda els primers dispositius amb Bluetooth, el terminal Ericsson T39 i el portàtil IBM ThinkPad A30.

### 1.2.2. Història de Bluetooth Low Energy

Com que el món de les comunicacions estava evolucionant cap als dispositius sense fils i alimentats amb bateria, era necessari adaptar les tecnologies existents per a les noves necessitats. L'any 2001 a Nokia es va començar a desenvolupar una versió de Bluetooth que fos similar però que reduís significativament el consum d'energia amb els compromisos mínims possibles. Aquest desenvolupament va culminar l'any 2004 amb la publicació de la *Bluetooth Low End Extension* [3]. Posteriorment, es va dur a terme múltiples millores juntament amb Logitech en el projecte d'investigació per part de la Unió Europea MIMOSA [4]. El resultat del projecte es va anunciar al 2006 amb el nom de Wibree, es pot veure el logo a la figura 1.1.



Figura 1.1: Logo de Wibree

Els membres del SIG després de negociar entre sí, van acordar incloure Wibree al estàndard de Bluetooth en l'especificació 4.0 amb el nom de *Bluetooth ultra low power technology* i publicitat com a Bluetooth Smart. El primer mòbil a incloure'l va ser l'iPhone 4S al 2011. Posteriorment es va canviar el nom per *Bluetooth Low Energy* (BLE d'ara endavant).

## 1.3. Bluetooth vs BLE

El Bluetooth Low Energy no té cap relació amb el Bluetooth Clàssic pel que fa a l'arquitectura de la tecnologia. Tot i que comparteixen l'ús de la banda freqüencial de 2,4 GHz, igual que altres protocols sense fils, i contenen el nom de Bluetooth, queda clar que no estem parlant del mateix amb el simple fet que no son compatibles entre sí. De fet, quant un dispositiu vol implementar les dues tecnologies (anomenat mode dual) ho ha de fer per separat ja que només comparteixen l'antena; les modulacions i els altres blocs són tots diferents.

## 1.4. MANETS (*Mobile Ad-Hoc Networks*)

Bluetooth Clàssic no permet tenir més d'una connexió establerta amb dispositius. I té el desavantatge tot i voler transmetre poques dades l'arquitectura fa que es consumeixi bastanta energia per establir i mantenir la connexió. Això no suposava problemes inicialment, ja que l'ús principalment era per la transferència de fitxers com contactes o fotografies.

Amb l'arribada dels telèfons intel·ligents, els auriculars inal·làmbrics van esdevenir populars i es va l'escaneig dominància de Bluetooth Clàssic per a escoltar música. Per poder transmetre música d'alta fidelitat es va millorar Bluetooth per ser més resistent a interferències i aconseguir velocitats superiors. D'altre banda, va començar a sorgir l'Internet of Things, basat en tenir molts dispositius connectats transmetent a taxes molt variades i de forma discontinua. Això va generar necessitats que no es podien cobrir amb les tecnologies existents. Era necessari tenir xarxes sense fils, amb consum molt baix d'energia i que cobrissin una gran distància (10-100 metres). Amb aquests requeriments, van sorgir les xarxes ad hoc o *Mobile Ad-Hoc Networks* (MANETS d'ara endavant).

### 1.4.1. Altres MANETS

BLE és una de les MANETs més importants però n'hi ha d'altres que competeixen i que s'analitzaran a continuació.

#### 1.4.1.1. Zigbee

L'estàndard Zigbee s'utilitza en entorns per l'automatització de la casa, xarxes de sensors, col·lecció de dades mèdiques entre d'altres. Zigbee està dissenyat per sobre del IEEE 802.15.4 i per tant, només defineix les capes superiors. No està pensat per a situacions amb gran mobilitat entre nodes sinó per a desplegaments més estàtics. Zigbee, per exemple, s'utilitza en els dispositius de Philips Hue que serveixen per controlar bombetes intel·ligents.

#### 1.4.1.2. 6LoWPAN

IPv6 over Low power Wireless Personal Area Networks és un protocol que permet enviar paquets d'Internet Protocol versió 6 a través de l'IEEE 802.15.4. Aquest protocol està orientat a aportar internet amb connexions sense fils. Una de les característiques destacades és la capacitat de comprimir les capçaleres dels paquets per així reduir el sobrecost que suposa. La utilització del 6LoWPAN és majoritàriament coneguda per l'estandardització del protocol Thread que l'utilitza per a domòtica.

#### 1.4.1.3. Z-Wave

Z-Wave és un protocol utilitzat principalment per domòtica. La comunicació es fa en una xarxa en malla que connecta els electrodomèstics i per tant, permet el seu control. La principal característica d'aquest protocol es que utilitza únicament les freqüències ISM de la banda 800 o 900 MHz (segons continent) i així evita les interferències que hi ha a 2.4 GHz de WIFI o Bluetooth. La utilització de freqüències més baixes permet més abast amb menys potència, especialment quan es tracta de penetrar parets. Un exemple de dispositius que utilitzen aquesta tecnologia són els productes de la marca Ring principalment coneguts pels seus vídeo-porters intel·ligents.

#### 1.4.1.4. Insteon

Insteon està orientat a domòtica i utilitza conjuntament radiofreqüència i les línies d'alimentació de casa per a la comunicació. Aquest tipus de sistema s'anomena de malla dual. El fet d'estar connectats tots els dispositius a la instal·lació elèctrica permet una molt bona sincronització entre els dispositius. Això permet, per exemple, que múltiples dispositius puguin retransmetre paquets alhora per tal de millorar la cobertura i reduir les retransmissions. SmartLabs és la companyia que fabrica i ven els productes que utilitzen Insteon. La tecnologia és pot controlar a través de Cortana, Alexa o Apple Watch entre d'altres.

#### 1.4.1.5. LoRa

El Long Range desenvolupat per Semtech està orientat a cobrir les distàncies més grans d'entre les MANETs. Ho aconsegueix reduint la taxa de transmissió de dades reals, augmentant així, el temps dedicat a cada símbol donant més oportunitats al receptor per detectar correctament el senyal. Aquesta tècnica s'anomena espectre eixamplat i LoRa la implementa amb *Chirps*. Els *Chirps* són els polsos que es transmeten, sempre s'envien més *Chirps* que bits i la relació ve definida pel factor d'eixamplat o *Spreading Factor*. LoRa permet configurar aquest paràmetre que defineix l'equilibri entre la taxa de dades i la distància, pot prendre valors entre 7 i 12. El LoRa també implementa protecció contra errors o FEC (*Forward Error Correction*) per incrementar encara més l'abast.

Pel que fa a la pila de protocols que s'utilitzen en la comunicació, LoRa únicament defineix les capes inferiors. Això suposa que només s'ha especificat com funciona la modulació dels senyals analògics i com es transformen en símbols digitals. Tot i així, la LoRa Alliance va definir el LoRaWAN que és un dels protocols que es poden utilitzar per a les capes superiors. El LoRaWAN està basat en un Control d'Accés al Medi (MAC en anglès).

### 1.4.2. Comparació

Tot i que els protocols tenen molt en comú, cada un es diferencia de la resta en els detalls. A continuació, veiem una comparació de les capacitats de cada un dels protocols

	Abast (m)	Velocitat Física (Kbps)	Tasa de dades (Kbps)	Consum (mA)	Banda
BLE 4	50	1000	800	15	2.4 GHz
BLE 5	200	2000	1430	15	2.4 GHz
6LoWPAN	100	250	162	15	800-900 MHz i 2.4 GHz
Zigbee	100	250	162	30	800-900 MHz i 2.4 GHz
Z-Wave	90	100	40	23	800-900 MHz
Insteon	120	4,5	2,8	—	915 MHz
LoRa	10 Km	27	5	18	400/800-900 MHz

Taula 1.1: Comparació entre MANETs

Tal i com s'ha vist BLE és classifica segons les seves capacitats en la seva versió 4 i en la 5. En el següent apartat es veurà perquè és així.

<sup>1</sup> Les bandes indicades com a 900 MHz varien depenent del país entre 800 MHz i 1000 MHz

## 1.5. Versions de BLE

característiques de la versió 4.0 amb la 5 ja que hi ha canvis significatius [5]. Les novetats que aporta la nova versió principalment tenen l'objectiu de millorar les capacitats del protocol, en els casos extrems, tant quan l'entorn és molt favorable com quan no es gens favorable. Les millores i noves opcions de BLE fan que s'utilitzi més eficientment l'espectre freqüencial i per tant depenent de la regió es pugui augmentar la potència de transmissió de 10 dBm, que era el límit de BLE 4.2, a 20 dBm en BLE 5.

### 1.5.0.1. LE 2M

LE2M (*Low Energy 2 Mega*) és el nom que te la nova capa física quan opera a 2 MBd<sup>1</sup> en la versió 5 enlloc dels 1 MBd que es podia anteriorment en la versió 4.0. L'augment de la velocitat dels símbols permet augmentar la quantitat de dades que es poden transmetre a nivell d'aplicació. Aquest canvi està pensat per facilitar la utilització de BLE en situacions més exigents pel que fa a volum de dades com per exemple, múltiples mesures del cos humà.

### 1.5.0.2. Abast

Tot i que la tecnologia BLE en la seva versió original (4.0) tenia una distància màxima de transmissió teòrica molt gran (100 metres sense obstacles), no era suficient per a tots els casos. Per exemple, era necessari tenir més abast en els sectors com el de les cases intel·ligents. El límit de l'abast ve definit per la potència màxima de transmissió (sigui per protocol o per normativa) i per la probabilitat d'error de bit BER (*Bit Error Ratio*) que es pot acceptar. Bluetooth defineix la BER màxima en 0.1% per tant quan un de cada mil bits és erroni es considera que la connexió és massa dolenta i que ja s'ha arribat al límit de distància.

Per tractar els errors en transmissió hi ha dues estratègies: detecció d'errors i correcció d'errors. En la versió 4.0 de Bluetooth només s'utilitzava la detecció d'errors. En Bluetooth 5 s'implementa la correcció d'errors d'aquesta manera sense augmentar la potència transmesa es pot incrementar l'abast màxim que pot tenir la comunicació. El desavantatge és que més bits dels que es transmeten es dediquen a corregir errors per tant queda menys espai per a dades de l'aplicació.

L'esquema de correcció d'errors (*Forward Error Correction*) es fa amb un codificador convolucional que pot generar a la sortida més bits dels que entren. El codificador pot funcionar amb dos esquemes diferents s'anomenen  $S = 2$  i  $S = 8$ . Amb els diferents esquemes s'utilitza una taxa de codi diferent que pot augmentar la capacitat de correcció i incrementar l'abast però disminuir la quantitat de dades que es poden transmetre. Es poden veure les capacitats dels diferents esquemes a la taula 1.2.

<sup>1</sup>El Baud és la unitat de mesura de la velocitat de modulació, que correspon al nombre de canvis de l'estat d'un senyal en un segon.

	LE 1M	LE Coded S=2	LE Coded S=8	LE 2M
Symbol Rate	1 Ms/s	1 Ms/s	1 Ms/s	2 Ms/s
Data Rate	1 Mbit/s	500 Kbit/s	125 Kbit/s	2 Mbit/s
Error Detection	CRC	CRC	CRC	CRC
Error Correction	NONE	FEC	FEC	NONE
Range Multiplier (approx.)	1	2	4	0.8
Bluetooth 5 Requirement	Mandatory	Optional	Optional	Optional

Taula 1.2: Comparació de diferents capes físiques

#### 1.5.0.3. Advertisement Extensions

En Bluetooth 4.0 els paquets d'anunci (*Advertisement Packets*) tenen 6 octets de capçalera i com a molt 31 de dades, és a dir, com a molt es poden transmetre 31 Bytes de cop. Normalment aquests paquets es transmeten pels 3 canals d'anuncis el 37, 38 i 39.

En Bluetooth 5 per poder difondre (*broadcast*) més dades el que es fa és enviar un paquet d'anunci on només s'indica la capçalera i un punter cap al canal per on s'enviarà el paquet complet. Posteriorment, pel canal per on s'ha indicat s'envia el paquet de dades i en cas de necessitar més dades s'encadenen paquets. I d'aquesta manera les dades només es transmeten una vegada ja que abans, calia transmetre-les en els tres canals d'anunci. Aquest procediment està detallat més endavant en l'apartat [1.7.1](#).

En la versió 4.0 sempre hi havia una component aleatòria que marcava en quin moment s'enviava el paquet, això és comú i serveix per evitar col·lisions periòdiques. El problema que genera és que les ràdios han d'estar més temps escoltant fins a rebre el paquet per culpa d'aquesta aleatorietat.

En BLE 5 el GAP defineix un mode síncron que permet establir un procediment d'establiment de paquets anunci sincronitzats. Existeix una nova capçalera (SyncInfo) on s'indica amb exactitud l'interval i variació (per evitar col·lisions) dels paquets.

#### 1.5.0.4. Slot Availability Masks

La tecnologia LTE (telefonía) està utilitzant cada vegada més espai freqüencial i en preparació de que s'utilitzi en freqüencials pròximes a la banda de 2.4 GHz ISM s'ha desenvolupat un sistema per indicar disponibilitat en el temps. Per evitar les possibles interferències que causin altres tecnologies, es defineix la SAM (*Slot Availability Mask*) que permet identificar aquelles ranures de temps hi ha disponibilitat així bloquejant aquells moments on hi

hagui interferències i així evitar-les.

#### 1.5.0.5. Improved Frequency Hopping

Els salts en freqüència que utilitza la versió 4.0 estan definits per 12 seqüències predeterminades. En BLE 5 s'utilitza una seqüència pseudo-aleatòria per determinar quins canals s'utilitzen. L'algorisme en aquest cas *channel selection algorithm #2* i és més efectiu a evitar interferències i esvaniments per propagació multi-camí.

## 1.6. Pila BLE

Un cop vistes les característiques generals del protocol cal entendre com funciona per dins. En entrar en detall queda clar com el protocol s'ha dissenyat de forma molt flexible.

