ESCUELA TÉCNICA SUPERIOR DE

INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD DE MÁLAGA

PROYECTO FIN DE CARRERA

DESARROLLO DE UN ENTORNO PARA LA CONFIGURACIÓN Y MONITORIZACIÓN DE REDES ZIGBEE/802.15.4

INGENIERÍA DE TELECOMUNICACIÓN

MÁLAGA, *2010* *SERGIO LILLO MORENO*

ESCUELA TÉCNICA SUPERIOR DE

INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD DE MÁLAGA

Titulación: Ingeniería de Telecomunicación

Reunido el tribunal examinador en el día de la fecha, constituido por:

D./Dª. _____

D./Dª. _____

D./Dª. _____

para juzgar el Proyecto Fin de Carrera titulado:

DESARROLLO DE UN ENTORNO PARA LA CONFIGURACIÓN Y MONITORIZACIÓN DE REDES ZIGBEE/802.15.4

del alumno/a D./Dª. *Sergio Lillo Moreno*

dirigido por D./Dª. *Eduardo Casilari Pérez*

ACORDÓ POR _____ OTORGAR LA CALIFICACIÓN DE _____

Y, para que conste, se extiende firmada por los componentes del tribunal, la presente diligencia

Málaga, a _____ de _____ de _____

El/La Presidente/a El/La Vocal El/La Secretario/a

Fdo.: _____ Fdo.: _____ Fdo.: _____

ESCUELA TÉCNICA SUPERIOR DE

INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSIDAD DE MÁLAGA

DESARROLLO DE UN ENTORNO PARA LA CONFIGURACIÓN Y MONITORIZACIÓN DE REDES ZIGBEE/802.15.4

REALIZADO POR:

Sergio Lillo Moreno

DIRIGIDO POR:

Eduardo Casilari Pérez

DEPARTAMENTO DE: *Tecnología Electrónica*

TITULACIÓN: Ingeniería de Telecomunicación

PALABRAS CLAVE: ZigBee, 802.15.4, CC2420, MSP430

RESUMEN: En este proyecto se realiza un estudio sobre las redes ZigBee/802.15.4 y se desarrolla una plataforma desde la cual configurar los parámetros básicos de una red y monitorizar tanto la topología de red existente en cada momento, como el intercambio de mensajes producido entre los distintos dispositivos que la componen.

Málaga*, FEBRERO 2010*

Agradecimientos

Quisiera empezar agradeciendo a mis padres, Juan Antonio y Pilar, el apoyo constante y la paciencia durante todos estos años. Sin su cariño, ayuda y consejo, jamás habría conseguido llegar hasta aquí.

En segundo lugar, me gustaría dar las gracias a mi tutor, Eduardo Casilari. Su rápida respuesta ante cualquier duda surgida, sus ánimos y muestras de interés hacia este proyecto me han facilitado mucho el trabajo durante los últimos meses.
También quiero incluir a José Manuel Cano. Gracias por atenderme y sacarme de cada uno de los atascos que me he ido encontrando a lo largo del desarrollo de este proyecto.

Agradecer a mis hermanos, Ignacio y María, y a Nines, sus constantes muestras de cariño y ánimo durante todos estos años de carrera. ¡Sí hermano, ya es hora de empezar a producir!

A mis tíos Jorge e Isa, que como personas del gremio que son, me han entendido y aconsejado mejor que nadie. Gracias además por todos los esfuerzos que realizáis para introducirme en este mundillo.

Gracias de forma muy especial a mi novia, Rocío, sin lugar a dudas el apoyo más importante durante ya casi cinco años. Su insistente lucha por centrarme en estudiar (casi lo conseguiste) son la causa de que esté ahora escribiendo esto y no tomando apuntes de cualquier asignatura de la carrera. ¡Mil gracias cariño!

A mis amigos de siempre: Guille, Carlos, Iago, Javi Arias, Javi "pijo", Elías, Ismael, David, Álvaro Olmedo y "Pape". ¡Porque no todo va a ser estudiar... ni lo será trabajar!

A mis amigos de teleco: Curro, Troya, Juanmi, Lidia, María, Javi "cordobés", Dani, Elvira, Jose, Patri, Miguel, Iván, Javi Fernández, Jorge, Miguel "basam", Moisés y David. Siempre nos quedará el recuerdo de lo que fue "Rechazo banda".
No quiero olvidarme de "Sebas", Jesús "Xuski", Nico y Aldo. Los primeros palos de la carrera nos los llevamos juntos… ¡y eso une mucho!

Por último agradecer a toda mi familia (Conchi, Cayetano y Antoñita, aquí os incluyo) vuestro continuo interés y apoyo, ya no sólo en la carrera, sino en todas las situaciones de mi vida.
A todos. GRACIAS.

Acrónimos

AES	Advance Encryption Standart
AF	Application Framework
AFH	Adaptative Frequency Hopping
APDU	Application Protocol Data Unit
APS	Application Support
BE	Backoff Exponent
BI	Beacon Interval
BO	Beacon Order
BP	Backoff Period
CAP	Contention Access Period
CBC-MAC	Cipher Block Chaining – Message Authentication Code
CCA	Clear Channel Assessment
CCM	Counter with CBC-MAC
CFP	Contention Free Period
CSMA-CA	Carrier Sense Multiple Access – Collision Advoidance
CW	Contention Window
DCO	Digitally Controled Oscilator
DSSS	Direct Sequence Spread Spectrum
DS-UWB	Direct Sequence – Ultra Wide Band
ED	Energy Detection
EDR	Enhance Data Rate

GTS	Guaranteed Time Slot
IEEE	Institute of Electrical and Electronics Engineers
IR	Infra Red
ISM	Industrial, Scientific and Medical
ITU	International Telecommunication Union
FCS	Frame Check Sequence
FFD	Full Function Device
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LQI	Link Quality Indicator
LR-WPAN	Low Rate – Wireless Personal Area Network
LSB	Least Significant Bit
MAC	Medium Access Control
MSB	Most Significant Bit
NB	Number of Backoff
OFDM	Orthogonal Frequency Division Multiplexing
OSI	Open Systems Interconnect
OTA	Over The Air
PER	Packet Error Rate
PPDU	Physical Protocol Data Unit
PSDU	Physical Service Data Unit
RF	Radio Frequency
RFD	Reduced Function Device
RISC	Reduced Instruction Set Computer

RSSI	Received Signal Strength Indicator
SD	Superframe Duration
SIG	Special Interest Group
SO	Superframe Order
SPI	Serial Peripherical Interface
SSP	Security Service Provider
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus
USCI	Universal Serial Communication Interface
UWB	Ultra Wide Band
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network
ZDO	ZigBee Device Object

Índice

Índice de figuras

Índice de tablas

CAPÍTULO 1

Introducción

1.1. Ubicación tecnológica del proyecto

Los continuos avances tecnológicos han permitido que, en los últimos años, las comunicaciones inalámbricas hayan experimentado un importante crecimiento tanto en el desarrollo de nuevas tecnologías, como en la demanda de productos capaces de satisfacer las actuales necesidades de portabilidad y movilidad. Así, han ido apareciendo en el mercado un amplio abanico de tecnologías capaces de ofrecer una comunicación eficaz y más cómoda, evitando el uso de cables y, por tanto, facilitando la autonomía de los usuarios. No se espera que la tecnología inalámbrica reemplace la cableada, pues esta última alcanza velocidades de transferencia muy superiores, pero sí que proporcione una serie de servicios que con una comunicación cableada no es posible o es excesivamente complejo conseguir, debido principalmente a estos tres inconvenientes:

- **Económicas:** La existencia de cables implica multiplicar los costes de instalación, ampliación y mantenimiento debido a la necesidad de conectar los dispositivos entre sí. Además, si estos dispositivos están alejados, habría que añadir repetidores para contrarrestar la atenuación producida por dicho cableado.

- **Portabilidad:** La obligatoriedad de estar físicamente conectados limita en gran medida la libertad de movimiento del usuario y atendiendo a la creciente necesidad de tener rápido y constante acceso a información y datos, esto se convierte en un problema muy importante.

- **Geográficas:** Dependiendo de la zona donde se requiera comunicación, la instalación de una red cableada puede resultar mucho más compleja que una red inalámbrica, ya que hay que tener en cuenta parámetros como pueden ser la orografía y extensión del terreno o la posibilidad de realizar las acciones necesarias para la instalación de los elementos.

Por estas y otras razones, han surgido cantidad de aplicaciones donde las tecnologías inalámbricas son la mejor opción, desde la creación de pequeñas redes personales con la capacidad de monitorizar un conjunto de sistemas, aportando servicios de gestión

energética, seguridad o comunicación, hasta grandes redes cuyos objetivos son la transferencia de gran cantidad de información (tanto de voz como de datos) entre equipos.

Las principales bandas de frecuencias utilizadas para establecer comunicaciones inalámbricas son las bandas infrarrojos (IR) y las de radiofrecuencia (RF). Las comunicaciones por IR proporcionan velocidades de trasferencia de hasta 4 Mbps, pero requieren de distancia entre dispositivos muy cortas, además de línea directa de visión. Por ello, su utilización se reduce a casos muy concretos. Sin embargo, las comunicaciones por RF consiguen, además de velocidades considerables, comunicaciones entre puntos muy distantes y sin la necesidad de que los elementos estén en línea directa de visión, lo que en el caso redes domóticas, inmóticas e industriales es un factor a tener muy en cuenta.

Dentro de la banda de RF se encuentra la banda ISM (*Industrial, Scientific and Medical*), definida por la ITU (*Unión Internacional de Telecomunicaciones*) y reservada internacionalmente para uso no comercial en áreas industriales, científicas y médicas, pero popularizadas por su utilización en comunicaciones WLAN (*Wireless Local Area Network*) o WPAN (*Wireless Personal Area Network*). Esta banda de frecuencias está abierta a todo el mundo sin necesidad de licencias siempre que se respeten ciertas limitaciones de potencia emitida.

Los sistemas diseñados para trabajar en bandas ISM, lo hacen principalmente a la frecuencia de 2.4 GHz o bien por debajo de 1 GHz (868 MHz en Europa, 915 MHz en Estados Unidos). La utilización de una u otra frecuencia depende de las características que se busquen conseguir en cada desarrollo en particular. Así, trabajar a 2.4 GHz será recomendable cuando se requiera interoperabilidad entre sistemas o su utilización en distintas zonas geográficas, ya que el uso de esta frecuencia es universal. Como inconveniente se ha de destacar que a esta frecuencia trabajan numerosas tecnologías (Bluetooth, ZigBee, etc) lo que provoca altos niveles de interferencia, además de un alcance menor debido a que las señales a mayores frecuencias son más fácilmente absorbidas por el entorno que las rodea. Por otro lado, trabajando a frecuencias menores de 1 GHZ, aunque mejoramos algunas de las características anteriores (mayor alcance, menos interferencias), se limita en gran medida la zona geográfica de utilización y la interoperabilidad entre sistemas.

1.2. ZigBee y otras tecnologías

En los últimos años, han aparecido una gran variedad de productos basados en tecnología inalámbrica. Principalmente tres son los estándares que hacen posible este hecho: Wi-Fi, Bluetooth y ZigBee, si bien, existen otros todavía en etapa de estudio, como es el caso de UWB (*Ultra Wide Band*), a los que se les augura un futuro prometedor. Cada uno estos estándares se diferencia por características tales como la tasa de transferencia de datos, alcance, tipo de modulación usada, etc, ofreciendo una gran variedad de opciones que permiten optimizar los recursos para cada aplicación en particular. En la tabla 1.1 se muestra una comparativa de dichos estándares con sus principales características:

Standard	Bluetooth	UWB	ZigBee	Wi-Fi
IEEE spec.	802.15.1	802.15.3a *	802.15.4	802.11a/b/g
Frequency band	2.4 GHz	3.1-10.6 GHz	868/915 MHz; 2.4 GHz	2.4 GHz; 5 GHz
Max signal rate	1 Mb/s	110 Mb/s	250 Kb/s	54 Mb/s
Nominal range	10 - 100 m	10 m	10 - 100 m	100 m
Nominal TX power	0 - 10 dBm	-41.3 dBm/MHz	(-25) - 0 dBm	15 - 20 dBm
Number of RF channels	79	(1-15)	1/10; 16	14 (2.4 GHz)
Channel bandwidth	1 MHz	500 MHz - 7.5 GHz	0.3/0.6 MHz; 2 MHz	22 MHz
Modulation type	GFSK	BPSK, QPSK	BPSK (+ ASK), O-QPSK	BPSK, QPSK COFDM, CCK, M-QAM
Spreading	FHSS	DS-UWB, MB-OFDM	DSSS	DSSS, CCK, OFDM
Coexistence mechanism	Adaptive freq. hopping	Adaptive freq. hopping	Dynamic freq. selection	Dynamic freq. selection, transmit power control (802.11h)
Basic cell	Piconet	Piconet	Star	BSS
Extension of the basic cell	Scatternet	Peer-to-peer	Cluster tree, Mesh	ESS
Max number of cell nodes	8	8	> 65000	2007
Encryption	E0 stream cipher	AES block cipher (CTR, counter mode)	AES block cipher (CTR, counter mode)	RC4 stream cipher (WEP), AES block cipher
Authentication	Shared secret	CBC-MAC (CCM)	CBC-MAC (ext. of CCM)	WPA2 (802.11i)
Data protection	16-bit CRC	32-bit CRC	16-bit CRC	32-bit CRC

Tabla 1.1: Comparativa de los protocolos Bluetooth, ZigBee, UWB y Wi-Fi. Fuente: [1]

A continuación se explicará de forma resumida los fundamentos básicos de estas tecnologías, si bien, ya que este proyecto se basa en la tecnología ZigBee, ésta será explicada en profundidad en el capítulo siguiente.

1.2.1. Bluetooth

Alrededor de 1997, un grupo de empresas líderes en su sector aúnan esfuerzos en la creación de una nueva tecnología inalámbrica, aparece así el SIG (*Special Interest Group*), una organización privada, sin ánimo de lucro cuya finalidad es el desarrollo de la tecnología Bluetooth. Si por entonces apenas siete compañías formaban dicho grupo de

investigación, en 2007 se han contabilizado más de 9000 empresas de telecomunicaciones, informática, automovilismo, etc, interesadas en el desarrollo de esta tecnología.

Bluetooth es un estándar de comunicaciones de corto alcance y bajo consumo establecido por la IEEE 802.15.1, que opera en la banda ISM a la frecuencia de 2.4 GHz con 79 canales de 1 MHz repartidos entre 2.402 GHz y 2.48 GHz. Debido a las altas probabilidades de interferencias en dicha banda de funcionamiento, Bluetooth incorpora un sistema diseñado para evitarlas, el AFH (*Adaptative Frequency Hopping*) a través de la cual, la señal salta de forma aleatoria entre 79 frecuencias en intervalos de 1 MHz, con la finalidad de excluir las secciones de frecuencias utilizadas por los dispositivos que están causando interferencias.

Una red basada en esta tecnología (popularmente conocida por piconet) acepta hasta ocho dispositivos en configuración de maestro-esclavo (uno de los nodos asume el papel de maestro y todos los demás dispositivos conectados, la de esclavos) y un máximo de 10 piconets en 10 metros.

En cuanto al alcance que proporciona esta tecnología, ésta depende de la clase de dispositivo que se utilice, diferenciándose cada clase en la potencia de transmisión utilizada. Así, las distancias oscilan entre 10 metros, transmitiendo a 1 mW y 100 metros transmitiendo a 100 mW. La velocidad de transmisión obtenida para la última versión de esta tecnología asciende de 3 Mbps gracias a la técnica EDR (*Enhanced Data Rate*), con la que, además, se mejora en robustez y seguridad.

1.2.2. Wi-Fi

Wi-Fi (*Wireless Fidelity*) es una de las tecnologías de comunicación inalámbrica más utilizada en la actualidad gracias a su capacidad de poder conectarse a Internet sin necesidad de medios físicos. Marca usada por Wi-Fi Alliance, también conocida por WLAN (*Wireless Local Area Network*) o estándar IEEE 802.11, fue diseñada para sustituir el equivalente a las capas físicas y MAC (*Medium Access Control*) de la norma 802.3 (Ethernet) por lo que es totalmente compatible con todos los servicios locales de cable Ethernet.

Al contrario que en el caso de Bluetooth, existen varios estándares Wi-Fi cuyas principales diferencias son tanto la banda de frecuencias en la que operan, como la velocidad de transferencia de información. En ese sentido estos son los estándares más destacados:

- **802.11b:** trabaja en la banda de 2.4 GHz con velocidad máxima de 11 Mbps. Utiliza como método de acceso al medio el protocolo CSMA/CA y, al igual que ocurría en el caso de Bluetooth, debido a la cantidad de interferencias que se pueden encontrar en esta banda de funcionamiento, utiliza una técnica de ensanchado de espectro basada en DSSS (*Direct Sequence Spread Spectrum*) con la que se mandan varios bits redundantes por cada bit que compone la señal.

- **802.11a:** opera en la banda de 5 GHz, una banda todavía poco usada y que por lo tanto introduce muy pocas interferencias. Utiliza 52 subportadoras con modulación OFDM (*Orthogonal Frequency Division Multiplexing*) consiguiendo velocidades de hasta 54 Mbps. Sus principales problemas son dos, por un lado, su alcance, al trabajar a mayor frecuencia, el alcance disminuye hasta una distancia de los 30 metros. Por otro su incompatibilidad con el estándar 802.11b.

- **802.11g:** se trata de una evolución del estándar 802.11b y por tanto, al igual que ésta opera en la banda de 2.4 GHz, pero consigue velocidades de hasta 54 Mbps utilizando modulación OFDM al igual que 802.11a y alcanza distancias de hasta 100 metros. 802.11g se diseñó específicamente para que fuera compatible con el estándar b, aunque redes bajo el estándar g con nodos del estándar b bajan considerablemente la velocidad de transmisión.

1.2.3. Ultra Wide Band (UWB)

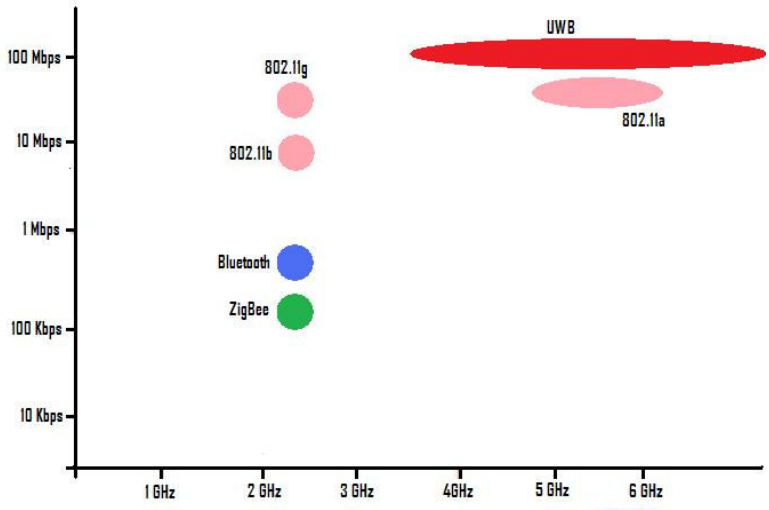
Tecnología que empezó a desarrollarse en 1950, consigue velocidades de transmisión muy superiores a las demás tecnologías inalámbricas, cercana a los 500 Mbps en un radio de 2 metros y de hasta 110 Mbps a una distancia máxima de 10 metros, transportando simultáneamente audio, video y datos, lo que la hace muy útil para sistemas de videocámaras digitales o reproductores mp3.

UWB hace uso de un gran ancho de banda (lo que le permite tales velocidades de transmisión) en el rango de frecuencias desde 3.1 GHz hasta 10.6 GHz, es decir, utiliza una banda de más de 7 GHz de anchura, transmitiendo a potencias muy reducidas, por lo que ofrece un consumo energético bajo.

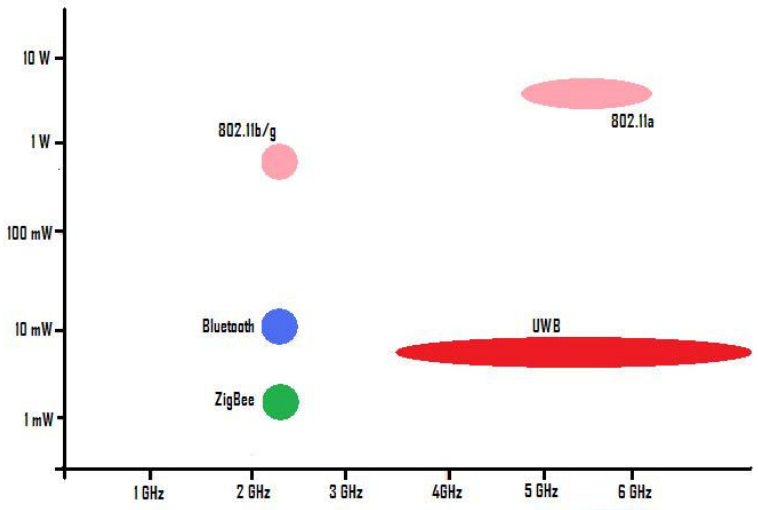
Actualmente existen dos estándares UWB compitiendo en el mercado, por un lado el foro UWB respalda un estándar basado en una secuencia directa (DS-UWB), por otro, la Alianza WiMedia defiende otro estándar basado en la modulación por división en frecuencia ortogonal (OFDM).

Esta tecnología, por sí misma, no parece estar dando los resultados esperados, pero sí que ha facilitado la evolución de otras tecnologías como Bluetooth al combinar ambas, así Bluetooth tiene intención de utilizar en un futuro cercano, UWB (con modulación OFDM) como capa física para desarrollar una versión de tecnología Bluetooth con opción a grandes anchos de banda.

A continuación se puede visualizar una comparativa entre estas cuatro tecnologías respecto a tasa de datos y consumo:



a)



b)

Figura 1.1: Comparativa entre las tecnologías ZigBee, Bluetooth, Wi-Fi y UWB respecto a: a) Tasa de datos y b) Consumo. Fuente [2].

Como se ha podido observar, Bluetooth proporciona una comunicación eficiente a cortas distancias, Wi-Fi aumenta la tasa de transferencia de datos y el rango de alcance,

ofreciendo una alternativa eficaz para conexiones a Internet y UWB es muy buena opción para comunicaciones que requieran transferencia de grandes cantidades de datos a cortas distancia. Pero queda un campo de aplicación donde estas tres tecnologías no son las óptimas dadas sus características, es el campo de la domótica, inmótica y control automatizado de plantas industriales. Una tecnología eficaz para realizar este tipo de funciones debe cumplir:

- Es necesario cubrir tanto distancias cortas para el caso de hogares (domótica), como distancias mayores en el caso de edificios (inmótica) o plantas industriales.

- No es necesario enviar gran cantidad de información, sino que valdría con el envío de mensajes cortos e instrucciones.

- Ya que los dispositivos pertenecientes a la red van, en su mayoría, a controlar sensores repartidos por un espacio determinado, no es necesario que estos dispositivos estén siempre activos, ya que los sensores sólo requerirán enviar información en momentos puntuales.

- Debido a estos sensores pueden no tener acceso a una red energética, se requiere que el consumo de los dispositivos sea mínimo para aumentar la autonomía de las baterías.

Estas y otras características son las que tratan de cumplir los dispositivos bajo el estándar ZigBee/802.15.4.

1.3. Objetivos del proyecto

Este proyecto se basa en el estudio del estándar ZigBee/802.15.4. El principal objetivo será la creación de una plataforma desde la cual se consiga manipular los dispositivos y los parámetros más característicos de la red ZigBee/802.15.4 a la que están asociados, así como de visualizar la red y observar la consecuencia de dichos cambios. Por tanto, este proyecto engloba dos grandes módulos.

- Configuración: por un lado, a través de esta plataforma, se podrá realizar diferentes ajustes de los principales parámetros existentes en una red ZigBee/802.15.4, para conseguir una topología de red determinada, u operar sobre los distintos dispositivos e interactuar con ellos mediante el envío de diferentes mensajes de forma que los demás dispositivos asociados sepan interpretarlos.

- Monitorización: por otro lado, esta plataforma proporcionará una herramienta visual que mostrará la monitorización de la red existente, la topología de la red creada y el intercambio de mensajes que se producen entre los dispositivos que forman parte de dicha red.

Para la realización de este estudio, se deberá, por tanto, tener en cuenta algunos parámetros como son:

- Tipo de nodo: como se verá en los siguientes capítulos, dentro de una red ZigBee/802.15.4 pueden coexistir tres tipos distintos de dispositivos, un Coordinador, Routers y dispositivos finales o End Devices. Cada tipo de dispositivo tendrá unas características propias y una funcionalidad diferente.

- Tipo de mensajes: en una red ZigBee/802.15.4 se producen distintos tipos de mensajes, ya sean de configuración (mensajes de balizado, asociación), de respuesta o mensajes de datos procedentes de la capa de aplicación.

1.4. Estructura de la memoria

En este apartado se pretende mostrar de forma ordenada los pasos a seguir para la realización de este proyecto. Lo primero será explicar el funcionamiento básico de una red basada en el estándar ZigBee/802.15.4, se expondrán los elementos que se han utilizado para conseguir dicho funcionamiento tanto software como hardware, se explicará la aplicación desarrollada para interactuar con la red y visualizarla y se obtendrán una conclusiones globales.

Más concretamente, la estructura de la memoria queda organizada de la siguiente manera:

- **Capítulo 1: Introducción.** Es el presente capítulo, en el que se han expuesto las principales tecnologías inalámbricas y su proyección de mercado. Además se han definido los objetivos del proyecto.

- **Capítulo 2: ZigBee/802.15.4.** En este capítulo se explicará en profundidad qué es ZigBee/802.15.4: funcionamiento, características y aplicaciones.

- **Capítulo 3: Sistema de desarrollo empleado.** Se describirán las herramientas, tanto hardware como software, utilizadas para la realización de este estudio.

- **Capítulo 4: Programa de gestión.** Capítulo en el que se expondrá el funcionamiento de la plataforma desarrollada y las modificaciones necesarias realizadas sobre el microcontrolador para hacer posible la interacción entre dispositivos y plataforma.

- **Capítulo 5: Plan de pruebas.** En este apartado, se mostrarán las principales pruebas realizadas a los dispositivos y plataforma, y los resultados obtenidos.

- **Capítulo 6: Conclusiones y líneas futuras de trabajo.** Se expondrán las conclusiones a las que hemos llegado tras el desarrollo del proyecto y se proporcionarán posibles líneas de trabajo futuras para la ampliación del mismo.

- **Anexo A: Manual de usuario.** Por último se añade un manual dónde se explica detenidamente el funcionamiento de todos y cada uno de los componentes de la plataforma para su correcto uso.

CAPÍTULO 2

ZigBee/802.15.4

2.1. Introducción a ZigBee/802.15.4

Como ya se ha visto, ZigBee es una tecnología de comunicaciones inalámbricas, creada específicamente para su aplicación en campos relacionados con la domótica, inmótica y control automático de plantas industriales, debido principalmente a estas tres características:

- Bajo consumo, ya que, como se observará más adelante en este capítulo, los nodos bajo este estándar poseen una capacidad especial de ahorro energético.
- Topología de red en mallas, que proporciona gran robustez en las comunicaciones.
- Facilidad de integración puesto que cada nodo requiere de muy poca electrónica y por tanto un coste muy reducido.

2.2. Tipos de dispositivos

Según su papel en una red, se distinguen tres tipos de dispositivos:

- **Coordinador:** Sólo existe un único dispositivo con esta funcionalidad por red ZigBee/802.15.4. Es el dispositivo más completo, pues realiza funciones de inicio, control y enrutamiento, por lo que requiere memoria y gran capacidad de computación.
- ***Router*:** Dispositivo cuya funcionalidad es extender la red gestionando nuevos caminos de comunicación entre dispositivos en el caso de detectar congestión o producirse algún problema en el enlace entre nodos. Es un nodo con complejidad similar al Coordinador pero no hay limitación de *routers* por red.
- **Dispositivo final (*End Device*):** Es el dispositivo más simple, su funcionalidad se reduce a comunicarse con su nodo padre (*router* o coordinador) y no puede gestionar nodos hijos. Esta simplicidad en sus funciones se traduce en un coste muy reducido.

Los dispositivos de una red ZigBee/802.15.4 obedecen también a otra clasificación:

- **FFD (*Full Function Device*):** También conocido como nodo activo, tiene funcionalidad completa por lo que puede ser usado como coordinador, router o dispositivo final.

- **RFD (*Reduced Function Device*):** También conocido como nodo pasivo, posee una funcionalidad muy limitada, básicamente son los sensores/actuadores de la red y por tanto, únicamente pueden ser utilizados como dispositivos finales.

2.3. Topología

Las redes ZigBee/802.15.4 se caracterizan por la posibilidad de funcionar en base a tres topologías distintas de red: topología en estrella, malla y árbol.

2.3.1. Topología en estrella

Esta topología de red consta de un dispositivo FFD funcionando como Coordinador y una serie FFD y/o RFD configurados como dispositivos finales. Todos los dispositivos finales están directamente conectados al coordinador, que hace las funciones de inicialización, control y gestión de la red; por tanto, toda comunicación existente entre cualquiera de los nodos, tiene que pasar necesariamente por el coordinador.

El principal inconveniente de esta configuración de red es que el alcance máximo de la red queda definido por el coordinador, por lo que no se pueden conseguir redes de más de 10 metros alrededor de éste.

2.3.2. Topología en malla

Es la topología característica de ZigBee/802.15.4. En este tipo de redes, el coordinador funciona como un *router* más, con la salvedad de que es éste quien inicializa y elige los principales parámetros de la red. Los *router* dan la posibilidad de ampliar la red tanto en número de nodos como en rango de alcance y el hecho de que no sólo el coordinador sea el encargado de gestionar la red sino que los *routers* también tengan esta funcionalidad hace que la red sea mucho más fiable.

Con esta configuración, se puede establecer comunicación entre cualquier par de nodos con la existencia, además, de varios caminos posibles. La elección de un determinado camino viene determinada por el nivel de red utilizando un protocolo de pregunta-respuesta, para seleccionar el camino óptimo.

Por contra, resulta muy difícil conseguir en este tipo de redes que los nodos pasen a un estado de bajo consumo.

2.3.3. Topología en árbol

Esta configuración es un caso particular de la topología en malla, en la que los dispositivos se organizan de manera jerárquica. El coordinador sería el primer nivel, al que pueden conectarse tanto dispositivos FFD como RFD (llamados nodos hijos). De cada FFD pueden seguir conectándose dispositivos estableciéndose los siguientes niveles de profundidad.

En esta topología, al igual que pasaba con la configuración mallada, tanto los *routers* como el coordinador tienen capacidad de encaminamiento y además, estos *routers* permiten expandir la red más allá del alcance máximo del coordinador, por lo que se pueden cubrir mayores áreas que en una topología en estrella.

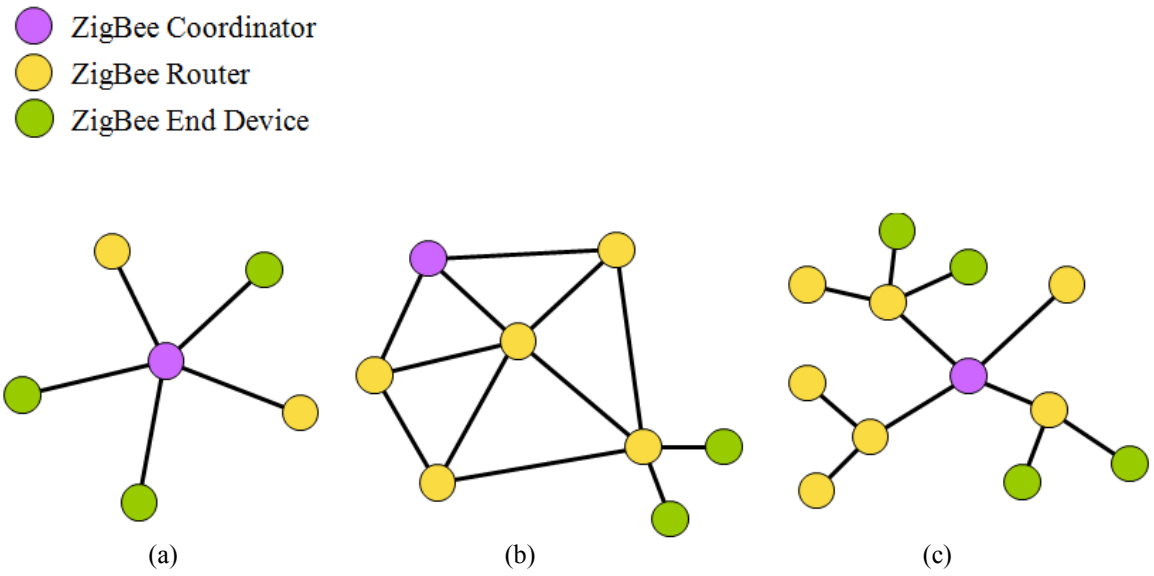


Figura 2.1: Topologías posibles en una red ZigBee/802/15.4.
(a) Estrella. (b) Malla. (c) Árbol. Fuente [8].

2.4. Pila de protocolos

Se define ZigBee/802.15.4 como una pila de protocolos que permite la comunicación de forma sencilla entre múltiples dispositivos. Especifica diversas capas adecuándose al modelo OSI (*Open Systems Interconnect*).

Las capas básicas, capas física y MAC (*Medium Access Control)* están definidas por el estándar IEEE 802.15.4, LR-WPAN (*Low Rate-Wireless Personal Area Network*) y se han diseñado pensando en la sencillez de la implementación y el bajo consumo sin perder

potencia ni posibilidades. Las capas superiores, capas de red y aplicación, están definidas por el estándar ZigBee, desarrollado por ZigBee Alliance, agrupación de más de 300 compañías que trabajan conjuntamente por convertir la tecnología ZigBee en un referente importante en el marco de aplicaciones ya expuesto en el capítulo anterior.

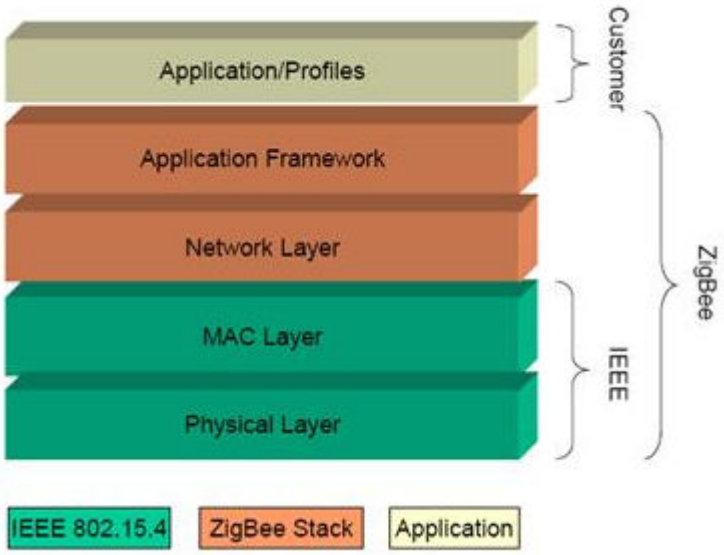


Figura 2.2: Pila de protocolo ZigBee/802.15.4. Fuente [10].

2.4.1. Capa física (PHY)

La capa física es la encargada de proporcionar un medio por el que transmitir y recibir datos. 802.15.4 ofrece la posibilidad de trabajar en tres bandas de frecuencia distintas, todas utilizando DSSS (*Direct Sequence Spread Spectrum*) como técnica de ensanchado de espectro: 868 MHz, 915 MHz y 2.4 GHz. La banda de 868 MHz proporciona un único canal de comunicaciones entre las frecuencias 868 y 868.6 MHz consiguiendo una velocidad de transmisión de 20 Kbps. En la banda de 915 MHz existen 10 canales de comunicación, repartidos uniformemente entre las frecuencias 902 MHz y 928 MHz con una separación entre canales de 2 MHz y obteniendo velocidades de transmisión de 40 Kbps. Estas dos bandas de frecuencias tienen como principal ventaja respecto a la banda de 2.4 GHz su mayor alcance debido a menores pérdidas de propagación, sin embargo, su mayor inconveniente es que no son bandas universales, la banda de 868 MHz es de uso exclusivo en Europa y la banda de 915 MHz en Estados Unidos y Australia, por lo que la movilidad y zonas de aplicación de dispositivos funcionando a estas frecuencias está altamente limitado. No es el caso de la banda de 2.4 GHz, cuyo uso está permitido en prácticamente todo el mundo. Proporciona 16 canales entre las frecuencias 2.405 GHz y 2.48 GHz, con una separación entre canales de 5 MHz y un ancho de banda de 2 Mbps,

alcanzando velocidades de transmisión de hasta 250 Kbps. En la siguiente figura se pueden observar los distintos canales radio existentes en cada una de las bandas de frecuencias:

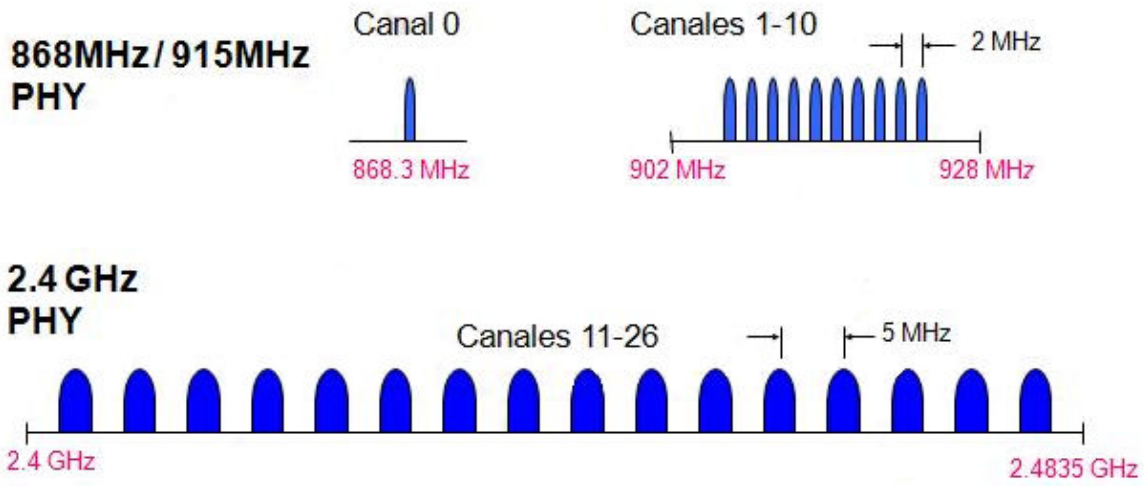


Figura 2.3: Canales radio en las tres bandas de frecuencias de trabajo. Fuente [10].

En la siguiente tabla se muestra un resumen de las principales características para cada una de las bandas de frecuencias utilizables:

PHY (MHz)	Banda de Frecuencias (MHz)	Modulación	Tasa de bit (kb/s)
868	868-868.6	BPSK	20
915	902-928	BPSK	40
2450	2400-2483.5	O-QPSK	250

Tabla 2.1: Resumen de las principales características para las distintas bandas de frecuencias.

Funciones de la capa física

Las funciones a realizar por la capa física son varias, aunque las más destacables son:

- **Activación y desactivación del transceptor radio:** El transceptor radio tiene tres posibles estados de funcionamiento: transmisión, recepción y modo dormido (*Sleeping*). Este último estado permitirá al dispositivo un importante ahorro de energía como se verá más adelante. Es la capa MAC la que decide el estado del transceptor en cada instante.