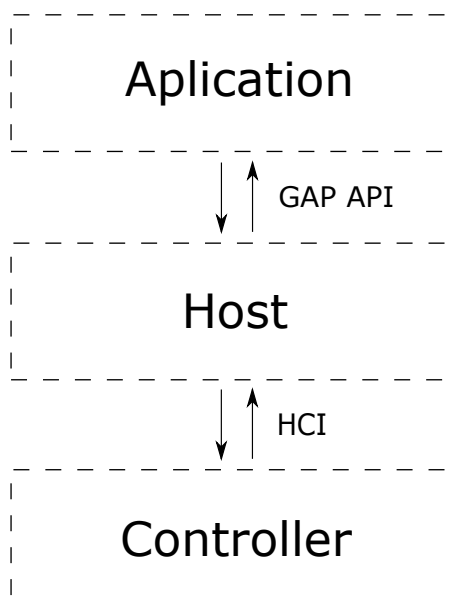


Figura 1.2: Pila de BLE

La pila pròpia de Bluetooth Low Energy està dividida en dues parts, tal i com es pot veure a la figura 1.2, el controlador i el *host*. Aquestes dues parts són independents i utilitzen el protocol *Host Controller Interface* (HCI d'ara endavant) per comunicar-se entre elles. Aquest protocol pot estar implementat amb qualsevol protocol de transport físic com USB o UART. La idea darrera separar la pila en dos serveix per fer compatible xips fets per diferents fabricants. Les dues parts de la pila poden estar implementades en el mateix xip anomenat configuració única o en xips separats anomenat configuració dual.

### 1.6.1. Controller

El controlador compren la capa física i la capa d'enllaç.

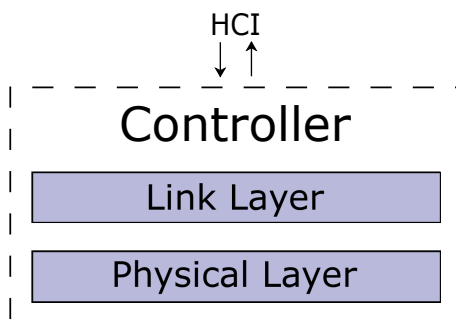


Figura 1.3: Controller Stack

#### 1.6.1.1. Capa Física

La capa física és la que s'encarrega de la comunicació anal·lògica modulant i desmodulant les senyals. Tal i com ja s'ha comentat abans treballa a la banda de 2.4 GHz en 40 canals diferents tal i com es mostra a la figura 1.4.

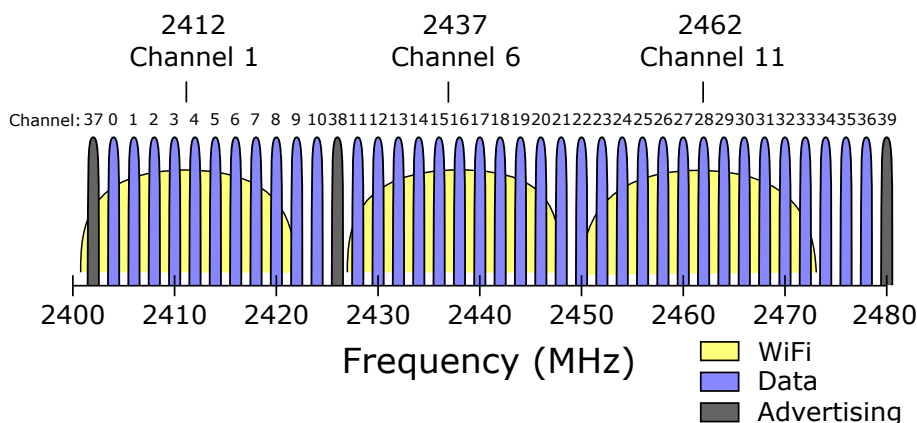


Figura 1.4: Canals BLE

Els canals es classifiquen en 37 de dades o també anomenats secundaris i 3 d'anunci (*Advertisement*) o primaris. En els canals d'anunci es vol tenir més qualitat ja que la informació que s'hi transmet és més important. Per exemple, són els canals que s'utilitzen per descobrir altres dispositius. És per això, que els canals d'anunci es troben en el buits que deixen els canals WiFi més comuns (1, 6 i 11), es pot veure a la figura 1.4.

El protocol BLE té l'objectiu de consumir el mínim d'energia possible i això s'aconsegueix, principalment, reduint el temps que s'està transmetent o escoltant a través de la ràdio. Si no s'ha de rebre o transmetre res, es pot apagar la ràdio i s'estalvia energia. La manera de tenir apagada la ràdio més temps és transmetre les dades el més ràpid possible. BLE és dels protocols que tenen la taxa de transmissió física <sup>2</sup> més alta, de fins a 1 Mbps originalment i de 2 Mbps en BLE 5.

La potència màxima de transmissió és de 20 dBm (100 mW) segons la especificació<sup>3</sup>. Però Aquesta potència de la ràdio es pot controlar, disminuint-la per consumir menys

<sup>2</sup>La taxa de transmissió física és aquella a la que transmeten la ràdio, no confondre amb la taxa de dades d'aplicació *Throughput*

<sup>3</sup>La placa utilitzada pot transmetre fins a 5 dBm de potència



energia. Tot i que això, només serà possible si l'entorn ho permet i el receptor pot rebre el senyal correctament. El paquets de BLE indiquen la potència amb que s'han transmès i junt amb el RSSI (*Received Signal Strength Indicator*), la potència rebuda, es pot estimar la distància fins al transmissor. Conèixer aquest valor és útil per a certes aplicacions, per exemple, la localització en espais interiors.

La modulació utilitzada per BLE és la GFSK (*Gaussian Frequency Shift Keying*), la mateixa que la majoria de MANETS que es basen en l'IEEE 802.11. Aquesta modulació és una de les més robustes, simples d'implementar i és el que permet a BLE, en part, tenir un abast molt més gran que Bluetooth Clàssic. El filtre gaussià redueix el consum pic d'energia [8] i també redueix les interferències en freqüències veïnes. En BLE 4.0 s'utilitza una desviació freqüencial de 185 kHz, en BLE 5 com que augmenta la velocitat de símbols també creix la interferència intersimbòlica. Per mitigar aquest efecte negatiu, la desviació de freqüència passa a ser de 370 kHz.

#### 1.6.1.2. Link Layer

La capa d'enllaç és l'encarregada d'escanejar, anunciar i gestionar les connexions amb altres dispositius.

Depenent del moment en la connexió, els dispositius es troben en diferents estats. Els estats són: Espera (*Idle*), Anunciador (*Advertiser*) o Escàner (*Scanner*) i Iniciador (*Initiator*), Esclau (*Slave*) o Mestre (*Master*). A la figura 1.5 s'hi pot veure el flux d'estats per arribar a una connexió.

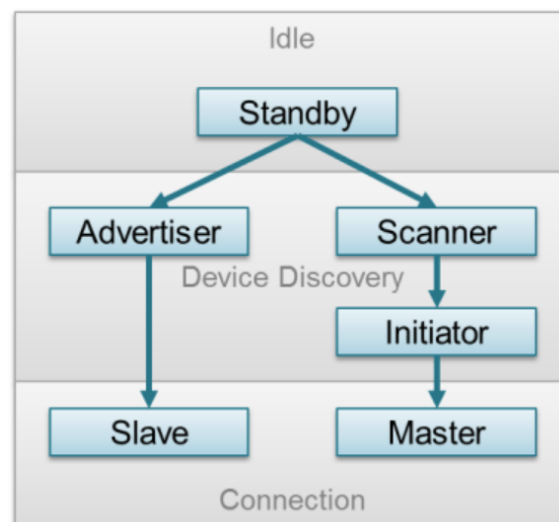


Figura 1.5: Estats de la capa d'enllaç [9]

Aquesta capa és l'encarregada d'implementar el salt en freqüència (*frequency hopping*) que permet mitigar l'impacte de les interferències estretes. Si s'utilitzés només un dels canals BLE i en aquell canal hi haguessin interferències no es podria establir comunicació. Al alternar múltiples canals, encara que hi hagi interferències en algun d'ells es considera que no n'hi haurà en tots i per tant hi podrà haver una bona comunicació. Els salts que es fan són des de 5 fins 16 canals per salt d'entre els dedicats a dades.

## 1.6.2. Host

BLE es va dissenyar amb la flexibilitat en ment, és per això que l'estructura de la pila del host permet no utilitzar certes capes que es consideren opcionals.

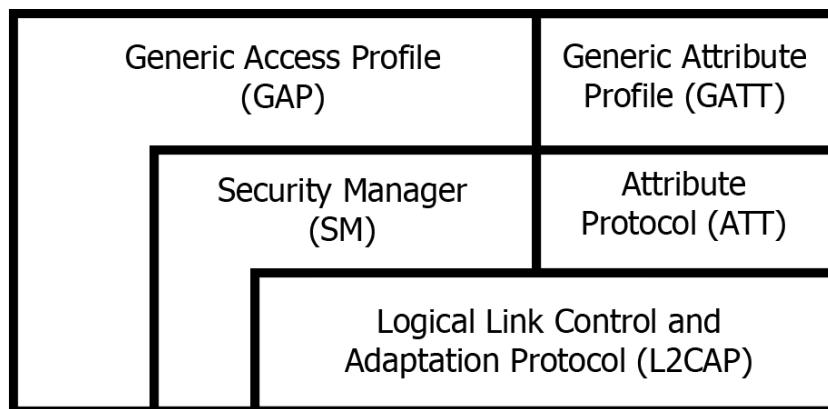


Figura 1.6: Pila del Host

### 1.6.2.1. L2CAP (Logical Link Control and Adaptation Protocol)

La *Logical Link Control and Adaptation Protocol* és la capa encarregada de l'establiment de la connexió lògica, multiplexament de protocols, segmentació i reassemblatge de paquets, i control de flux individual per canal.

La multiplexació de protocols és necessària per aconseguir que BLE sigui flexible. Permet que des de capes superiors s'utilitzin els protocols necessaris, i no només aquells determinats pel estàndard de BLE.

Per limitació física de l'arquitectura existeix una MTU<sup>4</sup> (*Maximum Transmission Unit*) i per tant és necessari segmentar els paquets de les capes superiors que es converteixen en paquets més petits per a les capes inferiors. Aquesta MTU es pot definir (dins dels límits establerts) per cada connexió així flexibilitzant la manera que s'utilitza el protocol per a cada cas.

La L2CAP també és la capa que fa el seguiment de la qualitat de la connexió i dels recursos utilitzats per assegurar-se que les necessitats dels serveis es compleixen.

### 1.6.2.2. SMP

La capa *Security Management Protocol* proveeix de diferents serveis relacionats amb la seguretat de la connexió. Aquests serveis són: autenticació i autorització de dispositius i també integritat, confidencialitat i privacitat de les dades. El protocol té tipus d'emparellament i generació de claus flexible per aconseguir reduir els requeriments de memòria i energia. Els diferents mètodes de seguretat es comentaran en més profunditat a l'apartat [1.10](#).

<sup>4</sup>La MTU és la mida màxima que pot tenir un paquet

### 1.6.2.3. ATT & GATT

El *Attribute Protocol* (ATT d'ara endavant) és el protocol d'aplicació més comú per a BLE i el *Generic Attribute Profile* defineix com utilitzar el protocol per oferir serveis a capes superiors. El ATT és un protocol dissenyat per a dispositius *Low Energy* amb l'objectiu de minimitzar la quantitat de dades transmeses. L'atribut està format per 4 elements, *handle*, UUID, permisos i *value*.

El *handle* fa la distinció única entre els diferents atributs, ocupa 16 bits i és comú que siguin valors seqüencials però no obligatori. És molt útil ja que s'utilitza per referenciar l'atribut amb el mínim de bits possibles. L'UUID (*Universal Unique Identifier*) identifica el tipus d'atribut, aquest numero pot ser de 16 bits si s'utilitza algun que ja estàndarditzat pel SIG o bé en tindrà 128 bits si està definit pel fabricant. En els permisos s'indicarà quin tipus d'accés té el client a la informació (només lectura, lectura i escriptura ...). També pot estar definit si requereix un nivell mínim d'encryptació o si es necessari l'autenticació. Per últim el valor de l'atribut serà on hi ha la informació, la seva interpretació i longitud (512 bytes com a molt) dependrà de l'UUID.

Des de la perspectiva del GATT els dispositius són clients i servidors. Habitualment és el client qui pren la iniciativa demanant dades, tot i així, el servidor també té la capacitat d'iniciar una comunicació per exemple notificant quan un valor ha canviat.

La definició del ATT és massa genèrica per si sola tal que seria comú que per fer el mateix es desenvolupessin múltiples definicions que fossin incompatibles entre sí. Per tal de tenir millor definits els serveis s'utilitza el GATT. El GATT permet definir perfils que agrupen múltiples atributs en un sol servei [11] tal i com es mostra en la figura 1.7.

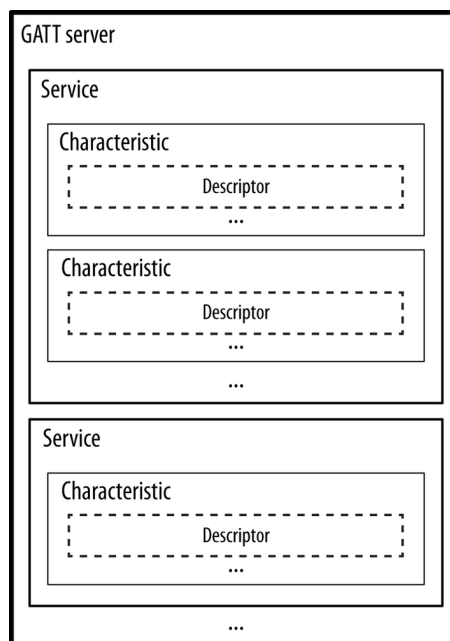


Figura 1.7: Jerarquia de GATT [10]

En un llistat de atributs el GATT identifica els serveis tenint en compte que cada servei comença amb un atribut amb l'UUID 0x2800. Aquest atribut amb UUID 0x2800 s'anomena Declaració de Servei i tots els atributs consecutius fins a una altra declaració de servei formen part d'aquest. En la declaració de servei el camp "valor" indica l'UUID que identifica

de quin servei es tracta.

Cada servei conté característiques [12] definides en els atributs amb UUID 0x2803. Aquests tipus d'atributs s'anomenen Declaració de Característica. El valor de la declaració de característica està format per un nou UUID que identifica la característica i per un *handle*. Aquest *handle* correspon al atribut on hi ha les dades de la característica.

Handle	UUID	Descripció	Valor
0x0100	0x2800	Battery Service	UUID 0x180F
0x0101	0x2803	Characteristic: Battery Level	UUID 0x2A19 Value handle: 0x0102
0x0102	0x2A2B	Battery Value	20
0x0103	0x2800	Custom Temperature Service	UUID 706676c8-3e49...
0x0104	0x2803	Characteristic: Temperature	UUID 0x2A6E Value handle: 0x0105
0x0105	0x2A6E	Temperature Value	25.45
0x0106	0x2803	Characteristic: date/time	UUID 0x2A08 Value handle: 0x0107
0x0107	0x2A08	Date/Time	1/1/1980 12:00

Taula 1.3: Exemple de possibles atributs

En aquest exemple de la taula 1.3 es pot veure que hi ha 2 serveis diferents, marcats en vermell, ja que hi ha 2 UUIDs 0x2800. També hi ha 3 característiques en total, marcades en groc, (una pel primer servei i dues pel segon) ja que hi ha 3 atributs amb UUID 0x2803. Com que els valors corresponents al nivell de bateria i la temperatura estan estandaritzats (veure [14][15], respectivament), no cal especificar a que es refereixen a percentatge de bateria restant i a graus Celsius.

En la declaració de servei de temperatura (*Handle* 0x0103) es pot veure que no forma part de l'estàndard ja que el seu valor té 128 bits. S'ha escollit per a aquest projecte un UUID completament aleatori, en concret el 706676c8-3e49-4ecc-9379-fa9851444e53. Aquest UUID identifica el servei que jo he desenvolupat in ten teoria hauria de ser únic. No es pot saber amb seguretat que algun altre desenvolupador no hagi escollit el mateix UUID i no hi ha una coordinació establerta. Tot i així es considera del tot improbable que hi hagi col·lisions degut a la longitud de 128 bits.

La condició que ha de complir l'UUID és que no sigui XXXXXXXX-0000-1000-8000-00805F9B34FB ja que aquest sufix correspon als que estan reservats segons l'estàndard. En cas de voler tenir un UUID global reservat es pot, amb \$2.500 de cost i també es poden veure tots els que ja s'han reservat per a empreses (veure [16]).

En aquest exemple no n'hi ha cap però les característiques poden tenir descriptors [13] que permeten aportar informació addicional sobre la característica que els precedeix. Aquests, per exemple, serveixen per poder subscriures a rebre notifikacions cada cop que una característica canvia de valor. Els atributs també tenen propietats d'accés que defineixen quines accions es poden prendre. Les propietats es comentaran en un exemple real més endavant 2.3.2..

#### 1.6.2.4. GAP

La *Generic Access Profile* és la capa superior de la pila BLE. Aquesta capa és la que interactua amb la aplicació i per tant ofereix la Interfície de Programació d'Aplicacions (API en anglès) amb la funcionalitat que aporta BLE.

El dispositiu sempre està en un<sup>5</sup> dels quatre rols: Emissor, Observador, Perifèric o Central. Si el rol és emissor, s'enviaran anuncis per poder ser descobert. Si és Observador s'escoltaran els canals per obtenir informació dels anuncis. Un cop s'estigui en una connexió el perifèric serà el node amb menys capacitats de processament o de bateria i és requerirà menys d'ell per mantenir la connexió, per exemple a un rellotge intel·ligent. En canvi el node Central serà el que tingui més recursos com pot ser un telèfon intel·ligent o un ordinador. Amb aquests rols i la especificació del GAP existeix el estàndard que permet els dispositius descobrir-se, connectar-se i emparellar-se entre d'altres.

## 1.7. Anunciaments

Quant els dispositius volen transmetre informació o volen connectar-se entre sí el primer que cal fer és anunciar-se. BLE és molt flexible alhora de configurar els paràmetres d'anunci i permet al desenvolupador adaptar el protocol per a les necessitats que tingui. A continuació, es concretarà quins són aquests paràmetres i com afecten a les prestacions finals del sistema.

Els paquets d'anunci com ja s'ha explicat anteriorment són aquells que serveixen a un dispositiu per donar-se a conèixer i alhora opcionalment transmetre informació [17].

### 1.7.1. Tipus

Hi ha 4 paquets d'aquest tipus i BLE 5 n'afegeix 4 més. Els 4 originals es classifiquen segons si permet connexió i si permeten escaneig, tal i com mostra la taula 1.4.

Connexió	Escaneig	Nom
Si	Si	ADV_IND
Si	No	ADV_DIRECT_IND
No	No	ADV_NONCONN_IND
No	Si	ADV_SCAN_IND

Taula 1.4: Tipus d'anunciaments

El ADV\_IND és el genèric, més comú i permet que qualsevol dispositiu pugui connectar-se. El ADV\_DIRECT\_IND serveix per a avisar un dispositiu específic, per exemple, si un rellotge intel·ligent que es vol connectar al telèfon al que està associat. El ADV\_NONCONN\_IND només indica que existeix i no rebrà informació. Això és útil, per exemple, per permetre localització de balises. El ADV\_SCAN\_IND també està orientat a balises però estarà escoltant per si rep missatges d'escaneig amb els que pot respondre amb poca informació. Això permet una comunicació bidireccional limitada sense necessitat d'establir connexió.

<sup>5</sup>En BLE 5.1 es permet certes combinacions de rols

Tots els paquets excepte el ADV\_DIRECT\_IND permeten transmetre 31 bytes de dades pròpies que han de seguir el format establert que s'explicarà a l'apartat 1.11.

En BLE 5 s'afegeixen 4 paquets més que permeten augmentar la quantitat de dades que es poden transmetre abans d'haver establert una connexió. El ADV\_EXT\_IND és el paquet que s'envia pels canals d'anunci i indica en quin canal secundari s'enviarà l'anunci. Aquest paquet no permet transmetre dades. Els paquets que s'envien per canals secundaris tenen el prefix "AUX" i permeten enviar fins a 254 bytes de dades pròpies. El AUX\_ADV\_IND és el paquet que s'envia per un canal secundari després que s'hagi indicat pel tipus de paquet anterior. En aquest paquet es pot configurar si es permet o no connexió i escaneig però no les dues. El AUX\_SYNC\_IND s'utilitza per indicar que s'enviaran anuncis periòdics pels canals secundaris. D'aquesta manera no cal utilitzar els canals d'anunci tant sovint.

Per poder enviar moltes dades sense necessitat d'establir una connexió es pot fer utilitzant el paquet AUX\_CHAIN\_IND. Amb aquest tipus de paquet un cop s'ha enviat l'anunci pel canal primari es poden enviar múltiples paquets d'anunci per canals secundaris tal i com es veu a la figura 1.8. En cada capçalera s'indica en quin moment i per quin canal es transmetrà el següent paquet.

Un cop explicats els paquets per separat, s'explicarà un exemple de com s'utilitzen aquests paquets per transmetre tanta informació com es vulgui sense necessitat d'establir una connexió. Aquest exemple és molt comú.

Primer de tot l'anunciador envia pels canals d'anunci (habitualment s'envia múltiples vegades per cada canal primari) el ADV\_EXT\_IND. Posteriorment es transmet pel canal secundari el AUX\_ADV\_IND que indica quan i on es transmetrà el següent paquet. Finalment es transmeten tants AUX\_CHAIN\_IND com siguin necessaris fins que s'hagi enviat totes les dades d'anunci que es vulgui.

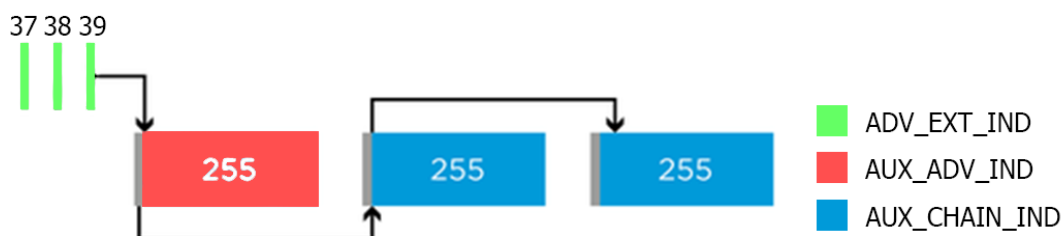


Figura 1.8: Exemple d'anunci estàs

### 1.7.2. Paràmetres

BLE permet seleccionar qualsevol combinació de canals per on transmetre els anuncis. Per consumir menys energia es pot transmetre en un únic canal però això es recomana no fer-ho ja que si el canal és sorollós els dispositius no el podran detectar.

Un paràmetre molt important que es pot configurar és l'interval d'anunci. Aquest valor defineix el temps que passa entre que es transmeten anuncis pel mateix canal tal i com es pot veure en la figura 1.9. La especificació de BLE defineix que pot estar entre 20ms i 10.24s amb salts de 0.625ms. Aquest paràmetre és crític depenent del servei que es vol

oferir. Si el valor és molt alt ajudarà a reduir el consum d'energia considerablement però això augmentarà la latència de descobriment de dispositius.

Un cop s'estableix la connexió ja es poden enviar dades ràpidament però per l'escaneig són necessaris múltiples intervals d'anunci i cal considerar que es possible que el receptor no detecti tots els anuncis. És per això que quan es desenvolupen serveis que interaccionen amb persones no és acceptable haver d'esperar desenes de segons, per tant, habitualment s'utilitzen valors d'entre 100 i 500 ms. En canvi, per a dispositiu que requereixen la mínima latència possible en les dades s'utilitzen valors entre 20 i 50 ms. Finalment, aquells dispositius que proporcionen dades que no varien sovint i que no requereixen interacció utilitzen valors des de 1 fins a 5 segons.

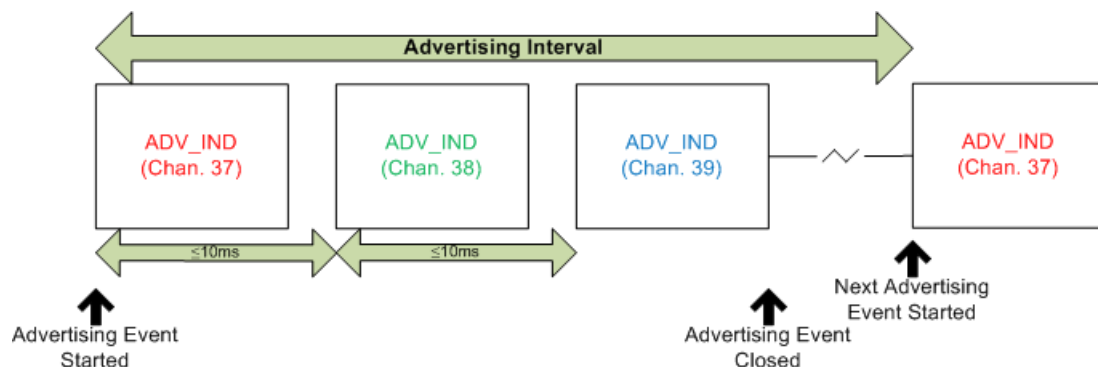


Figura 1.9: Paràmetres d'anunci [18]

Tot i que, es pot definir el valor de l'interval d'anunci cal tenir en compte que segons la especificació de BLE s'aplica un retard pseudo-aleatori que pot ser de fins a 10 ms i afecta als serveis que requereixen molt poca latència. Aquest retard aleatori s'implementa per evitar col·lisions entre dispositius que s'han sincronitzat involuntàriament i tenen el mateix interval o un múltiple entre si.

Tot i que, fins ara s'ha parlat de definir un únic valor d'interval d'anunci, habitualment s'estableix un règim on es va canviant aquest valor depenent de la situació. Quan el dispositiu s'encén o quan s'espera una connexió es disminueix l'interval d'anunci per tal de que la connexió s'estableixi més ràpid. En canvi, quan fa temps que no hi ha cap connexió s'augmenta l'interval d'anunci per reduir el consum d'energia.

## 1.8. Escaneig

Quan un dispositiu es vol anunciar, a part de transmetre anuncis, també ho pot fer fent escaneigs a dispositius que s'han anunciat prèviament. El procés d'escanejar en la especificació de BLE 5 s'anomena *Device Discovery* i es basa en dos tipus, l'escaneig actiu i el passiu. En cas de fer escaneig passiu només es pot obtenir informació a través dels anuncis d'altres dispositius. Si l'escaneig és actiu es poden enviar requeriments d'escaneig per demanar informació addicional a la que hi ha als anuncis.

### 1.8.1. Tipus

Per a l'escaneig actiu hi ha dos paquets 2 en la versió 4.0 i BLE 5 n'afegeix 2 més. El SCAN\_REQ serveix per requerir més informació a un dispositiu que s'està anunciant i que accepta aquest tipus de petició. En el paquet no hi van dades pròpies, només s'hi indica el origen i destí. El SCAN\_RSP és la resposta al paquet anterior, conté l'adreça pròpia de l'anunciador i fins a 31 bytes de dades. L'extensió que aporta BLE 5 defineix 2 nous tipus de paquets, el AUX\_SCAN\_REQ i el AUX\_SCAN\_RSP que tenen la mateixa funcionalitat que els anteriors respectivament. La diferència és que serveixen per a les comunicacions a través de canals secundaris. Conseqüentment la quantitat de dades passa a ser de fins a 254 bytes en la resposta.

### 1.8.2. Paràmetres

L'interval d'escaneig és el temps des de que es comença a escanejar en dos canals consecutius, es pot definir entre 10 ms i 10.24 s. La finestra d'escaneig és el temps en que s'està escoltant a un canal. Duració d'escaneig, és el temps que el dispositiu estarà escanejant. Es pot configurar des de 10 ms fins a 65 segons o escaneig indefinit. El període d'escaneig indica el temps entre que comencen les duracions d'escaneig i que serveix per poder fer pauses en que el dispositiu no escaneja. Tots aquests paràmetres es poden identificar en la figura 1.10.

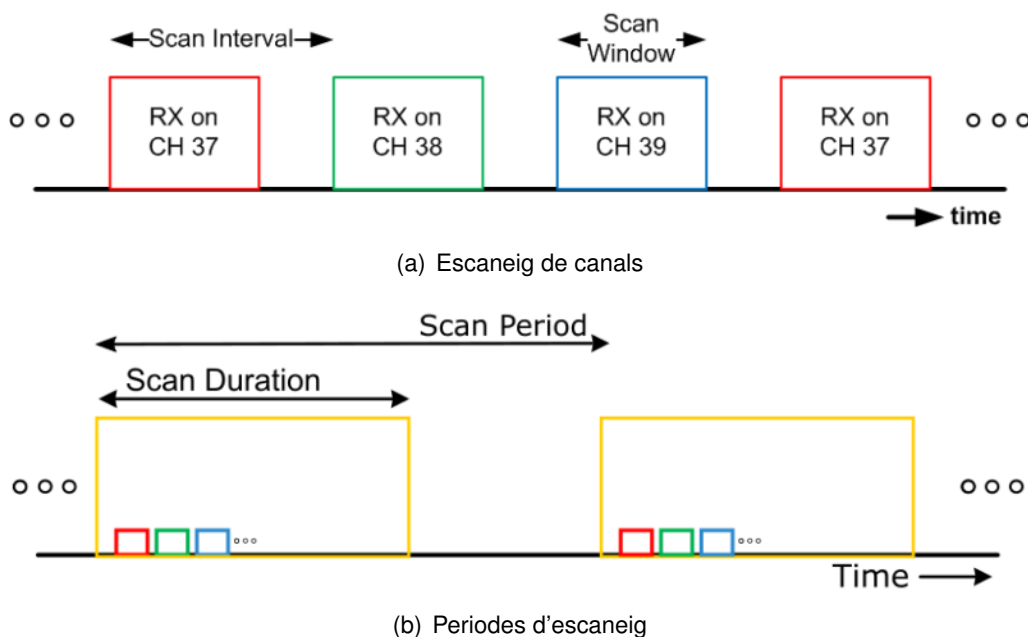


Figura 1.10: Paràmetres d'escaneig

## 1.9. Connexions

S'ha vist com no cal establir una connexió per a transmetre poca informació. Això permet desenvolupar implementacions molt simples per a situacions on es transmet amb molt po-



ca freqüència. Tot i així, BLE també permet transmetre moltes més dades quan s'estableix una connexió.