- **Detección de energía (ED):** La medición de la energía obtenida por el detector se utiliza en la capa de red como parte del algoritmo de selección de canal, así la capa física se encarga de realizar esta medición en un determinado canal. No se realiza decodificación alguna de señal, sólo se obtiene una estimación de la potencia recibida

en dicho rango de frecuencias que se traduce en un valor de 8 bits, cuyo rango varía entre 0x00, que indica que la potencia recibida es menor de 10 dB sobre la especificada por el receptor, y 0xFF.

- **Indicación de la calidad del enlace (LQI):** La medida del LQI es una caracterización de la potencia y/o calidad del paquete recibido. Esta medida se puede implementar utilizando el detector de energía, un estimador de relación señal a ruido o mediante una combinación de ambos métodos. El valor obtenido se traduce en 8 bits con representación entre 0x00 y 0xFF cuyos valores máximo y mínimo vienen determinados por la mayor y menor calidad de señal IEEE 802.15.4 detectable respectivamente, siguiendo los demás valores de calidad una distribución uniforme entre ambos límites.

- **Selección de canal:** Como ya se ha comentado, la capa MAC a través de un algoritmo decidirá el canal más adecuado para establecer una comunicación, pero es la capa PHY la que toma físicamente ese canal para realizar las transmisiones y/o recepciones de datos.

- **Evaluación de canal libre (CCA, *Clear Channel Assessment*):** La evaluación de canal libre se desarrolla utilizando uno de los siguientes tres métodos:

    - *Energy above threshold*: CCA avisará de la ocupación de un medio si detecta cualquier tipo de energía por encima de un umbral predeterminado.

    - *Carrier Sense Only*: CCA notificará sobre la ocupación de un medio si detecta una señal con la modulación y características de propagación propias de una señal IEEE 802.15.4. Esta señal puede recibirse con independencia del umbral.

    - *Carrier Sense with energy above threshold*: Unión de los dos métodos anteriores.

- **Transmisión y recepción de datos por el medio físico:**

Esta capa proporciona además, dos servicios: servicio de datos (PHY *data service*) y servicio de gestión (PHY *management service*) ejerciendo de interfaz entre la capa MAC y el medio físico en cuestión, en este caso, los canales radio. El servicio de datos permite la transmisión y recepción de paquetes PPDU (Physical Protocol Data Unit) cuyo formato se expone a continuación:

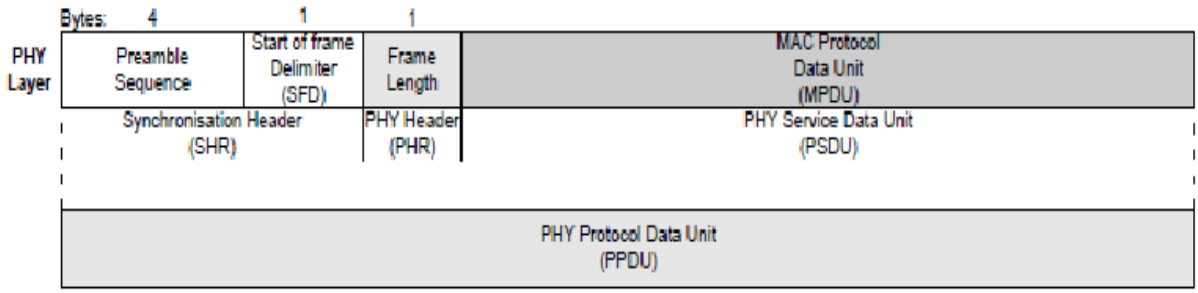


Figura 2.4: Formato de trama PPDU. Fuente [5].

Como se puede observar, un PPDU consta principalmente de tres campos:

- **SHR (Preámbulo de sincronización):** cuya función es la que sincronizar al dispositivo transmisor y el receptor.
- **PHR (Cabecera de nivel físico):** con la longitud de la trama.
- **PSDU:** campo variable con la trama de la capa MAC (carga útil).

2.4.2. Capa MAC (*Medium Access Control*)

La capa MAC o capa de acceso al medio es la responsable de las siguientes tareas:

- Generación de balizas (beacons) si el dispositivo es un coordinador.
- Sincronización de los beacons de la red.
- Gestión de las asociaciones y desvinculaciones de los dispositivos a la red.
- Empleo del algoritmo CSMA-CA (*Carrier Sense Multiple Access-Collision Avoidance*) para acceso al canal.
- Gestión de la técnica GTS (*Guaranteed Time Slot)*.
- Gestión de un enlace fiable entre las capas MAC de los nodos contiguos.

2.4.2.1. Modos de funcionamiento

La capa MAC permite dos modos de funcionamiento, la elección de uno u otro modo se realiza al configurar los dispositivos, aunque es el coordinador el encargado de informar a la red qué modo se utilizará en la nueva red creada. Estos dos modos son los siguientes:

- **Modo balizado (beacon-enabled network):** Modo de funcionamiento que consigue un importante ahorro energético. Está basado en la utilización por parte de los dispositivos FFD de balizas con las que marcan los tiempos en los que es posible la recepción y transmisión de información. Fuera de estos tiempos, todos los dispositivos (incluido el coordinador) pueden estar en modo "dormido", modo en el que se minimiza en consumo.
  La baliza se genera periódicamente por el coordinador y se distribuye por toda la red a través de los *routers*. Estas balizas se encargan de sincronizar los dispositivos, de modo que todos los dispositivos se "despierten" en un determinado instante en el cual se realiza la comunicación entre nodos.

- **Modo no balizado (non beacon-enabled network):** En este modo no existe sincronización entre dispositivos, por lo que los únicos nodos que pueden pasar al estado "dormido" son los dispositivos finales. Éstos se despertarán de forma periódica para preguntar si existen datos destinados a ellos o bien para mandar información.

Modo balizado

En el modo balizado, se utiliza una estructura de trama conocida por supertrama (*superframe*), esta estructura es definida por el coordinador y construida en base a:

- Intervalo de baliza (BI): variable que define el tiempo entre dos balizas consecutivas.

- Duración de supertrama (SD): indica la parte activa de BI. Está subdividido en 16 slots de tiempo en los que se permiten las transmisiones.

- Periodo inactivo: se define un intervalo de tiempo cuando BI supera a SD en el cual todos los nodos entran en modo "dormido", reduciendo así su consumo energético.

Los valores de BI y SD se definen mediante dos parámetros: *BeaconOrder* (BO) y *SuperframeOrder* (SO), de la siguiente manera:

$$\begin{cases} BI = aBaseSuperframeDuration \times 2^{BO} \\ SD = aBaseSuperframeDuration \times 2^{SO} \end{cases} \quad para\ 0 \leq SO \leq BO \leq 14$$

Donde el valor de *aBaseSuperframeDuration* = 15.36 ms (asumiendo que se está operando a 250 Kbps en la banda de 2.4 GHz) y cuando se asigne un valor de 15 a las variables SO y BO indica que se está trabajando en modo no balizado.

Como se ve en la Figura 2.5, la supertrama está compuesta por 16 slots temporales, el primero de ellos contiene la baliza y los siguientes se dividen en dos grupos:

- CAP (*Contention Access Period*), en el cual los dispositivos compiten por el acceso al medio utilizando CSMA ranurado.

- CFP (*Contention-Free Period*), esta etapa está pensada para aplicaciones en tiempo real, por lo que se reservan ciertos slots temporales para que aquellas aplicaciones que sean sensibles al tiempo tengan siempre la posibilidad de transmitir, sin necesidad de competir por el canal. Estos slots reservados toman el nombre de GTS (*Guaranteed Time Slot*).

En la siguiente figura se puede observar la estructura de trama en modo balizado:

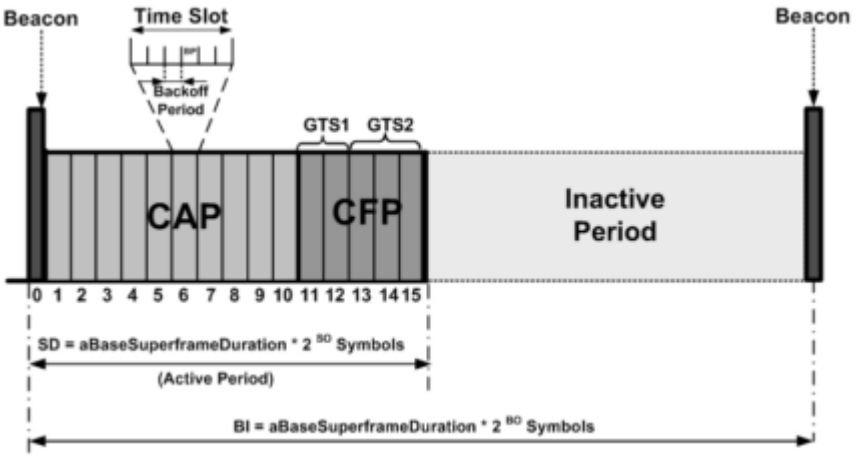


Figura 2.5: Estructura de la supertrama. Fuente [6].

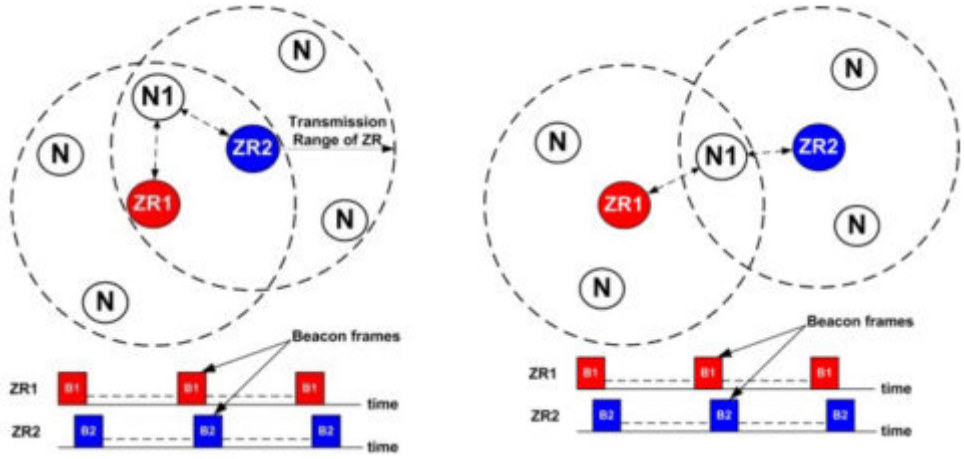
Es fácil darse cuenta que se puede conseguir un ciclo de trabajo reducido si se configura SO con un valor bastante menor que el de BO, consiguiendo así largos periodos de inactividad en los cuales todos los dispositivos quedarían en modo dormido. La gran ventaja de la sincronización en modo baliza es que todos los dispositivos se despiertan y se duermen al mismo tiempo, sin embargo, se verá a continuación el problema de utilizar este modo de funcionamiento en topologías de árbol o malla, en la que varios dispositivos

(coordinador y routers) envían balizas y por tanto puede aparecer el problema de las colisiones entre balizas.

Este problema surgido de la posible colisión entre balizas, ha sido profundamente estudiado por IEEE Task Group 15.4, grupo también conocido como LR-WPAN como ya se comentó en el capítulo 1. En este estudio se analizan las diferentes situaciones en las que se podría producir este problema y las consecuencias que conllevan. Así, en una topología de malla se consideran dos tipos distintos de posibles colisiones en modo baliza:

- Colisión directa: Ocurre cuando dos o más dispositivos con capacidad de mandar balizas (coordinador y/o routers) se encuentran dentro del rango de transmisión uno del otro y mandan la baliza aproximadamente al mismo tiempo. El nodo que recibe ambas balizas pierde la sincronización con el primer nodo que mandó la baliza, al producirse colisión de balizas. Ver figura 2.6.

- Colisión indirecta: En este caso los dispositivos con capacidad de mandar balizas no se escuchan (no están dentro del rango de transmisión respectivamente) pero el nodo que debe recibir las balizas sí se encuentra dentro del radio de acción de ambos nodos. Este dispositivo perderá la sincronización con ambos nodos por colisión de balizas. Véase la figura 2.6.



(a) (b)

Figura 2.6: Problema de colisión de balizas: (a) Colisión directa. (b) Colisión indirecta. Fuente [6].

Modo no balizado

El coordinador adopta este modo de funcionamiento si carga las variables BO y SO con el valor 15 como ya se ha comentado. En este caso no hay balizas, ni slots temporales, ni sincronización alguna entre nodos. Simplemente, cuando un dispositivo debe enviar algo aplica el algoritmo CSMA-CA y trata de hacerse con el canal. Sólo los mensajes de confirmación de recepción (mensajes ACK) son mandados sin necesidad de competir por el canal. En esta situación no existen por tanto, reserva de tiempo para la transmisión de información en tiempo real, es decir, no existe GTS.

En este caso, tanto el coordinador como los *routers* existentes en la red deben permanecer en modo despiertos todo el tiempo, ya que desconocen cuándo un dispositivo final conectado a éstos les mandará datos. El dispositivo final sí podrá pasar al modo dormido y sólo se despertará cuando necesite mandar información y/o periódicamente para ver si existen datos con éste como destino. Para ello manda un mensaje de petición de datos cada vez que despierta y el coordinador o *router* del que depende responde en el caso que tenga información para él.

Como se puede observar, este modo tiene un funcionamiento muy sencillo, pero a cambio el consumo energético del coordinador y *routers* es mucho mayor, por lo que se pierde eficiencia.

2.4.2.2. Algoritmo CSMA-CA

Al utilizar este algoritmo de acceso al medio, cada dispositivo anuncia su intención de realizar una transmisión antes de empezar con ella, de forma que sólo empezará a mandar información si encuentra el canal libre. Si varios dispositivos encuentran el canal ocupado, cada nodo espera un tiempo aleatorio antes de volver a pedir el canal con el fin de evitar colisiones. Un dispositivo en posesión del canal no lo libera hasta recibir la confirmación de recepción de su trama enviada. Por tanto, el proceso que sigue CSMA-CA se puede resumir, básicamente, en cuatro pasos:

1) El dispositivo con intención de enviar datos, escucha el canal para ver si éste está libre.
2) En caso de estar libre, envía la información. Si no lo está, espera un tiempo aleatorio antes de intentarlo de nuevo.
3) Una vez enviada la información, espera la llegada de una confirmación de recepción con la que asegurarse que la transmisión se ha realizado con éxito.
4) Tras esa confirmación, da la transmisión por concluida.

Existen dos versiones de este algoritmo, según se esté operando en modo balizado o en modo no balizado. Para el modo balizado se utiliza el método CSMA-CA ranurado mientras que para el modo no balizado se usa el método CSMA-CA no ranurado. La diferencia entre ambos mecanismos reside en la sincronización o no del tiempo de *backoff* con el balizado, siendo este tiempo de *backoff* la variable que define la periodicidad con la que se intenta acceder al medio. En el caso de CSMA-CA no ranurado, como no existe baliza alguna con la que sincronizarse, este tiempo de *backoff* se define de forma independiente para cada dispositivo.

Figura 2.7: Diagrama de bloques del algoritmo CSMA-CA. Fuente [5].

El algoritmo CSMA-CA se basa en una unidad básica de tiempo denominada *Backoff Period* (BP o periodo de *backoff*) que se corresponde con el valor de la variable *aUnitBackoffPeriod*, normalmente igual a 80 bits (suponiendo que se opera a 2.4 GHz y, por tanto, a 250 Kbps, corresponde a 0.32 ms). Este algoritmo depende principalmente de tres variables:

- ***Backoff Exponent* (BE):** determina el número de periodos de *backoff* que hay que esperar antes de volver a intentar hacerse con el canal tras un intento fallido. Este número suele ser aleatorio entre 0 y 2^{BE} para que cada dispositivo que se encuentre en esta situación espere un intervalo de tiempo distinto y así reducir otro posible intento con fallo.

- ***Contention Window* (CW):** representa el número de periodos de backoff (BP) que el canal debe permanecer sin actividad para que se considere libre y por tanto se pueda competir por él.

- ***Number of Backoff* (NB):** muestra el número de intentos que un dispositivo lleva acumulados para acceder al canal.

2.4.2.3. Inicio y mantenimiento de la red

Como ya se mencionó, la capa MAC es la encargada de inicializar una red así como de gestionar las conexiones y desconexiones de dispositivos en ésta. Para poder desarrollar estas tareas, se sirve de una serie de procedimientos:

- **Detección de energía de canal:** Con la finalidad de que el coordinador pueda elegir aquel canal que considere más limpio de interferencias, la capa MAC puede pedir a la capa física que realice una detección de energía de una serie de canales seleccionados por las capas superiores.

- **Escaneo activo de canal:** Cuando un *router* o dispositivo final se quiere asociar a una red utiliza este sistema, por el cual manda balizas periódicamente a través de cada uno de los canales en busca del coordinador. Si el coordinador se encuentra funcionando en modo no balizado, al recibir esta baliza, mandará una respuesta de baliza puntual, con la que el dispositivo en búsqueda detectará la presencia del coordinador y empezará el proceso de asociación. En el caso del modo balizado, el coordinador ignorará estas balizas y mandará periódicamente las suyas, el dispositivo en búsqueda detectará esta baliza y encontrará al coordinador.

- **Escaneo pasivo de canal:** En este caso, el nuevo dispositivo encendido, no manda balizas sino que se dedica a escuchar cada uno de los canales en busca de las balizas periódicas del coordinador. Como es lógico, es un tipo de escaneo utilizado sólo en el modo balizado.

- **Escaneo de orfandad:** Si las capas superiores reciben repetidas veces fallos de comunicación al solicitar datos, puede determinarse que el dispositivo ha quedado huérfano. En este caso, el dispositivo manda notificaciones de orfandad por aquel conjunto de canales que ordenen las capas superiores y desecha todos los datos entrantes a la capa física que no sean una respuesta por parte del coordinador a esta notificación durante un tiempo definido por la variable *aResponseWaitTime*. Si durante este tiempo el coordinador responde a esta notificación de orfandad, el dispositivo consigue volver a sincronizarse.

2.4.2.4. Formato de trama MAC

La capa MAC puede enviar hasta cuatro tipos de tramas distintas: trama de comandos MAC, trama de baliza, trama de datos y trama de confirmación de recepción o trama ACK. Todas estas tramas tienen una estructura común, compuesta por:

- **MHR:** Contiene información de control, número de secuencia, información de la dirección tanto de destino como de origen y un campo con datos sobre la seguridad utilizada.

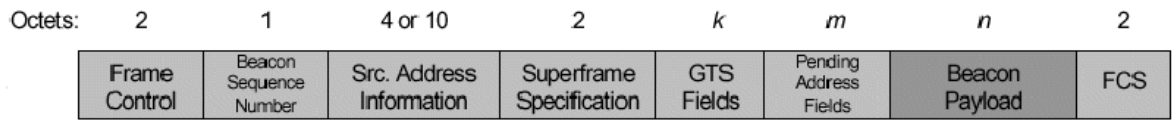
- **Carga útil:** De tamaño variable con información específica del tipo de trama. Las tramas ACK no contienen este campo.

- **MFR:** Campo que contiene el FCS (*Frame Check Sequence*), código redundante para la detección de errores en la trama.

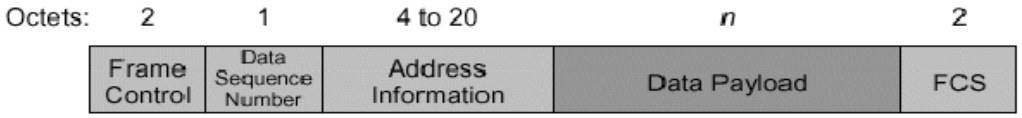
<table>
<tr><th>Octets: 2</th><th>1</th><th>0/2</th><th>0/2/8</th><th>0/2</th><th>0/2/8</th><th>0/5/6/10/ 14</th><th>variable</th><th>2</th></tr>
<tr><td rowspan="2">Frame Control</td><td rowspan="2">Sequence Number</td><td>Destination PAN Identifier</td><td>Destination Address</td><td>Source PAN Identifier</td><td>Source Address</td><td rowspan="2">Auxiliary Security Header</td><td rowspan="2">Frame Payload</td><td rowspan="2">FCS</td></tr>
<tr><td colspan="4">Addressing fields</td></tr>
<tr><td colspan="7">MHR</td><td>MAC Payload</td><td>MFR</td></tr>
</table>

Figura 2.8: Formato general de trama MAC. Fuente [5]

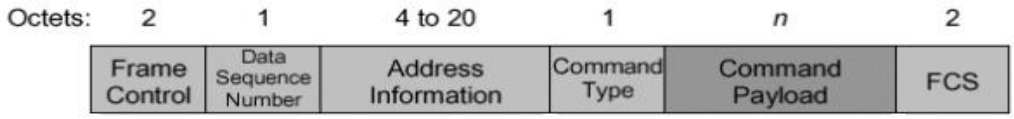
En la Figura siguiente se muestra el formato de trama para cada uno de los cuatro tipos de tramas posibles en la capa MAC:



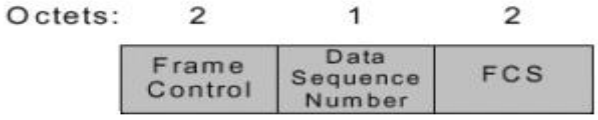
(a)



(b)



(c)



(d)

Figura 2.9: Tramas capa MAC.
(a) trama de baliza. (b) trama de datos. (c) trama de comandos. (d) trama de *ack*. Fuente [12]

2.4.3. Capa de red

2.4.3.1. Descripción general.

La capa de red tiene como misión procurar una correcta funcionalidad de las capas inferiores (física y MAC), además de servir como interfaz de servicio para la capa de aplicación, tanto en datos, como en gestión. Así, sus principales tareas se pueden resumir en:

- **Configuración de nuevos dispositivos:** para que éstos operen de la forma requerida, ya sea inicializando un dispositivo como coordinador o uniéndose a una red existente.

- **Inicialización de red:** La capa de red tiene la habilidad de establecer una nueva red.

- **Asociación, reasociación o abandono de una red:** solicitado tanto por el dispositivo afectado, como por parte del coordinador o algún *router*.

- **Direccionamiento:** esta capa otorga la capacidad tanto al coordinador como a los *routers* existentes en la red de proporcionar direcciones a los nuevos dispositivos que estén asociándose a la red.

- **Descubrimiento de vecinos:** tanto descubrimiento como almacenamiento de información y aviso sobre dispositivos vecinos cercanos.

- **Descubrimiento de ruta:** descubrimiento y registro de caminos entre dispositivos de forma que siempre se utilice el camino más óptimo entre un destino y un origen.

- **Control de recepción:** la capa de red permite controlar si un dispositivo tiene la recepción activada y por cuánto tiempo.

- **Enrutamiento:** proporciona la habilidad de utilizar diferentes mecanismos de enrutamiento como son: *unicast*, *broadcast* o *multicast* consiguiendo un intercambio de datos eficiente entre dispositivos.

A nivel de red existen dos tipos de direcciones, direcciones cortas (16 bits) y direcciones largas o direcciones IEEE (64 bits). Cada dispositivo posee una única dirección IEEE que es asignada al dispositivo a la hora de fabricarlo y en la red se le asigna de forma dinámica una dirección corta, única para dicha red, que se utilizará para todas las comunicaciones en las que forme parte el dispositivo.

2.4.3.2. Funciones.

Las funciones más destacadas realizadas por la capa de red se detallan a continuación:

- **Inicialización de una red:** esta función es sólo realizable por el dispositivo que actúa como coordinador, el cual al inicializarse realiza una serie de pasos:

  1) indica a la capa MAC que realice detección de energía de canal a un conjunto de canales (siempre y cuando no se imponga un único canal, en cuyo caso no es necesario este paso).

  2) La información obtenida se ordena de mayor a menor calidad, descartando aquellos canales con niveles de interferencias demasiado altos y se indica a la capa MAC que realice un escaneo activo en cada uno de estos canales.

  3) Se elige el canal con menor nivel de interferencia y que contenga menor número de redes ZigBee ya operando.

Una vez establecida la nueva red, se elige un identificador de red con un valor entre 0x0000 y 0xFFFF (normalmente preconfigurado en el dispositivo coordinador) y se

informa a la capa MAC. Después el coordinador se autoasigna la dirección corta 0x0000.

- **Permitir a dispositivos unirse a una red:** La capa de red tiene la posibilidad de modificar el parámetro *macAssociationPermit* el cual define la posibilidad de permitir que un nodo se asocie al dispositivo en cuestión. Como ya se mencionó anteriormente, sólo el coordinador y los *routers* tienen la capacidad de que nuevos dispositivos se asocien a ellos, por lo tanto este parámetro sólo tiene sentido modificarlo en estos tipos de elementos.

- **Unirse a una red:** Cuando se activa un dispositivo *router* o un dispositivo final, su primer objetivo es asociarse a una red ya existente, para ello realizan un escaneo activo por cada uno de los canales que tienen previamente configurados en busca de un coordinador que les responda. Tras este escaneo, almacena los distintos identificadores de red encontrados, denominados PANID (Lo normal es que un dispositivo esté preconfigurado para asociarse a una red en concreto, es decir, conoce la PANID a la que debe asociarse).

  Una vez encontrada su red, debe decidir a qué nodo de la red debe asociarse, para ello hace un listado de los posibles candidatos e intenta asociarse a los nodos en orden ascendente a la distancia con el coordinador, es decir, primero intentará asociarse al coordinador, si éste no permite nuevas incorporaciones se intentará asociar a un dispositivo router que "cuelgue" de este coordinador y así sucesivamente. En el caso de existir dos routers a la misma distancia del coordinador y ambos con permiso de asociación, el dispositivo se escoge el que tengo en primer lugar en el listado de candidatos. Para llevar a cabo esta asociación, el dispositivo nuevo manda su dirección IEEE al nodo elegido, el cual asociará esta dirección IEEE a una dirección corta de 16 bits que asignará a este nuevo dispositivo y será esta dirección la que se utilice para cualquier comunicación.

- **Mecanismos de asignación de una dirección corta (*short address*) en una topología en árbol:** El algoritmo utilizado para la asignación de direcciones se basa en la creación de subgrupos de direcciones. El coordinador siempre tiene asociada la dirección 0x0000. A cada uno de sus nodos "hijos" con capacidad de asignar direcciones, es decir *routers*, les asigna un conjunto de direcciones, asignando a este nodo *router* la primera de dicho conjunto y reservando las demás para posibles ampliaciones de la red. Una vez todos los *routers* "hijos" del coordinador tienen su

conjunto de direcciones asignados, se les otorga las siguientes direcciones disponibles a cada dispositivo final "hijo" del coordinador. El mismo proceso realiza cada *router* con su conjunto de direcciones y los dispositivos asociados a él.

Para decidir el tamaño de cada subconjunto de direcciones se tienen en cuenta una serie de parámetros definidos por el coordinador: máximo número de hijos por nodo (*nwkMaxChildren* o C_m), máxima profundidad permitida de red (*nwkMaxDepth* o L_m) o máximo número de *routers* que un determinado dispositivo puede tener como hijos (*nwkMaxRouters* o R_m). Siendo *d* como la profundidad de un dispositivo (distancia en saltos hasta llegar al coordinador) se define la función $C_{skip}(d)$:

$$Cskip(d) = \begin{cases} 1 + C_m \cdot (L_m - d - 1) & si\ R_m = 1 \\ \dfrac{1 + C_m - R_m - C_m \cdot R_m^{L_m - d - 1}}{1 - R_m} & en\ otro\ caso \end{cases}$$

$C_{skip}(d)$ es el tamaño del subconjunto de direcciones asignado a cada dispositivo, si esta variable obtiene el valor cero, indicará que ese dispositivo no acepta hijos.

La asignación de direcciones para el caso de un dispositivo final viene dada por la expresión:

$$Direc_n = Direc_{padre} + Cskip(d) \cdot R_m + n$$

En las especificaciones de ZigBee [12] se presenta el siguiente ejemplo de distribución de direcciones. Se han fijado el valor de las siguientes variables:

nwkMaxChildren = 8 *nwkMaxRouters* = 4 *nwkMaxDepth* =3

Obteniéndose los valores de $C_{skip}(d)$ que se muestran en la siguiente tabla y que corresponden con la topología de red mostrada en la figura:

Depth in the Network, *d*	**Offset Value, *Cskip(d)***
0	31
1	7
2	1
3	0

Tabla 2.2: Tamaño de subgrupos de direcciones en función de la profundidad. Fuente [12].

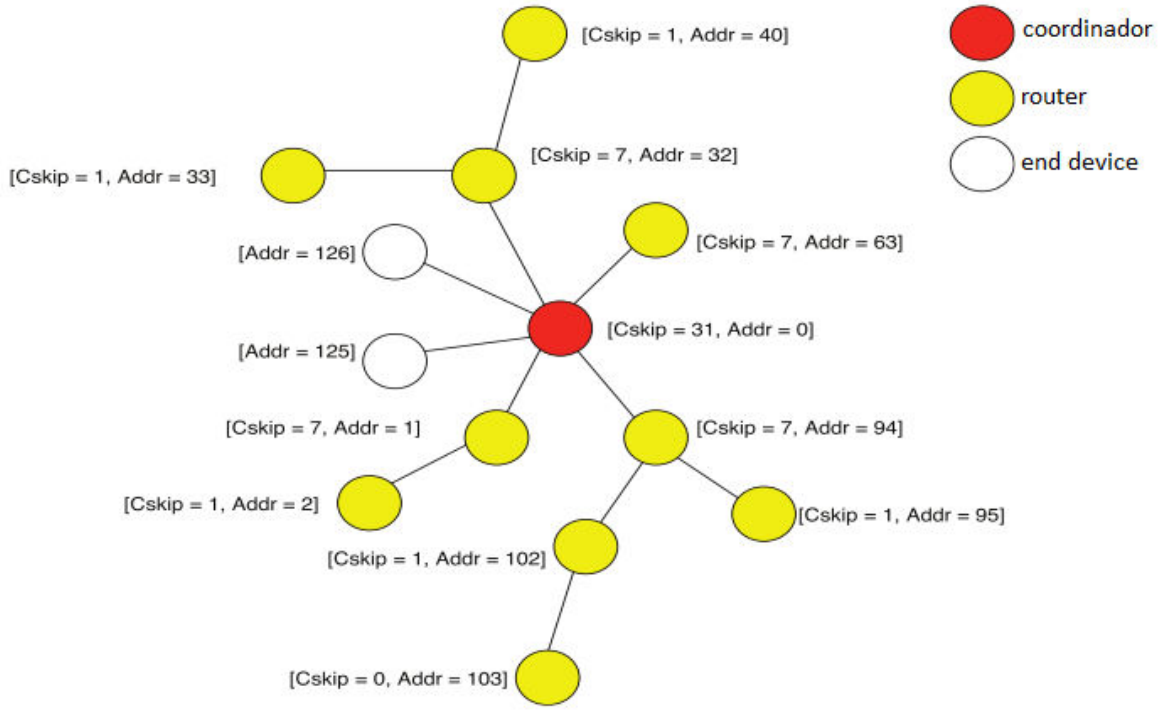


Figura 2.10: Asignación de direcciones y C_{skip} para el ejemplo de red. Fuente [12].

Las especificaciones del estándar ZigBee [12] también muestran un segundo método para la asignación de direcciones, se trata del método estocástico por el cual esta asignación se hace de manera aleatoria (excepto para el coordinador que mantiene la dirección 0x000).

- **Tabla de vecindades:** cada nodo posee una tabla con información sobre los nodos que tiene dentro de su radio de cobertura. Estas tablas son muy útiles en dos contextos: por un lado, cuando un dispositivo busca una red a la que asociarse, mira esta tabla para obtener el listado de posibles padres o bien cuando un dispositivo pierde la sincronización, la búsqueda de su nodo padre la realiza a través de estas tablas. Por otro lado es útil para almacenar información sobre la relación entre los distintos nodos o sobre el estado de cada uno de los enlaces.

  Cada dispositivo actualiza los datos almacenados en la tabla de vecindades cada vez que recibe una trama de un nodo vecino.

  Estas tablas ofrecen información sobre: dirección IEEE, dirección del dispositivo, tipo de dispositivo, relación con éste, LQI (*Link Quality Interface),* etc.

- **Abandono de red:** Pueden ocurrir dos casos, que un dispositivo decida abandonar la red o que un dispositivo padre expulse de la red a un hijo, en ambos casos si existieran hijos de este nodo borrado o expulsado, estos también se deben borrar. El comportamiento que tendrán entonces estos nodos hijos borrados a consecuencia de su

padre será el de reseteo y búsqueda de un nuevo padre al que asociarse (se comporta como si fuera un nodo nuevo, con una nueva dirección corta)

- **Enrutamiento:** Como ZigBee permite varias topologías, utiliza diferentes algoritmos según la red esté estructurada en configuración estrella, árbol o malla.

En una topología en estrella no tiene sentido hablar de enrutamiento ya que forzosamente todos los mensajes pasan por el coordinador. Para una topología en árbol un dispositivo final manda siempre el mensaje a su dispositivo padre, el cual chequea si tiene la dirección destino en su tabla de direcciones, en cuyo caso será capaz de enrutarlo. Si no es así, mandará el mensaje a su dispositivo padre y se volverá a realizar el mismo procedimiento. Para el caso de una topología en malla, el algoritmo de enrutamiento es bastante más complejo. Se basa en la utilización de tablas de enrutamiento, almacenando en éstas información sobre la dirección corta destino, el estado de los enlaces, la dirección del siguiente nodo, etc.

Para obtener la ruta óptima se realiza un cálculo del coste de cada camino en función del número de saltos que hay entre origen y destino y la calidad de los enlaces existentes entre ambos dispositivos.

2.4.3.3. Formato de trama

Una trama de la capa de red sigue el siguiente estructura de trama:

Octets: 2	**2**	**2**	**1**	**1**	**0/8**	**0/8**	**0/1**	**Variable**	**Variable**
Frame control	Destination address	Source address	Radius	Sequence number	Destination IEEE Address	Source IEEE Address	Multicast control	Source route subframe	Frame payload
NWK Header									Payload

Figura 2.11: Formato de trama de la capa de red. Fuente [12].

- **Campo de control de trama (*Frame Control*):** 16 bits que indican tipo de trama, versión de protocolo y si se emplea seguridad.
- **Dirección destino y origen.**
- **Radio (*Radius*):** cada vez que la trama pasa por un nodo se le resta uno al valor de este campo, la idea es limitar el número de saltos y por tanto evitar que una trama que no encuentra destino permanezca en la red indefinidamente.

- **Número de secuencia (*Sequence number*):** gracias al número de secuencia y a la dirección origen se puede identificar un trama de forma unívoca.
- **Dirección IEEE destino y origen:** campo opcional, su uso se indica en el campo de control de trama.
- **Control de Multicasting (*Multicast control*):** Define parámetros necesarios para la transmisión multicast. Uso opcional indicado en el campo control de trama.
- **Subtrama de ruta origen (*Source route subframe*):** campo opcional.
- **Carga útil (*Frame payload*):** con información de las capas superiores.

2.4.4. Capa de aplicación

2.4.4.1. Descripción general.

La capa de aplicación se subdivide en la subcapa APS (*Application Support*), la subcapa ZDO (*ZigBee Device Object)* y los objetos de aplicación definidos por cada uno de los fabricantes, denominada AF (*Application Framework*).

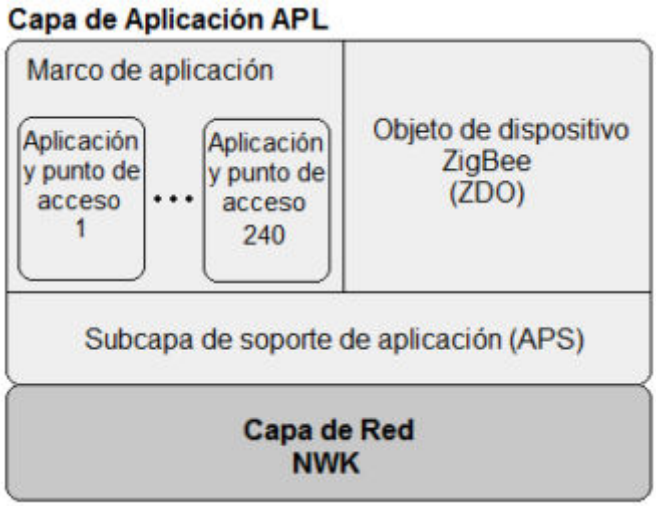


Figura 2.12: Capa de Aplicación.

A continuación se describirá de forma detallada la funcionalidad de cada una de estas subcapas, pero antes se deben definir algunos términos necesarios para comprender en qué se basa la funcionalidad de esta capa de aplicación:

- **Perfil de aplicación:** Describe los acuerdos sobre mensajes, formato de mensajes y acciones de procesado que permiten a los desarrolladores crear aplicaciones para un

grupo determinado de dispositivos. Este perfil de aplicación permite a las aplicaciones mandar comandos, pedir datos y procesar la información recibida.

- ***Clusters*:** Cada *cluster* hace alusión a una determinada aplicación, se identifica por la variable *ClusterID* y es la forma en la que los dispositivos indican a qué aplicación en concreto se refieren los comandos o acciones que invocan.

2.4.4.2. Subcapas

Subcapa de soporte de aplicación (APS)

La subcapa de soporte de aplicación proporciona un interfaz entre la capa de red y la capa de aplicación a través de un conjunto de servicios para ser utilizados tanto por la subcapa ZDO como por AF. Estos servicios son dos: los servicios de datos y los de gestión.

Algunas de las principales tareas que desarrolla esta capa son:

- **Generación de PDU:** a nivel de aplicación, denominada APDU.
- **Vinculación:** una vez que dos dispositivos están vinculados, la subcapa APS se encarga del intercambio de mensajes entre ambos dispositivos.
- **Filtrado de direcciones:** tiene la habilidad de realizar filtros sobre las posibles direcciones destino para crear subgrupos dentro de la red.
- **Fragmentación:** permite la segmentación y reensamblado de mensajes de longitud mayor a la carga útil de un mensaje simple de la capa de red.
- **Evita duplicado:** rechaza aquellos mensajes que entran por duplicado a la capa de aplicación.

Subcapa marco de aplicación (AF)

Es el entorno en el cual se gestionan las distintas aplicaciones definidas. Se permiten hasta 240 aplicaciones distintas en un mismo dispositivo, asociadas a los puntos de acceso 1 al 240. El punto de acceso 0 está reservado al nivel ZDO. Los puntos de acceso 241 al 254 se reservan para futuros usos y el 255 se utiliza para comunicaciones de tipo broadcast.

En esta subcapa también se definen los diferentes clusters que identificarán a cada una de las aplicaciones, denotados por el consiguiente ClusterID

Subcapa de Objeto de dispositivo ZigBee (ZDO)

Esta subcapa es responsable de las siguientes funciones:

- Inicialización de la subcapa APS, de la capa de red (NWK) y del proveedor de servicios de seguridad (SSP, *Security Service Provider*).
- Definición del tipo de dispositivo dentro de la red (coordinador, router o dispositivo final)
- Gestión de vínculos entre puntos de acceso.
- Asegurar conexiones seguras entre dispositivos.