Un cop s'hagi establert la connexió es podrà transmetre la informació molt eficientment, però establir una connexió suposa un cost energètic considerable. Per tant, a l'hora de definir els paràmetres de la connexió és important evitar, a ser possible, establir connexió cada vegada que es vol transmetre informació.

Com que les connexions estan basades en múltiples esdeveniments en instants determinats es considerarà una connexió síncrona. No hi ha límit de connexions segons l'especificació de BLE i per tant, dependrà de les capacitats del dispositiu amb el que es treballa.

Cal recordar, que per crear una connexió és necessari que un dispositiu tingui el rol GAP de central i un altre amb el rol de perifèric (els anunciadors i observadors no poden iniciar connexions).

El dispositiu central serà el que iniciarà la connexió i el perifèric el que l'acceptarà o no. El central serà sempre el mestre i el perifèric serà l'esclau de la connexió. Per iniciar una connexió el mestre envia un missatge de tipus `CONNECT_IND`<sup>6</sup> amb els paràmetres de la connexió. En aquests paràmetres, tal i com es veurà més endavant en l'apartat 1.9.1., hi ha la informació necessària per determinar el primer esdeveniment de la connexió. En cada esdeveniment, els dos dispositius tenen un temps assignat per transmetre i per rebre dades.

A la figura 1.11 es pot observar el procediment en que dos dispositius estableixen una connexió. Tenim el dispositiu A que inicialment està com a observador i el dispositiu B que s'està anunciant. Cal recordar que els anuncis no es detecten tots ja que els dispositius no sempre estan transmetent i escoltant en el mateix canal primari.

El dispositiu A decideix establir una connexió amb el dispositiu B i envia un requeriment de connexió en l'últim canal en que s'ha escoltat l'anunci. El dispositiu B després de transmetre un anunci per un canal escolta durant un temps determinat en el mateix canal a l'espera de requeriments de connexió. Quan el dispositiu B rep el requeriment decideix acceptar la connexió respon al requeriment de connexió.

En aquest moment el dispositiu A passa a ser el mestre amb els rols Central i Client i el dispositiu B passa a ser l'esclau amb rols de perifèric i servidor. La connexió està establerta i ja es possible transmetre dades entre ambdós dispositius durant els esdeveniments de connexió.

### 1.9.1. Paràmetres

L'interval de connexió és un dels paràmetres més importants en la connexió BLE, estableix com de sovint es comuniquen els dispositius. És el temps que hi ha entre esdeveniments de la connexió i va entre 7.5 ms i 4 s. Com que el que es configuren són preferències es determina l'interval mínim i el màxim que es vol. En cas que es vulgui fixar simplement d'indica el mateix valor per al mínim i al màxim.

BLE permet a l'esclau saltar-se esdeveniments per tal d'estalviar bateria, per exemple si no té dades a transmetre. La latència de l'esclau, que és un dels paràmetres de connexió, defineix la quantitat màxima d'esdeveniments que l'esclau es pot saltar i que la connexió

<sup>6</sup>Aquest paquet en la versió 4 de BLE s'anomena `CONNECT_REQ`

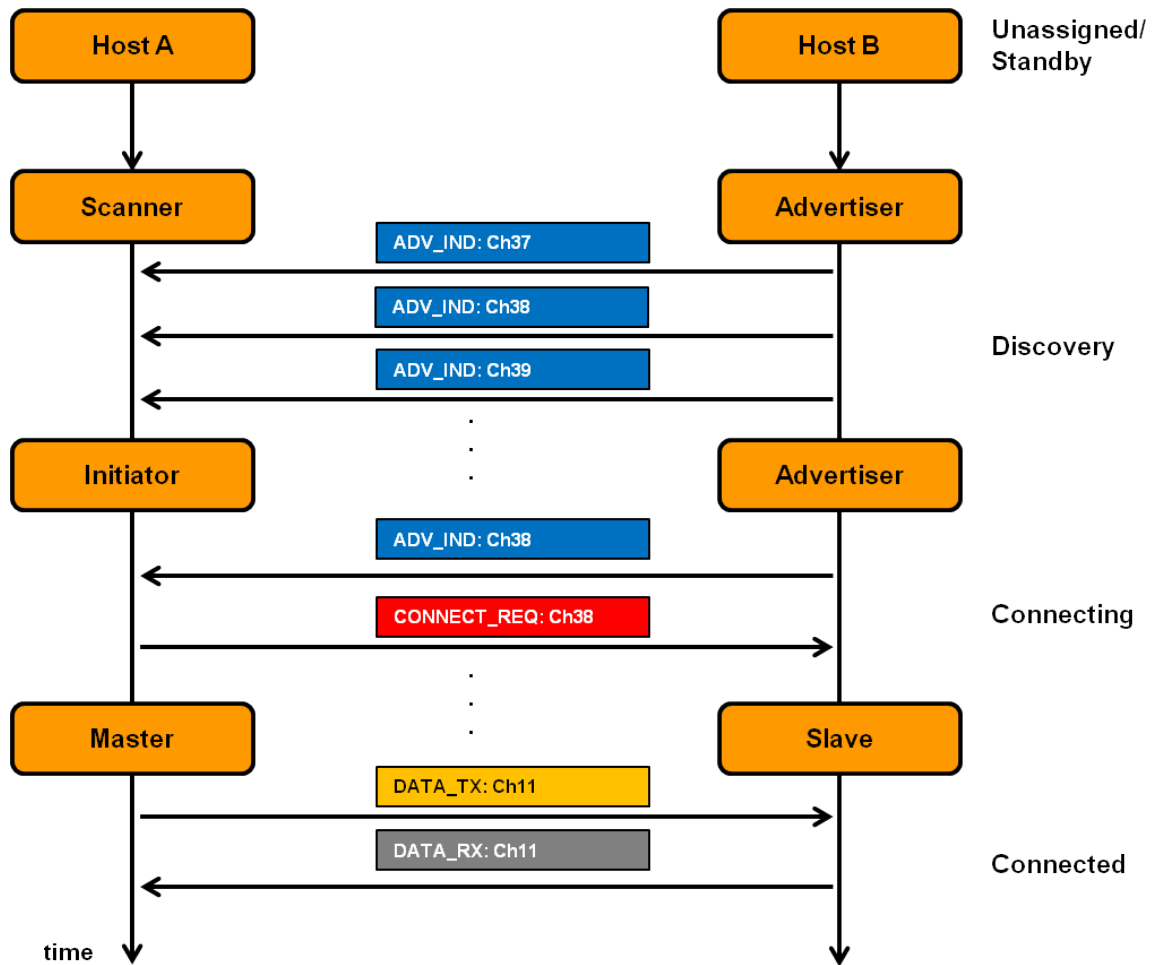


Figura 1.11: Establiment de connexió [19]

segueixi establerta. En la figura 1.12 es pot veure la comparació quan s'utilitza la latència d'esclau. Tot i així, el mestre que és el node que té més bateria, participa en tots els esdeveniments escoltant el canal per si l'esclau transmet. Quan s'utilitza aquest paràmetre permet tenir una connexió amb una latència baixa però amb un consum igualment reduït mentre no s'hagin de transmetre dades molt sovint.

L'últim paràmetre és el temps de supervisió, és el temps màxim acceptable que pot passar sense activitat en la connexió. Si es supera aquest temps es considera que la connexió s'ha acabat i per tant, per comunicar-se s'ha de tornar a iniciar. Això pot passar per exemple en entorns molt sorollosos o més sovint quan els dispositius surten del seu abast i la senyal no és detectable.

## 1.10. Seguretat

Com que l'ús de BLE és molt flexible per poder-lo utilitzar en certes aplicacions es requereix que l'intercanvi de dades sigui segur. Això significa que la connexió tingui resistència

Sense latència d'esclau

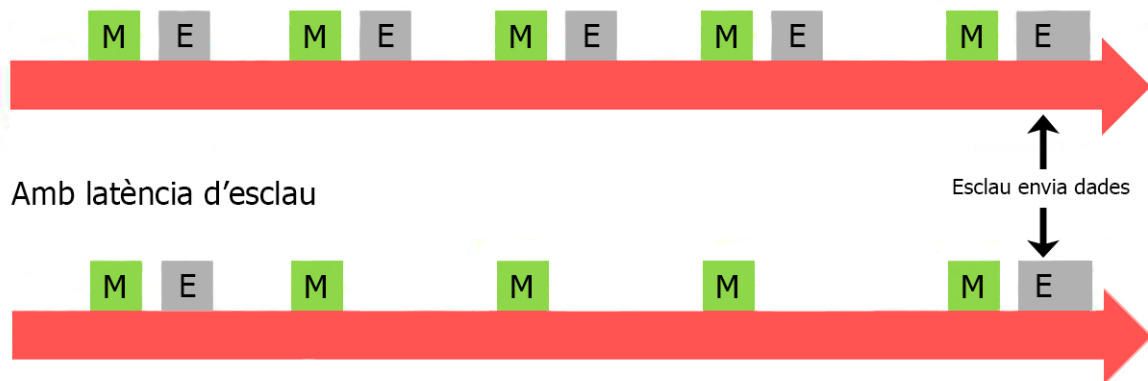


Figura 1.12: Comparació d'esdeveniments amb latència d'esclau [20]

a atacs de *eavesdrop*<sup>7</sup> passiu i d'intermediari<sup>8</sup> (*Man In The Middle* en anglès). BLE utilitza l'encryptació simètrica AES-CCM, però la seguretat rau en com els nodes s'intercanvien la clau.

Hi ha diferents maneres que el desenvolupador pot escollir utilitzar i que varien en seguretat i facilitat d'ús. Existeix el mètode sense seguretat anomenat *Just Works*. Aquest mètode és útil en casos en que no es requereix seguretat o que no es possible tenir-ne per les limitacions del dispositiu amb el que es vol establir connexió.

Un altre mètode es el *Out of Band* aquest permet utilitzar una altre tecnologia per l'intercanvi de claus, com per exemple NFC (comú en auriculars d'alta gama). La seguretat d'aquest mètode depèn completament en el nou sistema per l'intercanvi de claus.

El més habitual és el *Passkey* en que l'intercanvi de claus es basa en una contrasenya de 4 a 6 dígits. Cal tenir en compte que degut al baix número de combinacions dependent de la implementació aquest mètode serà sensible a atacs d'*eavesdropping* passiu.

Finalment un mètode similar al anterior és el de Comparació Numèrica, enlloc de ser l'usuari qui introdueix el codi es genera automàticament i es mostra en els dos dispositius. Cal comprovar que sigui el mateix número en els dos casos i això permet prevenir atacs de *Man in the Middle* però de nou, no protegeix contra el *eavesdropping*.

## 1.11. Format

En aquest apartat s'explicarà com estar format els paquets que s'envien en BLE i de quins camps està format. Cal distingir però entre la versió 4 i la 5 de BLE ja que degut a les noves funcionalitats el format canvia considerablement.

<sup>7</sup>Aquest tipus d'atac permet a l'atacant conèixer el contingut dels missatges que es transmeten.

<sup>8</sup>Aquest tipus d'atac permet modificar la informació que es transmet sense que sigui evident pel transmissor o el receptor.

### 1.11.1. BLE 4.2

En la versió de BLE 4.2<sup>9</sup> els paquets que es transmeten segueixen el format de la figura 1.13.

Preamble (1 octet)	Access Address (4 octets)	PDU (2 to 257 octets)	CRC (3 octets)
-----------------------	------------------------------	--------------------------	-------------------

Figura 1.13: Format del paquet en BLE 4.2 [21]

El preàmbul és una seqüència predefinida i que serveix per sincronisme. L'adreça d'accés identifica a quina connexió pertany aquest paquet, en cas de ser d'anunci i no pertànyer a cap connexió el valor és 0x8E89BED6. Aquest camp serveix al receptor per filtrar els paquets i només tractar aquells que interessa. El CRC (Cyclic Redundancy Check) s'utilitza per detectar errors en el paquet.

En la PDU (Protocol Data Unit) és on hi ha la informació pròpia del paquet i depenent de si és un paquet en un canal d'anunci o en un de dades té un format lleugerament diferent. En la figura 1.14 és poden veure les diferències. Cal destacar que en el cas de ser un anunci tot l'espai és dedica a informació, en canvi, en la PDU d'anunci es reserven 4 bytes per al comprovant d'integritat del missatge (*Message Integrity Check* en anglès).

Header	Payload
2 Bytes	0-37 Bytes

(a) Format PDU d'anunci [21]

Header	Payload	MIC*
2 Bytes	up to 255 Bytes (incl. MIC)	4 Bytes

(b) Format PDU de dades [21]

Figura 1.14: Format PDU

Pel que fa a la Càrrega Útil (*Payload*) serà diferent segons el tipus de paquet. En el cas del ADV\_IND està format per la adreça de l'anunciant seguit de un llistat d'estructures de dades d'anunci (*Advertisement Data Structure*). Aquestes estan formades per la longitud de la estructura, el tipus i les dades en si. Hi ha molts tipus d'estructura que estan definits en el estàndard [22].

En canvi, en el cas del ADV\_DIRECT\_IND simplement s'indica l'adreça de l'anunciador i la del dispositiu al que va dirigit el paquet.

<sup>9</sup>En les versions anteriors a la 4.2 la PDU tenia una mida màxima de 39 bytes.