2.4.4.3. Formato de trama

El formato de tramas de la capa de aplicación (APDU) sigue el siguiente esquema:

<table>
<tr><td>Octets:
1</td><td>0/1</td><td>0/2</td><td>0/2</td><td>0/2</td><td>0/1</td><td>1</td><td>0/
Variable</td><td>Variable</td></tr>
<tr><td rowspan="2">Frame control</td><td>Destination endpoint</td><td>Group address</td><td>Cluster identifier</td><td>Profile identifier</td><td>Source endpoint</td><td rowspan="2">APS counter</td><td rowspan="2">Extended header</td><td rowspan="2">Frame payload</td></tr>
<tr><td colspan="5">Addressing fields</td></tr>
<tr><td colspan="8">APS header</td><td>APS payload</td></tr>
</table>

Figura 2.13: Formato de trama de la capa de Aplicación. Fuente [12].

- **Campo de control de trama (*Frame Control*):** Campo de 8 bits que contiene información sobre si se utiliza seguridad, si se requiere mensaje de confirmación a nivel de aplicación y si se utiliza extensión de cabecera.
- **Dirección dispositivo destino (*Destination endpoint*)**: Contiene la dirección del dispositivo al que se dirige la trama.
- **Dirección de grupo (*Group address*):** indica la dirección del grupo de dispositivos al que se destina la trama. Si este campo tiene contenido, no lo tendrá el campo de punto de acceso destino ya que la trama se destinará a los puntos de acceso de los dispositivos pertenecientes al grupo señalado.
- **ClusterID (*Cluster identifier*):** indica el cluster y por tanto la aplicación a la que hace referencia el mensaje.

- **PerfilID (*Perfil identifier*):** identificador del perfil sobre el que se realiza la acción indicada en la trama.

- **Dirección dispositivo origen (*Source endpoint*):** dirección del dispositivo que manda la trama.

- **Contador APS (*APS counter*):** indica el número de trama enviada para evitar duplicados.

- **Extensión de cabecera (*Extender header*):** campo para extender la funcionalidad de la cabecera.

- **Carga útil (*Frame payload*):** contiene la información de las capas inferiores.

2.5. Seguridad.

Con todo método criptográfico, se pretende:

- **Autentificación de datos:** se debe asegurar que la información recibida no ha sido manipulada por un dispositivo ajeno a la comunicación.

- **Confidencialidad de los datos:** la información enviada por los dispositivos sólo debe llegar a los elementos pertenecientes a la red definidos como destinatarios.

Los estándares ZigBee y 802.15.4 definen mecanismos de seguridad para las capas MAC, de red y de aplicación, proporcionando métodos para el establecimiento y transporte de claves, cifrado de tramas y control de dispositivos.

La base de la arquitectura de seguridad son las claves, éstas son distribuidas normalmente por el coordinador, aunque pueden existir elementos en la red dedicados exclusivamente a ello, son los denominados centros de confianza (*trust center*). Es por tanto el coordinador el encargado de actualizar periódicamente las claves y cambiarlas si lo estima necesario.

ZigBee/802.15.4 utiliza claves de 128 bits. Pueden ser asociadas a una red o grupo de dispositivos o bien a un enlace entre dos elementos, el primero de los casos es un método mucho más simple pero existe el riesgo de sufrir un ataque por alguno de los nodos pertenecientes a dicho grupo.

Se utilizan tres tipos de claves:

- **Clave maestra:** Es la base en función de la cual se generan las claves de enlace. La seguridad de toda la red depende de ella ya que los distintos servicios utilizarán variaciones unidireccionales de la clave de enlace para evitar riesgos de seguridad.

- **Clave de enlace:** Dotan de seguridad las comunicaciones punto a punto a nivel de aplicación. Es una clave sólo conocida por los elementos que participan en una comunicación concreta.

- **Clave de red:** Clave utilizada a nivel de red y conocida por todos los elementos pertenecientes a ésta.

Existen dos tipos diferentes de niveles de seguridad:

- **Estándar:** en este nivel, la lista de los dispositivos pertenecientes a la red así como las diferentes claves pueden estar almacenadas en cada uno de los distintos dispositivos. En este caso, el centro de seguridad sólo se encarga de mantener una clave de red común y de controlar las políticas de admisión.

- **Alta seguridad:** en este caso, el centro de seguridad es el único dispositivo que almacena tanto el listado de nodos en la red como las distintas claves de seguridad. A medida que crece el número de dispositivos asociados a la red, debe crecer la capacidad de memoria disponible en el elemento que ejerce de centro de seguridad en la red, ya sea el coordinador o un elemento específico.

El mecanismo de cifrado utilizado se basa en el uso de claves simétricas, esto quiere decir que tanto el dispositivo origen como el destino utilizan la misma clave para descifrar un mensaje. En ZigBee/802.15.4, la encriptación de mensajes sólo protege el interfaz entre diferentes dispositivos, pero no existe cifrado alguno entre las distintas capas de un mismo nodo. Es lo que se conoce como modelo de confianza abierta (*open trust*) y posibilita la compartición de claves de forma mucho más económica.

2.5.1. Seguridad en la capa MAC

La capa MAC se encarga de la propia seguridad de sus tramas, aunque sean los niveles superiores los encargados de determinar el nivel de seguridad a utilizar. La figura siguiente muestra un ejemplo de los campos que tienen que ser incluidos en una trama MAC en los que se indica que se requiere seguridad:

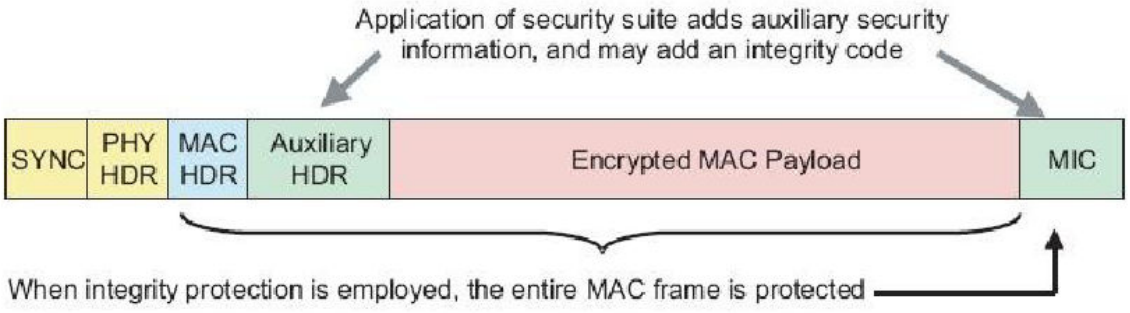


Figura 2.14: Trama con seguridad en capa MAC. Fuente [8].

2.5.2. Seguridad en la capa de red

Cuando una trama originada en la capa de red requiere de seguridad, se hace uso del estándar de encriptación avanzada (AES, *Advanced Encryption Standard*) y se utiliza el modo CCM*, variante del modo CCM (*Counter with CBC-MAC*). La capa de red es la responsable de realizar los pasos necesarios para asegurar la transmisión y recepción de tramas cuyo origen o destino sea esta capa. Las capas superiores controlan estas operaciones mediante la configuración de las claves apropiadas, contadores de tramas y establecimiento del nivel de seguridad a utilizar, esta última acción, a través de la variable *nwkSecurityLevel.*

En [12] (apartado 4.3, pag. 426) se puede obtener en detalle el proceso de realizado por la capa de red para asegurar tanto las tramas de transmisión como para certificar las tramas de recepción.

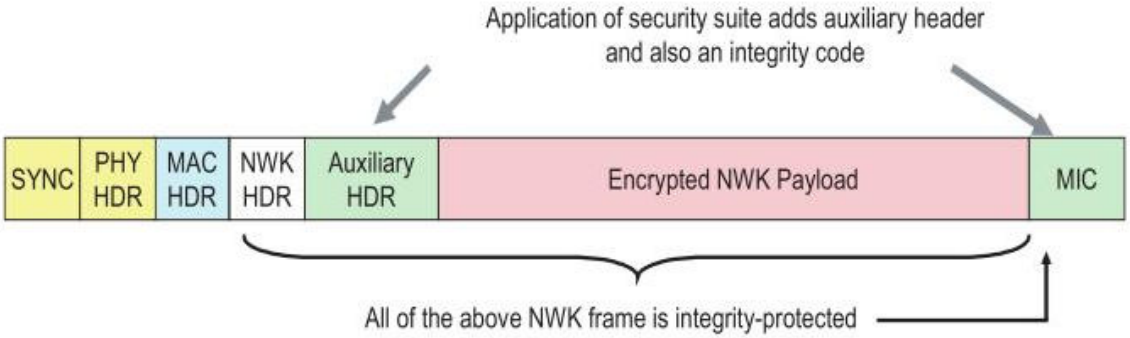


Figura 2.15: Trama ZigBee con seguridad a nivel de capa de red. Fuente [12]

2.5.3. Seguridad en la capa de aplicación

Cuando una trama originada o destinada a la capa de aplicación necesita ser segura, es la capa APS la que se encarga de gestionar todos los procesos necesarios. Es por tanto la capa APS la responsable de realizar los procedimientos de seguridad necesarios para transmitir y recibir tramas y de establecer y gestionar las claves criptográficas. La capa

APS permite basar la seguridad de las tramas en claves de red o claves de enlace, además es la encargada de proporcionar a las aplicaciones y a ZDO el establecimiento de claves, el transporte de claves y la gestión de servicios.

En [12] (apartado 4.4, pag. 434) se puede observar con detalle el procedimiento seguido por esta capa para realizar los procesos de seguridad, tanto en transmisión de tramas como en recepción de las mismas.

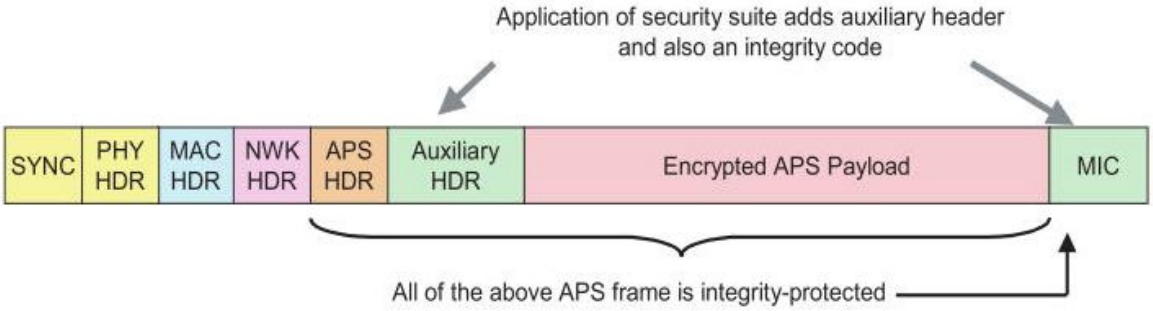


Figura 2.16: Trama ZigBee con seguridad a nivel de capa de aplicación. Fuente [12]

2.5.4. Modo CCM* (*Counter with CBC-MAC*)

CCM* es un modo de cifrado y autenticación de bloques. Usa bloques de encriptación de 128 bits basados en AES-128 (*Advanced Encryption Standard*). Es una variante de CCM en la que coincide en ofrecer autenticación y la posibilidad de encriptación, pero además soporta mensajes que sólo requieran encriptación.

CCM* se base en la combinación del modo contador para la encriptación con el modo de autenticación CBC-MAC. La idea principal es el uso de la misma clave para ambos modos, siempre que no exista colisión entre los valores utilizados en la codificación y los utilizados para la autenticación, es lo que se conoce como uso de claves simétricas por las que el destino y origen utilizan una misma clave para sus procesos de seguridad.

2.6. Implementaciones comerciales

En este apartado se muestran algunos ejemplos de dispositivos comerciales existentes actualmente en el mercado. Clasificando por fabricantes:

Freescale

- **MC1320X:** Transceptor RF basado en el estándar 802.15.4
- **MC1321X System in package:** integra el microcontrolador MC9S098GT con el transceptor MC1320X

- **MC1322X Platform in package:** combinación de transceptor y microcontrolador diseñado para optimizar el ahorro energético del transceptor.

Freescale también ofrece kits de demostración para nuevos usuarios en esta tecnología:

- **ColdFire MCF5282 Demostration Kit:** compuesto por cinco dispositivos y diferentes sensores, ya preparado para mostrar la funcionalidad de una red ZigBee.

Además, Freescale tiene su propia versión de la pila de protocolos ZigBee, conocida por BeeStack, y de la capa MAC, conocida por SMAC.

Ember

- **EM300 Series:** "System on chip" que combina una radio de 2.4 GHz basada en el estándar 802.15.4 con un microprocesador ARM Cortex M3.

- **EM250 SoC (*System on Chip*):** sistema diseñado para proporcionar un gran rango de alcance minimizando el consumo de energía gracias a su excelente sensibilidad y potencia de transmisión. Trabaja en la banda de 2.4 GHz.

- **EM260 Co-Processor:** combina transceptor a 2.4 GHz con un microcontrolador de capacidad limitada. Interfaz basado en conexión SPI/UART (*Serial Peripherical Interface/Universal Asynchronous Receiver Transmitter*) que permite interactuar con cualquier tipo de microprocesador.

Todos los sistemas nombrados trabajan usando una versión propia de ZigBee PRO denominada EmberZNet PRO.

Ember también ofrece kits de desarrollo basados en los tres tipos de elementos antes mencionados, además de software propio, llamado InSight Desktop, con el que programar y configurar éstos.

Jennic

Jennic fabrica tres tipos de microcontroladores, todos ellos basados en 802.15.4 en las capas inferiores y ZigBee PRO en las superiores:

- **JN5148**

- **JN5139**

- **JN5121**

Además proporcionan dos kits de desarrollo:

- **JN5148 ZigBee PRO Evaluation Kit:** contiene 5 dispositivos y todo el hardware y software necesarios para la creación de entornos basados en la pila completa de ZigBee PRO.

- **JN5139 IEEE 802.15.4/JenNet Evaluation Kit:** entorno para evaluar el estándar 802.15.4, montado bajo una capa de red de diseño propio, JenNet.

Texas Instruments (TI)

Texas Instruments proporciona una amplia gama de productos desarrollados para trabajar en entornos ZigBee/802.15.4. Este inventario se centrará en los que trabajan a 2.4 GHz:

- **CC2430/CC2431 SoC:** combina transceptor y microcontrolador 8051 en un único chip. Soporta trabajar con Z-Stack , versión propia del estándar ZigBee y con TI-MAC, versión propia de la capa MAC.

- **CC2420/CC2520 Single-Chip:** transceptor de bajo consumo basado en 802.15.4 y diseñado para trabajar en combinación con un microprocesador MSP430 de TI.

- **CC2480:** Coprocesador diseñado para trabajar junto con un microcontrolador. Está creado específicamente para albergar una versión muy reducida de ZigBee, denominada Z-Accel, con una funcionalidad parecida pero utilizando muchísimas menos funciones de las que el estándar ZigBee proporciona.

Además, TI ofrece al usuario un kit de desarrollo por cada uno de los dispositivos mencionados anteriormente, así, proporciona: CC2430DK (*Development Kit*), CC2431DK, CC2420DK y CC2520DK (ambos kits de desarrollo contiene un Experimenter Board con la posibilidad de utilizar hasta dos microcontroladores MSP430: , además de una serie de sensores, leds y buzzers para facilitar la creación de aplicaciones) y por último, para el CC2480: eZ430-RF2480.

La programación de los microcontroladores se realiza utilizando el entorno de desarrollo IAR Embedded Workbench, además se ofrece gran variedad de software para interactuar con las máximas funcionalidades posibles de estos desarrollos.

En el caso de este proyecto, los elementos utilizados para la creación del entorno de configuración y monitorización han sido kits de desarrollo de Texas Instruments, más

concretamente el CC2420DK. En primer lugar TI es una compañía con gran experiencia en el desarrollo de dispositivos electrónicos y más concretamente en el diseño de sistemas dedicados a comunicaciones inalámbricas. En segundo lugar, este kit de desarrollo escogido proporciona todo lo necesario para el correcto funcionamiento del diseño propuesto, tanto por la facilidad que nos ofrece para diseñar distintas aplicaciones, como por su sencillez a la hora de interactuar con un ordenador donde poder visualizar todos los procedimientos realizados.

CAPÍTULO 3

Sistema de desarrollo empleado

En este capítulo se expondrán todas aquellas herramientas que han sido imprescindibles para el correcto desarrollo de este proyecto. Por un lado se expondrán las características del sistema hardware que se ha utilizado, el kit de desarrollo CC2420DK de TI (*Texas Instruments*), compuesto por el dispositivo transceptor CC2420 y una placa experimental con dos microcontroladores integrados, el MSP430FG4618 y el MPS430F2013. Por otro lado se detallarán cada uno de los elementos software que han sido necesarios para crear y entender cómo funciona una red basada en el estándar ZigBee/802.15.4.

3.1. Kit de desarrollo CC2420DK

3.1.1. Introducción

Texas Instruments ofrece tres alternativas de arquitecturas hardware con las que crear redes ZigBee/802.15.4:

- Arquitectura basada en un transceptor 802.15.4 y un microprocesador. El transceptor se ocupa de la capa física y el microprocesador aloja la capa MAC, de red y aplicación. Esta estructura otorga una gran flexibilidad en el diseño de red ya que el desarrollador tiene el control de prácticamente toda la pila de protocolos y por tanto puede modificar la mayoría de los parámetros de configuración existentes.

- Arquitectura basada en un procesador que alberga las capas física, MAC y de red y un microprocesador que controla la capa de aplicación. En este caso, el desarrollador sólo tiene acceso a la capa de aplicación. Por lo tanto se pierde flexibilidad a favor de ganar simplicidad y de liberar al microprocesador de las tareas relacionadas con la red.

- Arquitectura SoC (*System on Chip*) en la que tanto el transceptor como el microprocesador están integrados en un único chip.

El kit de desarrollo utilizado en este proyecto pertenece al primer grupo; está compuesto por un módulo que contiene el transceptor radio funcionando según el estándar 802.15.4 denominado CC2420EM (*Evaluation Module*) y una placa experimental con dos microprocesadores MSP430, el MSP430FG4618 y el MSP430F2013.

En la figura 3.1 se muestran ambos elementos:

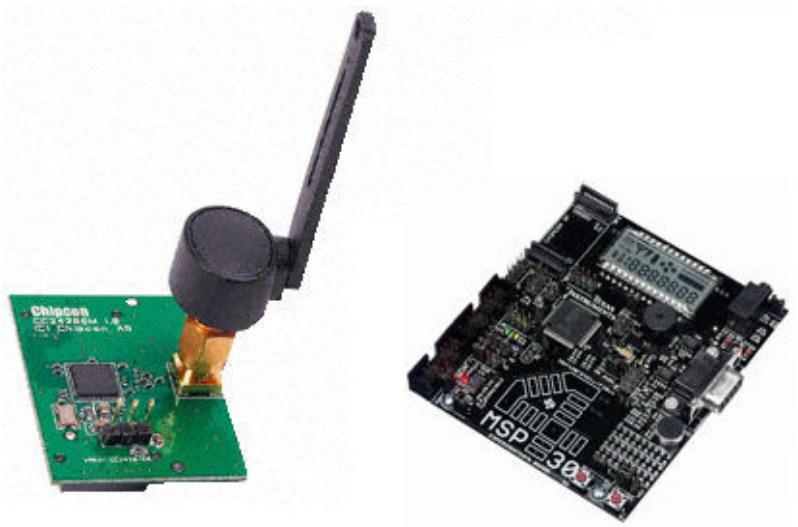


Figura 3.1: Componentes CC2420DK. Fuente [37]

El kit de desarrollo contiene además, el hardware necesario para poder programar los microprocesadores integrados en la placa experimental. Esta programación se realiza a través de un entorno software (IAR *Embedded Workbench*) y se carga en los microprocesadores utilizando un FET *Debugger* (*Flash Emulation Tool*), que introduce el código de forma que los microprocesadores entiendan las instrucciones creadas. Este depurador se muestra en la figura 3.2:

Figura 3.2: Depurador MSP-FET430UIF. Fuente [37].

3.1.2. Módulo CC2420EM

El módulo de evaluación CC2420EM contiene el chip CC2420 además de la circuitería externa necesaria para su correcto funcionamiento, entre esta circuitería se encuentra la antena, un cristal a 16 MHz y un *jumper* con el que seleccionar si se va a utilizar el regulador de tensión del chip CC2420 o el regulador de tensión de la placa experimental.

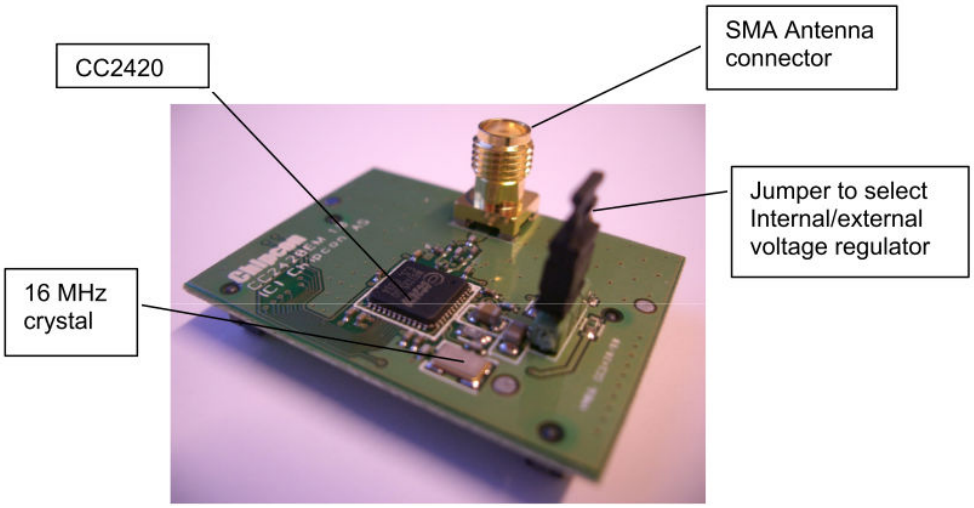


Figura 3.3: CC2420EM. Fuente [34].

3.1.3. Placa experimental

La placa experimental se utiliza como placa base sobre la que montar el módulo de evaluación CC2420EM, proporcionándole a éste tanto alimentación como conexiones externas. Además de los dos microprocesadores, la placa experimental contiene un conjunto de elementos muy útiles para la realización de pruebas que en este proyecto se han utilizado para crear diversas aplicaciones con las que entender, configurar y monitorizar una red ZigBee/802.15.4. Este subconjunto de elementos está compuesto por:

- 4 diodos LEDs (*Light Emitting Diodes*).
- Un zumbador (*buzzer*).
- Un micrófono y una salida de audio
- 2 pulsadores.
- Una pantalla LCD (*Liquid Crystal Display*).
- Superficie táctil capacitiva.

- Un puerto serie de 9 pines.
- 2 entradas JTAG para la programación de los microprocesadores.
- Control de alimentación.

La ubicación de todos estos elementos en la placa experimental se puede observar en la siguiente figura:

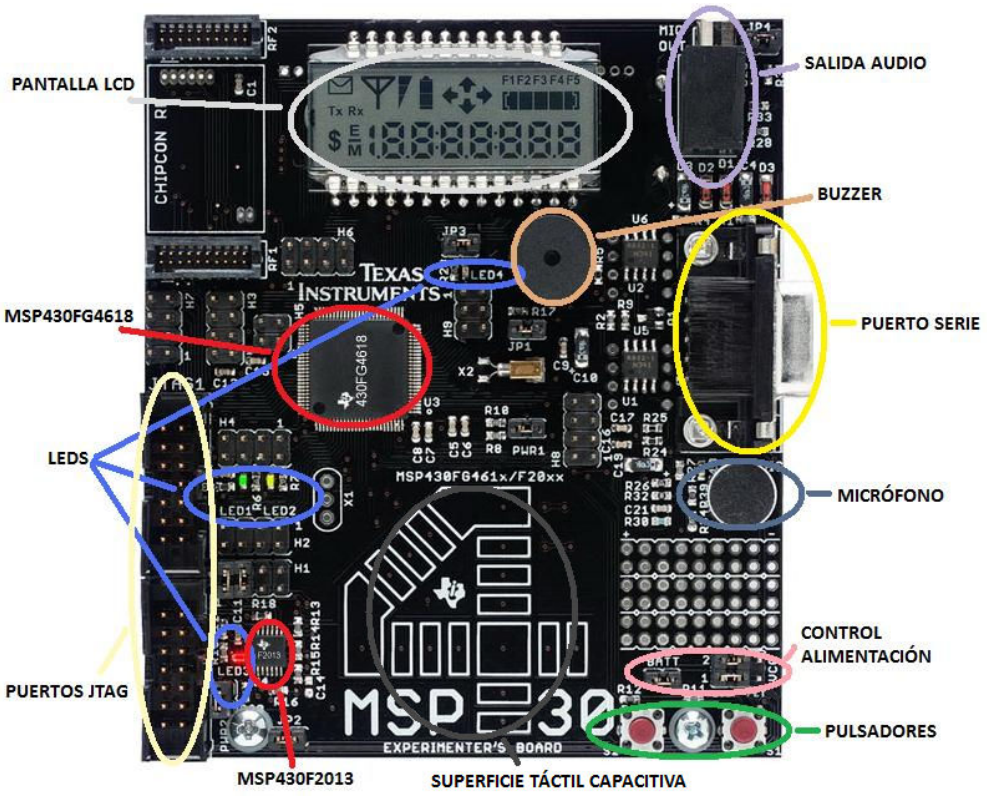


Figura 3.4: Placa experimental. Fuente [25].

A continuación se describirán más detalladamente cada uno de los elementos que constituyen la placa experimental [25], dividiéndolos en subgrupos según funcionalidad:

Alimentación

- **Control de alimentación:** La alimentación de la placa se puede obtener de tres posibles fuentes:
    - Mediante dos baterías AAA integradas en la placa.
    - Por el FET Debugger a través de la conexión JTAG.
    - Por una fuente externa.

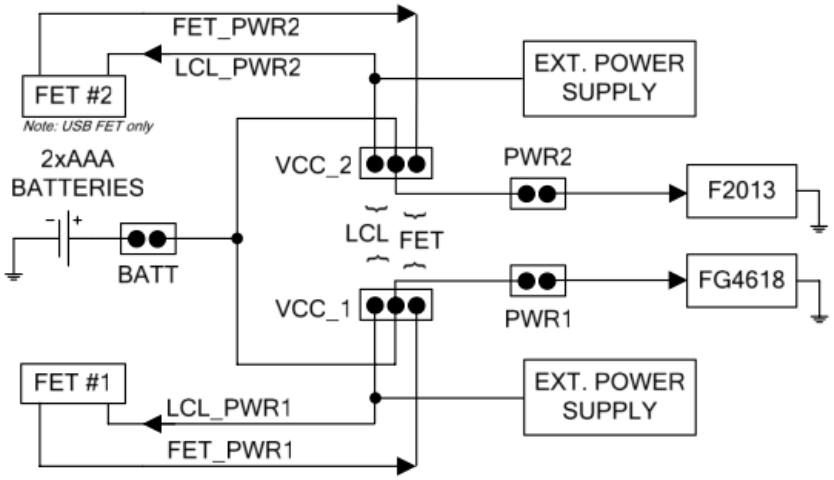


Figura 3.5: configuración de *jumpers* para selección de fuente de alimentación. Fuente [25]

El tipo de fuente de alimentación se selecciona utilizando dos *jumpers*. Conectando el etiquetado como BATT se activa la alimentación a través de las baterías AAA. Con los etiquetados VCC_1 y VCC_2 se selecciona el microprocesador (VCC_1 para el MSP430FG4618 y VCC_2 para el MSP430F2013) y una de las otras dos posibles fuentes según el *jumper* interconecte las dos posiciones más a la derecha (FET), en cuyo caso utilizaría alimentación a través del FET *Debugger* o las posiciones más a la izquierda (LCL), donde la alimentación se recibiría de una fuente externa.

Como se puede observar en la Figura 3.5, existen además dos *jumpers* etiquetados como PWR1 y PWR2, con ellos se decide si alimentar de forma individual cada uno de los microprocesadores: PWR1 para el MSP430FG4618 y el PWR2 para el MSP430F2013), además se pueden utilizar para realizar mediciones de consumo de cada MSP por separado.

A continuación se muestran los *jumpers* en una posible configuración que indicaría que la alimentación se produce a través de las baterías AAA.

Figura 3.6: Selección de *jumpers* para alimentación por baterías.

Interfaces

- **Pantalla LCD:** Controlable por el MSP430FG4618, soporta operaciones 4-MUX. En ella se mostrarán los diferentes tipos de mensajes que recibe un dispositivo además de indicar la acción que está realizando un nodo, ya sea transmisión o recepción.

- **Pulsadores:** Existen dos pulsadores etiquetados como S1 y S2 (ver Figura 3.4) que interrumpen, al ser pulsados, al microcontrolador MSP430FG4618. Cada uno de estos pulsadores tendrán asignados una determinada función en la aplicación propuesta por este proyecto que se detallará más adelante.

- **LED:** La placa experimental contiene cuatro diodos LED, los LED 1, 2 y 4 están controlados por el MSP430FG4618, mientras que el LED 3 se controla a través del MSP430F2013. Su función principal es la de visualización y se asociarán a ciertas funciones para comprobar el correcto funcionamiento de éstas, además de a determinados mensajes que encenderán un LED en un dispositivo asociado a la red.

- ***Buzzer*:** El zumbador está conectado a un puerto del MSP430FG4618 aunque puede ser desactivado a través del jumper J1. Se utilizará en este proyecto como recurso para el envío de cierto tipo de mensajes, los cuales activarán el zumbador en un determinado dispositivo.

- **Superficie táctil capacitiva:** Consta de 16 segmentos cuya actividad se monitoriza en el MSP430F2013. Los resultados de dicha monitorización se mandan al MSP430FG4618 a través de la interconexión de ambos microprocesadores.

Comunicación con periféricos

- **Módulo de evaluación:** Acepta los módulos CC2420EM, CC2500EM a 2.4 GHz y el CC1100EM en la banda de 868 MHz. Se conectan a través del USART del MSP430FG4618 configurado en modo SPI.

- **Puerto serie RS-323 de 9 pines:** Controlado por el MSP430FG4618 permite la comunicación del dispositivo con el PC (*Personal Computer*) configurado en modo UART. La tasa de transmisión y recepción se configuran vía software.

- **I2C/SPI:** Ambos microprocesadores soportan protocolos I2C y SPI. Estos protocolos se utilizan para la comunicación entre procesadores.

Señal analógica

- **Micrófono:** Conectado a un puerto del MSP430FG4618, puede ser activado y desactivado por dicho MSP430 con el fin de minimizar el consumo.

- **Salida analógica:** Controlado por el MSP430FG4618. Existen varias opciones de atenuación configurables en el MSP430 y con el jumper J4.

En la figura siguiente se muestra un esquema de la interconexión de cada uno de los elementos con el microprocesador del que depende su funcionamiento:

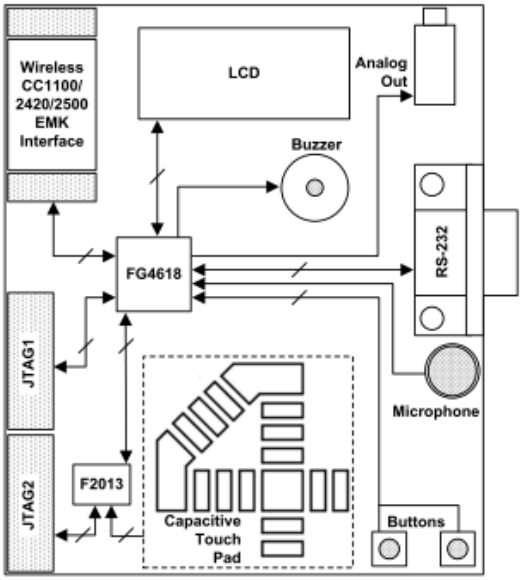


Figura 3.7: Interconexiones entre los elementos de la placa experimental. Fuente [25]

3.2. CC2420

3.2.1. Introducción.

El módulo CC2420 es un transceptor diseñado para operar en la banda ISM a 2.4 GHz, basándose en el estándar IEEE 802.15.4 para su funcionamiento. Es un dispositivo específicamente diseñado para el desarrollo de aplicaciones inalámbricas de muy bajo consumo, lo que lo hace idóneo para formar parte del hardware necesario para su utilización en redes ZigBee/802.15.4.

Su uso más frecuente se realiza junto a un microcontrolador y una serie de componentes pasivos externos, o circuito de aplicación, como se verá a continuación

3.2.2. Circuito de aplicación

El correcto funcionamiento del transceptor CC2420 requiere de algunos elementos externos, los cuales acompañan al dispositivo en la placa CC2420EM:

- **Entrada/salida**: La entrada/salida de RF es diferencial (pines RF_N y RF_P). El valor óptimo de carga diferencial para el puerto RF es 115 + j180 Ω. A través del pin TXRX_SWITCH se controla si se está en modo recepción o transmisión, de forma que en modo recepción dicho pin se conecta a tierra y en modo transmisión se haga a la tensión de polarización.

  Si se utiliza una antena desbalanceada (como en este caso) se requiere la utilización de un *balun* para optimizar el rendimiento, éste puede implementarse utilizando inductores y capacidades en combinación o no de líneas de transmisión (ver figura 3.8)

- **Resistencia de polarización**: una resistencia conectada al pin 45 (ver figura 3.8) se encarga de proporcionar la corriente exacta de polarización.

- **Cristal**: un cristal externo acompañado de dos capacidades de carga se utiliza como oscilador, tal y como se indica en la figura 3.8. Estas capacidades tienen un valor de 27 pF para una capacidad total vista desde los terminales del cristal C_L = 16 pF según la expresión: ($C_{parasita}$ = 2 pF – 5 pF)

$$\frac{1}{\frac{1}{C381}+\frac{1}{C391}} + Cparasita$$

  Capacidades C381 y C391 en serie y paralelas ambas a la $C_{parasita}$.

  Este cristal se conecta entre los terminales XOSC16_Q1 (pin 39) y XOSC16_Q2 (pin 38) y puede ser utilizado como referencia para la frecuencia principal a 16 MHz.

- **Regulador de tensión**: Un regulador de tensión proporciona 1.8 V a todas las entradas de alimentación internas. No puede ser utilizada para alimentación de elementos externos debido a que tiene una capacidad de entrega limitada. Este regulador de tensión se activa/desactiva a través del pin VREG_EN, aunque su desactivación implica la pérdida de la información almacenada en memoria. Requiere de una capacidad que proporcione estabilidad a dicho regulador.

- **Capacidades de desacoplo**: Una adecuada utilización de capacidades de desacoplo es necesaria para obtener un rendimiento óptimo. La disposición de estas capacidades se detalla en la documentación del fabricante.

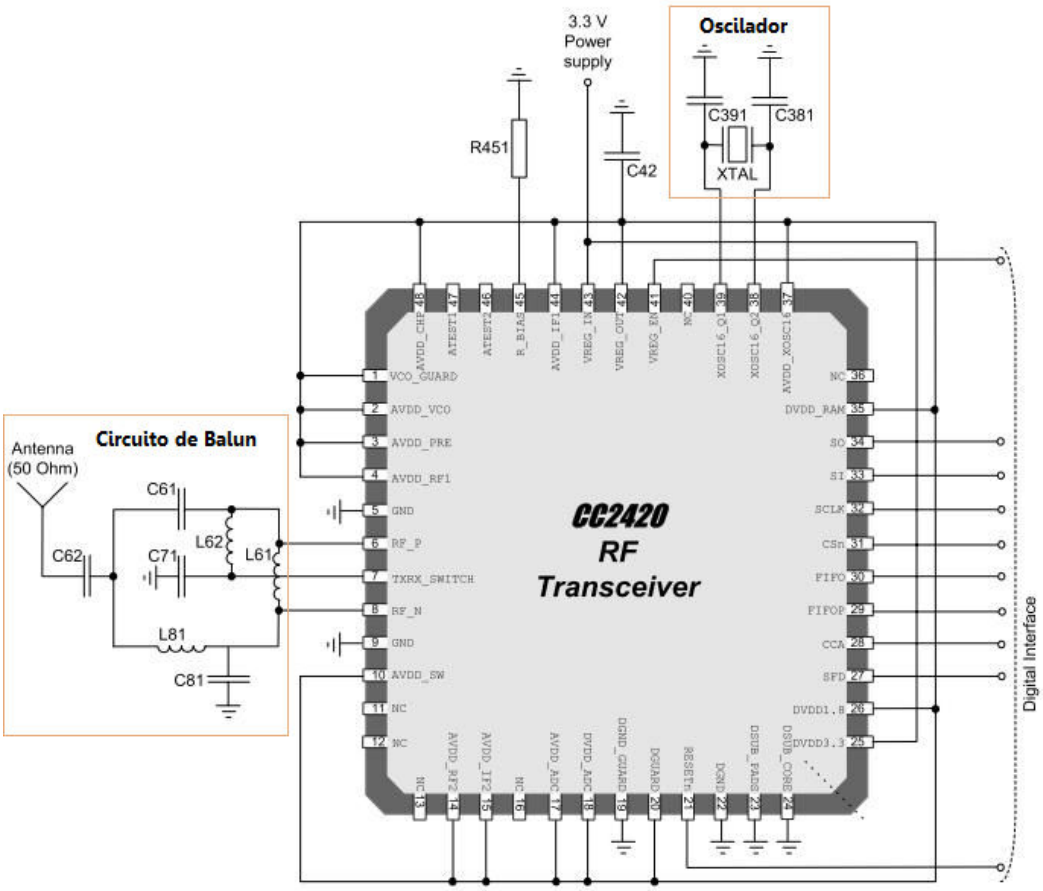


Figura 3.8: Circuito de aplicación típico con balun. Fuente [32].

3.2.3. Principales características del transceptor CC2420

En este apartado se pretenden destacar de forma resumida alguna de las principales características en las que se basa CC2420 para su funcionamiento.

Reconocimiento de direcciones

CC2420 tiene la capacidad de permitir o no el reconocimiento hardware de direcciones, con el fin de detectar si una trama recibida es realmente para un dispositivo concreto en cuyo caso la aceptará o no es para ese dispositivo y la rechazará. Este reconocimiento se basa en los siguientes requerimientos:

- El subcampo tipo de trama no debe contener ningún tipo de trama desconocido.

- Si el subcampo tipo de trama indica que se trama de una trama de baliza, el identificador PAN de la fuente debe contener el valor de la variable *macPANid* a menos que esta variable contenga el valor 0xFFFF (difusión) en cuyo caso se aceptará la trama de baliza siempre.

- Si se incluye el identificador de PAN destino, deberá coincidir con *macPANid* a menos que contenga el valor 0xFFFF.

- Si se incluye en la trama la dirección corta destino, esta coincidirá con *macShortAddress*.

- Por último, si sólo se incluye la dirección corta origen, se aceptará la trama sí el dispositivo es coordinador y su identificador de PAN fuente coincide.

Trama ACK

CC2420 permite el envío automático de tramas ACK, en cuyo caso realiza la acción cada vez que reciba una trama correcta, es decir, que supere el reconocimiento de dirección, que tenga la bandera de petición de trama ACK activa y su campo CRC válido.

RSSI (*Received Signal Strength Indicator*) / Detección de energía

CC2420 incorpora mecanismos de cálculo de RSSI ofreciendo un valor de 8 bits de resolución. Este valor se calcula como la media durante 8 periodos de símbolo (128 us) del valor de RSSI y se actualiza continuamente. Un bit de estado denominado RSSI_VALID indica si este valor es válido.

La medida de RSSI puede utilizarse como detector de energía ya que está relacionada con la potencia recibida de RF según la expresión:

$$P\ (Power) = RSSI\_VAL + RSSI\_OFFSET\ [dBm]$$

Donde RSSI_OFFSET se calcula de forma empírica durante el desarrollo del sistema y tiene un valor aproximado de -45 dBm.

LQI (*Link Quality Indicator*)

La medida de la calidad de un enlace hace referencia a la fuerza y/o calidad con la que se recibe un paquete de datos. Para realizar esta medida, CC2420 proporciona un valor de

correlación media basado en los 8 primeros símbolos tras el campo SFD. La correlación media (CORR) se convierte en un valor de LQI según la expresión:

$$LQI = (CORR - a) \times b$$

donde a y b vienen dadas mediante una estimación basándose en medias de PER (*Packet Error Rate*).

CCA (*Clear Channel Assessment*)

La señal de CCA se obtiene a partir de la medida de RSSI y un umbral programable. Es una función usada para poder implementar CSMA-CA (como ya se mencionó en el apartado anterior CSMA-CA es el protocolo de acceso al medio utilizado por ZigBee/802.15.4). CC2420 permite programar el umbral en saltos de 1 dB e implementa los tres posibles tipos de CCA existentes. La señal de CCA se obtiene en el pin de salida CCA activo a nivel alto.

3.2.4. Máquina de estados de control

CC2420 tiene integrada una máquina de estados que utiliza para conmutar entre los diferentes modos de operación (figura 3.9). Existe un registro de estados (FSMSTATE) donde se indica en cada momento el valor de aquellos comandos de los que depende la situación en un instante dado en la máquina de estados.

Antes de cualquier recepción o transmisión, el regulador de tensión y el oscilador (cristal) deben estar encendidos y permanecer estables. El oscilador es controlado por el comando SXOSON/SXOSOFF mientras que la estabilidad es indicada por un bit, XOSC16M_STABLE, en el registro de estados. Una vez que el bit XOSC16M_STABLE indica que del oscilador es estable se decide por transmitir o recibir según los comandos STXON para transmisión y SRXON para recepción.

CC2420 puede permanecer en modo de bajo consumo (modo *Power Down*) en cuya situación puede esperar hasta que exista una petición de inicio de sesión tanto de transmisión como de recepción. Además, en este modo se pueden realizar los ajustes necesarios en los registros de configuración para adaptar el dispositivo a la frecuencia y modos de trabajo necesarios.

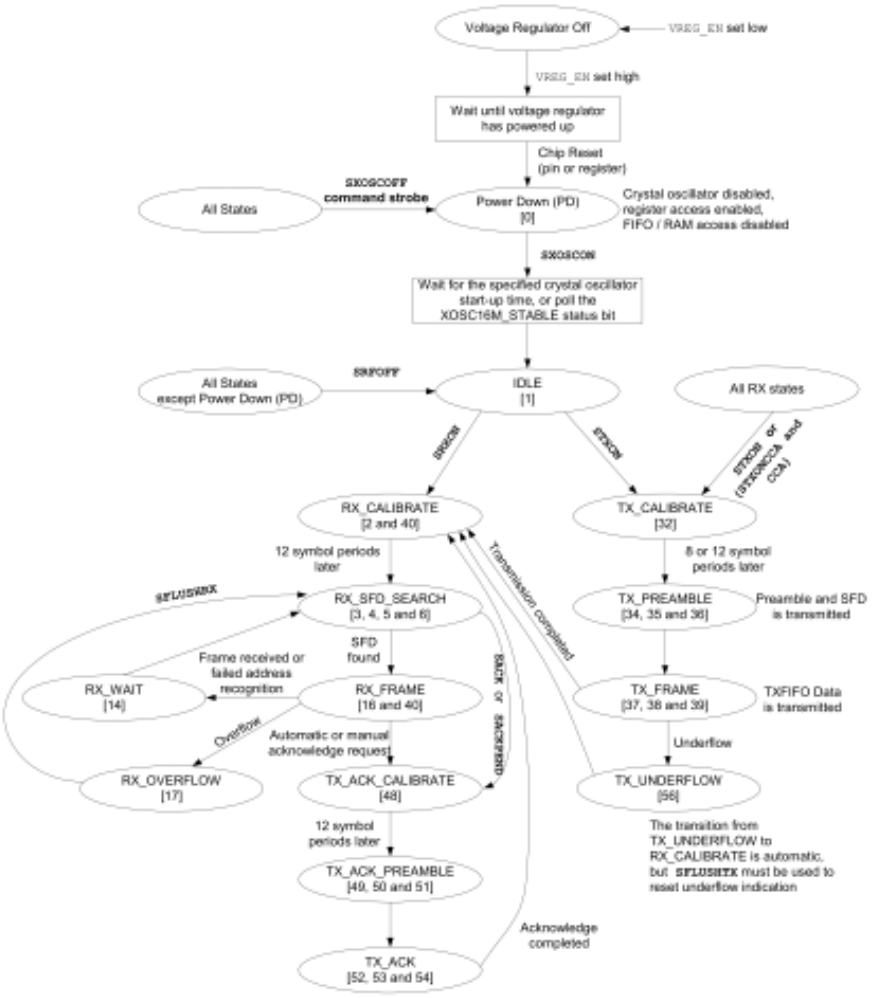


Figura 3.9: Máquina de estados de CC2420. Fuente [32].

3.2.5. Interfaz entre CC2420 y MSP430

La utilización habitual del CC2420 va acompañada de un microcontrolador, en el caso de este proyecto CC2420 + MSP430FG4618. Este microcontrolador debe permitir:

- Programar CC2420 en diferentes modos de funcionamiento, lectura y escritura de buffers de datos e información de estado vía bus serie SPI 4-wire

- Ofrecer un interfaz con las pilas FIFO tanto para recepción como transmisión utilizando los pines FIFO y FIFOP.

- Interfaz con el pin CCA (*Clear Channel Assessment*) para asignación de canal.

- Interfaz con el pin SFD (*Start of Frame Delimiter*) para información temporal, sobre todo para el caso de operar en redes balizadas.

3.2.5.1. Configuración del interfaz

Como se acaba de comentar, el microcontrolador debe usar 4 pines de entrada/salida para la comunicación serie en modo SPI, estos pines son: SI, SO, SCLK y CSn. SO debe conectarse a un pin de entrada en el microcontrolador mientras que los otros 3 van a pines de salida, como se puede observar en la figura 3.10.

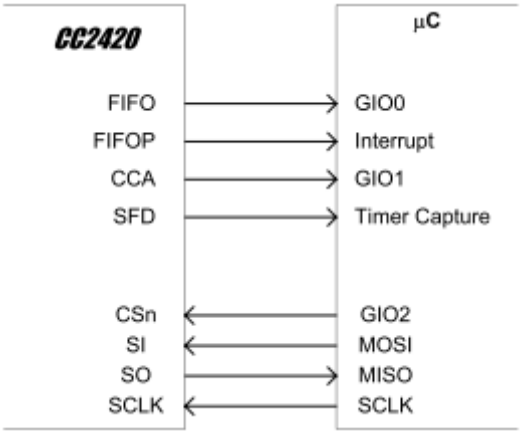


Figura 3.10: Interfaz CC2420-MSP430. Fuente [32].

3.2.5.2. Modo recepción

En modo recepción el pin SFD se activa a nivel alto tras la recepción del campo de trama SFD y vuelve a su estado inicial de reposo tras recibir el último byte del campo MPDU indicando por tanto que se ha recibido una nueva trama. Si en cualquier momento de la recepción existe algún fallo como por ejemplo que no se reconoce la dirección, el pin SFD pasaría inmediatamente al estado de reposo.

El pin FIFO se establece a nivel alto mientras existan datos en el registro RXFIFO y el pin FIFOP sólo pasará a nivel alto si se supera un umbral de bytes no leidos almacenados en el registro RXFIFO y en caso de estar activa la función de reconocimiento de dirección, nunca antes de haber superado dicho reconocimiento (aunque se supere dicho umbral). El registro RXFIFO tiene capacidad para almacenar hasta 128 bytes y puede estar rellena de múltiples tramas siempre que no se supere esta capacidad. En caso de existir *overflow*, esta situación queda reflejada con el pin FIFO a nivel bajo y el pin FIFOP a nivel alto.

En la siguiente figura se muestra el comportamiento de dichos pines en una situación de recepción de trama correcta sin problemas de *overflow*.

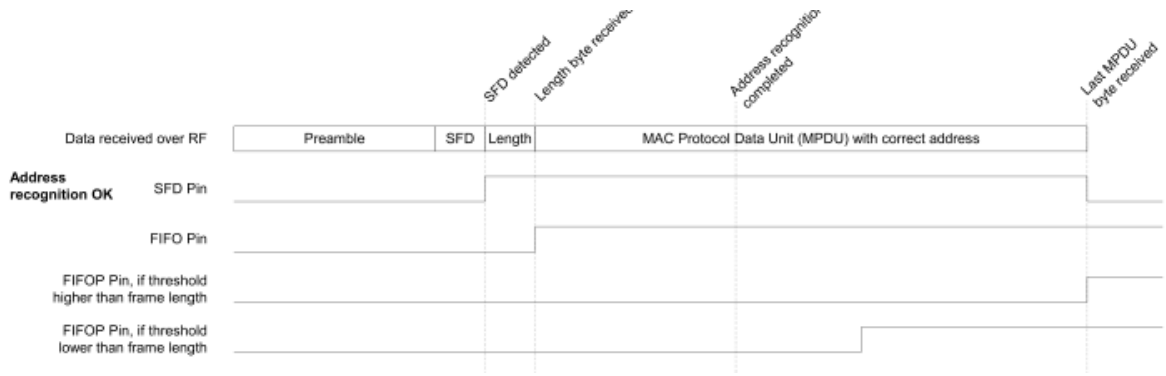


Figura 3.11: Estado de los pines del CC2420 en situación de recepción de trama. Fuente [32].

3.2.5.3. Modo transmisión

La transmisión de una trama se indica a través del pin SFD, éste permanecerá a nivel alto tras el envío del campo SFD y no pasará a nivel bajo hasta se que haya mandado el último byte de la trama o se detecte *underflow*. Todos los datos a enviar en dicha trama son obtenidos del registro de transmisión TXFIFO en el cual se han almacenado previamente. Los pines FIFO y FIFOP no tienen ninguna funcionalidad en la transmsión.

En la siguiente figura se puede observar el comportamiento de los pines en situación de transmisión:

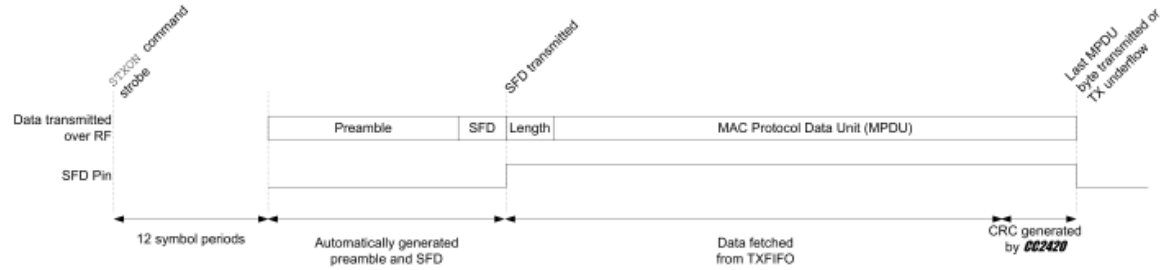


Figura 3.12: Estado de los pines del CC2420 en situación de transmisión de trama. Fuente [32].

3.3. MSP430FG4618

3.3.1. Introducción

Como ya se ha comentado en apartados anteriores, la placa experimental utilizada en este proyecto contiene integrados dos microprocesadores, el MSP430FG4618 y el MSP430F2213. Este segundo microprocesador funciona como apoyo al primero controlando el LED 3 y la superficie táctil capacitiva. Debido a que no ha sido necesaria la utilización de ninguno de estos dos periféricos, no se requirió el uso de este segundo

microprocesador. Por tanto a continuación sólo se comentarán las principales características del chip MSP430FG4618.

MSP430FG4618 es uno de los cuatro modelos de la familia MSP430FG461X, las diferencias entre unos y otros dispositivos reside en la capacidad de memoria de que disponen. En el caso del modelo FG4618, posee 116 Kbytes + 256 Bytes de memoria *flash* y 8 Kbytes de memoria RAM. También dispone de un interfaz para comunicaciones serie que permitirá la monitorización de su funcionalidad a través del PC y un controlador de LCD que se utilizará para la visualización de mensajes.

3.3.2. Arquitectura

Los aspectos más importantes de la arquitectura de un MSP430 son: (Véase la figura 3.13)

- Microprocesador de tipo RISC (*Reduced Instruction Set Computer*) de 27 instrucciones y 16 bits, es decir, todos sus registros internos y buses de datos son de 16 bits.

- Creado específicamente para destacar por su bajo consumo, recurre a un DCO (*Digitally Controled Oscilator)* que le permite pasar desde el modo de bajo consumo a modo activo en 6 us.

- Memoria programable *In-system* que proporciona gran flexibilidad para cambios en el código y actualizaciones.

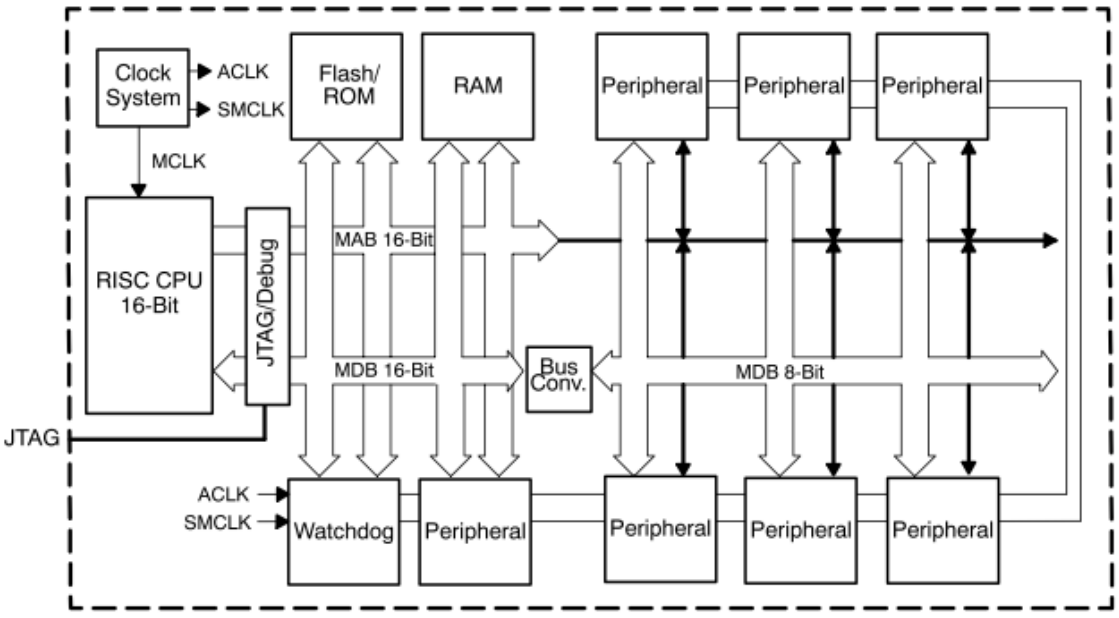


Figura 3.13: Arquitectura de un MSP430. Fuente [24]

3.3.3. Características

En este apartado se detallarán las características más importantes que aporta un chip MSP430, algunas de las cuales se han utilizado en el desarrollo de este proyecto.

Controlador de dispositivo LCD (*LCD_A Controller*)

Este controlador se encarga de gestionar el display de un LCD. Su configuración se define vía *software* por el usuario.

MSP430 acepta dispositivos LCD estáticos, bien de tipo 2-mux, 3-mux o 4-mux. En el caso de este proyecto se ha utilizado un LCD 4-mux, lo que quiere decir que cada dígito del LCD está gestionado por dos pines denominados SP1 y SP2 (*Segment Pin*) cada uno de los cuales controla 4 segmentos de dicho dígito. Así en combinación con las señales recibidas en los pines COM0, COM1, COM2 y COM3 se visualiza en la pantalla el dígito deseado.

La Figura 3.14 muestra este reparto de segmentos para los pines SPx y COMx:

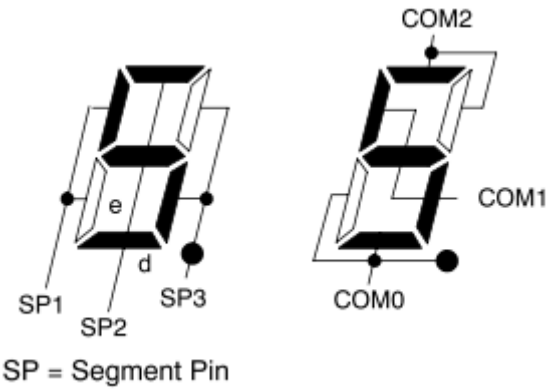


Figura 3.14: Función de pines SPx y COMx. Fuente [24].

Temporizador de vigilancia (WDT, *Watchdog Timer*)

El temporizador de vigilancia es un reloj de 16 bits cuya principal función es realizar un reinicio del sistema controlado cuando aparece algún problema de tipo *software*.

Este temporizador no es accesible directamente por *software* sino que para su configuración hay que acceder al registro WDTCTL, el cual está protegida por clave tanto para escritura como lectura.

Su funcionamiento se basa en la activación de un temporizador con un tiempo límite determinado por el usuario. Si antes de que este tiempo expire el usuario no realiza acciones de configuración, detención o eliminación de WDT o bien si se introduce una

clave errónea al intentar acceder al registro de configuración WDTCTL el sistema manda una señal a través del pin RST/NMI que implica el reinicio de todo el sistema. Así se pretende que ante la aparición de determinados problemas *software* (principalmente bucles infinitos) el sistema sea capaz de reaccionar y recuperar un estado de actividad normal.

Interfaces de comunicaciones serie

USCI (*Universal Serial Communication Interface*) soporta múltiples modos de comunicaciones serie en un único módulo *hardware*, estos modos son: UART (*Universal Asynchronous Receiver-Transmitter*), I^2C y SPI (*Serial Peripheral Interface*).

A continuación se detallarán de forma resumida las principales características de cada uno de ellos:

UART: En el modo asíncrono el MSP430 se conecta vía dos pines, UCAxRXD y UCAxTXD, a un sistema externo, seleccionando comunicación mediante UART a través del pin UCSYNC.

Este modo de comunicación es el utilizado para las transferencias de datos vía puerto seria hacia y desde el PC.

Las características de UART son las siguientes:

- 7 u 8 bits de datos con bit de inicio y posibilidad de añadir bit par/impar/no paridad, bit de modo de direccionamiento y uno o dos bits de parada. (Ver figura 3.15).

- Registros, *buffers* y capacidad de interrupciones independientes para transmisión y recepción.

- Detector de inicio de trama en el receptor para acciones de autoactivación en modos de funcionamiento de bajo consumo.

- Tasa de transferencia programable.

- Detector de errores y supresión de tramas.

- Programación del orden en el envío de datos de una trama, LSB (*Least Significant Bit*) o MSB (*Most Significant Bit*) primero.

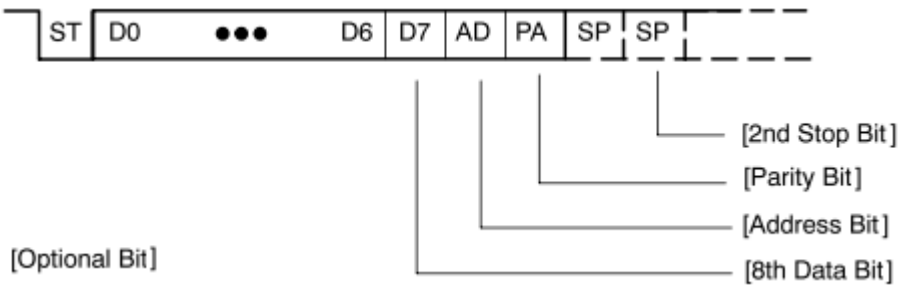


Figura 3.15: Formato de trama UART. Fuente [24].

I²C: Este modo de funcionamiento se utiliza para comunicar el MSP430 con un dispositivo compatible mediante un bus serie I^2C. En el caso del MSP430FG4618 en la placa experimental utilizada, éste hace uso de dicho bus de datos para comunicarse con el otro microprocesador disponible en la placa, el MSP430F2213. La forma de comunicación es mediante el mecanismo maestro-esclavo en el que el MPS430 toma el rol de maestro mientras que los demás dispositivos se comunican con él como esclavos.

Sus principales características son:

- Modos de direccionamiento de 7 y 10 bits.
- Función de inicio, reinicio y parada.
- Modo de transmisión/recepción multi-master.
- Modo de transmisión recepción esclavo.
- Tasa de transferencia de datos superiores a 100 Kbps en modo estándar y superiores a 400 Kbps en modo rápido.
- Frecuencia en modo maestro programable.
- Diseñado para bajo consumo permite estado "dormido" de los dispositivos, utilizando la detección de inicio como señal para "despertar".

Un dispositivo conectado a un bus I^2C puede ser considerado maestro o esclavo cuando se lleva a cabo una transferencia de datos. Este reparto de roles se lleva a cabo de la siguiente manera: el maestro inicia la comunicación de datos y genera una señal de reloj SCL (*Serial Clock*). Cualquier dispositivo direccionado por el maestro es considerado por éste como su esclavo.

La comunicación mediante el bus I^2C se realiza a través del pin SDA (*Serial Data*) y el reloj serie se trasmite por el pin SCL. Ambos pines son bidireccionales.

En la siguiente figura se muestra un diagrama de conexionado típico.

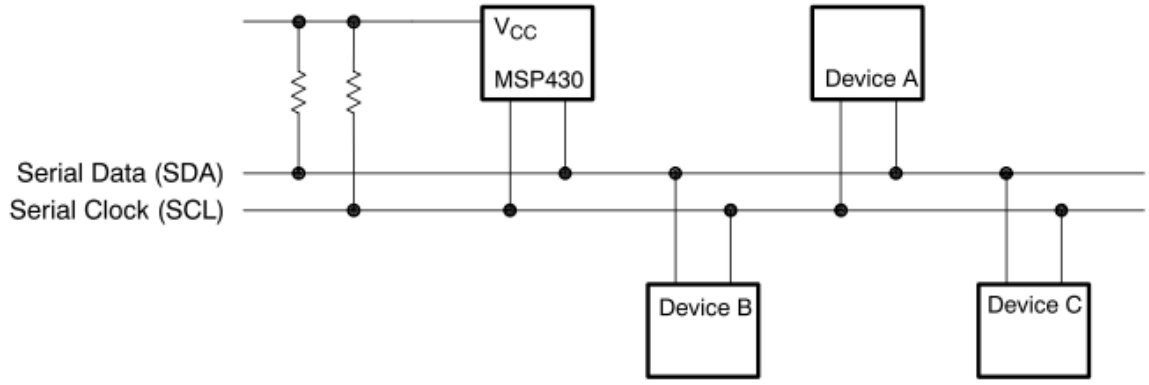


Figura 3.16: Diagrama de conexión del bus I^2C. Fuente [24].

SPI: En modo síncrono el MSP430 se conecta a un sistema externo a través de 3 ó 4 pines: UCxSIMO, UCxSOMI, UCxCLK y UCxSTE. Este modo de funcionamiento se selecciona mediante el pin UCSYNC.

En el caso de este proyecto, se utiliza este modo de comunicación para conectar el MSP430 con el transceptor CC2420.

Las características del modo SPI son:

- Modo de funcionamiento según mecanismo de Maestro-Esclavo.
- Capacidad de selección de funcionamiento con 7 ó 8 bits de datos.
- Programación por parte del usuario del orden de envío de bits de una trama, primero LSB o MSB.
- *Buffers*, registros y capacidad de interrupción independientes para transmisión y recepción.
- Polaridad de reloj y control de fase seleccionable.

En este modo de comunicación, todos los dispositivos esclavos comparten una señal de reloj proporcionada por el maestro. El intercambio de datos se realizada de acuerdo con 3 ó 4 pines (depende de configuración de la comunicación serie) cuya funcionalidad se expone a continuación:

- UCxSIMO: si el dispositivo es el maestro, esta será la línea de datos de salida y si el dispositivo se configura como esclavo, este pin será de entrada de datos.

- UCxSOMI; si el dispositivo es maestro, será línea de datos entrantes y si el dispositivo es esclavo será línea de datos de salida.

- UCxCLK: es la línea por la que circula la señal de reloj, con ella se sincronizan todos los dispositivos y por tanto es un pin de salida para el caso del maestro, pues es éste quien genera la señal y un pin de entrada para los esclavos.

- UCxSTE: si se utiliza una transmisión a 4 pines, se está posibilitando la existencia de varios dispositivos maestros. Este pin se utiliza en dicho caso para permitir o no a un esclavo la transmisión. En el modo de operación a 3 pines no se utiliza.

Digital I/O

MSP430 contiene hasta 10 puertos digitales de entrada/salida implementados. Cada uno de los puertos requiere de 8 pines individualmente configurados para ser entrada o salida de información. Esta configuración se realiza a través de *software* por parte del usuario. Los puertos 1 y 2 tienen la posibilidad de habilitar interrupciones, es por ello que el puerto 1 se dedica en la placa experimental utilizada en este proyecto para el control de los pulsadores.

En la siguiente figura se muestra la distribución de pines de un chip MSP430FG4618.

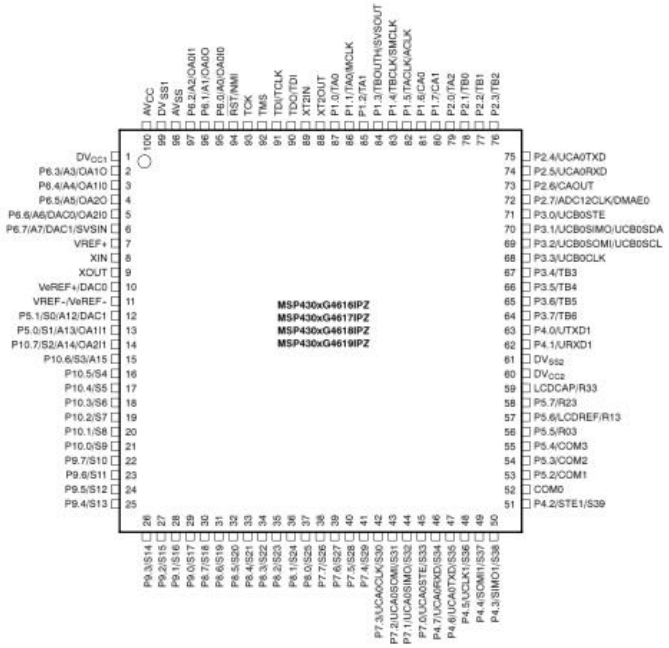


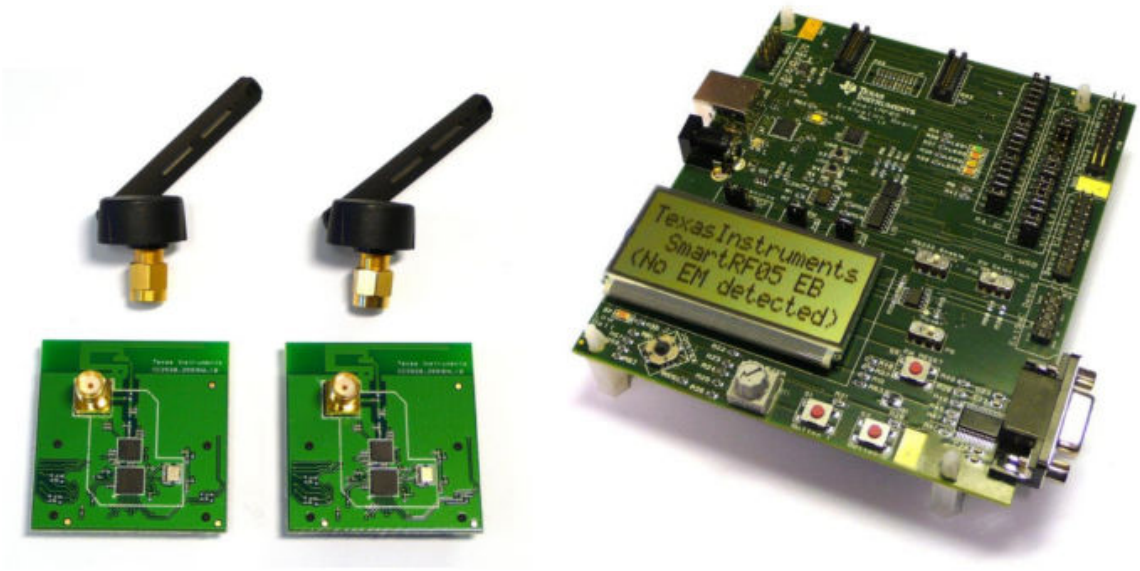
Figura 3.17: Distribución de pines del chip MSP430FG4618. Fuente [26].

3.4. CC2520EMK (*Evaluation Module Kit*) y SmartRF05EB (*Evaluation Board*).

Para llevar control sobre la información que se trasmite por la red, se ha hecho uso de un *sniffer*. Este *sniffer* está compuesto por componentes *hardware,* para lo que se ha utilizado el CC2520EMK junto con la placa de evaluación SmartRF05 y un componente *software* se que explicará en el apartado 3.5.2.

CC2520EMK está formado por un transceptor CC2520, evolución del CC2420 utilizado en este proyecto, y toda la circuitería necesaria para su correcto funcionamiento. La placa de evaluación SmartRF05 es la encargada que proporcionar la conectividad con el PC (a través de puerto USB) y facilitar determinadas configuraciones a través de distintos *jumpers* y una pantalla LCD.

Para obtener más información sobre el dispositivo CC2520EMK es recomendable utilizar la hoja de características del transceptor CC2520 [27] y la guía de inicio rápido del kit CC2520EMK [28]. Para información sobre la placa de evaluación, existe una guía de usuario.



(a) (b)

Figura 3.18: Componentes del sniffer. (a) CC2520EMK. (b) SmartRF05EB. Fuente [37].

3.5. Herramientas *software*

3.5.1. IAR Embedded Workbench

IAR es un entorno desarrollado específicamente para trabajar sobre microprocesadores MSP430 de Texas Instruments. Esta herramienta permite programar dichos microprocesadores en un lenguaje de alto nivel (C, C++), mucho más cómodo para el desarrollador, traduciéndolo a un lenguaje ensamblador entendible por los microprocesadores. Además ofrece una serie de funcionalidades como la simulación de código, inserción de puntos de ruptura o seguimiento de variables que lo convierten en una herramienta muy útil para obtener un funcionamiento óptimo del sistema a crear.

Para la realización de este proyecto se ha utilizado la versión 4.10 de la herramienta, cuyo interfaz con el usuario tiene el siguiente aspecto:

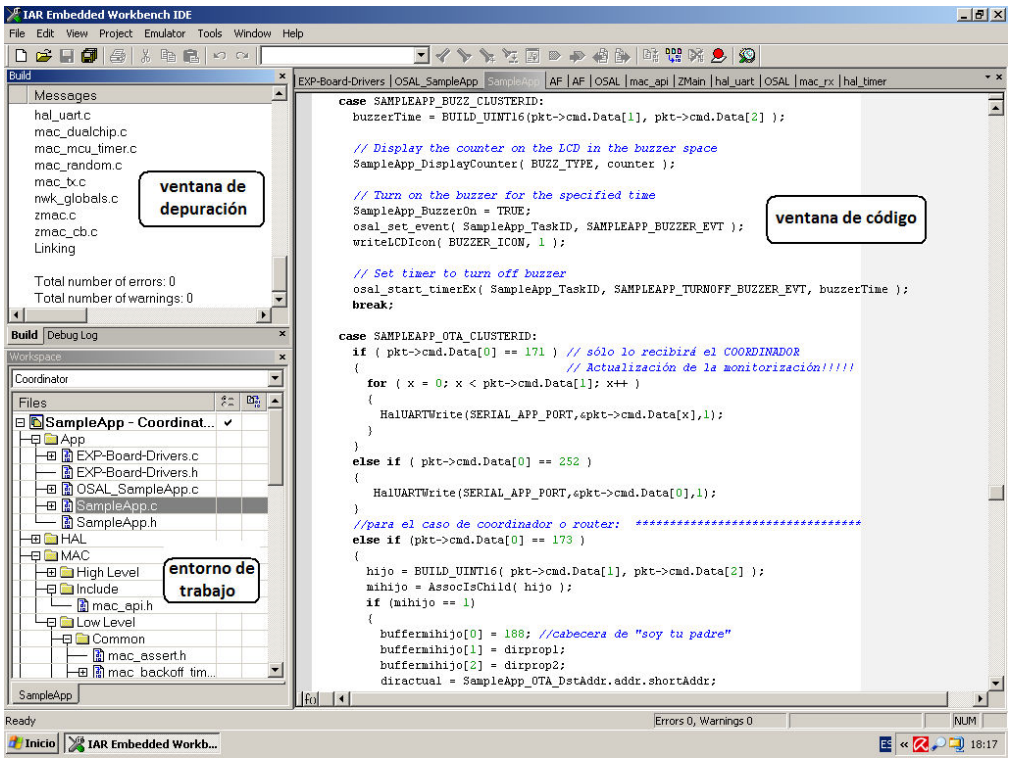


Figura 3.19: Vista principal de IAR Embedded Workbench.

Como se puede observar en la figura 3.19, el interfaz con el usuario consta principalmente de 3 ventanas:

- Ventana de entorno de trabajo, donde se muestran las carpetas y los ficheros de código que componen el total del sistema.

- Ventana de código, en la que se visualiza e introduce por parte del desarrollador el código necesario para conseguir el funcionamiento deseado.

- Ventana de depuración, donde se detallan los posibles errores o avisos detectados al compilar el código y fallos en el ensamblaje del conjunto de códigos que completan el sistema.

Una vez que el código ha sido correctamente cargado en el microprocesador, aparece un segundo interfaz en tiempo de ejecución, cuyas ventanas y funciones más características se exponen y comentan a continuación:

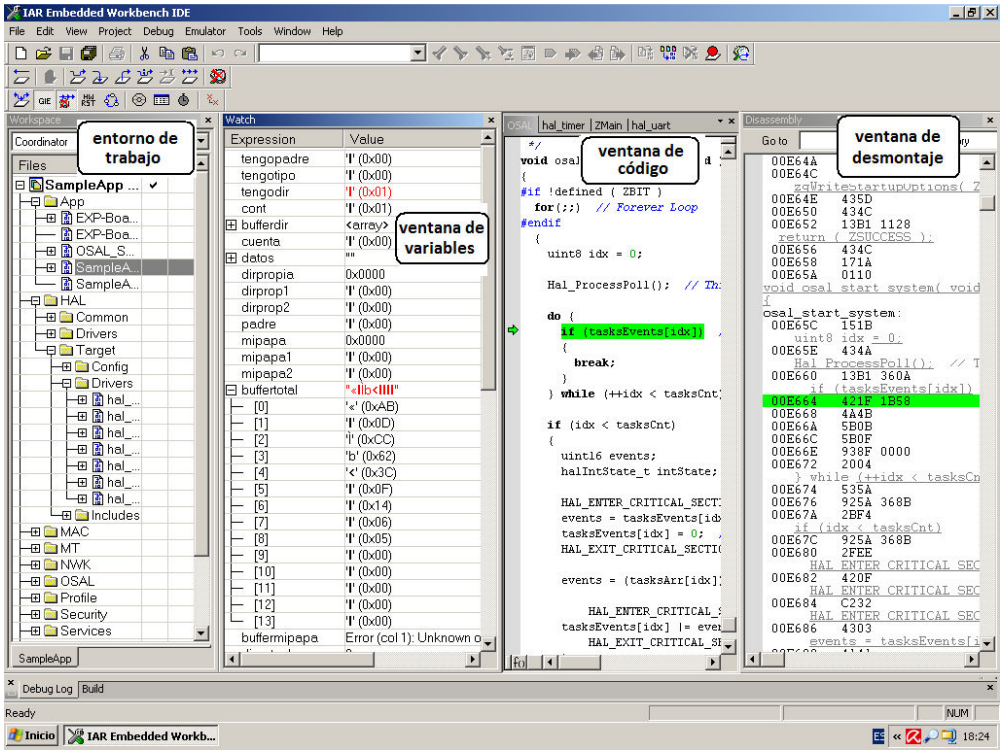


Figura 3.20: Vista de depurador de IAR Embedded Workbench.

Desde esta vista se puede ejecutar normalmente el código, o bien, en el caso de ser necesario, ejecutar el programa instrucción a instrucción para monitorizar el valor de las variables deseadas. Además, permite introducir hasta ocho puntos de ruptura con los que hacer seguimiento de los pasos que realiza el código en ejecución y detectar posibles anomalías.

Las principales ventanas visibles en este interfaz son:

- Ventana de entorno de trabajo.

- Ventana de código, en el que se visualiza instrucción a instrucción la evolución del código.

- Ventana de desmontaje, donde es posible ver la traducción de código de alto nivel a código ensamblador.

- Ventana de variables. Aquí se pueden visualizar aquellas variables globales cuyo valor en un instante determinado sea de interés.

Para una descripción más detallada de esta herramienta se recomienda utilizar el tutorial oficial de IAR [31].

3.5.2. *Packet Sniffer*

Esta aplicación permite (haciendo uso del kit de evaluación CC2520EMK junto con la placa de experimentación SmartRF05, ya expuestos en el apartado 3.4) capturar todos los paquetes que se envían desde los distintos dispositivos ZigBee/802.15.4 y monitorizarlos en un PC. De esta forma se puede obtener mayor control sobre todas las acciones que se realizan en la red.

Packet Sniffer es capaz de estructurar una trama capturada diferenciando qué información proviene de la capa MAC, capa de red y capa de aplicación además de mostrar datos sobre la temporización entre dos tramas consecutivas, la intensidad con la que se recibe o si la trama tiene errores.

En la figura siguiente se muestra el aspecto de la aplicación en un determinado instante de su ejecución:

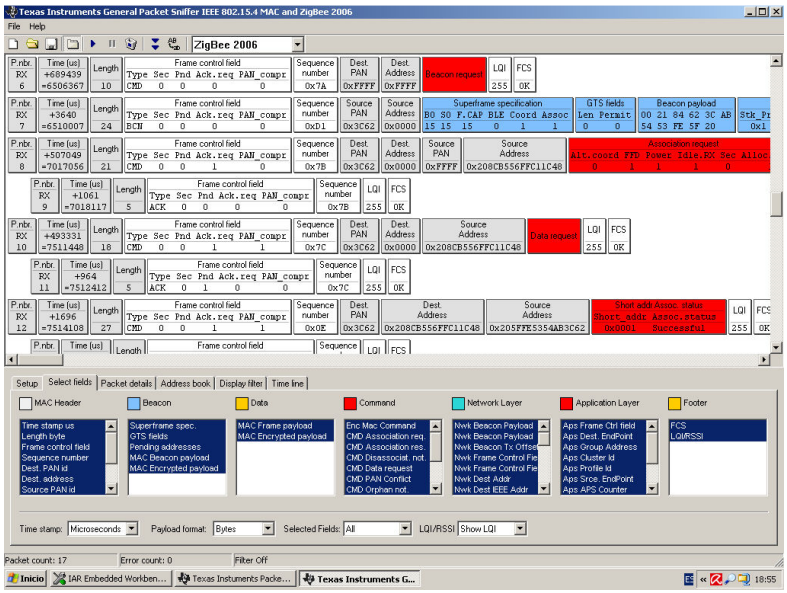


Figura 3.21:Vista de *Packet Sniffer*.

Este software así como su manual de instrucciones [30] se pueden descargar gratuitamente en la web de Texas Instruments [37].