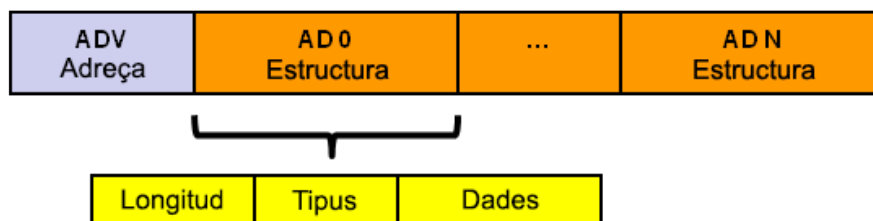


Figura 1.15: Payload de ADV\_IND [21]

### 1.11.2. BLE 5

Els canvis per a BLE 5 corresponen a les noves possibilitats que s'han comentat a l'apartat 1.5.. Primer, la funcionalitat per poder transmetre anuncis als canals secundaris que s'ha comentat a 1.5.0.3. i enumerat els tipus de paquets a 1.7.1..

Payload			
Extended Header Length (6 bits)	AdvMode (2 bits)	Extended Header (0 - 63 octets)	AdvData (0 - 254 octets)

Figura 1.16: Format de la Extended Advertising Payload [23]

Tots els paquets nous de BLE 5 d'anunci es basen en aquest format flexible. L'*Extended Header Length* defineix la longitud del *Extended Header*. L'*AdvMode* defineix una sèrie de valors que identifiquen si l'anunciador accepta connexions o es pot escanejar. L'*Extended Header* indica informació similar a la versió 4.2 com adreça de l'anunciador. Però també indica tot allò necessari per BLE 5, com en quin canal secundari es transmetrà part de la informació del paquet o amb quin tipus de capa física es transmetrà aquesta informació (LE 1M, LE 2M o LE Coded) entre d'altres.



## CAPÍTOL 2. DESENVOLUPAMENT

Un cop vist com funciona el protocol del Bluetooth Low Energy en aquest capítol s'analitzarà el dispositiu que l'implementa. S'explicarà tant els components i parts que formen part de la placa com el programari que s'utilitza per desenvolupar projectes utilitzant-la.

## 2.1. LaunchXL CC1352R1

Per analitzar el protocol BLE i veure les seves característiques s'ha utilitzat el kit LaunchXL pel desenvolupament ràpid del microcontrolador CC1352R1.

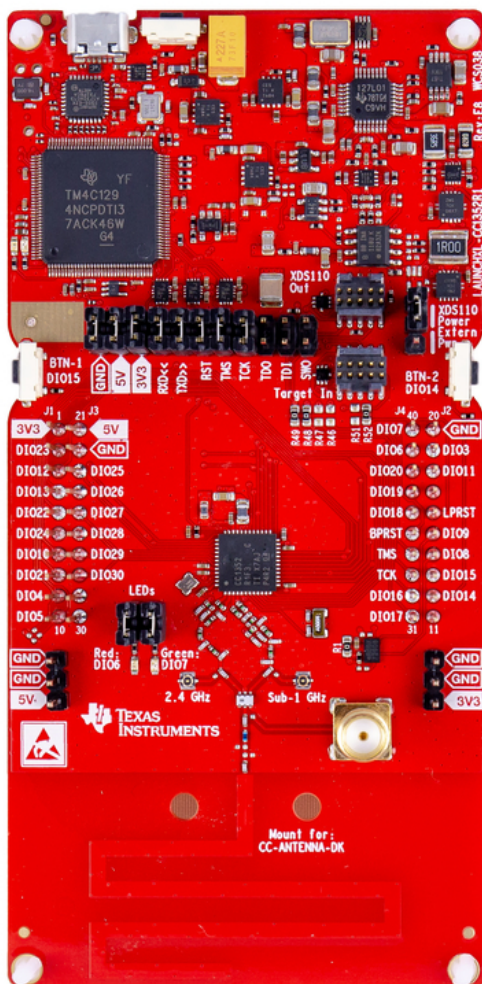


Figura 2.1: Placa [24]

Aquesta placa permet el desenvolupament d'aplicacions en BLE utilitzant el microcontrolador CC1352 de Texas Instruments a continuació es descriuen les seves característiques principals [26].

El dispositiu CC1352R és multiprotocol i multibanda orientat a 2.4 GHz o sub-1GHz que serveix per a Thread, Zigbee, Bluetooth 5 Low Energy, IEEE 802.15.4g i 6LoWPAN. Pel que fa a memòria té 352 KB flash programable, 256 KB de ROM per a protocols i llibreries, 8 KB de Cache SRAM i 80 KB de RAM protegida amb paritat. Pel que fa als perifèrics el més important és el ADC de 12 bits i 8 canals amb freqüència de mostreig de 200 Kmostres/s (multiplexat). També té Relotge de Temps Real (RTC en anglès), acceleradors d'operacions criptogràfiques i generador de números aleatoris. La radio multibanda que té té un receptor amb sensibilitat de -121 dBm per a sub-1GHz i de -110 dBm a 50 Kbps o -105 dBm a 125 Kbps. El transmissor pot transmetre fins a 14 dBm sub-1GHz i 5dBm a 2.4 GHz.

## 2.2. Software

Per al desenvolupament de projectes per a la placa s'ha utilitzat el entorn de desenvolupament Code Composer Studio 9. El CCS el distribueix Texas Instruments i està basat en Eclipse [25], que és de codi obert. Aquest entorn de desenvolupament està orientat al desenvolupament per a processadors incrustats (*embedded*) de les famílies fabricades per Texas Instruments. El gran benefici d'aquest programa es que permet la depuració de programes basats en JTAG que es el cas d'aquesta placa.

Texas Instruments té la plataforma SimpleLink que conté tant les llibreries necessàries per desenvolupar projectes com els recursos necessaris per a l'apranentatge de les tecnologies que es volen implementar. Per poder realitzar projectes amb la placa d'aquest treball és necessari utilitzar el SimpleLink CC13X2 versió 2.30.00.45<sup>1</sup>.

## 2.3. Project 0

El Project 0 és el projecte instal·lat amb que les plaques vénen de fàbrica. Aquest projecte exposa certs serveis a través de BLE i permet fer una comunicació simple entre la placa i un dispositiu mòbil. Aquesta comunicació es bidireccional, des del mòbil es poden controlar els LEDs que té la placa. I en el mòbil es pot veure si qualsevol dels dos botons està pitjat.

Aquest projecte serveix per tenir un bon exemple de com està dissenyada l'arquitectura dels serveis amb les seves característiques amb una relativa simple funcionalitat. A continuació s'analitzaran diferents parts dels atributs.

### 2.3.1. Serveis de Botons i LEDs

En la taula 2.1 es poden veure tots els valors que hi ha tal i com estan en la taula d'atributs del servidor GATT. Per facilitar la visualització de les dades s'han tret els zeros finals dels UUIDs propis però cal recordar que en total tenen 16 parells de caràcters (128 bits en representació hexadecimal) i no només els 9 parells que surten a la taula. Els atributs

---

<sup>1</sup>La versió utilitzada no és la més nova per a que sigui compatible amb la revisió C del xip que s'està utilitzant.



Handle	UUID	Valor	Propietats
0x0022	0x2800	::B0:00:40:51:04:10:11:00:F0	
0x0023	0x2803	0E:24:00::B0:00:40:51:04:11:11:00:F0	0x0E
0x0024	0xF000111104514000B0	0	0x0E
0x0025	0x2803	0E:26::B0:00:40:51:04:12:11:00:F0	0x0E
0x0026	0xF000111204514000B0	0	0x0E
0x0027	0x2800	::B0:00:40:51:04:20:11:00:F0	
0x0028	0x2803	12:29:00::B0:00:40:51:04:21:11:00:F0	0x12
0x0029	0xF000112104514000B0	0	0x12
0x002A	0x2902	00:00	
0x002B	0x2803	12:2C:00::B0:00:40:51:04:22:11:00:F0	0x12
0x002C	0xF000112204514000B0	0	0x12
0x002D	0x2902	00:00	

Taula 2.1: Atributs del Project 0

en si no es poden identificar per tant és necessari tenir en compte la documentació del Project 0. En la documentació hi ha inclosa una taula on es relaciona cada UUID amb el corresponent significat.

UUID	Which ATT Field	Description	Usage
F000 <b>1110</b> -0451-4000-B000-000000000000	Value	LED Service	<i>Service declaration</i>
F000 <b>1111</b> -0451-4000-B000-000000000000	Type/Value	LED0 State	Read state or write 01 or 00.
F000 <b>1112</b> -0451-4000-B000-000000000000	Type/Value	LED1 State	Read state or write 01 or 00.
F000 <b>1120</b> -0451-4000-B000-000000000000	Value	Button Service	<i>Service declaration</i>
F000 <b>1121</b> -0451-4000-B000-000000000000	Type/Value	BUTTON0 State	Read state or subscribe to notifications
F000 <b>1122</b> -0451-4000-B000-000000000000	Type/Value	BUTTON1 State	Read state or subscribe to notifications
F000 <b>1130</b> -0451-4000-B000-000000000000	Value	Data Service	<i>Service declaration</i>
F000 <b>1131</b> -0451-4000-B000-000000000000	Type/Value	String char	Read/Write a long string
F000 <b>1132</b> -0451-4000-B000-000000000000	Type/Value	Stream char	Send or receive WriteNoRsp/Notification

Figura 2.2: Definició dels UUIDs

El primer que cal tenir clar és que en taula 2.2 els valors estan representats amb l'ordre en que es reben i en general els serveis estandarditzats (i també aquest servei propi) utilitza *little-endian* per enviar la informació. Això resulta en que els caràcters hexadecimal queden (per parelles, ja que, dos caràcters representen un byte) ordenats al revés.

Al analitzar els atributs es pot veure com aquells que tenen el UUID 0x2800 en el seu valor només tenen un UUID corresponent al les definicions de serveis de LEDs i de botons. Seguidament, als atribut amb UUID 0x2803 hi ha les definicions de les característiques, en el seu valor hi ha múltiples parts. El primer parell hexadecimal correspon a les propietats. Just després hi ha dos parells hexadecimal que identifiquen el *Handle* del atribut on hi ha la característica. I per últim la resta de caràcters corresponen al UUID que identifica la característica.

### 2.3.2. Propietats

Les propietats d'accés són un valor que identifica quines operacions i procediments es poden fer sobre la característica. El camp té 8 bits per defecte i cada un d'ells identifica una operació o procediment. Es poden combinar qualsevol quantitat d'aquests bits per habilitar múltiples opcions.

Binary	Hex	Property
0000 0001	0x01	Broadcast
0000 0010	0x02	Read
0000 0100	0x04	Write w/o Response
0000 1000	0x08	Write
0001 0000	0x10	Notification w/o ACK
0010 0000	0x20	Indication with ACK
0100 0000	0x40	Signed Writes
1000 0000	0x80	Additional Properties

Taula 2.2: Valors de les propietats

Primer de tot cal destacar la implementació de comunicacions amb o sense reconeixement. D'aquesta manera si preferim reduir la quantitat de missatges per estalviar bateria enlloc d'assegurar-se que s'ha rebut la informació es dona l'opció.

Pel que fa les propietats en el Project 0 les característiques dels LEDs tenen el valor 0x0E que resulta en 0000 1110 per tant, es permet: llegir, escriure i escriure sense resposta. En canvi les característiques del servei de botons té el valor 0x12 que suposa 0001 0010, per tant, es permet lectura i notificació sense reconeixement.

La notificació i indicació són funcions que ajuden a reduir el consum de recursos. Quan volem saber en quin estat estan els botons de la placa contínuament es podria anar llegint el estat de la característica constantment però això suposarien molts missatges i reduiria el temps que els dispositius es poden adormir i així ser més eficients. Per evitar aquest cas es pot configurar la característica tal que sigui la mateixa placa qui envii el missatge automàticament quant el estat de la característica canviï. D'aquesta manera el receptor només cal que escolti els missatges de la placa per tal de saber en quin estat està el botó.

En cas de que es vulgui indicació amb reconeixements és possible configurar-ho d'aquesta manera a través del atribut Configuració de Característica del Client identificat amb el UUID 0x2902 i estandarditzat en la especificació de Bluetooth. Si el valor és 0 no hi ha transmissions, si el valor és 1 s'envien notificacions (sense reconeixement) i si el valor és 2 s'envien indicacions (notificacions amb reconeixement).

El bit d'extensió de propietats, en cas que sigui 1 indica que existeix un atribut: Descriptor Propietats Esteses de Característica. Aquest atribut té el UUID 0x2900 i el que permet és afegir més propietats de les que poden existir amb el espai limitat de 8 bits que hi ha per defecte. Aquest sistema permet afegir fins a 16 bits més per indicar propietats, actualment els dos primers es defineixen segons el estàndard i la resta estan reservats per a ús futur [29].

Aquests 2 primers que estan definits són escriptura fiable i escriptura auxiliar. L'escriptura fiable permet escriure valors amb un procediment diferent al habitual que permet

assegurar-se que el valor que es vol modificar s'ha escrit i no hi han hagut errors. Pel que fa al'escriptura auxiliar permet escriptura al descriptor de característica.

## 2.4. Client BLE

Per poder interactuar amb un servidor de BLE és necessari tenir una implementació de client. Com que els mòbils intel·ligents tots tenen BLE hi ha moltes aplicacions que permeten visualitzar els serveis amb les seves corresponents característiques. També interactuar, escrivint o llegint aquells valors en que estigui permès. Tot i així, no totes les aplicacions permeten rebre notificacions o utilitzar seguretat. I les aplicacions no permeten interactuar directament amb el controlador (a través de HCI), per exemple, no poden canviar la potència de transmissió.

Per poder tenir un control total del sistema l'equip de desenvolupament de programari que proporciona Texas Instruments conté l'aplicació BTool. S'utilitza conjuntament amb el projecte Host Test del mateix equip per convertir la placa en un client BLE. El projecte Host Test permet controlar les diferents capes de la pila BLE del xip a través de la interfície sèrie. Totes les possibles peticions, tant del servidor GAP com els missatges HCI.

### 2.4.1. Btool

L'aplicació BTool utilitza l'API del projecte Host Test per executar les crides a les diferents capes BLE. Primer de tot, BTool necessita saber per quina interfície ha d'interactuar amb la placa. En l'administrador de dispositius de Windows es poden trobar els ports utilitzats per la placa i els paràmetres de la connexió els indica Texas Instruments[30]. Un cop s'inicia la comunicació entre BTool i la placa es pot escanejar, iniciar connexió i descobrir la taula d'atributs del servidor. En tot moment es poden veure tots els paquets que s'envien per la interfície amb la seva corresponent interpretació en la consola. Com que en aquesta consola apareixen tots els esdeveniments de BLE 5, permet veure quan es reben notificacions. Un cop s'ha establert la connexió es pot interactuar amb la taula d'atributs directament.