3.5.3. Hyperterminal

La comunicación entre el dispositivo coordinador y el PC se ha realizado mediante el puerto serie RS-232. Por ello ha sido de gran utilidad la utilización de la aplicación Hyperterminal. Esta aplicación monitoriza todos los datos que el PC recibe por su puerto serie mostrándolos por pantalla en lenguaje ASCII.

Hyperterminal tiene diversas opciones de configuración, desde la selección del puerto a monitorizar, hasta la configuración de la tasa de bits, bits de datos, si existe o no bits de paridad, cuantos bits de parada se utilizan o si se utiliza algún mecanismo de control de flujo.

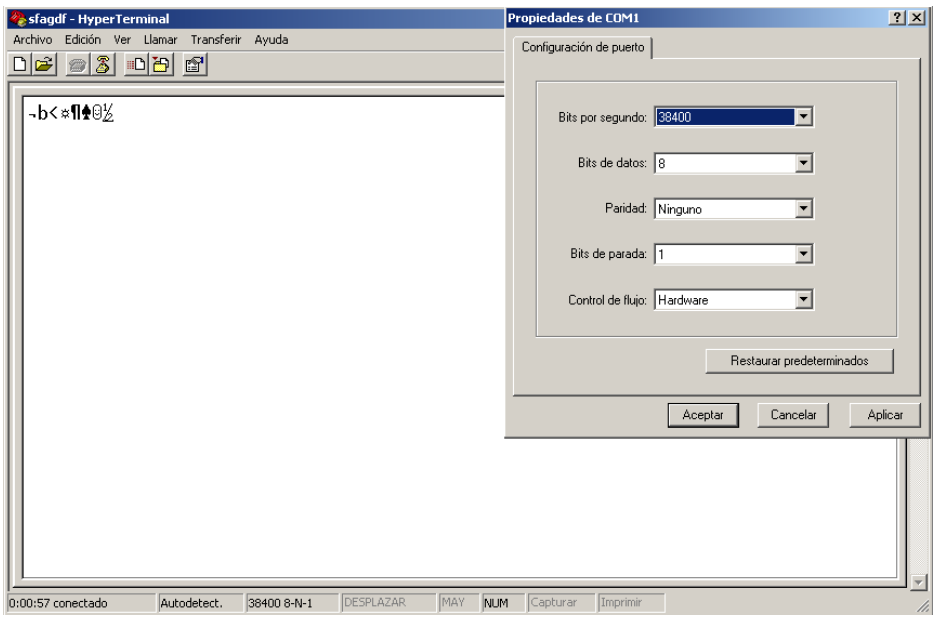


Figura 3.22: Vista de Hyperterminal.

La figura 3.22 muestra una captura de un instante en el funcionamiento de Hyperterminal así como su menú de opciones de configuración.

3.5.4. Xvi32

Como se ha comentado en el apartado anterior, Hyperterminal visualiza los datos entrantes a un PC por el puerto serie en lenguaje ASCII, lenguaje absolutamente ininteligible por un usuario.

La aplicación Xvi32 traduce dicha información de código ASCII a hexadecimal, con lo que se facilita sobremanera la comprensión y visualización de cada una de las tramas transmitidas.

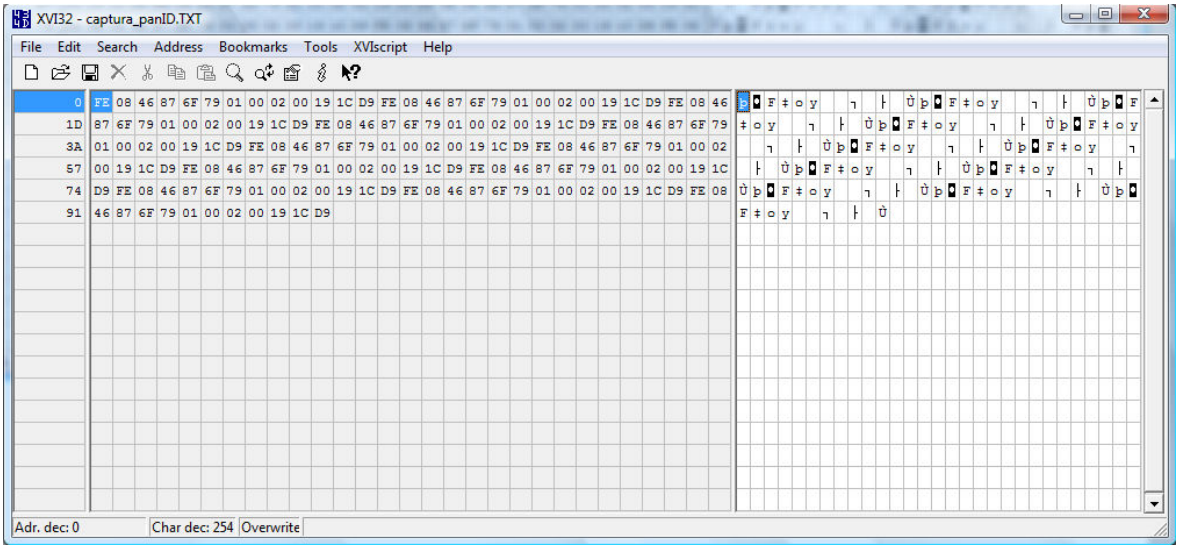


Figura 3.23:Visualización de Xiv32.

Xvi32 es una aplicación *freeware* y por tanto descargable de forma gratuita de páginas web como [39].

3.5.5. Microsoft Visual Basic 2008 Express Edition

La creación de la plataforma de configuración y monitorización de redes ZigBee/802.15.4, objetivo de este proyecto, se ha desarrollado utilizando la herramienta *Microsoft Visual Basic 2008 Express Edition* y, por tanto, haciendo uso del lenguaje de programación Visual Basic.

Esta herramienta proporciona un entorno muy fácil e intuitivo con el que crear, compilar y depurar el código necesario para el correcto funcionamiento de una aplicación. La interfaz está compuesta principalmente por cinco ventanas: una principal con la visualización del entorno gráfico que se está creando, la ventana Explorador de Soluciones que muestra la estructura de la aplicación (número de formularios y archivos adjuntos), la ventana de Propiedades con información de configuración de cada uno de los elementos seleccionados, una cuarta ventana con la lista de errores y advertencias detectados a tiempo real y, por último, el Cuadro de Herramientas, que no es más que un menú con todos los elementos de que se dispone para crear la interfaz gráfica. En la figura 3.24 se muestra la pantalla principal del entorno de programación *Microsoft Visual Basic 2008 Express Edition.*

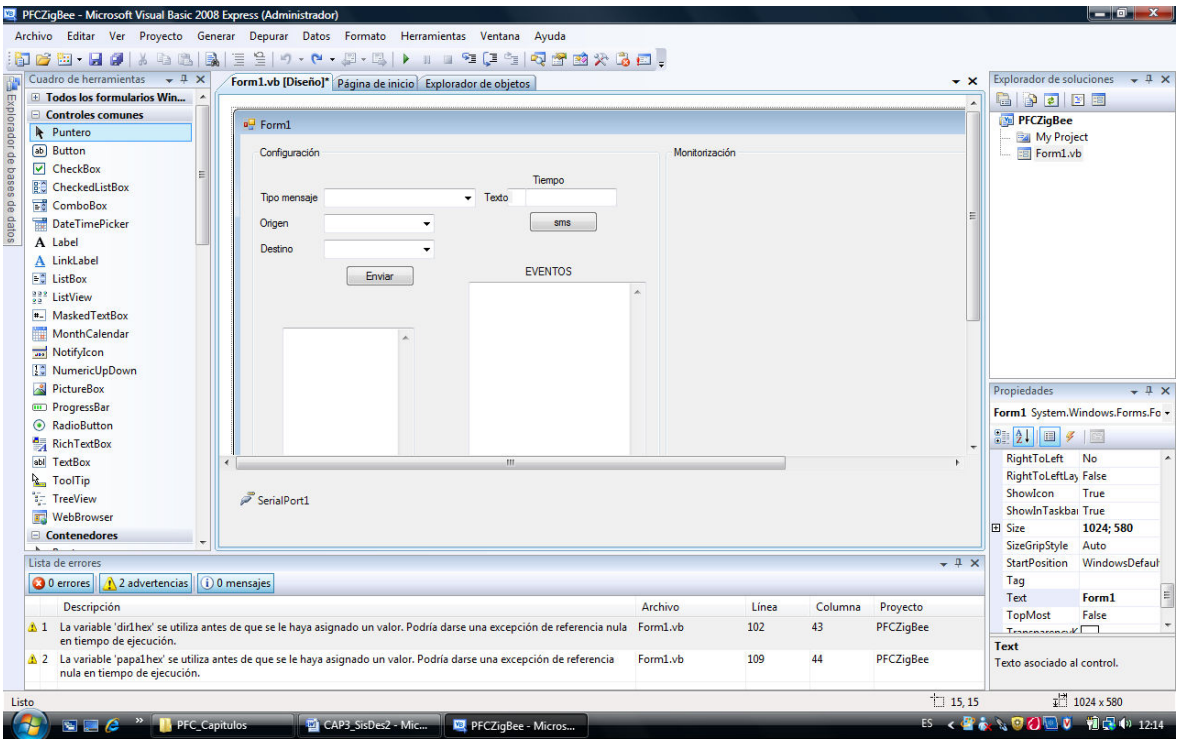


Figura 3.24: Vista entorno de programación de *Microsoft Visual Basic 2008 Express Edition*.

Pulsando en cualquiera de los elementos utilizados, se abre el entorno propio de programación donde se introduce el código que regirá las acciones a realizar por cada elemento en particular y la interacción entre ellos.

Un tutorial muy útil para iniciarse en este lenguaje de programación es [41] aunque existen numerosas publicaciones al respecto. Además se proporciona una ayuda en línea a través de Internet [42] donde se resuelven la gran mayoría de dudas que puedan surgir.

CAPÍTULO 4

Programa de gestión

4.1. Especificaciones

Para el desarrollo de este proyecto se requirieron principalmente dos elementos. Por un lado se debía ser capaz de crear una red ZigBee/802.15.4. Para ello se partía del kit de desarrollo CC2420EMK de la compañía *Texas Instruments* facilitado por la Universidad en el cual habría que cargar el código necesario para su correcto funcionamiento como dispositivos basados en el estándar ZigBee/802.15.4. Por otro lado, se pidió crear una plataforma con la que poder configurar los parámetros básicos de esta red ZigBee/802.15.4 a la vez que se monitorizara dicha red con el fin de observar la topología existente en cada momento y cómo se produce el intercambio de información entre los distintos nodos que la componen, ya sea cuando un nuevo nodo quiere incorporarse a la red o debido a algún evento interno o externo preconfigurado. Este entorno debía ser diseñado en un PC y comunicarse con los dispositivos ZigBee/802.15.4 a través de puerto serie (impuesto por la propia configuración de la placa experimental del kit de desarrollo).

Para la creación de esta plataforma no se impuso, de partida, el empleo de ningún lenguaje de programación en especial, se exigía sólo que permitiera desarrollar un entorno muy gráfico con el que un usuario pudiera interactuar de forma fácil e intuitiva.

Debido a que la finalidad del sistema creado tiene carácter didáctico, no se observó necesario crear aplicaciones con una funcionalidad compleja, sino que bastaba con interactuar con aquellos periféricos que proporciona la placa experimental (LED, LCD, *buzzer*) con el objetivo de crear distintos mensajes entre nodos y poder visualizar cómo son enrutados dichos mensajes en función del tipo de dispositivo transmisor y receptor y cómo estos nodos son capaces de interpretar dichos mensajes y actuar en consecuencia. Por la misma razón, la monitorización a diseñar tendría una capacidad máxima diez nodos (aunque una red ZigBee/802.15.4 tenga capacidad para 2^{16} nodos conectados) ya que se entiende que es esta una cantidad suficiente para demostrar la funcionalidad de estas redes y sus características, pudiendo lograr la configuración de cualquier tipo de topología.

4.2. Decisiones de diseño

Hubo que tomar varias decisiones importantes a la hora de afrontar el diseño. La primera de ellas fue qué aplicación utilizar para la programación de los microcontroladores. Existen dos alternativas en el mercado compatibles con los dispositivos de *Texas Instruments*: IAR *Embedded Workbench* (IAR EW) y *Code Composer Essentials* (CCE). Ambos entornos se ofrecen gratuitamente con capacidad de código reducido en la web de TI [37]. Dicha reducción de capacidad hizo imposible utilizar estas versiones para este proyecto, por lo que el hecho de que el Departamento de Tecnología Electrónica tuviera licencia de IAR EW junto con que la mayoría de los códigos ejemplos disponibles en la web estaban preparados para esta aplicación hizo que nos decantáramos por la utilización de este entorno de programación. El lenguaje aceptado por IAR es C por lo que no tuvimos margen de decisión al respecto, pero sí hubo que elegir qué entorno y lenguaje se iban a utilizar en la creación de la plataforma de configuración y monitorización.

Tras un proceso de investigación, se observó muy atractiva la utilización de la plataforma de programación .NET a través de la aplicación *Visual Studio 2008* de *Microsoft.* Esta plataforma está especialmente indicada para trabajar mediante eventos, la forma habitual de operar en entornos Windows, por lo que la convierte en un entorno muy cómodo para que un usuario pueda interactuar con él. La herramienta .NET y sus librerías ofrecen funciones específicas para la utilización de puertos serie, el uso de gráficos sencillos y la programación de botones, temporizadores y ventanas de visualización de texto que hacen muy sencilla la creación de entornos como el que se requiere en este proyecto y por tanto facilita enormemente la tarea del programador en este caso.

Microsoft proporciona las denominadas *Express Editions*. Éstas son ediciones básicas de licencia libre pensadas para uso no profesional, separadas por lenguajes de programación y sin algunas de las características avanzadas que sí poseen las versiones comerciales. Sin embargo es más que suficiente para las tareas a programar en la plataforma de configuración y monitorización que se ha llevado a cabo.

Estas *Express Editions* permiten la programación en Visual C++, Visual C#, ASP .NET y Visual Basic .NET aunque se han desarrollado extensiones para abarcar muchos más lenguajes. En este caso el lenguaje escogido ha sido Visual Basic .NET ya que, aunque la programación en este tipo de entornos basados en eventos no requiere de excesiva complejidad, la familiaridad previa del programador con este lenguaje lo hacía el más indicado en este caso.

4.3. Desarrollo

El desarrollo de este proyecto consta de dos etapas. Una primera en la que se programa la forma de actuar de los dispositivos según el estándar ZigBee/802.15.4 introduciendo el código necesario en los microcontroladores MSP430 y teniendo en cuenta la existencia de tres tipos de dispositivos con funcionalidades muy diferentes (coordinador, *router* y dispositivo final). Y una segunda etapa en la que se programan las plataformas creadas para monitorizar y configurar estos dispositivos de forma que se puedan visualizar las distintas topologías de red posibles, mostrando cada uno de los dispositivos que la forman junto con el tipo de dispositivo e información más relevante, y el intercambio de mensajes que se produce en dicha red para su configuración y correcto funcionamiento.

La figura 4.1 muestra un diagrama de la interacción existente entre el usuario, los dispositivos y la plataforma a desarrollar:

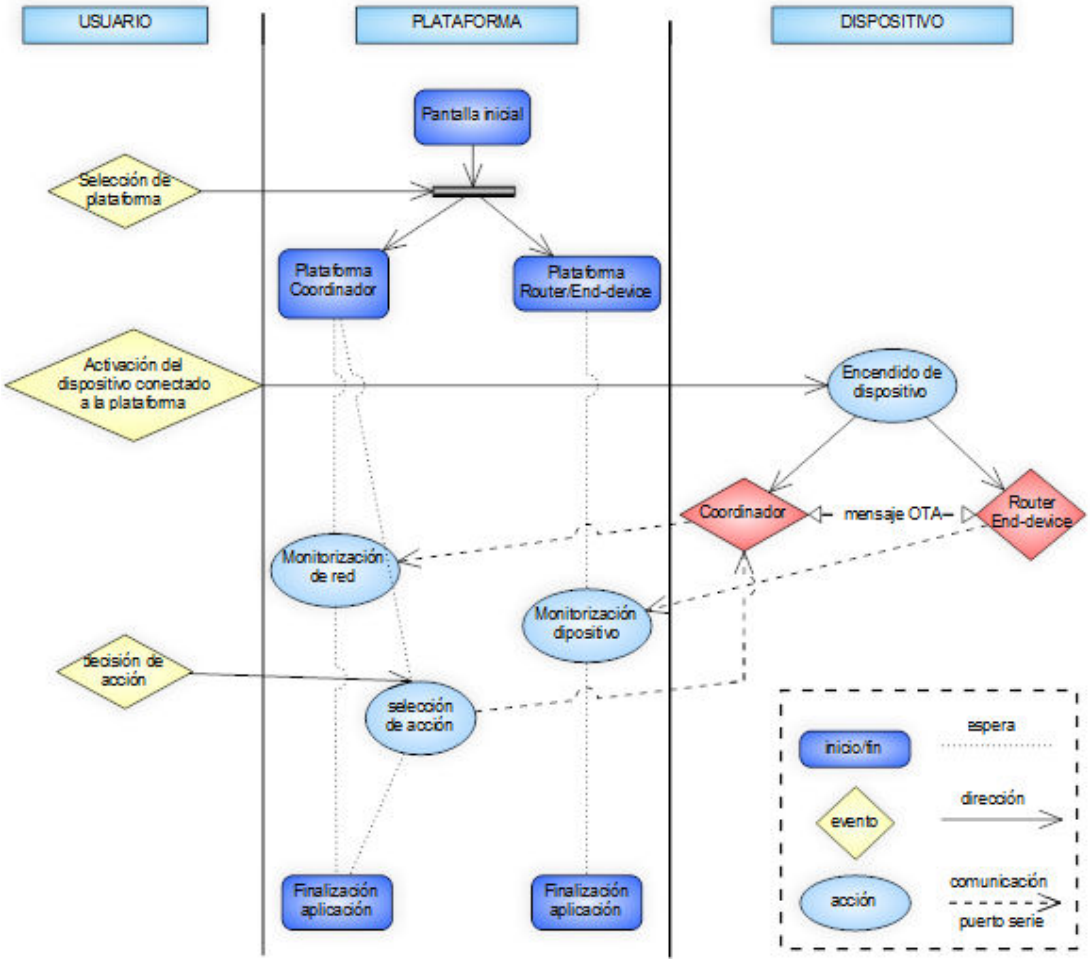


Figura 4.1: Diagrama de flujo de la interacción usuario-plataforma-dispositivo.

4.3.1. Programación del microcontrolador MSP430

Antes de explicar cómo y qué se ha programado, se detallará cómo está implementado el estándar ZigBee/802.15.4 en el microcontrolador.

La aplicación que se ha utilizado como base para este proyecto consta de las siguientes capas, basabas en la utilización de la pila *Z-Stack* desarrollada por TI [43], las cuales representan cada una de las capas del estándar ZigBee/802.15.4:

- Capa de **Aplicación**: Desde esta capa se realiza todo el proceso de identificación de mensajes recibidos a nivel de aplicación y la preparación para la transmisión de mensajes cuando se requiera, tomándose las decisiones de actuación adecuadas según el mensaje. Aquí se configuran los *clusterId,* los temporizadores y la prioridad con la que deben tratar las distintas tareas existentes en la ejecución del código. También se identifican por un lado y crean por otro las tramas que se comunican con la plataforma de configuración y monitorización desarrollada y activa en un PC a través del puerto serie. En esta capa están además desarrolladas las funciones para inicializar el *buzzer* y la visualización en el LCD o las principales características de configuración del puerto serie.

- Capa **HAL**: Corresponde a la capa física y desde ella se controlan los aspectos básicos de los diferentes periféricos como botones, LED, LCD, puerto serie, conversor analógico-digital, etc, estando aquí desarrolladas las funciones con las que se controla su funcionamiento.

- Capa **MAC**: En esta capa se encuentra todo el código que hace referencia a la configuración de la capa MAC. Aquí están implementadas las funciones utilizadas para la transmisión, recepción y los parámetros de radio a utilizar. Se incluye la API de esta capa con la definición de las funciones que marcan el estándar ZigBee/802.15.4.

- **MT** (Monitor Test): Las herramientas incluidas aquí han sido diseñadas por TI y se utilizan para el testeo de cada una de las capas a través de programas que proporciona esta misma empresa. Con ello se trata de facilitar el desarrollo del sistema a diseñar.

- Capa **NWK** (de red): En esta carpeta se implementan las funciones propias de la capa de red y la subcapa APS. Desde aquí se tiene acceso a las listas de asociación de nodos (equivalente a las listas de vecindades), a las tablas de vinculación, la gestión de direcciones, la creación de subgrupos para transmisiones de difusión y variables y *buffers* necesarios para el correcto funcionamiento de ambas capas.

- **OSAL:** Capa utilizada por el sistema para la gestión de tareas, gestión de consumo y memoria (equivalente a un pequeño sistema operativo).

- Capa **AF**: Implementa las funciones propias de la capa AF de ZigBee/802.15.4 y las variables necesarias para su funcionamiento.

- **Security**: Aquí se implementan las opciones de seguridad que proporciona (de forma opcional) ZigBee/802.15.4.

- **Services**: En la carpeta *services* están desarrolladas las funciones básicas para el tratamiento y manejo de direcciones, así como las funciones de copia y comparación tanto de direcciones cortas como extendidas.

- **ZDO**: Implementa las funciones que marcan el funcionamiento de la ZDO de ZigBee/802.15.4 tanto a nivel de aplicación, como de configuración o gestión de la seguridad.

- Capa **ZMAC**: Versión desarrollada por *Texas Instruments* que implementa la capa MAC utilizada por la pila Z-Stack. Es la capa donde se realiza realmente la creación e identificación de todos aquellos mensajes recibidos y transmitidos por el dispositivo a nivel de capa MAC. Esta capa se encargará de mandar a los niveles superiores aquellos mensajes recibidos que así lo requieran y de enviar al nivel físico el formato de trama correcto para su transmisión. Es aquí, por tanto, donde se realiza todo el procesado de aquellos mensajes que no van más allá de la capa MAC como los mensajes de balizas o de asociación.

- **ZMain**: Contiene las funciones necesarias para la correcta inicialización de los dispositivos. Además incluye funciones para la interacción con la placa experimental como la gestión de los botones.

Nuestro trabajo se ha centrado en la programación de la capa de aplicación para la gestión de mensajes entre nodos y, desde ahí, el control e interacción con los periféricos disponibles, pero también se ha tenido que programar algunos aspectos de la capa ZMAC para gestionar aquellos mensajes cuyo origen y destino no van más allá de esta capa como son las tramas de balizado, los parámetros de asociación o las tramas *ack* y la capa NWK para la visualización de otras tramas de comandos que ZigBee/802.15.4 utiliza para la configuración de los elementos de la red. Además, ha sido imprescindible realizar ciertas modificaciones en el código de la capa física, en aquellas funciones que definen la

recepción y transmisión de datos por el puerto serie, para acomodarlas al flujo de datos creado por dicho periférico.

A continuación se detalla más detenidamente el trabajo que van a desempeñar las capas modificadas, las funciones implementadas para lograr dichos trabajos y los cambios ejecutados en las funciones ya existentes para cada una de las capas nombradas en el párrafo anterior.

Capa de Aplicación

Esta capa es, de cara a la comunicación con nuestra plataforma, la capa más importante, pues es aquí donde se identifican y analizan todos aquellos mensajes, tanto vía OTA (*Over The Air*) como por puerto serie, que no sean tramas de configuración y sincronización propias de una red ZigBee/802.15.4, es decir todas aquellas tramas cuyo contenido sean datos. En este proyecto se utilizan 4 tramas de datos distintas a transmitir y recibir vía OTA: tramas de mensaje de encendido de LED, trama de mensaje de activación de *buzzer*, trama de configuración de mensajes periódicos y tramas de datos propiamente dicho. Estos últimos contienen instrucciones que es necesario transmitir entre unos dispositivos y otros para llevar a cabo distintas acciones. Son ejemplos de tramas de datos aquellas con información sobre un mensaje de texto a visualizar en el LCD destino o los mensajes de actualización de datos para la monitorización de la red en la plataforma.

Como se ha mencionado, en esta capa también se identifican y analizan, en el caso de que el dispositivo sea el coordinador (este nodo es el único que acepta configuración de parámetros a través de la plataforma) las tramas recibidas a través del puerto serie. Estas tramas pueden ser de 3 tipos: tramas que indican si un nodo perteneciente a la red podrá o no aceptar asociaciones, tramas para la configuración del número máximo de hijos, *routers* y profundidad aceptada por el nodo y tramas con instrucciones sobre el envío de los mensajes vía OTA, definidos en el párrafo anterior, a otros dispositivos.

Todos estos tipos de tramas se detallan en profundidad en apartados posteriores.

Es también trabajo de esta capa la ejecución de temporizadores que marcarán el funcionamiento de las distintas aplicaciones desarrolladas. Así se ha configurado el uso de 4 temporizadores: uno para controlar la periodicidad de los mensajes periódicos, otro para el apagado de LED, desactivación del *buzzer* y limpieza del LCD, otro para controlar el periodo de envío de mensajes de actualización y un último temporizador para observar periódicamente si se ha recibido algo por el puerto serie.

Y por último, en esta capa se realiza la definición de las funciones a desarrollar cuando se ejecute la pulsación de cualquiera de los 2 botones existentes en la placa de experimentación. Las acciones que se han decidido realicen son el envío de un mensaje de activación de *buzzer* en el caso de pulsar el botón 1 (etiquetado como S1) y el envío de un mensaje de encendido de LED en el caso del pulsar el botón 2 (etiquetado como S2), ambos mensajes de difusión.

Capa ZMAC

En esta capa se han modificado aquellas funciones que indican recepción y transmisión de mensajes de sincronización entre dispositivos, como son los mensajes de balizado, los mensajes *ack* y aquellos que hacen referencia a una asociación entre nodos, con el fin de crear tramas que poder mandar por el puerto serie y visualizar en la plataforma conectada al dispositivo en tiempo real, además de visualizarlas también en el LCD de cada uno de los dispositivos para tener una monitorización general del sistema aunque no estén todos los nodos conectados a plataformas. Esto permite observar cómo se realiza el proceso de intercambio de mensajes entre dos o más dispositivos.

Aunque se trabaje en modo no balizado, cuando un dispositivo nuevo intenta conectarse a la red realiza un escaneado por los distintos canales (aquellos previamente configurados) en los que ZigBee/802.15.4 trabaja. En cada uno de estos canales el nuevo dispositivo realiza una petición de baliza con el fin de encontrar el dispositivo coordinador de la red, éste y todos aquellos *routers* que escuchen dicha petición, envían una baliza para indicar los posibles nodos con los que puede iniciar el proceso de asociación.

Capa NWK (de red)

En esta capa obtenemos información sobre posibles errores en la asociación de un dispositivo, o notificaciones de orfandad. El primer caso ocurre cuando un dispositivo al que se está asociado deja de funcionar y por lo tanto se pierde el enlace con su nodo padre y en consecuencia el contacto con la red. Cuando esto ocurre, se procede a enviar notificaciones de orfandad a la espera de que algún nodo que sí permita asociación lo acoja como nodo hijo a través del envío de una baliza con la que sincronizarse.

Los cambios realizados en esta capa de la pila ZigBee/802.15.4 están relacionados con la creación de tramas para la monitorización de estos mensajes a través del puerto serie en la plataforma conectada y a través del LCD del dispositivo.

Capa Física

En la capa física se ha necesitado configurar la recepción de datos por el puerto serie, modificando para ello la función encargada de realizar dicha tarea. Su funcionamiento es el siguiente: Cuando se produce un evento de recepción de datos por puerto serie, es decir se detectan datos en el *buffer* de recepción, la rutina de ejecución salta a esta función, en la cual se recogen y almacenan los datos en memoria de forma que la capa de aplicación pueda acceder a ellos. Las modificaciones realizadas tienen por objetivo sincronizar la detección de datos de recepción de forma que se puedan capturar tramas completas con el fin de analizarlas y actuar según la información contenida en ellas.

En el apartado siguiente se explica con detalle el funcionamiento de esta y otras funciones desarrolladas en el entorno de programación IAR EW pasa su posterior carga en el microcontrolador.

4.3.1.1. Funciones desarrolladas

Capa de Aplicación

Las funciones desarrolladas en esta capa se detallan a continuación:

uint16 SampleApp_ProcessEvent(uint8 task_id, uint16 events)

Es la función principal de nuestro código. Cuando se detecta un evento, la ejecución salta a esta función, en la que se analiza qué evento es el causante de los siete programados.

Entradas:

Task_id: byte con el identificador de la tarea

Events: mapa de 16 bits que contiene todos los eventos a procesar.

Salidas:

Mapa de 16 bits en el que se señalan los eventos realizados mediante un AND lógico entre el mapa de bits que forma la variable *events* contenedora de todos los posibles eventos y el evento realizado.

Los eventos existentes son los siguientes:

- **Mensaje del sistema**: pueden ser tres las causas por las que se active un evento de mensaje del sistema: la inicialización del dispositivo, la interrupción producida por pulsación de uno de los botones de la placa experimental o la recepción de un mensaje

OTA. En el caso de inicialización del dispositivo, se recopilará toda la información necesaria para el relleno de la trama de monitorización que se enviará por puerto serie (en el caso de ser el dispositivo coordinador) o se enviará hacia el coordinador con el fin de mandarlo a la plataforma de monitorización, trama cuyo formato y contenido se detallan en el apartado 4.3.3.1, además de inicializar los distintos temporizadores configurados para proporcionar eventos de carácter periódico. Si la causa para que se produzca este evento es la pulsación de un botón o la recepción de mensajes OTA, se ejecutará la función desarrollada para tratar dichos casos, las cuales se expondrán más adelante.

- **Mensaje periódico**: Se ha programado que cada cierto tiempo, configurable por el usuario a través de la plataforma e inicialmente desactivado, se envíe un mensaje de carácter periódico. Esta periodicidad está controlada por un temporizador y la variable SAMPLEAPP_SEND_PERIODIC_MSG_TIMEOUT, la cual proporciona el tiempo en segundos entre eventos de este tipo. La activación de este mensaje produce la ejecución de la función *SampleApp_SendPeriodicMessage*.

- **Visualización de Iconos en el LCD:** Este evento se utiliza para mostrar en el LCD los iconos que indican transmisión y recepción de mensajes. La ejecución de este evento produce el salto a la función *SampleApp_FlashIconEvt*.

- **Oscilación de *buzzer* (zumbador):** Este evento estará activo siempre que se haya recibido un mensaje de *buzzer* y no se haya cumplido el tiempo configurado de duración del sonido.

- **Apagado de periféricos:** Se ha utilizado un evento para la restauración de los periféricos, es decir, apagado de LED tras el tiempo configurado de duración de encendido, apagado de *buzzer* tras el tiempo configurado de duración del sonido y borrado de la parte textual del LCD tras el tiempo configurado de duración de muestra de un texto en el mismo.

- **Actualización periódica:** Con la finalidad de actualizar la monitorización de la plataforma, cada 10 segundos se crea una trama de actualización con los datos sobre el dispositivo que se envía por puerto serie en caso de ser coordinador o hacia el coordinador si se trata de un *router* o un dispositivo final para que éste lo mande vía puerto serie hacia la plataforma.

- **Poll de Hardware:** Se ha denominado así a un evento de carácter periódico (cada 3 segundos) en el que se busca si han llegado datos por el puerto serie. En el caso de que efectivamente haya datos en el *buffer* de entrada, éstos se procesan y se actúa según proceda. En el apartado 4.3.3.1 se muestran con detalle los posibles mensajes que pueden llegar por puerto serie, y formato y contenido.

*void SampleApp_MessageMSGCB(afIncomingMSGPacket_t *pkt)*

Función que procesa cualquier mensaje OTA recibido. Es aquí donde se clasifica dicho mensaje y se toman las decisiones de actuación. Para algunos mensajes, no importará el contenido del campo de datos pues si la funcionalidad queda completamente definida en el campo *ClusterId*, no ocurrirá lo mismo para otro tipo de mensajes donde todo su contenido útil se encontrará en los valores de los bytes del campo de datos y el campo *ClusterId* sólo servirá para diferenciarse de los demás mensajes.

Entrada:

Pkt: variable que contiene todos los campos a nivel de aplicación de la trama recibida

Salida:

Ninguna.

La identificación del tipo de mensaje de datos se realiza a través del campo *ClusterId* según la siguiente tabla:

Tipo de mensajes	**ClusterId** (Hexadecimal)
Mensaje periódico	0x01
Mensaje *buzzer*	0x02
Mensaje LED	0x03
Mensaje datos	0x04

Tabla 4.1: valores del campo *ClusterId*.

Si el tipo de mensaje detectado es un mensaje de datos, habrá que realizar otra identificación del tipo de datos que contiene. Esta identificación se realiza analizando el primer byte del campo de datos, que corresponde a la cabecera de la trama integrada en el campo de datos. Los valores de dicha cabecera se muestran en la tabla 4.2.

Tipo mensaje datos	**Valor Cabecera** (Hexadecimal)
Monitorización	0xAB
Aviso a la plataforma de inicialización de dispositivo	0xFC
Mensaje configuración	0xFE

Tabla 4.2: Cabeceras para tipo de mensaje en tramas de datos.

Una vez detectado el tipo de mensaje recibido, se realizan los procesos necesarios para cada tipo de mensaje, como el encendido de LED o *buzzer* si es el caso, la visualización en el LCD del tipo de mensaje, el envío por puerto serie de la información recibida para su muestra a través de la plataforma o la configuración de los parámetros solicitada.

void palabratoLCD(uint8 pal, uint8 posletra)

Cuando se quiere mandar un mensaje con un texto a visualizar en el LCD de algún dispositivo (mensaje cuyo origen será siempre la plataforma), cada letra que forma el texto se codifica asignándole un número correspondiente con su orden alfabético. La función *palabratoLCD* realiza la operación inversa, ya que a partir de los bytes recibidos cuyo contenido es numérico, identifica la letra asociada y la visualiza por pantalla en la posición indicada.

Entradas:

Pal: Byte que contiene el número a identificar por su letra asociada.

Posletra: Posición que ocupa dicha letra en la palabra para mostrarla por pantalla.

Salida:

Ninguna.

void SampleApp_SendPeriodicMessage(void)

Función que envía un mensaje periódico.

Entradas:

Ninguna.

Salida:

Status: indica el estado de la acción solicitada.

void SampleApp_SendLedMessage(uint16 ledTime)

Función que realiza el envío de un mensaje de tipo LED.

Entradas:

ledTime: Duración de encendido del LED 2 de la placa experimental.

Salida:

Status: indica el estado de la acción solicitada.

void SampleApp_SendBuzzerMessage(uint16 buzzTime)

Función que envía un mensaje de activación de *buzzer*.

Entradas:

buzTime: Duración de sonido del *buzzer*.

Salida:

Status: indica el estado de la acción solicitada.

*void SampleApp_SendData(uint8 *buf, uint8 len)*

Función que realice el envío OTA de un mensaje con datos.

Entradas:

Buf: Buffer que contiene los datos a enviar en el mensaje.

Len: Longitud en bytes del mensaje.

Salida:

Status: indica el estado de la acción solicitada.

Estas cuatro últimas funciones de envío de mensajes realizan dicha acción a través de la siguiente llamada:

*afStatus_t AF_DataRequest(afAddrType_t *dstAddr, endPointDesc_t *srcEP, uint16 cID, uint16 len, uint8 *buf, uint8 *transID, uint8 options, uint8 radius)*

dstAddr: Dirección destino.

srcEP: Dirección origen.

cID: ClusterId que define el tipo de mensaje.

len: Número de bytes de datos contenidos en el siguiente parámetro.

buf: Datos a enviar.

transID: Número de secuencia del mensaje.

options: Máscara de bits que define las opciones de transmisión.

radius: Alcance máximo del mensaje.

void SampleApp_DisplayCounter(uint8 counterType, uint8 counter)

Función que se encarga de mostrar en el LCD el tipo de mensaje que ha llegado en el caso de ser un mensaje de LED, *buzzer* o periódico. En caso de mensaje de texto ya se ha visto anteriormente que se muestra en el LCD a través de la función *palabratoLCD.*

Entradas:

CounterType: Variable que contiene el tipo de mensaje a visualizar:

CounterType	**LCD**
BUZZ_TYPE	BUZZER
LED_TYPE	LED
PERIODIC_TYPE	PERIODIC

Tabla 4.3: valores de la variable CounterType y resultado en el LCD.

Len: Longitud en bytes del mensaje.

Salida:

Ninguna.

Capa física

Como se ha mencionado, la función que se ha requerido modificar en esta capa es aquella relacionada con la recepción de datos por el puerto serie. La función se muestra a continuación:

```
Void Hal_UART_RxProcessEvent ( uint8 port, uint8 rxReady, uint8 ch )
{
   unit8 datoserie;
   if ((rxReady) && !(Hal_UART_RxBufferIsFull (port)))
   {
      Hal_UART_RxInsertBuffer (port, ch);
      datoserie = ch;
      if (datoserie == 254)
      {
         cuenta = 0;
         datos[cuenta] = datoserie;
         cuenta = cuenta + 1;
      }
      else
      {
         if ((cuenta >= 1) & (cuenta <15))
         {
            datos[cuenta] = datoserie;
            cuenta = cuenta + 1;
         }
         else
            cuenta = 0;
      }
   }
}
```

port: indica el puerto serie donde se reciben los datos.

rxReady: indica si la recepción está preparada.

ch: byte recibido por el puerto serie.

Esta función busca la cabecera de la trama, valor fijo a 0xFE, para una vez encontrada, almacenar en orden los datos en la variable datos[15] para su posterior uso en la capa de aplicación.

4.3.2. Programación de la plataforma de configuración y monitorización

Existen dos plataformas según el nodo a conectar sea el coordinador, en cuyo caso habría que activar la aplicación diseñada para el coordinador, o un *router* o dispositivo final, en cuya situación se procedería a activar la aplicación diseñada para éstos. La plataforma que actúa sobre el coordinador consta de una ventana principal dividida en dos grupos, el lado derecho se utiliza para la monitorización del sistema (topología y situación actual de los principales parámetros de red) y el lado izquierdo para monitorización de mensajes de comandos e información sobre los eventos ocurridos en la red además de la configuración de mensajes a enviar entre dispositivos (para obtener una información más

detallada sobre la ejecución de esta aplicación, recurrir a *Anexo A: Manual de usuario* de este mismo documento, donde se explica con detenimiento el funcionamiento de ambas plataformas y las opciones que ofrece).

Configuración

Desde este entorno se permite la variación de parámetros que afectan directamente a la topología de red, tales como el número máximo de hijos que un nodo puede tener, cuántos de estos hijos pueden ser *routers* o la máxima profundidad que se permite a la red (sólo en caso de que el nodo sea el coordinador o un *router*). También se tiene opción de configurar si se permite o no que un nodo acepte más asociaciones, con lo que se puede ir modelando la topología de la red a medida que ésta va creciendo. Por otro lado, se ha diseñado un menú desde el cual poder mandar cualquiera de los cuatro tipos de mensajes que ya con anterioridad se han mencionado: LED, *buzzer*, periodicidad o texto. Este menú permite decidir quién será el dispositivo origen del mensaje y quién el dispositivo destino, por lo que se puede crear una comunicación entre cualesquiera dos o más puntos de la red y observar así cómo se realiza el enrutamiento según el tipo de dispositivo que realice la acción. De esta forma se podrá observar cómo si el dispositivo emisor es un *router* o el coordinador, estos tienen capacidad para poder conocer la dirección del dispositivo receptor y mandar el mensaje directamente, o si el dispositivo emisor es un dispositivo final, cómo envía siempre el mensaje a su nodo padre para que este último lo distribuya según sus tablas internas de enrutamiento y vecindades.