## 2.5. Crear un Perfil propietari

Per poder implementar una xarxa de sensors amb BLE serà necessari dissenyar un esquema propi en el servidor GATT. Tal i com s'ha comentat anteriorment el servidor GATT està format per perfils que alhora agrupen serveis. Aquests serveis són els que contenen les característiques amb les que s'interactuarà per intercanviar informació. A continuació s'explicarà com es poden implementar perfils en l'entorn de Texas Instruments BLE Stack.

### 2.5.1. Generar Fitxers

Texas Instruments proporciona un Generador de Fitxers [31] que evita haver de començar de zero. En aquesta eina s'han de definir el nom del servei, els noms de les carac-

terístiques amb la seva longitud i les propietats i els corresponents UUIDs. Aquesta eina està pensada per a ser utilitzada junt amb el Project Zero, per tant, aquest serà el projecte base per aquest treball. Un cop s'han generat els fitxers cal copiar-los dins de Application/services en el projecte.

A parts dels fitxers generats (font i capçalera<sup>2</sup>) també cal que l'aplicació cridi a la funció que inicialitza el servei registrant-los al GATT. Aquesta és la funció [Perfil].AddService(..) que cal afegir dins la funció ProjectZero\_init() en la secció Inicialització de Serveis.

Tot i que aquest generador de fitxers és molt útil a l'hora de tenir una estructura bàsica del projecte cal analitzar les parts importants per entendre com funciona per dins i quina relació té cada part amb BLE.

## 2.5.2. Definir el Perfil Propi

Per poder desenvolupar un perfil propi cal seguir els següents passos.

Cada característica necessita: un identificador per poderla classificar, el UUID que té assignat i la longitud de dades que tindrà. Aquests valors es defineixen en el fitxer capçalera del perfil.

```
#define TEMPERATURE_UUID 0x2345
#define temperatureID 0
#define temperatureLen 2

#define humidityID 1
#define humidityLen 2
#define HUMIDITY_UUID 0x3456

#define heartRate 2
#define heartRateLen 2
#define HEARTRATE_UUID 0x4567

#define bloodOxygen 3
#define bloodOxygenLen 2
#define BLOODOXYGEN_UUID 0x5678
```

En aquest cas tots els UUIDs tenen 4 caràcters que significa 16 bits. Però només es poden tenir UUIDs de 16 bits si estan reservats i aquest no és el cas. Aquests 4 caràcters acabaran només sent part del UUID final que tindrà 128 bits un cop s'hagui afeïgit farciment. Això es fa utilitzant la funció TI\_BASE\_UUID\_128 que converteix els 16 bits a 128 amb el format F000XXXX-0451-4000-B000-000000000000 que és el que ve per defecte.

```
CONST uint8_t environmentalServiceUUID[ATT_UUID_SIZE] =
{ TI_BASE_UUID_128 (ENVIRONMENTALSERVICE_SERV_UUID) };

CONST uint8_t temperatureUUID[ATT_UUID_SIZE]=
{ TI_BASE_UUID_128 (TEMPERATURE_UUID) };

CONST uint8_t humidityUUID[ATT_UUID_SIZE]=
{ TI_BASE_UUID_128 (HUMIDITY_UUID) };
```

<sup>2</sup>Els fitxers de capçalera en C s'identifiquen per l'extensió .h i contenen les declaracions de funcions que es poden compartir entre múltiples fitxers font

```

CONST uint8_t heartrateUUID[ATT_UUID_SIZE]=
{TI_BASE_UUID_128(HEARTRATE_UUID)};

CONST uint8_t bloodoxygenUUID[ATT_UUID_SIZE]=
{TI_BASE_UUID_128(BLOODOXYGEN_UUID)};

```

Posteriorment cal assignar l'espai necessari per poder emmagatzemar les dades de les característiques.

```

static uint8_t temperatureVal[temperatureLen] = {0x00};
static uint8_t tempProps = GATT_PROP_READ;

static uint8_t humidityVal[humidityLen] = {0x00};
static uint8_t humidityProps = GATT_PROP_READ;

static uint8_t heartRateVal[heartRateLen] = {0x00};
static uint8_t heartRateProps = GATT_PROP_READ;

static uint8_t bloodOxygenVal[bloodOxygenLen] = {0x00};
static uint8_t bloodOxygenProps = GATT_PROP_READ;

```

Tal i com es pot veure s'assigna tant espai com s'ha definit prèviament. Els valors se'ls inicialitza a 0 i es defineix quines propietats tindran.

Una de les parts més importants de BLE i conseqüentment del servei és la taula d'atributs. A continuació es pot veure com queda la taula definida en el codi<sup>3</sup>.

```

static gattAttribute_t environmentalServiceAttrTbl[] =
{
    { { ATT_BT_UUID_SIZE, primaryServiceUUID },
      GATT_PERMIT_READ,
      0,
      (uint8_t *)&environmentalServiceDecl },
    { { ATT_BT_UUID_SIZE, characterUUID},
      GATT_PERMIT_READ,
      0,
      &tempProps },
    { { ATT_UUID_SIZE, temperatureUUID},
      GATT_PERMIT_READ,
      0,
      temperatureVal },
    { { ATT_BT_UUID_SIZE, characterUUID},
      GATT_PERMIT_READ,
      0,
      &humidityProps },
    { { ATT_UUID_SIZE, humidityUUID},
      GATT_PERMIT_READ,
      0,
      humidityVal
    },
};

```

La taula en codi està formada per un llistat d'atributs, la estructura d'aquests ve definida en el fitxer gatt.h i s'anomena gattAttribute.t. Cada objecte està format per quatre parts que són les següents:

- El UUID, al que cal afegir com a prefix la longitud que té.

<sup>3</sup>En aquest extracte de codi només s'inclouen la declaració del servei i les dues primeres característiques.

- Les propietats que té l'atribut, que determinen quin tipus de permisos.
- El *Handle* que cal deixar a 0 ja que s'assigna internament per al servidor.
- El valor en si del atribut que té una longitud màxima de 512 bytes

En cas de voler assignar més d'una propietat a la característica es pot fer utilitzant l'operador OR (en C correspon al símbol `|`). Un exemple seria el següent:

```
GATT_PERMIT_READ | GATT_PERMIT_WRITE
```

El fitxer `gatt.h`, que s'inclou dins del fitxer font, conté les definicions de tots els possibles permisos amb els seus valors en bits corresponents.

Per llegir o modificar els valors de les característiques és important fer-ho de forma segura i així evitar problemes com el desbordament de memòria intermèdia (*Buffer Overflow* en anglès) que poden ocórrer si s'escriuen valors no vàlids. Es per això que, en cada servei hi ha funcions de *callback*. Les funcions `get` i `set` s'utilitzen per llegir o modificar els valors que s'exposen a través de BLE. Quant en algun punt del codi és necessari interactuar amb aquests valors es farà amb aquestes funcions. Un exemple de la funció `get` és el següent:

```
bStatus_t EnvironmentalService_GetParameter(
    uint8_t param, uint8_t len, void *value )
{
    bStatus_t ret = SUCCESS;
    switch ( param )
    {
        case temperatureID:
            memcpy(value, temperatureVal, len);
        case humidityID:
            memcpy(value, humidityVal, len);
        default:
            ret = INVALIDPARAMETER;
    }
    return ret;
}
```

En aquest exemple, es passa en els arguments: l'identificador de la característica (*param*), la longitud d'aquesta (*len*) i un punter on s'hi copiarà el valor (*\*value*). La funció retorna un valor on indica si s'ha trobat o no la característica.

Per poder implementar notifiacions cal que la característica tingui un *Client Characteristic Configuration Descriptor* o CCCD. El CCCD és un atribut més en la taula que cal posar just després del atribut on hi ha el valor de la característica i en codi quedaria de la següent manera<sup>4</sup>.

```
{
    { ATT_BT_UUID_SIZE, clientCharCfgUUID },
    GATT_PERMIT_READ | GATT_PERMIT_WRITE,
    0,
    (uint8_t *)&bs_BUTTON0Config
},
```

<sup>4</sup>En aquest cas, l'exemple és del CCCD corresponent al boto 0 de la placa que es troba al servei de botons del Project 0

Com que el UUID correspon al de un CCCD cal que sigui 0x2902 que esta a la variable *clientCharCfgUUID*. La longitud del UUID és de dos bytes i s'indica amb la variable *ATT\_BT\_UUID\_SIZE*. La configuració per defecte es que s'enviaran notificacions a aquells dispositius que escriguin un 1 o Indicacions als que escriguin un 2. Però per que el servidor sàpiga que ha de realitzar aquestes operacions cal executar la funció *GATTServApp\_ProcessCharCfg* que ha d'estar en la funció set del servei.





## CAPÍTOL 3. PROJECTE DE SENSORS

En aquest capítol es realitzarà un escenari real on s'implementarà la tecnologia BLE per transmetre dades. Primerament es realitzaran proves per mesurar les capacitats del protocol i de la placa utilitzats.

### 3.1. Experimentació

A fi d'entendre millor com afecten els diferents paràmetres configurables corresponents a BLE i també les prestacions dels perifèrics de la placa, s'han realitzat els següents escenaris.

#### 3.1.1. Abast

Com ja s'ha mencionat anteriorment l'abast teòric que té BLE és considerable comparat amb tecnologies similars. Però, cal entendre que, al estar utilitzant la banda de 2.4 GHz, l'abast de BLE dependrà de l'entorn, on pot haver-hi molta variabilitat d'interferències. Degut a això, en un escenari realista l'abast que es pot assolir pot ser molt diferent del teòric. Les LAUNCHXL-CC1352R1 utilitzades en aquest projecte, per si soles, no són l'eina perfecte per fer proves d'abast. Això es deu, en part, a que no es possible transmetre a la màxima potència permesa per l'estàndard que és de fins a 20 dBm. El límit és de 5 dBm i per defecte només s'utilitzen 0 dBm de potència en transmissió.

Per realitzar un experiment que fos més precís seria avantatjós utilitzar una antena externa més directiva. Enlloc d'analitzar l'abast de la tecnologia en aquest apartat es farà un balanç comparatiu per comprovar les diferències reals d'utilitzar les diferents capes físiques de BLE.

Per a aquest apartat es va realitzar un experiment pràctic en un lloc relativament aïllat i on es pogués tenir una suficient distància amb visibilitat directa. Es va escollir un pàrquing i es va utilitzar la potència per defecte de 0 dBm per poder treballar amb distàncies més curtes.

El procediment per realitzar l'experiment va ser col·locar una de les plaques sobre un cotxe i amb l'altre connectada a una altre ordinador anar retrocedint fins que es perdés la connexió. Per forçar que es perdés la connexió quan ja no es podia establir una comunicació des de la placa que s'anava movent s'anaven enviant peticions de lectura d'atributs.

Els resultats obtinguts, que es poden veure en la figura 3.1, indiquen que la distància màxima d'abast del dispositiu en una capa M2 és de 75 metres, mentre que en una capa S=8 aquesta distància s'extén fins els 135. S'han tingut en compte únicament aquestes capes físiques ja que al no haver-hi interferències significatives i sempre s'ha mantingut la visibilitat directe, els resultats de la capa 1M són similars als de 2M i els de Coded S=2 són similar als de S=8.

Un cop realitzat l'experiment s'han validat, tal i com s'havia comentat anteriorment, els avantatges que aporta la nova capa física Coded que no existia abans de la versió 5.0 de BLE. En aquest rang amb visibilitat directe es pot considerar que BLE podrà assolir

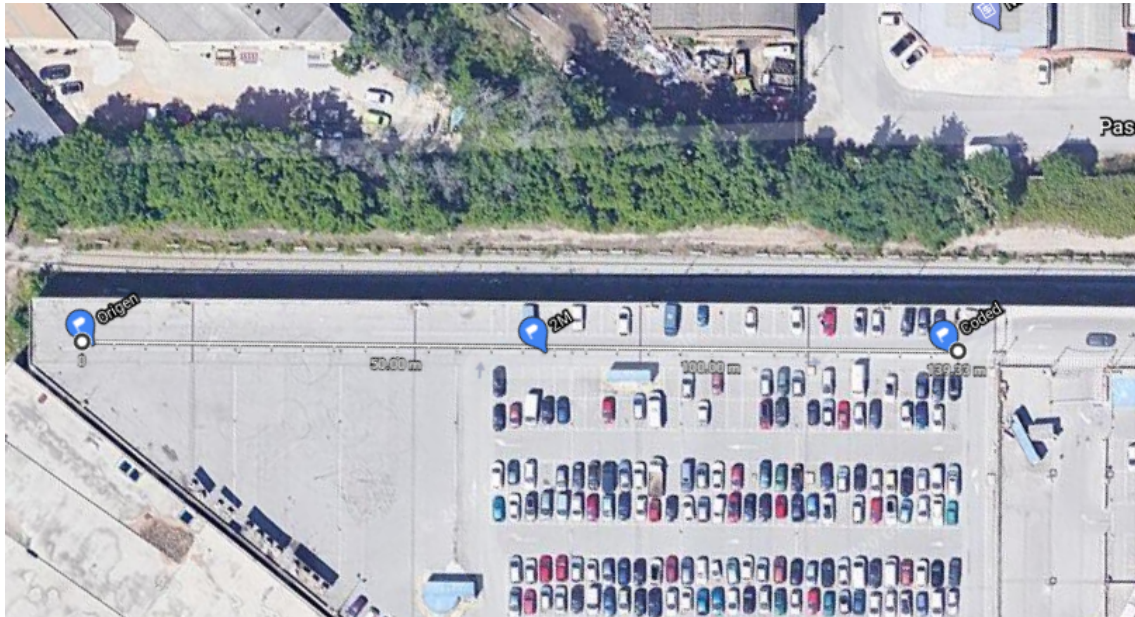


Figura 3.1: Abast real de BLE

cobertura suficient dins les cases a través de parets.

Així doncs, en aquest apartat s'ha vist una de les millores de la capa física Coded, l'abast. Però també cal recordar que on és extremadament útil és en el de resistència a les interferències.

### 3.1.2. Consum d'Energia

El consum dels dispositius que utilitzen BLE és la característica més significativa, ja que, aquesta tecnologia està dissenyada per a consumir la menor quantitat d'energia possible. Les plaques utilitzades tenen punts extraïbles, tal i com es pot veure en la figura 3.2, que permeten desconectar els components que no són essencials per al funcionament del xip CC1352. D'aquesta manera, es pot utilitzar un analitzador de potència per mesurar l'energia consumida pel xip. Aquesta configuració, produeix una mesura molt precisa i permet calcular el cicle de vida del dispositiu que s'està desenvolupant. Per contra, no es poden utilitzar les eines de desenvolupament com la depuració del codi.

Per poder desenvolupar fàcilment un projecte i analitzar els canvis que el codi produeix en l'ús d'energia, existeix una eina anomenada Energy Trace. D'aquesta manera, es pot analitzar comparativament en quin moment s'està consumint l'energia i es poden adaptar els paràmetres de la connexió per observar quin serà l'estalvi que s'aconseguirà. És per això que, es poden prendre les decisions de quins sacrificis són assumibles, per exemple, augmentar la latència a canvi de reduir el consum. Aquesta eina però, no substitueix la solució explicada anteriorment amb l'analitzador de potència ja que l'Energy Trace resulta molt menys precisa.

Per provar el funcionament de l'EnergyTrace s'ha utilitzat durant l'execució en el xip del Project Zero. A la figura 3.3 es pot observar una captura d'un segon de duració del consum aproximat de corrent.

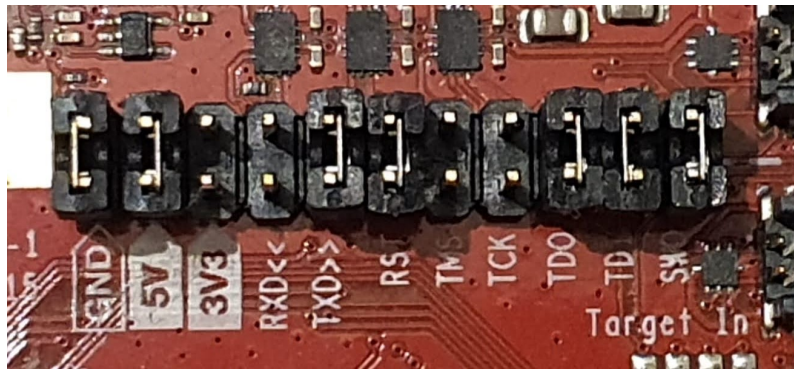


Figura 3.2: Ponts extraïbles de la placa

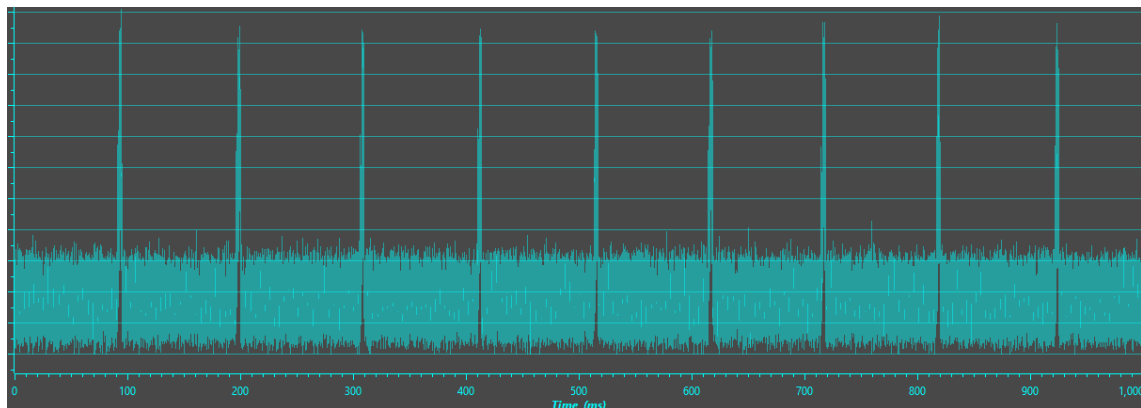


Figura 3.3: Captura d'Energy Trace

És possible visualitzar com hi ha pics de consum cada 100 ms, aquests corresponen als instants en que el dispositiu està enviant els anuncis. Aquests 100 ms corresponen a l'interval d'anunci que s'ha configurat per a aquest projecte.

Un cop s'estableix una connexió amb el dispositiu es realitza una altre captura d'un segon que es pot observar a la figura 3.4.

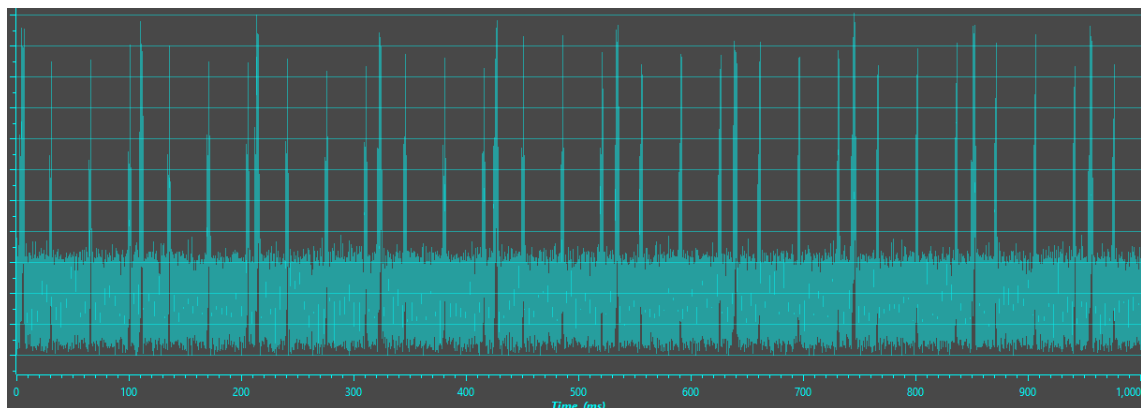


Figura 3.4: Captura d'Energy Trace 2

A la gràfica es possible veure com segueixen estant els pics cada 100 ms però també n'hi ha cada 40 ms. Aquests, darrers, es deuen als esdeveniments de connexió que en aquest

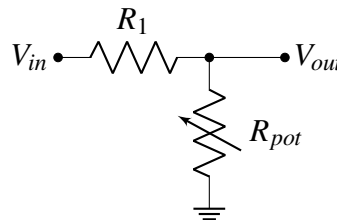
cas estan configurats per a que es produeixen cada 40 ms.

Aquesta eina, per tant, està orientada a facilitar una idea comparativa de en quins instants es consumeix més energia. D'aquesta manera, és més fàcil pel desenvolupador comprovar com els canvis en el codi afecten al consum d'energia.

### 3.1.3. Simulador de sensors

La placa té un convertidor analògic-digital que ens permetrà enviar senyals analògiques un cop s'han mostrejat. En aquest treball no es tractaran directament els sensors sinó que es simularan els senyals que produirien. Com que la placa i el seu ADC treballen a 3.3V es crearà un circuit que permeti controlar un voltatge d'entre 0 i 3.3V.

Aquest circuit es basarà en un divisor de voltatge simple format per una resistència i un potenciòmetre. El potenciòmetre permet canviar la resistència de l'element a través d'un cargol i junt amb el circuit que l'envolta permetrà canviar el voltatge a la sortida.



Aquest circuit segueix l'estructura d'un divisor de voltatge per tant es pot calcular el voltatge a la sortida segons la següent fórmula.

$$V_{in} \cdot \frac{R_{pot}}{R_1 + R_{pot}} = V_{out} \quad (3.1)$$

Per escollir els components cal tenir en compte utilitzar resistències de valor alt per reduir el consum d'energia. S'utilitzarà un potenciòmetre de 10kΩ per tant serà una resistència que es podrà modificar des de 0Ω fins 10kΩ. Per trobar l' $R_1$  que compleixi els requisits queda la següent fórmula.

$$R_1 = 5V \cdot \frac{10k\Omega}{3.3V} - 10k\Omega \approx 5151\Omega \quad (3.2)$$

Per a l' $R_1$  s'utilitzaran resistències de la sèrie E12, els valors que més s'acosten a 5,121Ω són 5.6kΩ i 4.6kΩ. Per assegurar-se que no es superen els 3.3V a la entrada del ADC que podria malmetre el dispositiu s'escull la resistència superior, per tant el circuit final s'ha dissenyat de la següent manera.

Com que s'utilitzaran 4 canals del ADC per mesurar, es necessiten 4 circuits com el que s'ha dissenyat. S'ha implementat el circuit en una protoboard i soldat els components de tal manera que cal connectar el voltatge d'entrada (a 5V) en vermell, terra en negre i finalment quatre cables blaus on hi haurà el voltatge controlat pels quatre potenciòmetres. A la figura 3.5 es pot observar aquesta implementació real del circuit.

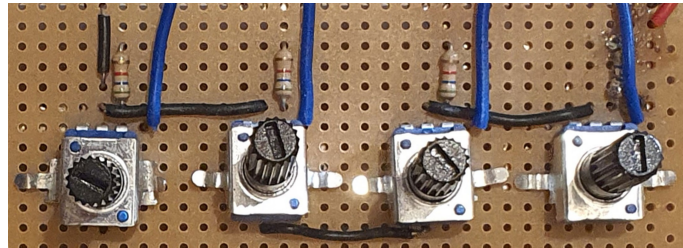
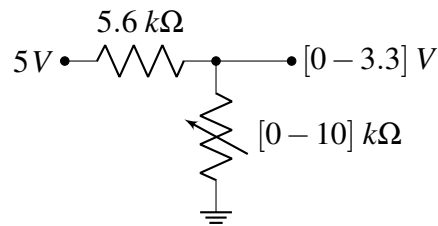


Figura 3.5: Protoboard amb el circuit

## 3.2. Lectura del ADC

Tal i com s'ha comentat anteriorment la placa que s'està utilitzant conté un ADC que s'utilitza per llegir valors de voltatge.

En aquest exemple, l'objectiu és aconseguir la lectura dels valors del voltatge d'un pin de la placa. El següent codi permet fer una mesura del canal que s'indiqui a través del argument i retorna el resultat en miliVolts.

```
static uint16_t takeMeasurement(uint_least8_t adcIndex){
    ADC_Handle adc;
    ADC_Params params;
    ADC_init();
    ADC_Params_init(&params);
    adc = ADC_open(adcIndex, &params);
    if (adc == NULL){
        Log_info0("ADC start Failed");
        while(1);
    }
    int_fast16_t res;
    uint16_t adcValue;
    res = ADC_convert(adc, &adcValue);

    if (res == ADC_STATUS_SUCCESS) {
        Log_info1("ADC result %d mV", adcValue);
    }
    else {
        Log_info0("ADC status failure");
    }
    ADC_close(adc);
    return adcValue;
}
```

Finalment cal canviar el codi de BLE per a que quan es llegeixin els atributs es retorni el valor mesurat al ADC. Per tant cal canviar la funció `environmentalService_ReadAttrCB` tal i com es veu a continuació.



```
if(!memcmp(pAttr->type.uuid, temperatureUUID, pAttr->type.len))
{
    uint16_t adcValue = takeMeasurement(Board_ADC0);
    *pLen = (uint16_t)temperatureLen;
    memcpy(pValue, &adcValue, *pLen);
}
else if(!memcmp(pAttr->type.uuid, humidityUUID, pAttr->type.len))
{
    uint16_t adcValue = takeMeasurement(Board_ADC1);
    *pLen = (uint16_t)humidityLen;
    memcpy(pValue, &adcValue, *pLen);
}
```

### 3.3. Aplicació mòbil

Per veure un exemple simplificat d'un client que consumís els serveis oferits per la placa s'ha desenvolupat una aplicació per a Android. Aquesta aplicació llegeix continuament els valors dels quatre atributs que té la placa relacionas amb el sensors. El codi de la app es pot trobar a [27].

Per poder interactuar amb la placa el primer que cal és tenir en compte l'adreça del dispositiu i els identificadors, tant del servei, com dels atributs. Aquests es poden veure a la taula 3.1.

Identificador	Valor
deviceAddress	00:81:F9:4A:4D:B3
serviceUUID	f0001234-0451-4000-b000-00000000
temperatureUUID	f0002345-0451-4000-b000-000000000000
humidityUUID	f0003456-0451-4000-b000-000000000000
heartRateUUID	f0004567-0451-4000-b000-000000000000
bloodOxygenUUID	f0005678-0451-4000-b000-000000000000

Taula 3.1: Tipus d'anunciaments

Les dades que s'estan transmetent per cada atribut corresponen al valor del voltatge. Tot i així, per caracteritzar aquestes dades a l'aplicació s'interpreten com si fossin valors plausibles de les mesures que s'estan prenent. Igualment, es mostra tant una barra de progrés com el valor real de milivolts per cada mesura de la placa. A continuació es pot observar una captura de la aplicació en funcionament.

### 3.4. Escenari

Finalment, un cop desenvolupat el circuit per simular sensors, el codi necessari per mostrar i transmetre els valors i una aplicació per rebre aquests valors, l'escenari és el següent.

El circuit que simula els 4 sensors es connecta als pins de la placa que tenen ADC segons el manual de la placa que es pot trobar a [28] seguint la taula 3.2.

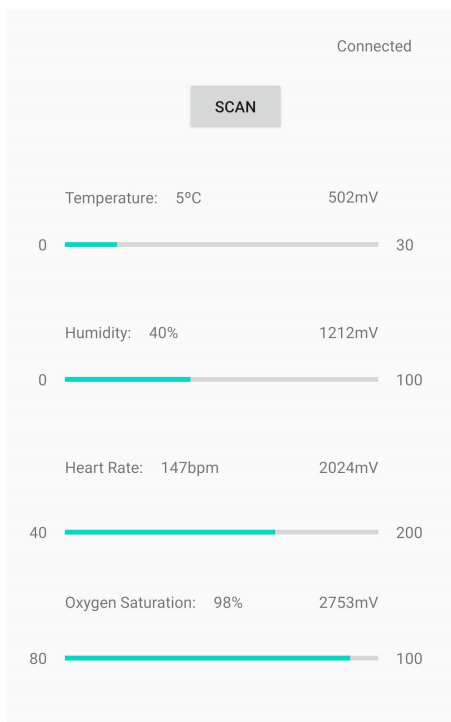


Figura 3.6: Ports placa

ADC	DIO	PIN
0	23	2
1	24	6
2	25	23
3	26	24

Taula 3.2: Taula dels pins amb ADC

Les connexions des del circuit fins la placa es fan a través dels ports que proporciona la placa a la seva part de darrera que es veuen a la figura 3.7.

Un cop fetes les connexions, cal executar el codi de la placa i clicar el botó d'escanejar de l'aplicació. Cal tenir en compte que durant el procés de d'escaneig degut a la configuració de BLE sempre és possible (tot i que poc probable) que el mòbil no descobreixi la placa un cop transcorregut el temps establert. En aquest cas, cal tornar a clicar el botó d'escaneig. Un cop l'aplicació està escanejant es poden canviar les posicions dels potenciòmetres i s'observa com els valors corresponents en el mòbil canvien amb poca latència.