Monitorización

La plataforma permite observar el número de dispositivos que contiene la red, qué tipo de dispositivos son y su topología en cada instante de forma gráfica y, por tanto, muy cómoda y visual. La identificación de cada dispositivo es rápida ya que se basa en la utilización de colores para distinguir la naturaleza de cada uno y la dirección corta identificativa del nodo en la red aparece junto a la representación del dispositivo de forma fácilmente detectable. Además permite la visualización de los principales parámetros característicos de la red y de cada nodo en particular simplemente posando el cursor sobre un icono junto a la representación visual del nodo. Igualmente se permite acceder al menú de configuración pulsando otro de los iconos que aparecen en el campo de monitorización.

4.3.2.1. Funciones desarrolladas

A continuación se detallarán las funciones con mayor relevancia en el desarrollo de la plataforma y por tanto se omitirán todas aquellas de uso más general aunque su presencia en el código sea abundante. Es el caso del evento producido por la pulsación de un botón, el menú de una *ComboBox* o el uso de las *TextBox*.

Private Sub SerialPort1_DataReceived (ByVal sender As Object, ByVal e As System.IO.Ports.SerialDataReceivedEventArgs) Handles SerialPort1.DataReceived

Función que se ejecuta tras producirse un evento de recepción de datos por el puerto serie configurado, en nuestro caso el puerto COM1. Esta función se encarga de ir recogiendo byte a byte todos los datos almacenados en el *buffer* de recepción para su identificación. Para ello lo primero es detectar la cabecera de la trama para identificar cada uno de los campos posteriores a ésta con la información correspondiente. Se han utilizado tres cabeceras de trama distintas para tres posibles mensajes a recibir:

Cabecera (Hexadecimal)	**Tipo de mensaje**
0xAB	Actualización monit.
0xAF	Comando
0xBF	Datos

Tabla 4.4: Cabeceras de tramas recibidas por puerto serie.

Para la captura de los bytes recibidos se hace uso de la función
octeto = SerialPort1.ReadByte

Desde esta función, una vez recogida toda la información de la trama correspondiente, se realiza la monitorización de mensajes de comandos y mensajes de datos en las ventanas de EVENTOS DE MONITORIZACIÓN e INTERCAMBIO DE MENSAJES. (Véase Anexo A). Además también se preparan los datos para la monitorización visual de los dispositivos y la topología de red.

Private Sub GroupBox2_Paint (ByVal sender As Object, ByVal e As System.Windows.Forms.PaintEventArgs) Handles GroupBox2.Paint

Esta función se encarga de la representación visual del dibujo de cada nodo, de las direcciones cortas asociadas, de los enlaces que los unen y de los iconos que dan lugar a las distintas funciones de monitorización y configuración.

Para realizar dichos gráficos, se utilizan cuatro funciones propias de la herramienta *Visual Studio*:

- *e.Graphics.DrawEllipse(Color, Coordenada X, Coordenada Y, Tamaño X, Tamaño Y)*
  Función que dibuja el contorno de una elipse del color marcado en la variable *Color*, en la posición indicada por los campos *Coordenada X* y *Coordenada Y* y del tamaño indicado en los campo *Tamaño X* y *Tamaño Y*.

- *e.Graphics.FillEllipse(Color, Coordenada X, Coordenada Y, Tamaño X, Tamaño Y)*
  Función que rellena del color indicado en la variable *Color* el interior de una elipse en la posición indicada por las variables *Coordenada X* y *Coordenada Y* y del tamaño señalado con las variables *Tamaño X* y *Tamaño Y*.

- *e.Graphics.DrawString(String, Fuente, Color, Coordenada X, Coordenada Y)*
  Esta función escribe lo contenido en la variable *String*, con la fuente declarada en *Fuente*, del color que indique la variable *Color* y en la posición indicada por las variables *Coordenada X* y *Coordenada Y*.

- *e.Graphics.DrawLine(Color, Origen X, Origen Y, Destino X, Destino Y)*
  Función que dibuja una línea del color indicado en *Color*, cuyas coordenadas origen son *Origen X* y *Origen Y* y coordenadas destino *Destino X* y *Destino Y*.

Para el manejo de los datos y su visualización, se ha hecho uso de las dos siguientes funciones de conversión, de hexadecimal a byte y viceversa:

```
Private Function ConvertAsciiHexToByte(ByVal asciiHexToConvert As String) As Byte
    Dim convertedValue As Byte
    convertedValue = Convert.ToByte(asciiHexToConvert, 16)
    Return convertedValue
End Function
```

```
Private Function ConvertByteToAsciiHex(ByVal byteToConvert As Byte) As String
    Dim convertedValue As String
    convertedValue = Hex$(byteToConvert)
    Return convertedValue
End Function
```

Con la finalidad de obtener un código ordenado e inteligible se han definido las siguientes clases:

```
Public Class Nodo
    Public pos(9) As Integer
    Public actualiza(9) As Integer
    Public addr(9) As String
    Public addr1(9) As Byte 'LO de short address
    Public addr2(9) As Byte 'HI de short address
    Public color(9) As Pen
    Public relleno(9) As Brush
    Public padre(9) As String
    Public panid(9) As String
    Public disp(9) As String
    Public modobeacon(9) As Byte
    Public maxhijos(9) As Byte
    Public maxrouter(9) As Byte
    Public maxprof(9) As Byte
    Public canallogico(9) As Byte
End Class
```

La clase *Nodo* contiene toda la información a visualizar sobre cada uno de los dispositivos que forman parte de la red.

```
Public Class Trama
    Public longi As Byte
    Public addr As String
    Public color As Pen
    Public relleno As Brush
    Public dir1 As Byte
    Public dir2 As Byte
    Public papa1 As Byte
    Public papa2 As Byte
    Public padre As String
    Public panid1 As Byte
    Public panid2 As Byte
    Public panid As String
    Public modobeacon As Byte
    Public maxhijos As Byte
    Public maxrouter As Byte
    Public maxprof As Byte
    Public canallogico As Byte
End Class
```

La clase *Trama* contiene almacenados los datos recibidos en la última trama de actualización de monitorización. Estos datos serán procesados antes de que pasen a forman parte del registro de datos de la clase *Nodo*.

```
Public Class Comandos
    Public tipotrama As Byte
    Public addr1 As String
    Public addr2 As String
    Public dir1 As Byte
    Public dir11 As Byte
    Public dir2 As Byte
    Public dir22 As Byte
    Public ack As Byte
    Public bo As Byte
    Public so As Byte
    Public assoc As Byte
  End Class
```

La clase *Comandos* contiene los datos recibidos en la última trama de comandos y se utilizará para su visualización en la plataforma.

```
Public Class TramaDatos
    Public tipotrama As Byte
    Public clusterid As String
    Public dir1 As Byte
    Public dir2 As Byte
    Public addr As String
    Public ack As Byte
    Public lqi As Byte
  End Class
```

La clase *TramaDatos* almacena los datos recibidos en la última trama de datos y se utilizará para su visualización en la plataforma.

4.3.3. Comunicación entre dispositivo ZigBee/802.15.4 y plataforma

Como ya se ha mencionado anteriormente, Existen dos plataformas, una para el coordinador, mucho más completa y desde donde se pueden realizar acciones de configuración y monitorización completa de la red, y otra de apoyo para los *routers* y dispositivos finales que permite la monitorización de los mensajes de comandos y aplicación que transmite y recibe el dispositivo conectado, además de ofrecer información detallada sobre la configuración del nodo.

El puerto serie se ha configurado para operar a una tasa de 38400 bits por segundo, con 8 bits por dato, sin paridad y con un único bit de parada. Estas características cargadas tanto en el dispositivo como en la configuración del puerto serie del PC permitirá el correcto entendimiento entre ambos sistemas.

El esquema de conexionado se representa en la figura 4.2:

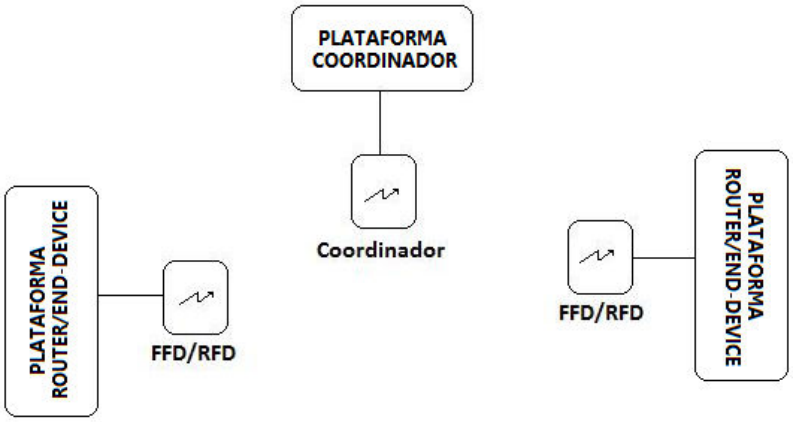


Figura 4.2: Diagrama de bloques sobre conexionado entre plataformas y dispositivos.

La comunicación establecida entre los dispositivos y sus respectivas plataformas está basada en la transmisión y recepción de tramas de tamaño variable (según la información contenida) y diferentes según el sentido de dicha comunicación sea dispositivo → plataforma o plataforma → dispositivo. En la dirección dispositivo → plataforma, los mensajes tendrán como funcionalidad la monitorización ya sea de tramas de comandos, datos o del estado de la red en el caso del coordinador. En la dirección opuesta, plataforma → dispositivo, los mensajes tendrán como objetivo la configuración de algún parámetro o bien ordenar el establecimiento de algún tipo de intercambio de información entre dos o más nodos.

En la figura 4.3 se detallan todos los posibles mensajes creados en la plataforma y cómo éstos son analizados y procesados por el coordinador para realizar las acciones solicitadas o para su transmisión, en caso de ser necesario, hacia los restantes dispositivos pertenecientes a la red. Se trata por tanto de comunicación en sentido plataforma → dispositivo.

La figura 4.4 muestra los distintos mensajes que los dispositivos utilizan para comunicarse con sus respectivas plataformas a través del puerto serie y cómo la plataforma los analiza y opera con éstos según el caso.

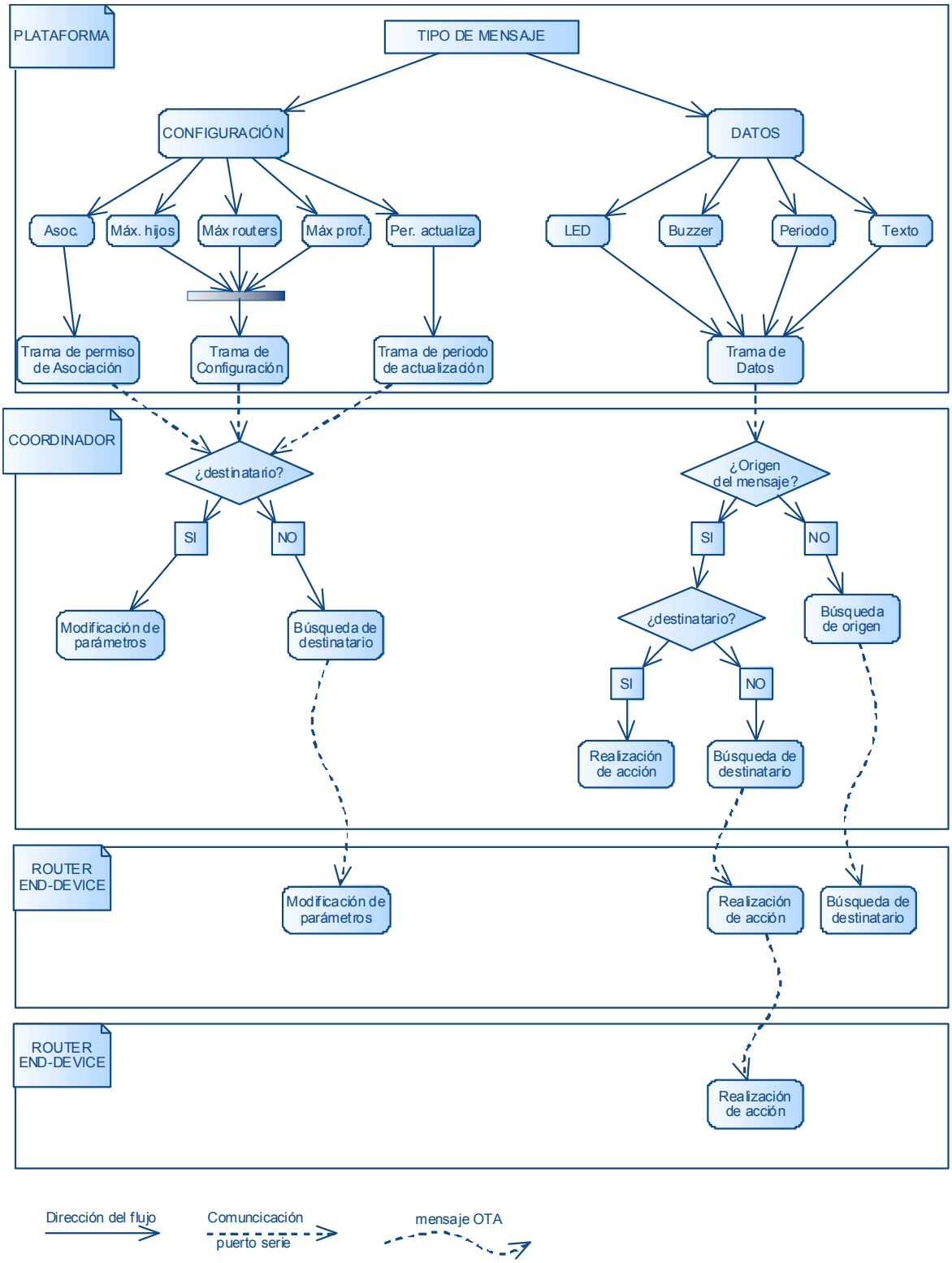


Figura 4.3: Mensajes creados en la plataforma y análisis por parte de los dispositivos.
(sentido de la comunicación: plataforma → dispositivo coordinador).

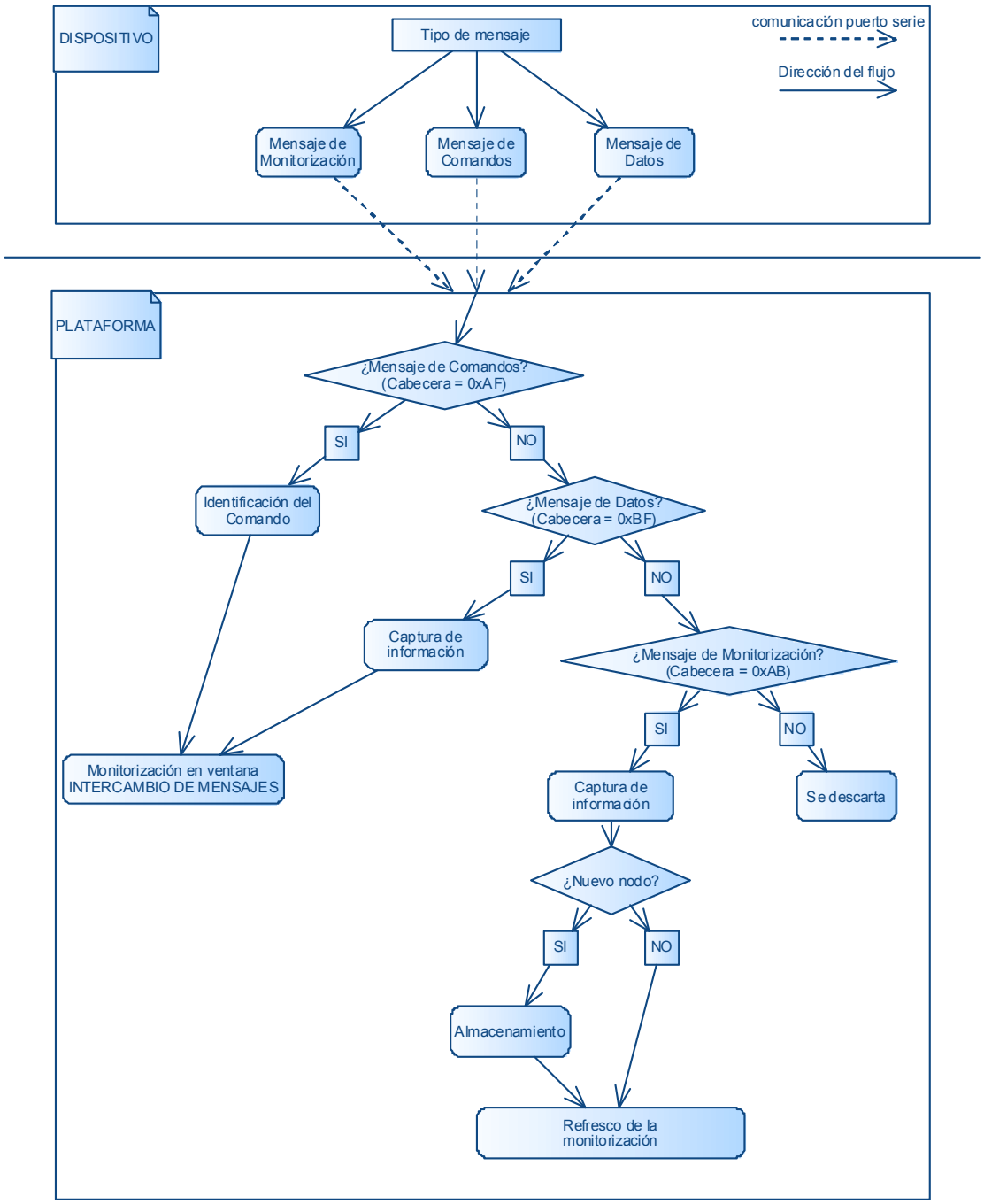


Figura 4.4: Mensaje creados por los dispositivos y análisis por parte de la plataforma.
(sentido de la comunicación: dispositivo → plataforma).

4.3.3.1. Formato de tramas

A continuación se definen los formatos de trama utilizados para cada uno de los mensajes que se usarán para la comunicación por puerto serie entre los dispositivos y las plataformas, distinguiendo, como ya se ha mencionado con anterioridad, entre ambos sentidos de la comunicación.

Mensajes en el sentido plataforma → dispositivo

En este sentido de la comunicación se pretende transportar valores que determinen la configuración de algunos parámetros característicos de la red además de órdenes para el envío de cierto tipo de mensajes entre unos dispositivos y otros.

Los parámetros de configuración que se ha decidido tener capacidad de modificar comprenden aquellos que afectan directamente a la configuración de la topología de red. Estos parámetros son 4:

- Permitir/No permitir asociaciones en los nodos con capacidad para ello, es decir, el coordinador y los *routers*.

El formato de trama utilizado para este tipo de mensajes se detalla en la figura 4.2:

CAMPO	Cabecera	Asoc/No asoc	DirDestino
TAMAÑO	1 Byte	1 Byte	2 Bytes

Figura 4.5: Formato de trama para mensajes de permiso de asociación.

  - **Cabecera:** valor fijo a 0xFE para indicar el inicio de la trama.

  - **Asoc/No asoc:** byte con la información sobre si se permite o no asociación en el nodo destino, los posibles valores se muestran en la siguiente tabla:

Asoc/No asoc	Valor (Hexadecimal)
Asociación permitida	0x05
Asociación no permitida	0x06

Tabla 4.5: valores del campo Asoc/No asoc.

  - **DirDest:** Dirección del nodo destino del mensaje.

- Configuración del número máximo de hijos que a un nodo se le permite tener.

- Configuración del número máximo de *routers* que un nodo puede tener asociado entre sus hijos.

- Configuración de la profundidad máxima para cada uno de los nodos que forman parte de la red.

El formato de trama para estos tipos de mensajes queda definido a continuación:

CAMPO	Cabecera	TipoTrama	DirDest	Hijos	Router	Prof
TAMAÑO	1 Byte	1	2	1	1	1

Figura 4.6: Formato de trama de configuración.

- **Cabecera:** Valor fijo a 0xFE para indicar el inicio de una trama.
- **TipoTrama:** 1 byte fijo a 0x07 con el que se indica al receptor que se trata de una trama de configuración.
- **DirDest:** 2 bytes para indicar la dirección hacia donde se quiere enviar la trama y por tanto el dispositivo que va a sufrir las modificaciones.
- **Hijos:** 1 byte cuyo valor indica el número máximo de hijos que se ha configurado para el nodo en cuestión (máximo configurable: 255 hijos, aunque la plataforma no permite monitorizar más de 3 hijos para el coordinador ni más de 2 hijos para los *routers*).
- **Router:** 1 byte que indica el número máximo de router que pueden estar asociados al nodo destino de la trama (máximo configurable: 255 hijos, aunque la plataforma no permite monitorizar más de 3 hijos para el coordinador ni más de 2 hijos para los *routers*).
- **Prof:** 1 byte con la profundidad máxima configurada para el nodo destino de la trama.

Se ha decidido tener opción de configurar la periodicidad de los mensajes de actualización de la monitorización con el fin de aumentar o disminuir ésta en función de la actividad que se desea para cada uno de los nodos, así se optimiza el consumo de energía de los dispositivos en cada caso particular. El formato de trama con el que se configura este parámetro se detalla en la figura 4.6:

CAMPO	Cabecera	TipoTrama	DirDest	Tiempo
TAMAÑO	1 Byte	1	2	1

Figura 4.7: Formato de trama para el control de la periodicidad de las actualizaciones de la monitorización.

- **Cabecera:** 1 byte con valor fijo a 0xFE para indicar el inicio de la trama.
- **TipoTrama:** 1 byte con valor fijo a 0x08 que indica que se trata de una trama de configuración de la periodicidad de los mensajes de actualización.

- **DirDest:** 2 bytes con la dirección del nodo destino del mensaje.

- **Tiempo:** campo de un byte que contiene el tiempo de periodicidad en segundos (tiempo máximo configurable: 255 segundos)

También se pueden configurar las órdenes para el envío de mensajes entre dos o más dispositivos pudiendo decidir el dispositivo emisor del mensaje y el/los dispositivos receptores del mismo. 4 son los posibles mensajes a enviar:

- Mensaje que ordene a un dispositivo o al conjunto de toda la red el encendido de un LED de la placa experimental.
- Mensaje de activación del zumbador (*buzzer*).
- Mensaje de texto de una palabra de no más de 6 caracteres a visualizar en el LCD del dispositivo destino.
- Mensaje de periodicidad configurable por el usuario.

La finalidad de estos mensajes es la creación de tráfico visualizable en la red y la de poder observar cómo trabajan los dispositivos ZigBee/802.15.4 en la identificación del tipo de mensaje, tratamiento de la información contenida en el mensaje (ClusterIDs) y el enrutamiento a través de la red. Para ello se recomienda el uso de la plataforma creada en este proyecto además de un *sniffer* con el que poder observar con más detalle el contenido de cada uno de los campos de las tramas enviadas OTA entre los distintos nodos.

Estas tramas con las órdenes para envío de mensajes tienen en los casos de mensaje de LED, mensaje de *buzzer* y mensaje de configuración de periodicidad un tamaño fijo de 7 bytes siguiendo el siguiente formato:

CAMPO	Cabecera	Tipo mensaje	Dir. Origen	Dir. Destino	Tiempo
TAMAÑO	1 Byte	1 Byte	2 Bytes	2 Bytes	1 Byte

Figura 4.8: Formato de trama de mensajes.

- **Cabecera**: 1 byte con el valor en hexadecimal 0xFE, se utilizará para indicar el inicio de una trama.

- **Tipo mensaje**: 1 byte que contiene la información sobre el tipo de mensaje que a enviar. Los posibles valores de este campo se muestran en la tabla 4.4.

Tipo de mensajes	Valor (Hexadecimal)
Conf. de periodicidad	0x01
Buzzer	0x02
LED	0x03
Texto	0x04

Tabla 4.6: valores del campo Tipo de mensaje.

- **Dirección Origen**: 2 bytes con la dirección corta identificativa del dispositivo ZigBee/802.15.4 que debe enviar la trama. Un valor de 0xFFFF indicará que dicha trama debe ser enviada por todos los dispositivos de la red.

- **Dirección Destino**: 2 bytes con la dirección corta identificativa del dispositivo destinatario de la trama. Un valor de 0xFFFF indicará que dicha trama va destinada a todos los dispositivos de la red.

- **Tiempo**: Este campo sólo será útil cuando el tipo de mensaje indique que se trata de un mensaje de configuración de periodicidad. Un valor de 0x00 indicará entonces que no existan mensajes periódicos y cualquier otro valor distinto de 0x00 se traducirá en tiempo de periodicidad en segundos.

Para el caso de que el tipo de mensaje sea mensajes de texto, el formato de trama difiere levemente en el hecho de que ahora el tamaño de la trama será variable y dependerá de la palabra a enviar, en este caso la trama queda configurada de la siguiente forma:

Cabecera	Tipo mensaje	Dir. Origen	Dir. Destino	Longitud	Dato.1	...	Dato.n
1 Byte	1 Byte	2 Bytes	2 Bytes	1 Byte	1 Byte		1 Byte

Figura 4.9: Formato de trama para mensajes de texto.

- Los campos **Cabecera**, **Tipo mensaje**, **Dirección origen** y **Dirección destino** tienen la misma funcionalidad explicada en el formato de trama anterior.

- **Longitud**: 1 byte con el que se indica el número de bytes que siguen a este campo y que contienen la palabra a enviar.

- **Dato.1 … Dato.n**: *n* bytes con la palabra a enviar. Cada letra ocupa 1 byte que contiene una traducción numérica que corresponde con una letra asociada, así la letra “a” se traduce por 0x01, “b” por 0x02 y sucesivamente.

Mensajes en el sentido dispositivo → plataforma

En este sentido de la comunicación se realiza el intercambio de tramas necesario para la monitorización de la red. Se han creado tres tipos de tramas según la función a desarrollar en la plataforma. Por un lado las tramas cuya información sirve para la monitorización de la topología de red y obtención de los datos referentes a los distintos nodos que la conforman. Por otro lado están las tramas de monitorización de mensajes propios de una red ZigBee/802.15.4, ya sean éstos de configuración o sincronización y por último quedan los mensajes de datos, con información sobre el nodo emisor, el tipo de mensaje y la calidad del enlace.

- En el primer caso cada nodo manda periódicamente hacia el coordinador una trama informando sobre su presencia en la red con una actualización de sus principales parámetros. El coordinador, al detectar que se trata de una trama de monitorización, manda directamente dicha información por el puerto serie hacia la plataforma donde se estudia la información recogida y se actualizan los datos recibidos. El formato que rige este tipo de tramas se presenta a continuación:

cabecera	**long**	**tipodisp**	**PANID**	**canal**	**modoB**	**hijos**	**router**	**prof**	**dirOrig**	**dirPadre**
1 Byte	1	1	2	1	1	1	1	1	2	2

Figura 4.10: Formato de trama de mensajes de monitorización.

  - **Cabecera:** 1 byte cuyo valor siembre es 0xAB e indica el inicio de la trama.

  - **Long:** 1 byte para indicar el número de bytes que componen la trama, como es una trama de tamaño fijo su valor será siempre constante e igual a 0x0E (14 en decimal).

  - **Tipodisp:** 1 byte con el tipo de dispositivo que envía la trama. Los posibles valores para este campo se muestran en la siguiente tabla:

TipoDisp	**Valor** (Hexadecimal)
Coordinador	0xCC
Router	0xAA
End-Device	0xBB

Tabla 4.7: Posibles valores del campo TipoDisp.

- **PANID:** 2 bytes con la dirección identificativa de la red a la que se está conectado.
- **Canal:** 1 byte para indicar el canal lógico en el cual se está operando.
- **ModoB:** 1 byte con el que se indica si se trabaja en modo balizado o no balizado. Un valor de 15 en este campo indica modo no balizado, cualquier otro valor menor que 15 indica que se opera en modo balizado relacionado con el tiempo configurado entre balizas (este último modo de funcionamiento no se implementa en la plataforma).
- **Hijos:** Este byte contiene información sobre el número máximo de hijos que permite el nodo en cuestión, es un valor configurable por el usuario a través de la plataforma.
- **Router:** Byte con la información sobre el número de routers que el nodo permite tener asociados. Configurable a través de la plataforma.
- **Prof:** Byte que indica la profundidad máxima que el nodo permite tener. Configurable por el usuario a través de la plataforma.
- **DirOrig:** 2 bytes con información sobre la dirección corta asignada al dispositivo. Dicha dirección será la identificativa de cualquier acción que realice el nodo en la red.
- **DirPadre:** 2 bytes que indican la dirección corta identificativa del nodo "padre" del dispositivo que manda la trama, es decir el nodo al que está directamente asociado.

- El segundo caso está formado por tramas de tamaño fijo más reducido cuya información se manda en tiempo real a la plataforma a la que esté conectado cada dispositivo por puerto serie para su visualización. Dicha información contiene el aviso de la ejecución de algún comando característico de ZigBee/802.15.4 además de la información más relevante que aporta ese comando. El formato de este tipo de tramas se muestra a continuación.

CAMPO	Cabecera	TipoTrama	Dir1	Dir2	Ack	Bo	So
TAMAÑO	1 Byte	1	2	2	1	1	1

Figura 4.11: Formato de trama de mensajes de comandos.

- **Cabecera:** 1 byte fijo a 0xAF para informar sobre el inicio de la trama.
- **TipoTrama:** 1 byte para indicar qué comando se ha enviado:

TipoTrama	Valor (Hexadecimal)
PETICIÓN BEACON	0x01
TX BEACON	0x02
RX BEACON	0x03
PETICIÓN DE ASOCIACIÓN	0x04
RESPUESTA A LA ASOCIACIÓN	0x05
CONFIRMACIÓN DE ASOCIACIÓN	0x06
ASOCIACIÓN DENEGADA	0x07
NOTIFICACIÓN DE ORFANDAD	0x08
RESPUESTA A LA ORFANDAD	0x09
TX ACK	0x0A
RX ACK	0x0B
PETICIÒN DE DATOS	0x0C

Tabla 4.8: Posibles valores del campo TipoTrama.

- **Dir1:** 2 bytes que contienen una dirección. El significado de dicha dirección depende del tipo de trama.
- **Dir2:** 2 bytes que contienen una dirección. El significado de dicha dirección depende del tipo de trama.
- **Ack:** 1 byte que indica si el comando requiere de trama *ack*.
- **Bo:** 1 byte con la variable *BeaconOrder*. Este campo sólo se rellena en caso de que la trama sea de tipo Recepción de baliza. En cualquier otro caso se rellena con el valor 0x00.

- **So:** 1 byte con la variable *SuperframeOrder* Este campo sólo se rellena en caso de que la trama sea de tipo Recepción de baliza. En cualquier otro caso se rellena con el valor 0x00.

- Y por último, el tercer tipo de mensajes, los de datos, se envían por puerto serie cada vez que un nodo transmite o recibe una trama de datos desde o hacia cualquier otro dispositivo perteneciente a la red. Son tramas de tamaño fijo con la información básica sobre el contenido de dichos mensajes y cuyo formato se detalla en la figura 4.8.

CAMPO	Cabecera	Tx/Rx	ClusterId	DirOrig	Ack	LQI
TAMAÑO	1 Byte	1	1	2	1	1

Figura 4.12: Formato de trama de mensajes de datos.

- **Cabecera:** 1 byte con valor fijo a 0xBF para indicar el inicio de la trama.
- **Tx/Rx:** 1 byte con el que se señala si la trama es por transmisión o por recepción de datos.

Tx/Rx	Valor (Hexadecimal)
Tx	0x01
Rx	0x02

Tabla 4.9: Tabla de valores del campo Tx/Rx.

- **ClusterId:** 1 byte para indicar el tipo de trama de datos.

ClusterId	Tipo Mensaje
0x01	Periódico
0x02	LED
0x03	Buzzer
0x04	Texto

Tabla 4.10: Correspondencia entre variable ClusterId y tipo de mensaje.

- **DirOrig:** 2 bytes para indicar la fuente del mensaje.
- **Ack:** 1 byte con el que se informa si el mensaje requiere o no trama *ack*
- **LQI:** Un último byte para señalar la calidad del enlace por el que ha circulado el mensaje. Su valor oscila entre 0 y 255.

4.3.4. Comunicación entre dispositivos ZigBee/802.15.4

Las plataformas creadas son capaces de monitorizar todos aquellos mensajes, ya sean de comandos o de aplicación, que circulan por la red. A continuación se detallarán estos mensajes, definiendo brevemente su utilidad y mostrando su formato de trama y posibles valores para los principales campos que contienen:

Mensaje de petición de baliza (*Beacon request*)

Mensaje utilizado por los dispositivos para localizar una red al inicializarse o al perder sincronización.

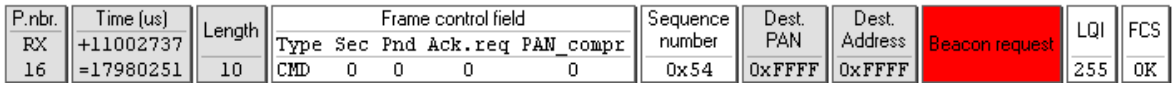


Figura 4.13: Mensaje de petición de baliza.

Mensaje de baliza (*Beacon frame*)

Mensaje con el que un nodo coordinador o *router* informan de su existencia y proporcionan sincronización a un nuevo dispositivo detectado

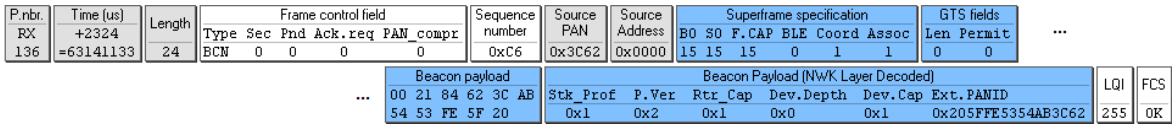


Figura 4.14: Mensaje de baliza.

Campos destacados:

- *Source Address*: Dirección del dispositivo que emite la baliza
- *Superframe specification*: Contiene información referente al modo de funcionamiento de la red y configuración del nodo emisor:
    - *BO* (*BeaconOrder*): indica si se está trabajando en modo balizado o no.
    - *SO* (*SuperframeOrder*): en caso de estar operando en modo balizado, esta variable indica la duración de la supertrama.
    - *Coord*: indica si el dispositivo emisor de la baliza es el coordinador de la red.
    - *Assoc*: variable que indica si el dispositivo permite o no asociaciones.

Mensaje de petición de asociación (*Association request*)

Mensaje con el que un dispositivo inicia el proceso de asociación a un nodo "padre".

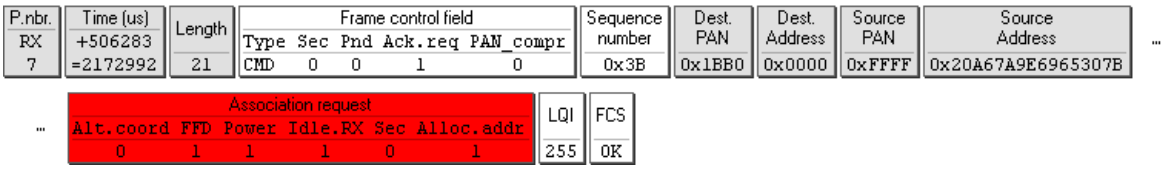


Figura 4.15: Mensaje de petición de asociación.

Mensaje de respuesta de asociación (*Association response – Association Successful*)

Con este mensaje un nodo informa a un nuevo "hijo" que se ha establecido una asociación con éste y le proporciona una dirección corta con la que identificarse en la red.

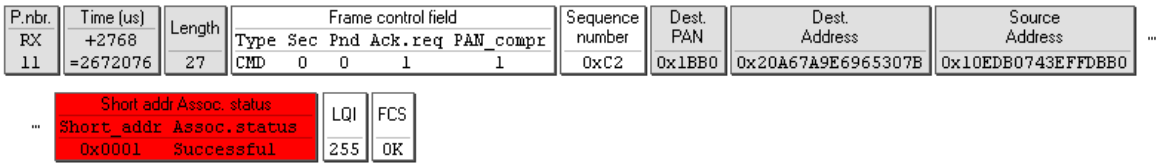


Figura 4.16: Mensaje de respuesta de asociación.

Mensaje de asociación denegada (*Association response – Association denied*)

Mensaje utilizado por un dispositivo para informar a un nodo que trata de asociársele de que dicha asociación no se ha podido establecer.

Figura 4.17: Mensaje de asociación denegada.

Mensaje de notificación de orfandad (*Orphan notification*)

Mensaje emitido por un dispositivo cuando pierde comunicación con su nodo "padre" para indicar que necesita un dispositivo al que asociarse.

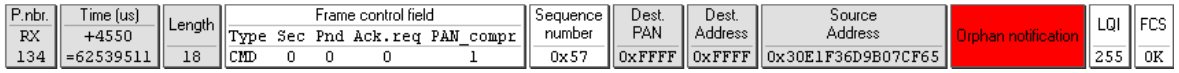


Figura 4.18: Mensaje de notificación de orfandad.

Mensaje de petición de datos (*Data request*)

Mensaje con el que un dispositivo indica que está preparado para la recepción de datos.

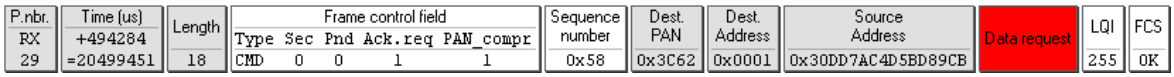


Figura 4.19: Mensaje de petición de datos.

Mensaje de *ack (ack frame)*

Respuesta para la confirmación de recepción de un mensaje. El requerimiento de tramas *ack* es configurable para cada tipo de mensajes.

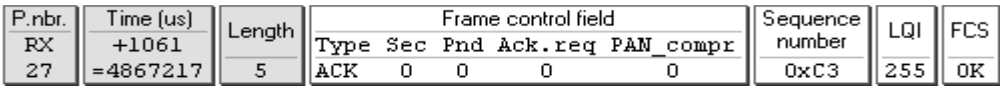


Figura 4.20: Mensaje de *ack*.

Mensaje de encendido de *Buzzer*

Mensaje con el que se ordena al destino la activación de un buzzer.

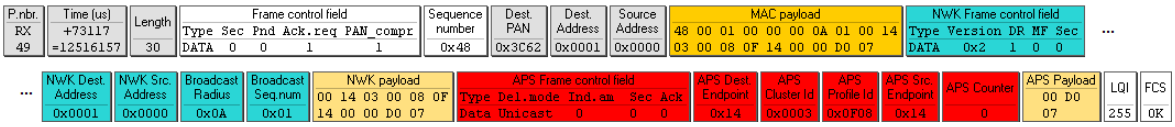


Figura 4.21: Mensaje de activación de *buzzer*.

Campos destacados:

- *Frame control field*: Campo con información de control sobre la trama transmitida:
  - *Type*: indica el tipo de trama transmitida.
  - *Sec*: indica si el mensaje está provisto de seguridad para dicha capa.
  - *Ack req*: indica si el mensaje requiere de trama *ack* de confirmación de recepción.
- *Dest. Address*: Dirección destino del mensaje.
- *Source Address*: Dirección origen del mensaje.
- *Radius Broadcast*: Campo con información sobre la distancia de alcance del mensaje.

- *APS Frame control field*: Añade información sobre si el mensaje es de difusión (*broadcast*), es hacia un solo nodo (*unicast*) o hacia un grupo de nodos (*Group*).