### 3.5. Continuitat del projecte

Un cop realitzat el projecte i entès tant la tecnologia Bluetooth Low Energy com l'entorn de desenvolupament, hi ha aspectes en els que és possible aprofunditzar més. Aquestes es podrien abarcar en un altre futur treball.

Tal i com s'ha esmentant quant s'ha realitzat l'estudi del consum d'energia utilitzant Energy Trace, els valors obtinguts no es poden considerar absoluts. És per això que per poder

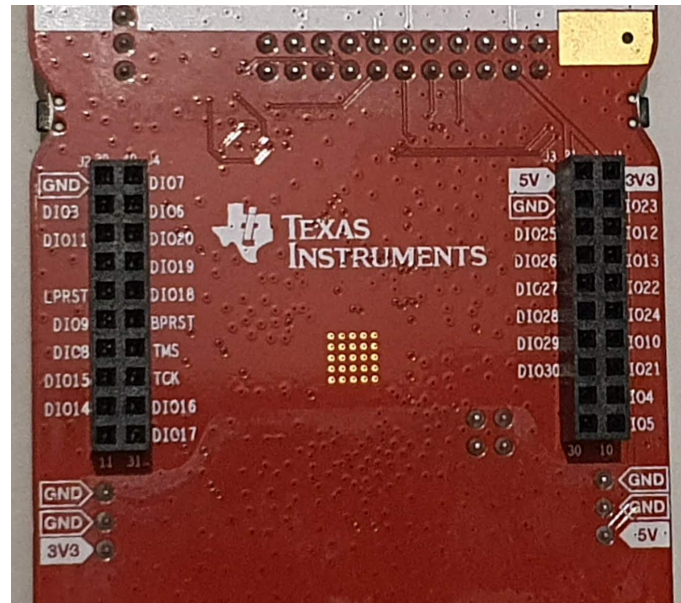


Figura 3.7: Ports placa

determinar el temps de vida amb diferents configuracions del BLE s'hauria de mesurar el consum amb un oscil·loscopi mentre la placa esta configurada amb la font d'alimentació externa.

La implementació pràctica de BLE que s'ha realitzat no utilitza cap servei propietari. Seria interessant implementar un servei estandaritzat i tenir algun dispositiu comercial que pogués connectar-se a la placa.

Finalment tot i haver-se desenvolupats projectes que permeten mesurar els voltatges dels ADCs i emetre notifikacions, per part de BLE, no s'ha aconseguit que sigui alhora.



# CONCLUSIONS

Aquest projecte serveix per considerar si BLE és la tecnologia adequada per a les aplicacions de mesures mediambientals o biològiques.

## Tassa de dades necessària

Un dels dubtes al inici del treball era si aquesta tecnologia tenia una capacitat de transmissió suficient per a xarxes de sensors. Tot i que Bluetooth Low Energy té bastant menys capacitat per transmetre dades, les dades necessàries en aquest tipus de xarxes acostumen a ser molt reduïdes. La tecnologia BLE no es capaç de transmetre vídeo o àudio en temps real però és suficient per transmetre fluxos de dades de sensors amb mesures biològiques o mediambientals.

## Abast

Com s'ha comentat l'abast de la tecnologia BLE ve limitat tant per la distància entre transmissor i receptor però també per les interferències. En mesures biològiques es pot considerar que el transmissor i el receptor estaran relativament a prop per tant no hi haurà problemes d'abast. En canvi per a mesures ambientals si considerem instal·lacions privades en habitatges, cal tenir en compte els obstacles que s'oposen al senyal. També cal considerar la interferència més que probable de les tecnologies amb les que BLE comparteix espectre amb WiFi, altres dispositius intel·ligents i amb Bluetooth Clàssic mateix. Per combatre aquests desavantatges serà suficient augmentant la potència de transmissió tot i que reduirà la vida de la bateria en cas que en tingui.

## Energia Consumida

BLE és dels protocols que permet consumir menys energia per a transmetre informació. Això només serà un avantatge quan els sensors estiguin alimentats per bateria i la transmissió sigui el que consumeix més energia de tot el sistema. Si els sensors tenen pantalles, s'alimenten de la instal·lació elèctrica, fan un processament de les dades dels sensors o emmagatzemen les dades localment, és possible que el consum de la transmissió no sigui significatiu. En aquests casos no s'ha d'escollir el protocol de transmissió segons el consum ja que el temps entre recàrregues no es veu afectat pel protocol.

## Penetració del mercat

És evident que la presència de BLE en tots els mòbils intel·ligents facilita l'ús i simplifica la arquitectura d'un desplegament. En cas de no utilitzar BLE no hi ha cap tecnologia de baix consum que es pugui connectar directament amb els telèfons dels usuaris. Per tant, és necessari un dispositiu entremig entre els sensors i els usuaris que s'anomena concen-

trador o *hub*. En architectures simples suposa un cost extra però si es fa un desplegament extens, el cost és marginal.

En el cas de connexió directa fins al terminal del usuari BLE és perfecte ja que es troba en tots els telèfons intel·ligents. En total es preveu la producció de 7.500 milions de dispositius entre 2020 i 2024 amb un creixement del 26% anual segons el SIG[32].

# BIBLIOGRAFIA

- [1] Bluetooth Special Interest Group “Bluetooth Core Specification” (Revision 5.2)
- [2] UNIVERSITAT POLITÈCNICA DE CATALUNYA; TERMCAT, CENTRE DE TERMINOLOGIA; ENCICLOPÈDIA CATALANA. Diccionari de telecomunicacions [en línia]. Barcelona: TERMCAT, Centre de Terminologia, cop. 2017. (Diccionaris en Línia) (Ciència i Tecnologia) <https://www.termcat.cat/ca/diccionaris-en-linia/235>
- [3] Honkanen, M.; Lappetelainen, A.; Kivekas, K. (2004). *Low end extension for Bluetooth*. 2004 IEEE Radio and Wireless Conference 19–22 September 2004. IEEE. pp. 199–202. [Consulta: Octubre 2020] <https://ieeexplore.ieee.org/document/1389107> 4
- [4] Arxiu del lloc web de MIMOSA. [Consulta: Octubre 2020] <https://web.archive.org/web/20160804035320/http://www.mimosa-fp6.com/> 4
- [5] Milllores de BLE 5 respecte BLE 4. [Consulta: Octubre 2020] [https://www.bluetooth.com/wp-content/uploads/2019/03/Bluetooth\\_5-FINAL.pdf](https://www.bluetooth.com/wp-content/uploads/2019/03/Bluetooth_5-FINAL.pdf) 7
- [6] Casals Ibáñez, Lluís & Mir Masnou, Bernat & Vidal Ferré, Rafael & Gomez, Carles. (2017). *Modeling the energy performance of LoRaWAN*. Sensors. 17. 2364. 10.3390/s17102364. [Consulta: Octubre 2020] <https://www.researchgate.net/publication/320435869>
- [7] Latré, Benoît; De Mil, Pieter; Moerman, Ingrid; Dierdonck, Niek; Dhoedt, Bart; De-meester, Piet. (2005). [Consulta: Octubre 2020] [Maximum Throughput and Minimum Delay in IEEE 802.15.4](#). 3794. 866-876. 10.1007/11599463\_84.
- [8] Gomez, C.; Oller, J.; Paradells, J. [Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology](#). Sensors 2012, 12, 11734-11753. [Consulta: Octubre 2020] 11
- [9] Link Layer States [Consulta: Octubre 2020] [https://dev.ti.com/tirex/explore/content/simplelink\\_academy\\_cc13x2\\_26x2sdk\\_4\\_10\\_01\\_00/modules/ble5st-link-layer\\_xi](https://dev.ti.com/tirex/explore/content/simplelink_academy_cc13x2_26x2sdk_4_10_01_00/modules/ble5st-link-layer_xi), 11
- [10] O'Reilly online learning Chapter 4. GATT (Services and Characteristics) [Consulta: Octubre 2020] <https://www.oreilly.com/library/view/getting-started-with/9781491900550/ch04.html> xi, 13
- [11] Bluetooth Special Interest Group “GATT Services Specification” [Consulta: Octubre 2020] <https://www.bluetooth.com/specifications/gatt/services/> 13
- [12] Bluetooth Special Interest Group “GATT Characteristics Specification” [Consulta: Octubre 2020] <https://www.bluetooth.com/specifications/gatt/characteristics/> 14
- [13] Bluetooth Special Interest Group “GATT Descriptors Specification” [Consulta: Octubre 2020] <https://www.bluetooth.com/specifications/gatt/descriptors/> 14

- [14] Especificació de la característica de nivell de bateria.  
[Consulta: Octubre 2020]  
[https://www.bluetooth.com/wp-content/uploads/Sitecore-Media-Library/Gatt/Xml/Characteristics/org.bluetooth.characteristic.battery\\_level.xml](https://www.bluetooth.com/wp-content/uploads/Sitecore-Media-Library/Gatt/Xml/Characteristics/org.bluetooth.characteristic.battery_level.xml) 14
- [15] Especificació de la característica de temperatura. [Consulta: Octubre 2020]  
<https://www.bluetooth.com/wp-content/uploads/Sitecore-Media-Library/Gatt/Xml/Characteristics/org.bluetooth.characteristic.temperature.xml> 14
- [16] Llistat de UUIDs reservats per a empreses. [Consulta: Octubre 2020]  
<https://www.bluetooth.com/specifications/assigned-numbers/> 14
- [17] Blog oficial sobre els anuncis en BLE. [Consulta: Octubre 2020]  
<https://www.bluetooth.com/blog/bluetooth-low-energy-it-starts-with-advertising/> 15
- [18] Diagrama dels paràmetres d'anunci. [Consulta: Octubre 2020]  
[https://dev.ti.com/tirex/explore/content/simplelink\\_academy\\_cc13x2\\_26x2sdk\\_4\\_10\\_01\\_00/modules/ble5stack/parameters](https://dev.ti.com/tirex/explore/content/simplelink_academy_cc13x2_26x2sdk_4_10_01_00/modules/ble5stack/parameters) xi, 17
- [19] Diagrama del procés de connexió. [Consulta: Octubre 2020]  
<https://microchipdeveloper.com/wireless:ble-link-layer-connections> xi, 20
- [20] Diagrama de la latència d'esclau. [Consulta: Octubre 2020]  
[http://software-dl.ti.com/lprf/simplelink\\_cc2640r2\\_latest/docs/blestack/ble\\_user\\_guide/html/ble-stack-3.x/gap.html#connection-parameters](http://software-dl.ti.com/lprf/simplelink_cc2640r2_latest/docs/blestack/ble_user_guide/html/ble-stack-3.x/gap.html#connection-parameters) xi, 21
- [21] BLUETOOTH SPECIFICATION Version 4.2 [Vol 6, Part B] apartat 2  
[Consulta: Octubre 2020] xi, 22, 23
- [22] Estructures de les dades d'anunci possibles  
[Consulta: Octubre 2020]  
<https://www.bluetooth.com/specifications/assigned-numbers/generic-access-profile/> 22
- [23] BLUETOOTH CORE SPECIFICATION Version 5.2 — Vol 6, Part B apartat 2.3.4 xi, 23
- [24] Fotografia del LaunchXL CC1352R [Consulta: Octubre 2020]  
[http://www.ti.com/diagrams/med.launchxl-cc1352r1\\_launchxl-cc1352r1.jpg](http://www.ti.com/diagrams/med.launchxl-cc1352r1_launchxl-cc1352r1.jpg) xi, 25
- [25] Web del entorn de desenvolupament Eclipse. [Consulta: Octubre 2020]  
<https://www.eclipse.org/ide/> 26
- [26] CC1352R SimpleLink Datasheet [Consulta: Octubre 2020]  
<https://www.ti.com/tool/LAUNCHXL-CC1352R1#technicaldocuments> 25
- [27] Respositori amb l'aplicació Android desenvolupada.  
<https://github.com/garretaserra/sensorReader> 40
- [28] Manual del LaunchPad CC1352R [Consulta: Octubre 2020]  
<https://www.ti.com/lit/pdf/swru525> 40

- [29] Characteristic Extended Properties [Consulta: Octubre 2020]  
[https://www.bluetooth.com/wp-content/uploads/Sitecore-Media-Library/Gatt/Xml/Descriptors/org.bluetooth.descriptor.gatt.characteristic\\_extended\\_properties.xml](https://www.bluetooth.com/wp-content/uploads/Sitecore-Media-Library/Gatt/Xml/Descriptors/org.bluetooth.descriptor.gatt.characteristic_extended_properties.xml)  
28
- [30] Configuració del port sèrie [Consulta: Octubre 2020]  
[http://dev.ti.com/tirex/explore/content/simplelink\\_academy\\_cc13x2\\_26x2sdk\\_4\\_10\\_01\\_00/modules/ble5sta](http://dev.ti.com/tirex/explore/content/simplelink_academy_cc13x2_26x2sdk_4_10_01_00/modules/ble5sta)  
29
- [31] Generador de fitxers per a nous serveis [Consulta: Octubre 2020]  
[http://dev.ti.com/tirex/explore/content/simplelink\\_academy\\_cc13x2\\_26x2sdk\\_4\\_10\\_01\\_00/modules/ble5sta](http://dev.ti.com/tirex/explore/content/simplelink_academy_cc13x2_26x2sdk_4_10_01_00/modules/ble5sta)  
service-generator 29
- [32] Anàlisi del mercat de Bluetooth [Consulta: Octubre 2020]  
[https://www.bluetooth.com/wp-content/uploads/2020/03/2020\\_Market\\_Update-EN.pdf](https://www.bluetooth.com/wp-content/uploads/2020/03/2020_Market_Update-EN.pdf)  
44



# **APÈNDIXS**





# APÈNDIX A. TAULA DE ATRIBUTS DEL PROJECT 0

Prova d'apèndix

Handle	UUID	Valor	Propietats
0x0022	0x2800	::B0:00:40:51:04:10:11:00:F0	
0x0023	0x2803	0E:24:00::B0:00:40:51:04:11:11:00:F0	0x0E
0x0024	0xF000111104514000B0	0	0x0E
0x0025	0x2803	0E:26::B0:00:40:51:04:12:11:00:F0	0x0E
0x0026	0xF000111204514000B0	0	0x0E
0x0027	0x2800	::B0:00:40:51:04:20:11:00:F0	
0x0028	0x2803	12:29:00::B0:00:40:51:04:21:11:00:F0	0x12
0x0029	0xF000112104514000B0	0	0x12
0x002A	0x2902	00:00	
0x002B	0x2803	12:2C:00::B0:00:40:51:04:22:11:00:F0	0x12
0x002C	0xF000112204514000B0	0	0x12
0x002D	0x2902	00:00	