- *APS Cluster Id*: 2 bytes que indican el tipo de mensaje de datos que contiene la trama. 0x0002 indica mensaje de activación de *buzzer*.

- *APS Payload:* Contiene datos adicionales para la ejecución de la acción indicada por el mensaje en destino.

Mensaje de encendido de LED

Mensaje con el que se ordena al destino el encendido de un diodo LED.

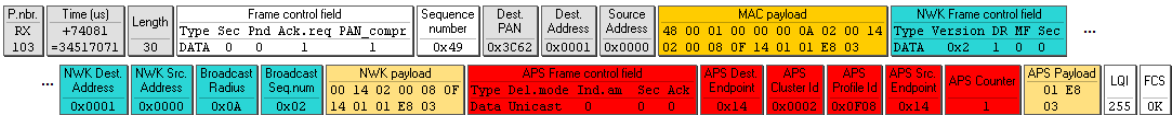


Figura 4.22: Mensaje de activación del LED.

Mensaje de monitorización de texto en LCD

Mensaje para la monitorización en el LCD del dispositivo destino de un texto corto.

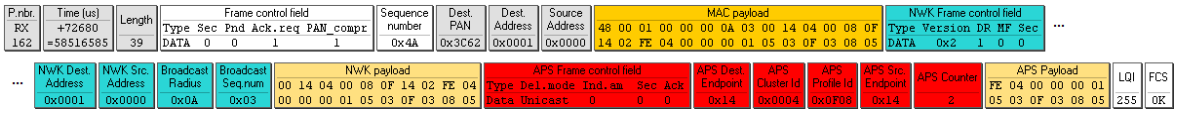


Figura 4.23: Mensaje de monitorización de texto en LCD.

- *APS Payload:* En este caso el contenido de este campo es una trama completa con el mensaje corto a visualizar en el LCD del dispositivo destino (Ver figura 4.9).

Mensaje periódico

Trama utilizada para el envío periódico de un mensaje. En principio no contiene información adicional aunque ésta puede ser añadida en el campo de datos (*APS Payload*).

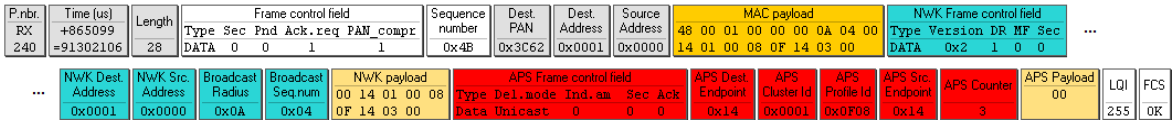


Figura 4.24: Mensaje periódico.

Mensaje de actualización de monitorización de la plataforma

Mensaje periódico con el que se actualiza la información almacenada y mostrada en la plataforma de monitorización y configuración.

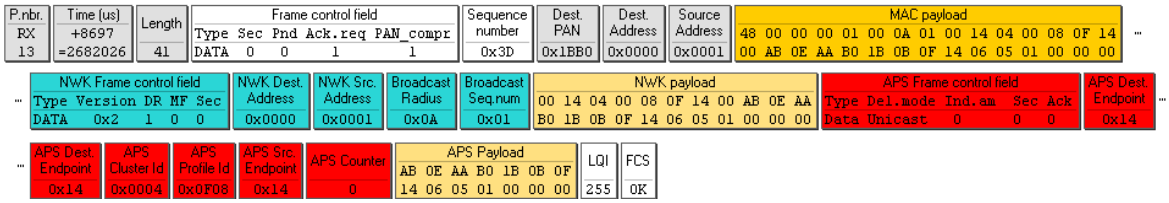


Figura 4.25: Mensaje de actualización de monitorización.

- *APS Payload:* El contenido de este campo es una trama completa con los datos para la monitorización de la red en la plataforma del coordinador. (Ver figura 4.10).

CAPÍTULO 5

Plan de pruebas

En este capítulo se detallan las distintas pruebas realizadas sobre la plataforma diseñada con el fin de demostrar su correcto funcionamiento en un número amplio de situaciones. Para estas pruebas se ha utilizado un PC con las siguientes características:

Intel Pentium 4 CPU 3.20GHz
3.20GHz, 1.00GB de RAM

El sistema operativo utilizado ha sido Windows XP, aunque también se ha ejecutado la aplicación en Windows Vista para comprobar que no existen problemas de compatibilidad.

Las pruebas realizadas se han clasificado en 4 tipos según el objetivo de los mismos:

- Pruebas de funcionalidad básica: se testea el correcto funcionamiento de los dispositivos físicos: periféricos (LCD, LED, *buzzer*, botones, puerto serie) y alimentación.

- Pruebas de configuración: Se describen las pruebas realizadas para distintas combinaciones de valores de los parámetros que afectan a la configuración y topología de la red.

- Pruebas de mensajes: Se detallan los resultados obtenidos en las pruebas realizadas para todos los tipos de mensajes y las posibles combinaciones de dispositivos origen y destino.

- Pruebas de capacidad: Se testea la monitorización de la plataforma al máximo de capacidad disponible.

5.1. Pruebas de funcionalidad básica

Se inicia este plan de pruebas testeando la correcta programación del interfaz visual de los dispositivos físicos, es decir, el LCD. Para ello se comprueba que tanto los mensajes programados para ser visualizados en la zona de texto del LCD como los distintos iconos que contiene dicho LCD funcionan correctamente y se muestran cuando así se requiere.

Los resultados de estas pruebas se han presentado en las tablas 5.1 y 5.2, indicándose con OK que la respuesta a la prueba fue la esperada.

LCD (Mensajes de comandos):

LCD	Significado	Resultado Test
Mensaje BEAC_PT	Mensaje de petición de baliza	**OK**
Mensaje BEAC_RX	Mensaje de recepción de baliza	**OK**
Mensaje ASOC_PT	Mensaje de petición de asociación	**OK**
Mensaje ASOC_RS	Mensaje de respuesta de asociación	**OK**
Mensaje ASOC_CF	Mensaje de confirmación de asociación	**OK**
Mensaje ASOC_DN	Mensaje de asociación denegada	**OK**
Mensaje ORF_NOT	Mensaje de notificación de orfandad	**OK**

Tabla 5.1: Pruebas de mensajes en LCD

LCD (Iconos):

LCD	Resultado Test
DOLAR (Coordinador)	**OK**
TX	**OK**
RX	**OK**
SOBRE (RX LED)	**OK**
SOBRE (RX BUZZER)	**OK**
SOBRE (RX TEXTO)	**OK**
ANTENA	**OK**
LQI (Alto)	**OK**
LQI (Medio)	**OK**
LQI (Bajo)	**OK**
BATT (Alto)	**OK**
BATT (Medio)	**OK**
BATT (Bajo)	**OK**
ENCENDIDO	**OK**

Tabla 5.2: Pruebas iconos en LCD.

En las tablas 5.3, 5.4, 5.5, 5.6 y 5.7 se detallan los resultados obtenidos en las pruebas realizadas a los diferentes periféricos utilizados en este proyecto. Estas pruebas se basan en la comprobación del encendido y apagado de LED, activación y desactivación del *buzzer*, correcta respuesta a la pulsación de botones y funcionamiento adecuado tanto en transmisión como en recepción por puerto serie:

LED

LED	Resultado Test
LED 1	**OK**
LED 2	**OK**

Tabla 5.3: Pruebas de LED.

Buzzer

BUZZER	Resultado Test
Encendido/Apagado	**OK**

Tabla 5.4: Pruebas de *buzzer*.

Botones

Botones	Resultado Test
Botón 1 (envío mens. *Buzzer*)	**OK**
Botón 2 (envío mens. LED)	**OK**

Tabla 5.5: Pruebas de botones.

Puerto serie

Puerto serie	Resultado Test
Transmisión	**OK**
Recepción	**OK**

Tabla 5.6: Pruebas de puerto serie.

Por último dentro de esta agrupación de pruebas, se testea que ambas fuentes de alimentación (baterías AAA y alimentación externa) funcionan correctamente:

Alimentación

Tipo Alimentación	Estado
BATERIAS (Interna)	**OK**
FET (Externa)	**OK**

Tabla 5.7: Pruebas de alimentación.

5.2. Pruebas de configuración

Para la realización de este tipo de pruebas se han establecido 6 escenarios con los que se demuestra el correcto funcionamiento de la plataforma y los dispositivos conectados ante las distintas opciones de configuración programadas. En estos escenarios se han establecido valores extremos para los parámetros con el fin de comprobar no sólo que los dispositivos se configuran según lo decidido en la plataforma, sino también que se comportan como deben bajo dicha configuración. Así mismo se demuestra que la plataforma monitoriza correctamente dicha configuración y sus consecuencias.

Para cada uno de los 6 escenarios se adjuntan las tramas capturadas por un *sniffer* junto con las capturas que muestran el comportamiento de la plataforma, de este modo se puede observar, comparando ambos sistemas, el adecuado funcionamiento de la aplicación.

Partimos de la siguiente topología inicial (figura 5.1) para proseguir con distintos escenarios.

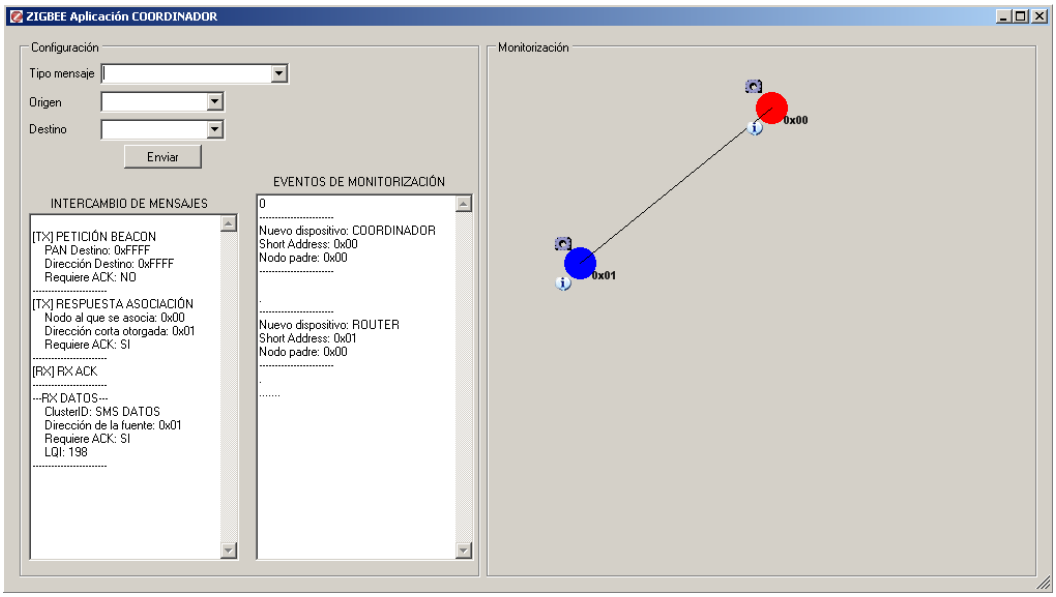


Figura 5.1: Situación inicial para plan de pruebas.

Escenario 1: *Creación de una red en la que tanto el coordinador como el router aceptan asociaciones*

Inicializamos ahora un dispositivo final. Ambos dispositivos, coordinador y *router*, mandan balizas para sincronizarse con el nuevo nodo. Estas balizas contienen el valor 0x01 en el campo *Assoc* indicando que permiten la asociación de un nuevo dispositivo. El dispositivo final pide asociación al coordinador y se vincula a él creándose una topología

de árbol de profundidad uno en la que el coordinador tiene dos hijos: el *router* y el dispositivo final.

Tramas capturadas por un *sniffer*:

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Dest. PAN	Dest. Address		LQI	FCS
RX 24	+4121933 =54172385	10	Type Sec Pnd Ack.req PAN_compr CMD 0 0 0 0	0x17	0xFFFF	0xFFFF	Beacon request	255	OK

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Source PAN	Source Address	Superframe specification	GTS fields	Beacon payload
RX 25	+1870 =54174255	24	Type Sec Pnd Ack.req PAN_compr BCN 0 0 0 0	0x39	0x3C62	0x0000	BO SO F.CAP BLE Coord Assoc 15 15 15 0 1 1	Len Permit 0 0	00 21 84 62 3C 54 53 FE 5F 20

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Source PAN	Source Address	Superframe specification	GTS fields	Beacon payload
RX 26	+3885 =54178140	24	Type Sec Pnd Ack.req PAN_compr BCN 0 0 0 0	0xBB	0x3C62	0x0001	BO SO F.CAP BLE Coord Assoc 15 15 15 0 0 1	Len Permit 0 0	00 21 8C 62 3C 54 53 FE 5F 20

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Dest. PAN	Dest. Address		LQI	FCS
RX 27	+746831 =54924971	10	Type Sec Pnd Ack.req PAN_compr CMD 0 0 0 0	0x18	0xFFFF	0xFFFF	Beacon request	255	OK

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Source PAN	Source Address	Superframe specification	GTS fields	Beacon payload
RX 28	+2228 =54927199	24	Type Sec Pnd Ack.req PAN_compr BCN 0 0 0 0	0xBC	0x3C62	0x0001	BO SO F.CAP BLE Coord Assoc 15 15 15 0 0 1	Len Permit 0 0	00 21 8C 62 3C 54 53 FE 5F 20

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Source PAN	Source Address	Superframe specification	GTS fields	Beacon payload
RX 29	+1512 =54928711	24	Type Sec Pnd Ack.req PAN_compr BCN 0 0 0 0	0x3A	0x3C62	0x0000	BO SO F.CAP BLE Coord Assoc 15 15 15 0 1 1	Len Permit 0 0	00 21 84 62 3C 54 53 FE 5F 20

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Dest. PAN	Dest. Address		LQI	FCS
RX 30	+806441 =55735152	10	Type Sec Pnd Ack.req PAN_compr CMD 0 0 0 0	0x19	0xFFFF	0xFFFF	Beacon request	255	OK

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Source PAN	Source Address	Superframe specification	GTS fields	Beacon payload
RX 31	+1919 =55737071	24	Type Sec Pnd Ack.req PAN_compr BCN 0 0 0 0	0xBD	0x3C62	0x0001	BO SO F.CAP BLE Coord Assoc 15 15 15 0 0 1	Len Permit 0 0	00 21 8C 62 3C 54 53 FE 5F 20

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Source PAN	Source Address	Superframe specification	GTS fields	Beacon payload
RX 32	+1511 =55738582	24	Type Sec Pnd Ack.req PAN_compr BCN 0 0 0 0	0x3B	0x3C62	0x0000	BO SO F.CAP BLE Coord Assoc 15 15 15 0 1 1	Len Permit 0 0	00 21 84 62 3C 54 53 FE 5F 20

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Dest. PAN	Dest. Address	Source PAN	Source Address	Association request
RX 33	+508853 =56247435	21	Type Sec Pnd Ack.req PAN_compr CMD 0 0 1 0	0x1A	0x3C62	0x0000	0xFFFF	0x30756BA24FC790A7	Alt.coord FFD Power Idle.RX 0 0 0 0

P.nbr.	Time (us)	Length	Frame control field	Sequence number	LQI	FCS
RX 34	+1060 =56248495	5	Type Sec Pnd Ack.req PAN_compr ACK 0 0 0 0	0x1A	255	OK

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Dest. PAN	Dest. Address	Source Address		LQI	FCS
RX 35	+493324 =56741819	18	Type Sec Pnd Ack.req PAN_compr CMD 0 0 1 1	0x1B	0x3C62	0x0000	0x30756BA24FC790A7	Data request	255	OK

P.nbr.	Time (us)	Length	Frame control field	Sequence number	LQI	FCS
RX 36	+964 =56742783	5	Type Sec Pnd Ack.req PAN_compr ACK 0 1 0 0	0x1B	255	OK

P.nbr.	Time (us)	Length	Frame control field	Sequence number	Dest. PAN	Dest. Address	Source Address	Short addr Assoc. status
RX 37	+1239 =56744022	27	Type Sec Pnd Ack.req PAN_compr CMD 0 0 1 1	0x95	0x3C62	0x30756BA24FC790A7	0x205FFE5354AB3C62	Short_addr Assoc.status 0x796F Successful

Figura 5.2: Escenario 1, captura del *sniffer*.

Monitorización realizada por la plataforma:

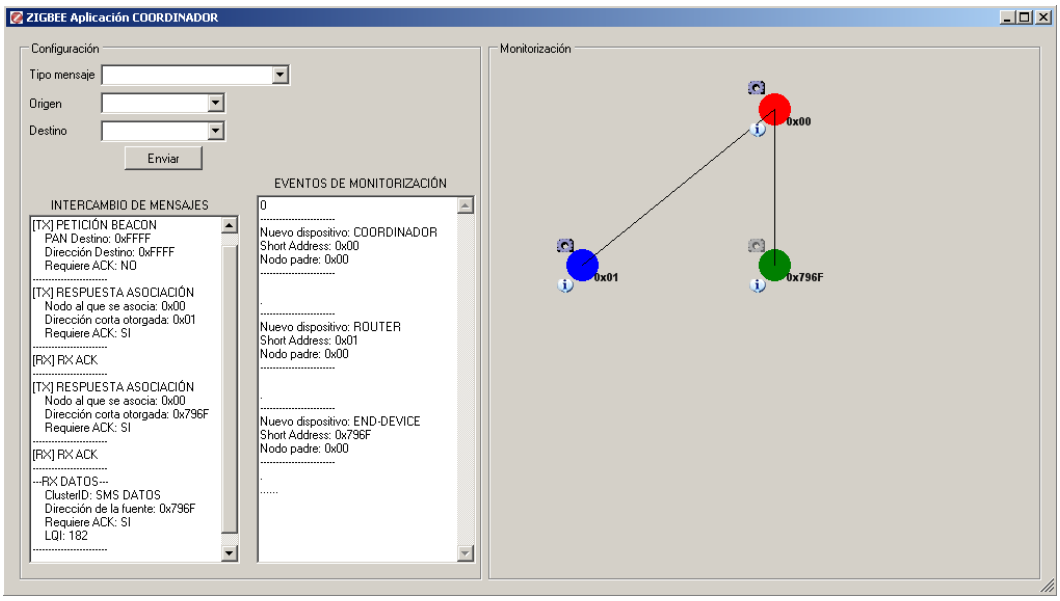
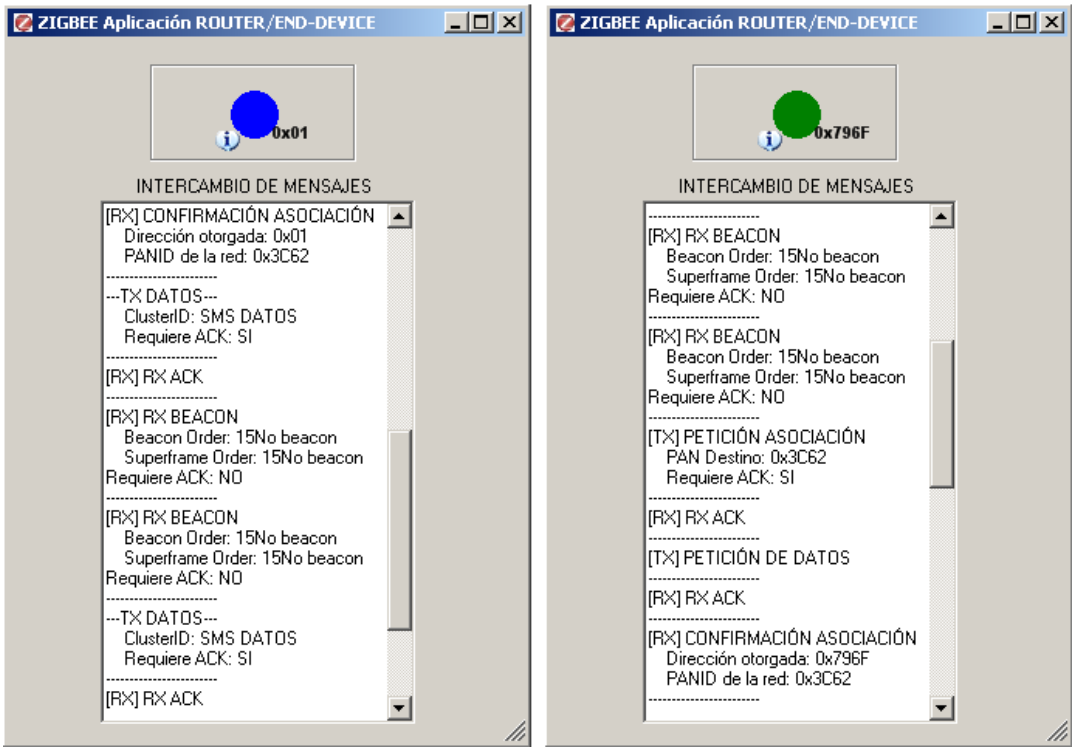


Figura 5.3: Escenario 1, captura de la plataforma del coordinador.

En la ventana INTERCAMBIO DE MENSAJES se muestra cómo el coordinador responde a ambas asociaciones, la del dispositivo *router* y la del dispositivo final, asumiendo su papel de nodo padre y otorgando direcciones cortas a ambos. Además la ventana EVENTOS DE MONITORIZACIÓN detalla la situación actual de la red ofreciendo información sobre el tipo de dispositivo detectado, su dirección corta asignada y la dirección corta del dispositivo que asume el papel de nodo padre. Dicha información es obtenida de la trama de actualización de monitorización recibida por el puerto serie.



(a) (b)

Figura 5.4: Escenario 1, captura de la plataforma del *router* y *end-device.* (a) *Router*. (b) *End-device*.

En la plataforma de monitorización del dispositivo final (imagen de la derecha) se observa cómo este dispositivo realiza una petición de asociación y poco después recibe la confirmación de asociación con la dirección corta otorgada.

Escenario 2: *Creación de una red en la que el coordinador no acepta más asociaciones y el router sí*

Partiendo de nuevo de la situación inicial (figura 5.1), inicializamos ahora el dispositivo final. El coordinador y el *router* mandan balizas, pero en este caso el coordinador manda en la baliza información de que no acepta nuevas asociaciones al contener el campo *Assoc*

el valor 0x00. El nuevo nodo inicia el proceso de asociación con el *router* vinculándose a éste y creándose una topología de árbol de profundidad 2.

Tramas capturadas por un *sniffer:*

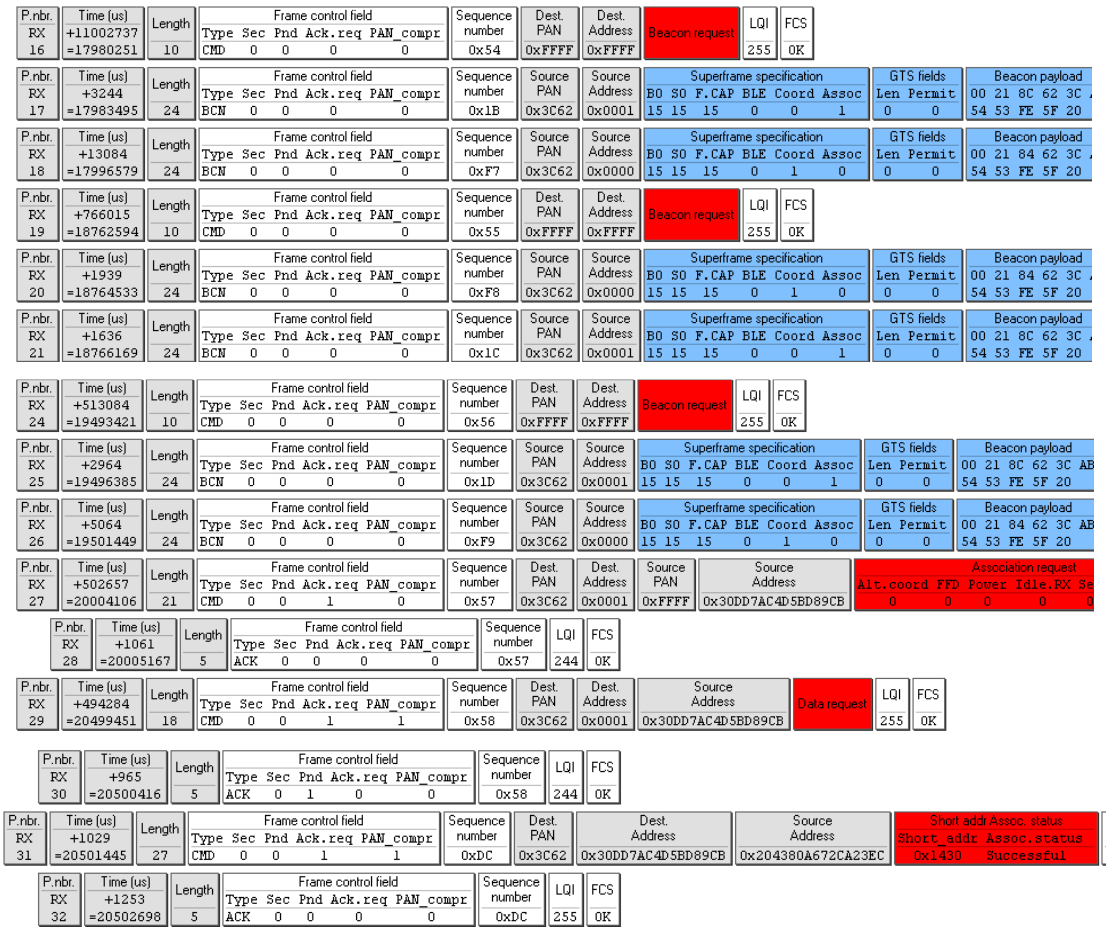


Figura 5.5: Escenario 2, captura del *sniffer*.

Monitorización realizada por la plataforma:

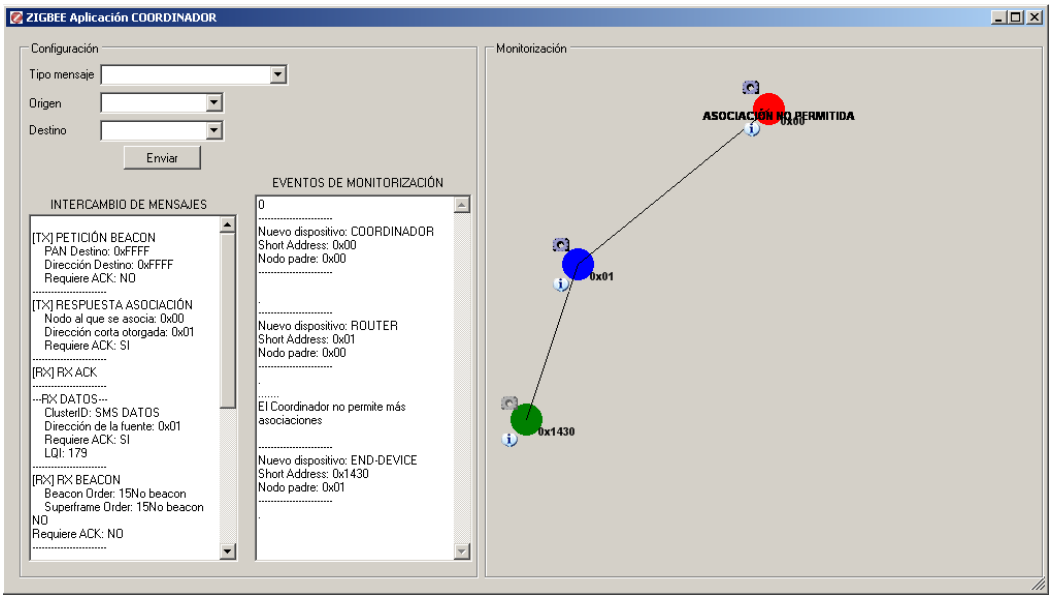
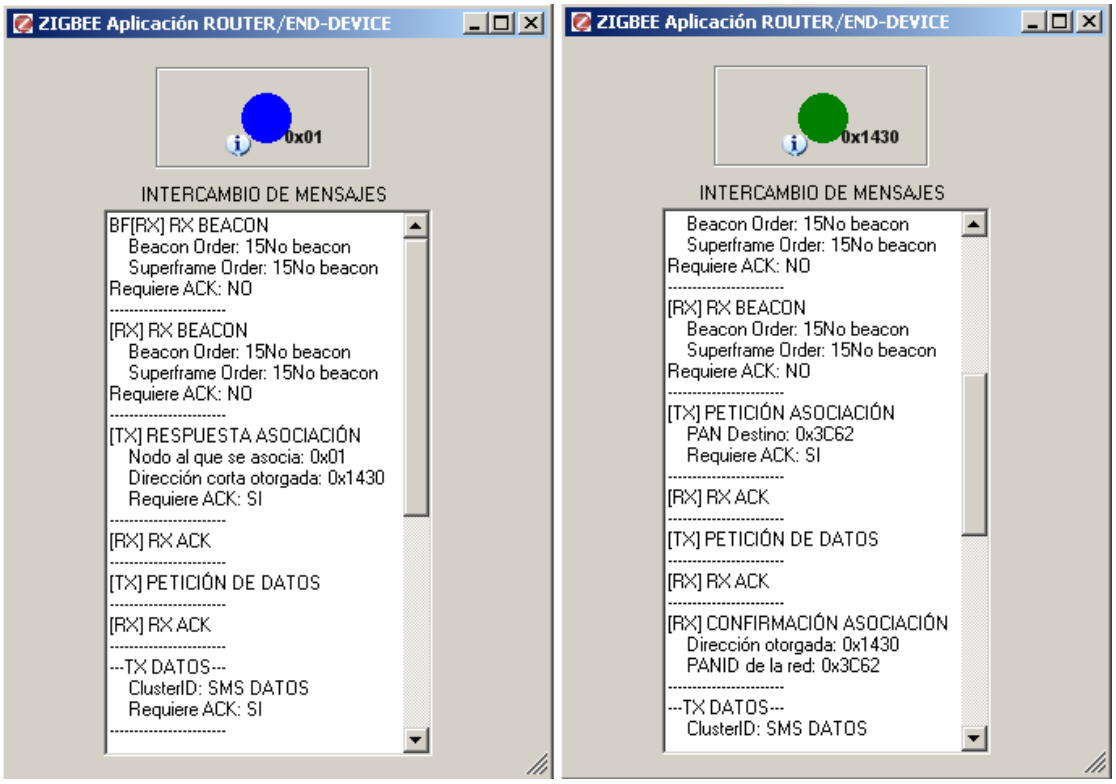


Figura 5.6: Escenario 2, captura de la plataforma del coordinador.

Se puede observar cómo la ventana de EVENTOS DE MONITORIZACIÓN registra la entrada de un nuevo dispositivo de tipo END-DEVICE cuyo nodo padre es el 0x0001. Así mismo se observa que esta topología se ha conseguido por el correcto funcionamiento de la opción "No permitir asociación" configurada a través de la plataforma.



(a) (b)

Figura 5.7: Escenario 2, captura de la plataforma del *router* y *end-device*. (a) *Router*. (b) *End-device*.

En la imagen de la izquierda se puede ver como es el dispositivo *router* con dirección corta 0x0001 el que asume el papel de nodo padre y el que otorga la dirección corta al nuevo dispositivo conectado. En la imagen de la derecha se observa cómo el dispositivo final recibe una confirmación de asociación con dicha dirección corta otorgada.

Escenario 3: *Tanto el coordinador como el router no aceptan asociación*

Inicializamos ahora el dispositivo final. Ambos dispositivos, coordinador y *router*, mandan balizas indicando que no aceptan asociaciones, es decir, con el campo *Assoc* a 0x00. El nuevo dispositivo no podrá asociarse a ninguno de los nodos existentes en la red y seguirá buscando indefinidamente un nodo al que conectarse, es decir, continuará mandando peticiones de balizas hasta conseguir un nodo al que asociarse.

Tramas capturadas por el *sniffer*:

P.nbr.	Time (us)	Length	Frame control field								Sequence number	Dest. PAN	Dest. Address	Source PAN	Source Address	Superframe specification						GTS fields		Beacon payload								LQI	FCS
			Type	Sec	Pnd	Ack.req	PAN_compr									BO	SO	F.CAP	BLE	Coord	Assoc	Len	Permit										
RX	+211921		CMD	0	0	0	0				0x1A	0xFFFF	0xFFFF			Beacon request															255	OK
31	=16782133	10																														
RX	+2066		BCN	0	0	0	0				0x6F	0x3C62	0x0001			15	15	15	0	0	0	0	0	00	21	8C	62	3C	AB				
32	=16784199	24																						54	53	FE	5F	20				
RX	+1708		BCN	0	0	0	0				0x9A	0x3C62	0x0000			15	15	15	0	1	0	0	0	00	21	84	62	3C	AB				
33	=16785907	24																						54	53	FE	5F	20				
RX	+821446		CMD	0	0	0	0				0x1B	0xFFFF	0xFFFF			Beacon request															255	OK
34	=17607353	10																														
RX	+1865		BCN	0	0	0	0				0x9B	0x3C62	0x0000			15	15	15	0	1	0	0	0	00	21	84	62	3C	AB				
35	=17609218	24																						54	53	FE	5F	20				
RX	+5971		BCN	0	0	0	0				0x70	0x3C62	0x0001			15	15	15	0	0	0	0	0	00	21	84	62	3C	AB				
36	=17615189	24																						54	53	FE	5F	20				
RX	+739950		CMD	0	0	0	0				0x1C	0xFFFF	0xFFFF			Beacon request															255	OK
37	=18355139	10																														
RX	+608322		CMD	0	0	0	0				0x1D	0xFFFF	0xFFFF			Beacon request															255	OK
39	=18965655	10																														
RX	+2841		BCN	0	0	0	0				0x9D	0x3C62	0x0000			15	15	15	0	1	0	0	0	00	21	84	62	3C	AB				
40	=18968496	24																						54	53	FE	5F	20				
RX	+4052		BCN	0	0	0	0				0x72	0x3C62	0x0001			15	15	15	0	0	0	0	0	00	21	8C	62	3C	AB				
41	=18972548	24																						54	53	FE	5F	20				
RX	+796248		CMD	0	0	0	0				0x1E	0xFFFF	0xFFFF			Beacon request															255	OK
42	=19768796	10																														
RX	+1892		BCN	0	0	0	0				0x9E	0x3C62	0x0000			15	15	15	0	1	0	0	0	00	21	84	62	3C	AB				
43	=19770688	24																						54	53	FE	5F	20				
RX	+2132		BCN	0	0	0	0				0x73	0x3C62	0x0001			15	15	15	0	0	0	0	0	00	21	8C	62	3C	AB				
44	=19772820	24																						54	53	FE	5F	20				
RX	+753681		CMD	0	0	0	0				0x1F	0xFFFF	0xFFFF			Beacon request															255	OK
45	=20526501	10																														

Figura 5.8: Escenario 3, captura del *sniffer*.

Monitorización realizada por la plataforma:

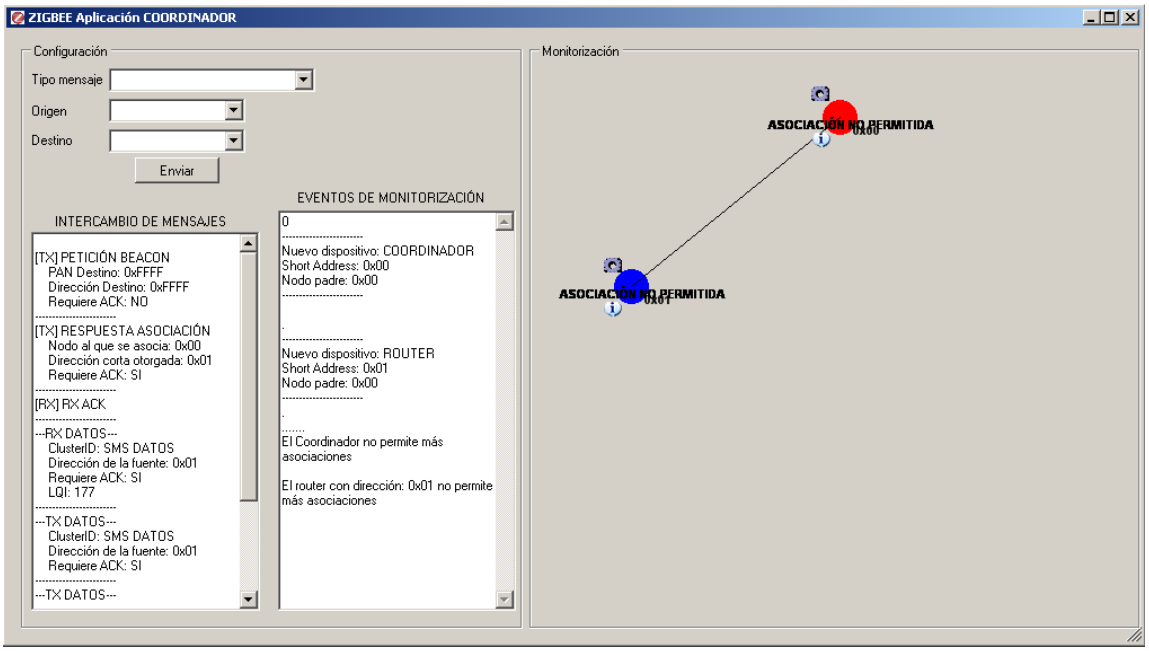


Figura 5.9: Escenario 3, captura de la plataforma del coordinador.

En la figura 5.9 se aprecia como ambos dispositivos, coordinador y *router* han sido configurados para no permitir nuevas asociaciones. El *end-device* que quiere asociarse no encuentra un nodo al que hacerlo y por tanto no es capaz de conectarse a la red. Dicha información es mostrada tanto en la zona de monitorización como en la ventana de EVENTOS DE MONITORIZACIÓN.

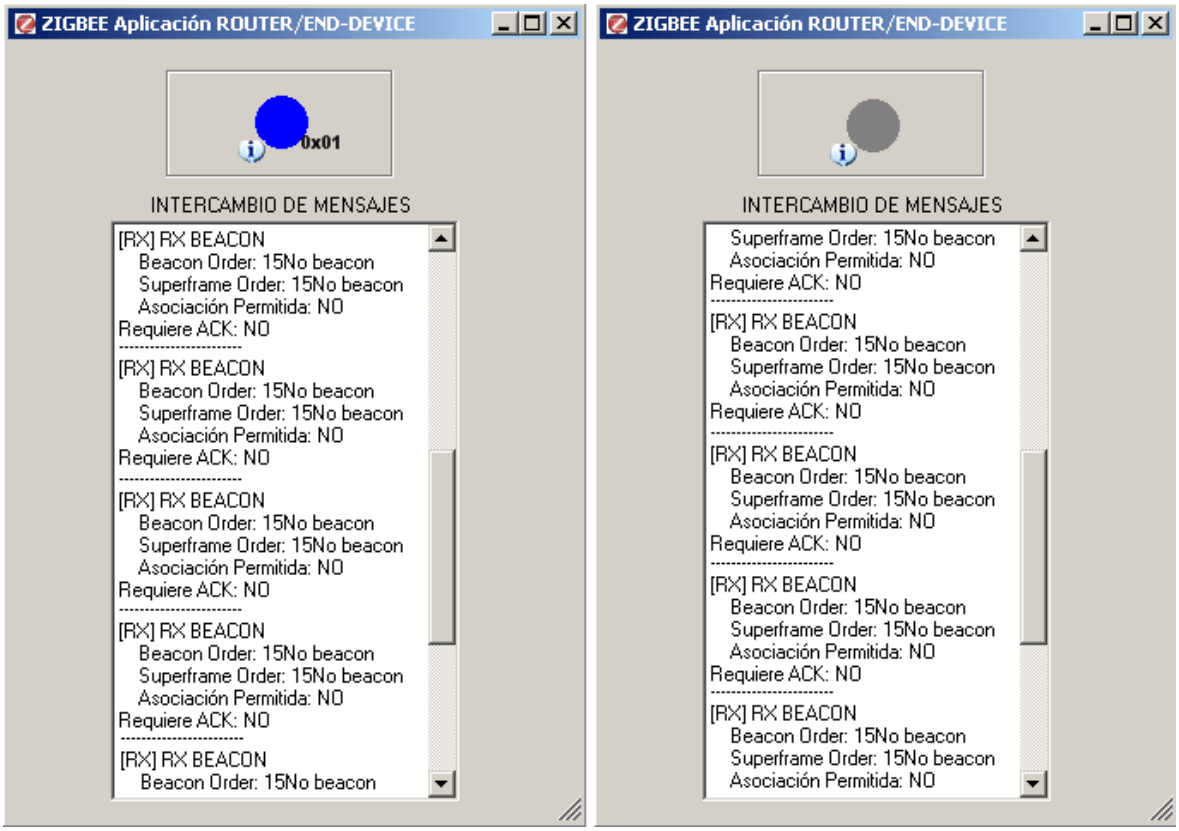


Figura 5.10: Escenario 3, captura de la plataforma del *router* y *end-device*.

Todos los dispositivos reciben balizas enviadas por coordinador y *router*, pero el dispositivo final no consigue asociarse a ninguno de los 2 nodos.

Escenario 4: *El coordinador sólo acepta un hijo y ya lo tiene, el router no acepta asociaciones*

En esta situación, como respuesta a la petición de balizas realizada por el nuevo dispositivo, el coordinador y el *router* mandarán éstas, el *router* indicando que no permite nuevas asociaciones en el campo *Assoc* y el coordinador indicando que, en principio, sí acepta dicha asociación. El nuevo dispositivo inicia el proceso de asociación y el coordinador le responde con una denegación de asociación indicada en el campo *Assoc. Status*. El dispositivo se queda sin conectarse a la red e inicia un nuevo proceso de petición de balizas en busca de algún nodo al que vincularse.

Tramas capturadas por el *sniffer*:

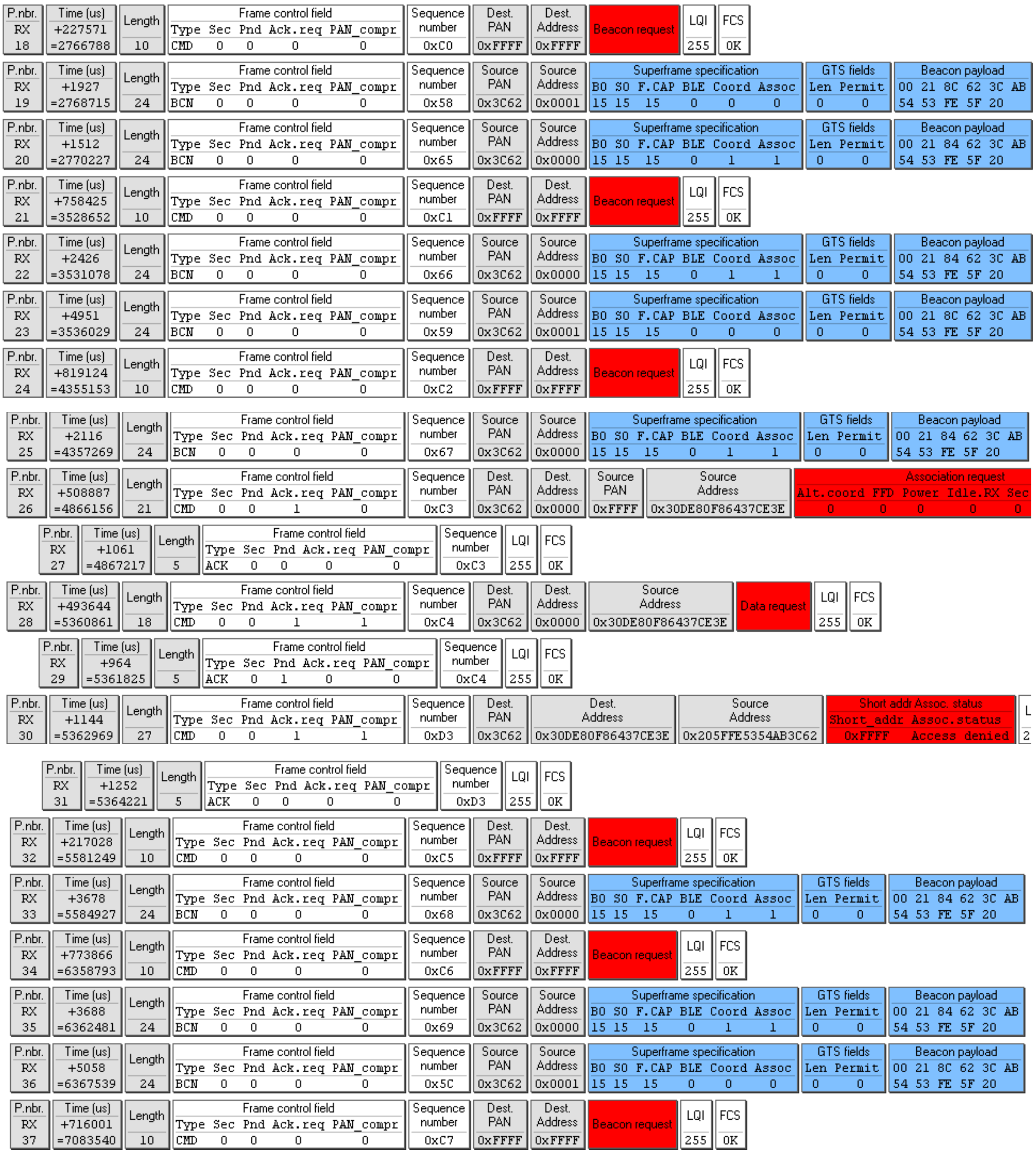


Figura 5.11: Escenario 4, captura del *sniffer*.

En la figura 5.11 se ve cómo un dispositivo inicia el proceso de asociación (trama 26) con el nodo coordinador, pero este último acaba denegándolo (trama 30).

En la figura 5.12 se puede observar que la plataforma detecta un nuevo dispositivo, que el *router* no acepta nuevas asociaciones y que el coordinador está configurado para tener un máximo de un hijo y que este hijo puede ser un *router*, sin embargo no se termina de establecer asociación alguna.

Viendo el intercambio de tramas en la figura 5.11 y en la plataforma del dispositivo final (figura 5.13) se demuestra que los resultados de configurar los parámetros "número máximo de hijos" y "número máximo de *routers*" son correctos.

Monitorización realizada por la plataforma:

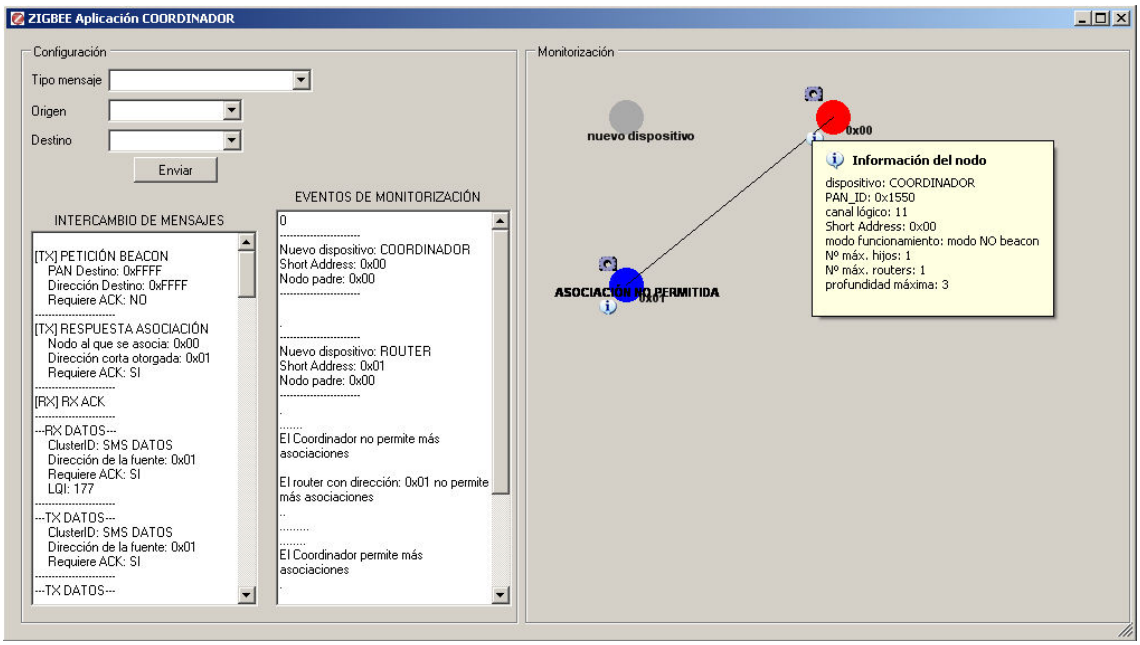


Figura 5.12: Escenario 4, captura de la plataforma del coordinador.

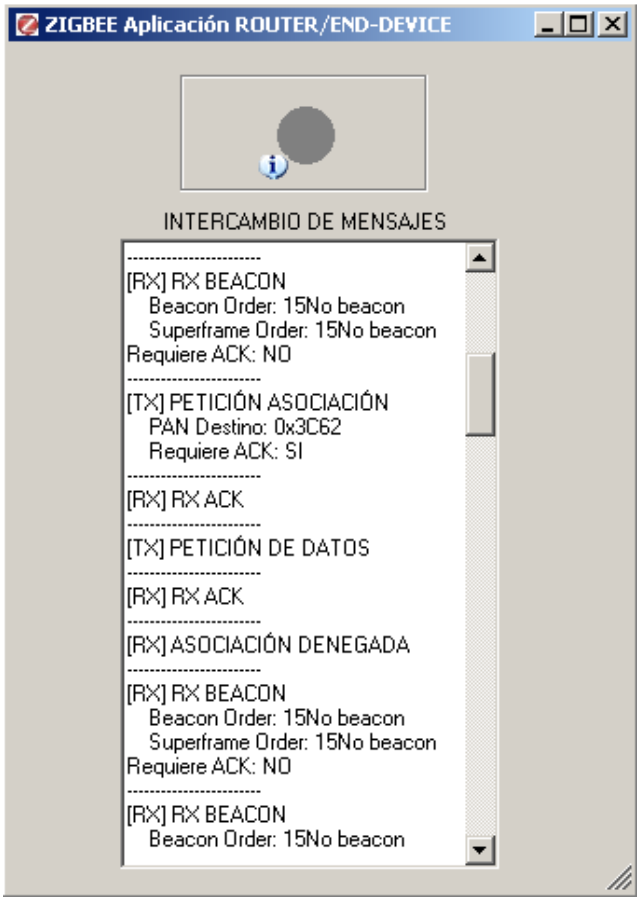


Figura 5.13: Escenario 4, captura de la plataforma del nuevo dispositivo.

Escenario 5: *El coordinador sólo acepta un hijo y ya lo tiene, el router está configurado para no aceptar hijos*

En este escenario ocurre exactamente lo mismo que en el escenario 4, sólo que ahora ambos nodos, coordinador y *router* aceptan en principio, asociaciones. El nuevo dispositivo intentará asociarse al coordinador y en el proceso de asociación se le denegará dicha asociación al no poder tener más hijos el coordinador.

Tramas capturadas por el *sniffer*:

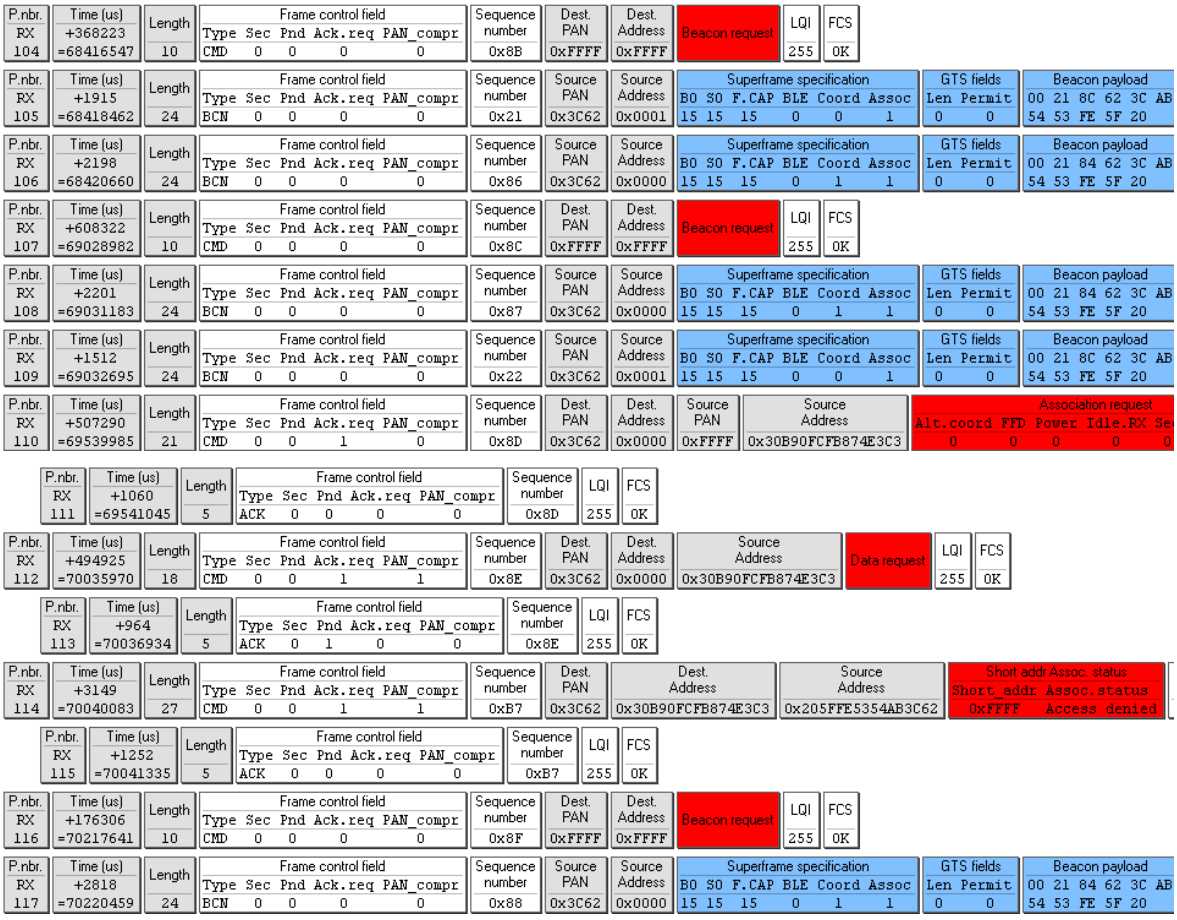


Figura 5.14: Escenario 5, captura del *sniffer*.

En este escenario se comprueba la correcta configuración de los parámetros “número máximo de hijos”, “número máximo de *routers*” y “profundidad máxima” para un nodo distinto al coordinador.

Se observa cómo los resultados obtenidos son similares a los del escenario 4 tanto en las tramas capturadas, como en la monitorización de las plataformas del coordinador y dispositivo final.

Monitorización de la plataforma:

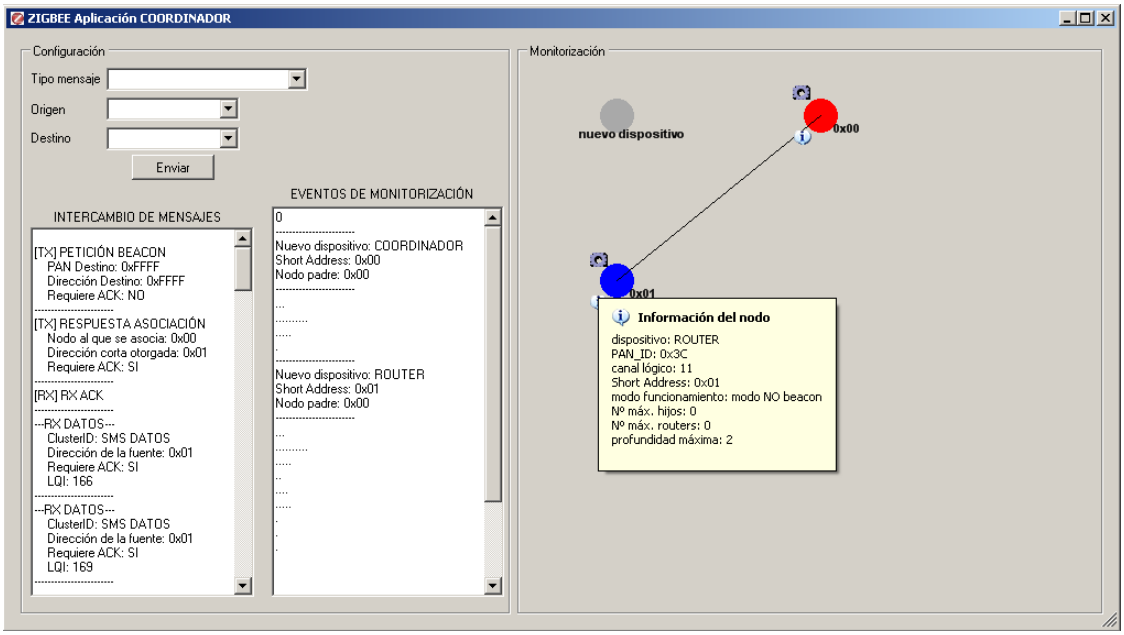


Figura 5.15: Escenario 5, captura de la plataforma del coordinador.

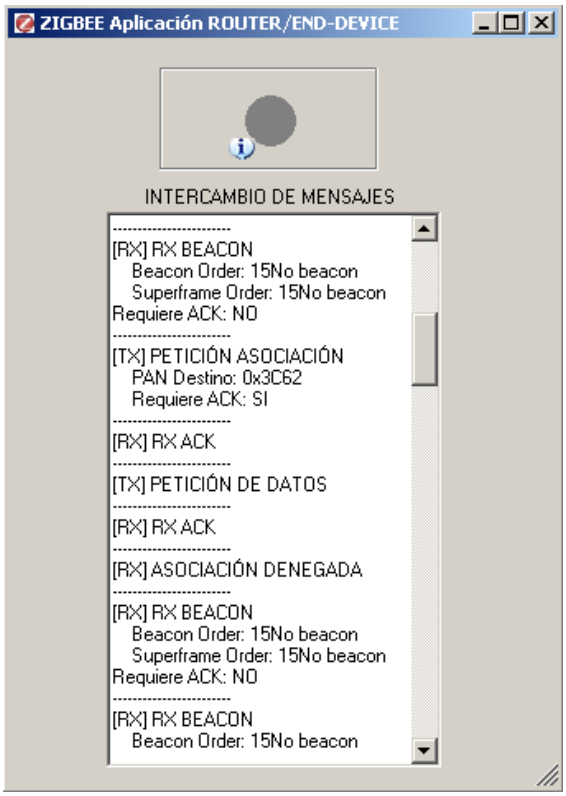


Figura 5.16: Escenario 5, captura del nuevo dispositivo.

Escenario 6: *Partiendo de la topología obtenida en el Escenario2 y con el coordinador aceptando asociaciones, se pierde comunicación con el router intermedio.*

El dispositivo final pierde la comunicación con su nodo padre, inmediatamente manda una notificación de orfandad a la que el coordinador responde acogiendo dicho dispositivo como nodo hijo.

Tramas capturadas por un *sniffer*:

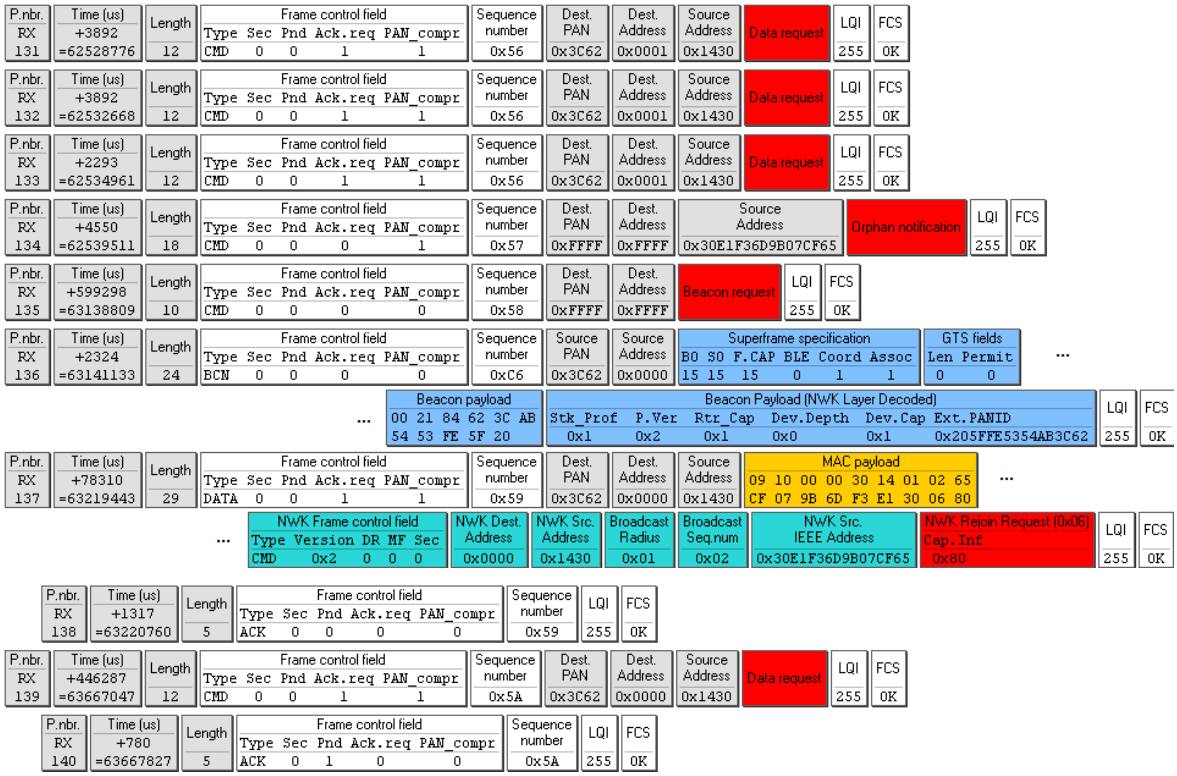


Figura 5.17: Escenario 6, captura del *sniffer*.

Se puede observar cómo inicialmente se realiza una petición de datos por parte del dispositivo *end-device* con destino el dispositivo con dirección 0x0001 (su nodo padre). Al no recibir tramas *ack,* detecta que su nodo padre no está operativo, por lo que emite una trama de notificación de orfandad (trama 134). El dispositivo con dirección 0x0000 (el coordinador) responde a dicha notificación (trama 136) con una baliza informando de que sí acepta asociaciones y el nodo huérfano se asocia a éste. Se observa cómo a partir de este cambio de nodo padre, las peticiones de datos por parte del dispositivo *end-device* tienen como nodo objetivo el direccionado por 0x0000 (ejemplo: trama 139).

Monitorización realizada por la plataforma:

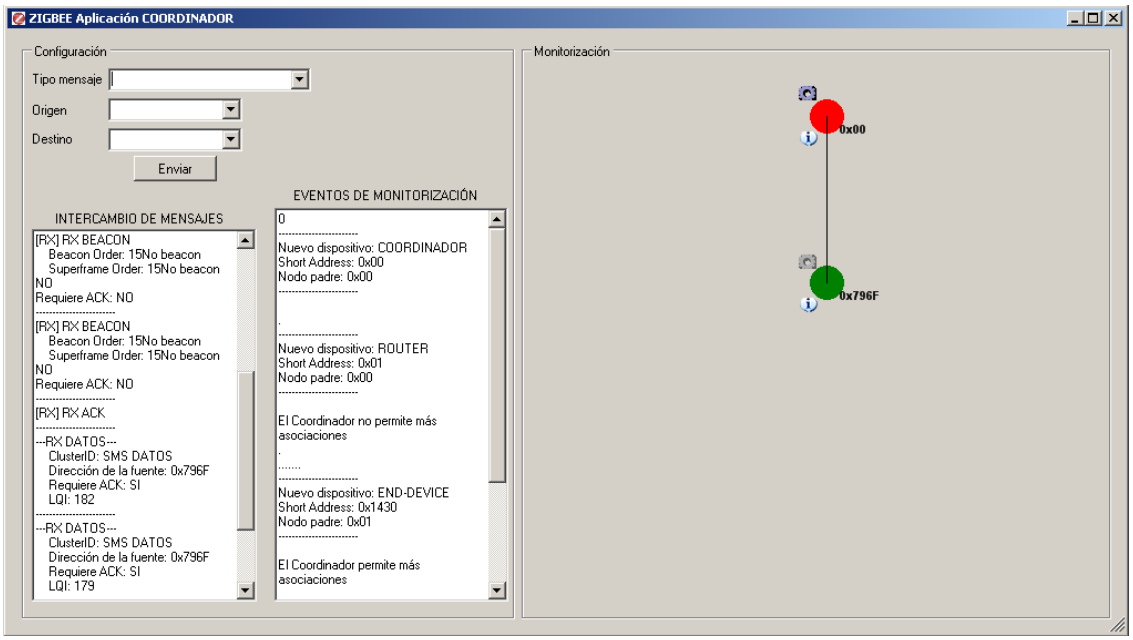


Figura 5.18: Escenario 6, captura de la plataforma del coordinador.

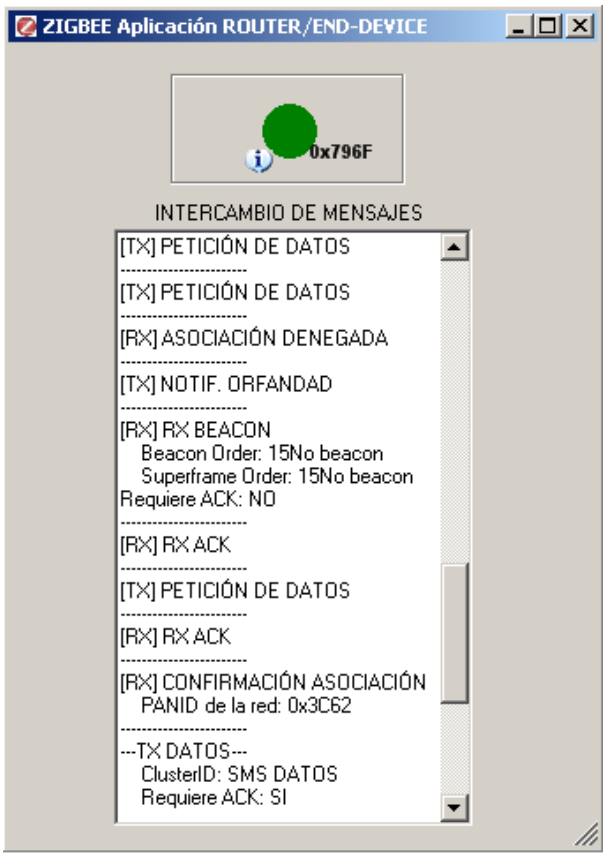


Figura 5.19: Escenario 6, captura de la plataforma del *end-device* huérfano.

En la figura 5.19 se muestra la transmisión por parte del dispositivo final de una notificación de orfandad y la consiguiente recepción de una confirmación de asociación cuando el coordinador se hace cargo de la paternidad del nodo.

5.3. Pruebas de mensajes

A continuación se muestran los resultados obtenidos en las pruebas de envío y recepción de mensajes a nivel de aplicación.

Mensaje de LED

Pruebas (Coordinador-*Router*-*End Device*)	**Resultado Test**
Visualización de transmisión de mensaje en plataforma nodo origen	**OK**
Visualización de recepción de trama *ack* en nodo origen	**OK**
Visualización de recepción de mensaje en plataforma nodo destino	**OK**
Encendido de LED en nodo destino	**OK**
Visualización en el LCD de recepción de mensaje de LED	**OK**
Encendido en el LCD de los iconos correspondientes	**OK**

Tabla 5.8: Pruebas en la transmisión y recepción de mensaje de LED.

Mensaje de *Buzzer*

Pruebas (Coordinador-*Router*-*End Device*)	**Resultado Test**
Visualización de transmisión de mensaje en plataforma nodo origen	**OK**
Visualización de recepción de trama *ack* en nodo origen	**OK**
Visualización de recepción de mensaje en plataforma nodo destino	**OK**
Encendido de *buzzer* en nodo destino	**OK**
Visualización en el LCD de recepción de mensaje de *buzzer*	**OK**
Encendido en el LCD de los iconos correspondientes	**OK**

Tabla 5.9: Pruebas en la transmisión y recepción de mensajes de *buzzer*.

Mensaje Periódico

Pruebas (Coordinador-*Router*-*End Device*)	**Resultado Test**
Visualización de transmisión de mensaje en plataforma nodo origen	**OK**
Visualización de recepción de trama *ack* en nodo origen	**OK**
Visualización de recepción de mensaje en plataforma nodo destino	**OK**
Comprobación de periodicidad configurable	**OK**
Visualización en el LCD de recepción de mensaje periódico	**OK**
Encendido en el LCD de los iconos correspondientes	**OK**

Tabla 5.10: Pruebas de transmisión y recepción de mensajes periódicos.

Mensajes de Texto

Pruebas (Coordinador-*Router-End Device*)	Resultado Test
Visualización de transmisión de mensaje en plataforma nodo origen	**OK**
Visualización de recepción de trama *ack* en nodo origen	**OK**
Visualización de recepción de mensaje en plataforma nodo destino	**OK**
Visualización en el LCD del mensaje de texto recibido	**OK**
Encendido en el LCD de los iconos correspondientes	**OK**

Tabla 5.11: Pruebas de transmisión y recepción de mensajes de texto.

5.4. Pruebas de capacidad

La plataforma principal está diseñada para monitorizar una red de hasta 10 dispositivos, compuesta por un coordinador y cualquier combinación de *routers* y dispositivos finales hasta un máximo de 9 dispositivos en total.

A continuación se testea el comportamiento de la plataforma a máxima capacidad. Debido a la falta de tal cantidad de dispositivos, se ha realizado una emulación vía software de una red de 10 dispositivos con la que poder demostrar el correcto funcionamiento de la monitorización de la plataforma. El resultado de dicha emulación se presenta a continuación:

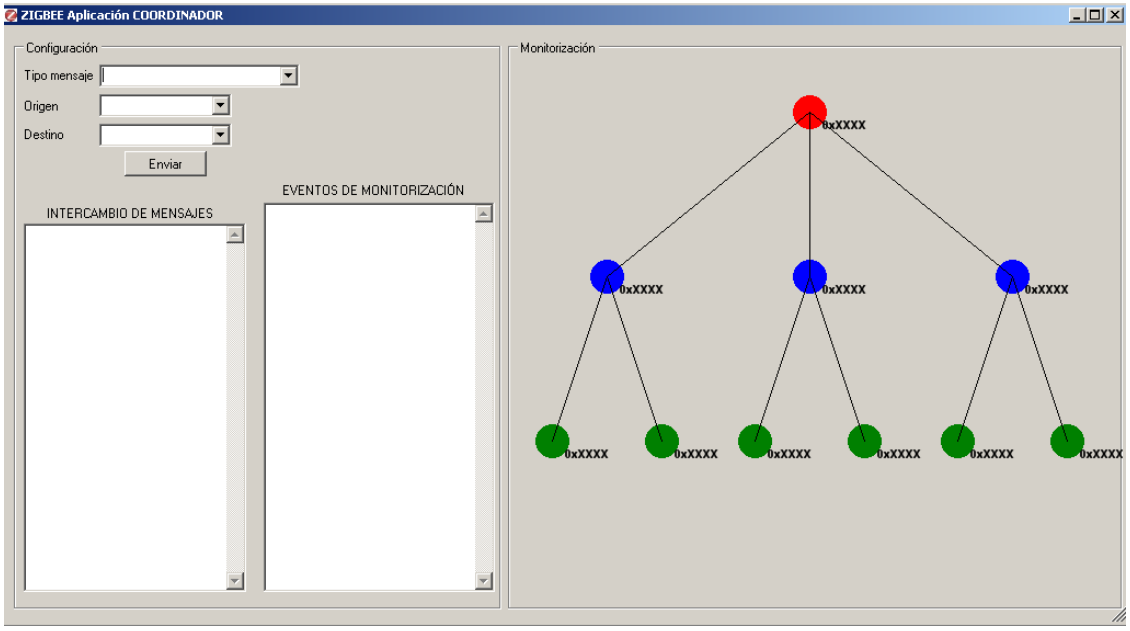


Figura 5.20: Captura plataforma a máxima capacidad.

CAPÍTULO 6

Conclusiones y líneas futuras de trabajo

En el presente capítulo se expondrán las conclusiones deducidas del desarrollo de este proyecto en cuanto a la tecnología ZigBee/802.15.4 se refieren. Además se mencionarán posibles líneas de investigación y estudio que ampliarían este mismo, de forma que otros alumnos puedan utilizar en beneficio propio.

6.1. Conclusiones

Tras la creación de la plataforma para la configuración y monitorización de redes ZigBee/802.15.4 a la que se refiere este proyecto y el consecuente estudio del comportamiento de dichas redes, se pueden destacar las siguientes ventajas sobre la utilización de la tecnología ZigBee/802.15.4:

- La principal característica de estos dispositivos es el bajísimo consumo, que aporta una gran autonomía de sus baterías y por tanto una vida útil bastante larga. Ello hace que esta tecnología sea idónea para su utilización en redes domóticas, inmóticas y control industrial, donde los accesos a energía eléctrica pueden ser muy limitados.

- Simplicidad en la creación de aplicaciones. Como se ha podido observar a lo largo de este proyecto, existen numerosas herramientas de apoyo que permiten el diseño de aplicaciones sin la necesidad de un conocimiento excesivo sobre programación de microcontroladores, si bien el entendimiento de la funcionalidad de cada una de las capas que componen el estándar ZigBee/802.15.4 es bastante más complejo.

- Bajo coste. Para desarrollar un dispositivo básico ZigBee/802.15.4 no se necesita más que un microcontrolador, un transceptor y una antena. Elementos que se pueden encontrar en el mercado a muy bajo precio, lo que hace que esta tecnología sea muy accesible para todo tipo de desarrolladores.

- Campo de aplicación. La domótica, inmótica y control industrial son campos de aplicación de muy reciente descubrimiento y en pleno auge, por lo que la existencia de una tecnología específicamente desarrollada para cubrir dichas funciones como es el caso de ZigBee/802.15.4 le proporciona una inmejorable situación para su evolución y desarrollo.

Por otro lado, se han detectado ciertos inconvenientes, los cuales se detallan a continuación:

- Como se acaba de mencionar, la principal característica de esta tecnología es su bajo consumo, sin embargo es realmente difícil encontrar una implementación en la que se desarrolle el modo de funcionamiento qué optimiza dicho consumo, el modo balizado. Esto se debe a que los principales fabricantes y desarrolladores de ZigBee/802.15.4 se han centrado en el modo no balizado priorizando la sencillez en el diseño y no la minimización del consumo de los dispositivos enrutadores. Existe en TI una alternativa para el desarrollo del modo balizado, es la utilización de la pila TIMAC (en lugar de la pila Z-Stack), la cual es un protocolo que se limita a definir la capa MAC y no implementa nivel de red, por lo que sólo permite topología en estrella, aunque sí da la opción de creación de aplicaciones en modo balizado. Aún así, su funcionamiento es muy reducido y sólo opera correctamente con dispositivos específicos.

- Debido al amplio mercado que se prevé pueda surgir en el ámbito domótico e inmótico, están desarrollándose otros estándares similares que amenazan la evolución de esta tecnología, ejemplos de ello son las tecnologías Z-Wave [44] y Bluetooth *Low Energy Technology* [36].

6.2. Líneas futuras de trabajo

ZigBee/802.15.4 es una tecnología relativamente reciente, lo que permite numerosas opciones de estudio. En cuanto a lo que se refiere a este proyecto, a continuación se describen posibles ampliaciones y alternativas al desarrollo aquí expuesto:

- Monitorización y Configuración de redes ZigBee/802.15.4 en modo balizado. Este proyecto se basa en el estudio de redes ZigBee/802.15.4 operando en modo no balizado. Una alternativa sería realizar dicho estudio en modo balizado, permitiendo observar este tipo de redes en un modo de funcionamiento que permite mucha mayor autonomía y un importantísimo ahorro energético.

- Estudio del consumo de dispositivos en redes ZigBee/802.15.4. Debido a que la principal característica de ZigBee/802.15.4 es su bajo consumo, es recomendable realizar un estudio detallado de dicho consumo tanto en modo balizado como en modo no balizado.

- Comparativa de tecnologías inalámbricas. En la introducción de este proyecto se ha realizado una pequeña comparativa de las tecnologías inalámbricas existentes actualmente en el mercado: ZigBee/802.15.4, Bluetooth, Wi-Fi y UWB. Se propone la realización de una comparativa detallada de dichas tecnologías con el fin de acotar los campos de aplicación de cada una de ellas.

- Estudio de interferencias entre ZigBee/802.15.4, Bluetooth y Wi-Fi. Estas tres tecnologías operan en la misma banda de frecuencias, 2.4 GHz, es por tanto muy interesante observar los posibles efectos interferentes que se producen al compartir una misma región de trabajo.

- Creación de una aplicación para funcionamiento en modo balizado. En este proyecto se han creado aplicaciones en modo no balizado. Se propone crear aplicaciones en modo balizado que permitan observar cómo se llevan a cabo los procesos de acceso al medio y la utilización de los diversos recursos que se ofrecen en una red balizada como los *slots* reservados.

ANEXO A

Manual de usuario

A.1. Instalación

Para poder utilizar la plataforma de configuración y monitorización, sólo se requiere copiar la carpeta *PFCZigBee* en un ordenador con puerto serie disponible (si el ordenador tiene varios puertos serie disponibles, se requiere conectar al puerto COM1 ya que la aplicación está configurada para trabajar sobre dicho puerto. En esta carpeta se encuentra el archivo ejecutable *PFCZigBee.exe* de la aplicación, además de las carpetas con el contenido necesario para el correcto funcionamiento de todos los elementos contenidos en el programa. Un doble *click* en el archivo ejecutable bastará para poner la aplicación en marcha.

Se ha comprobado el correcto funcionamiento de la aplicación en los siguientes sistemas operativo: Windows XP y Windows Vista.

A.2. Navegando por la plataforma

Una vez ejecutada la aplicación aparecerá la pantalla principal con el título de la plataforma y logotipo de ZigBee donde seleccionar el tipo de dispositivo que se va a conectar para su monitorización:

Figura A.1: Pantalla principal de la plataforma de monitorización y configuración.

- Pulse el botón **COORDINADOR** si el dispositivo conectado al puerto serie es el coordinador de la red.

- Pulse el botón **ROUTER/END-DEVICE** si el dispositivo conectado al puerto serie es un router o un dispositivo final.

IMPORTANTE: Asegúrese de tener realizadas las conexiones del puerto serie tanto en el ordenador como en el dispositivo antes de pulsar cualquiera de las dos opciones y de que el dispositivo no esté encendido.

Si el dispositivo conectado es un **coordinador**, al pulsar 

aparecerá la siguiente ventana de aplicación:

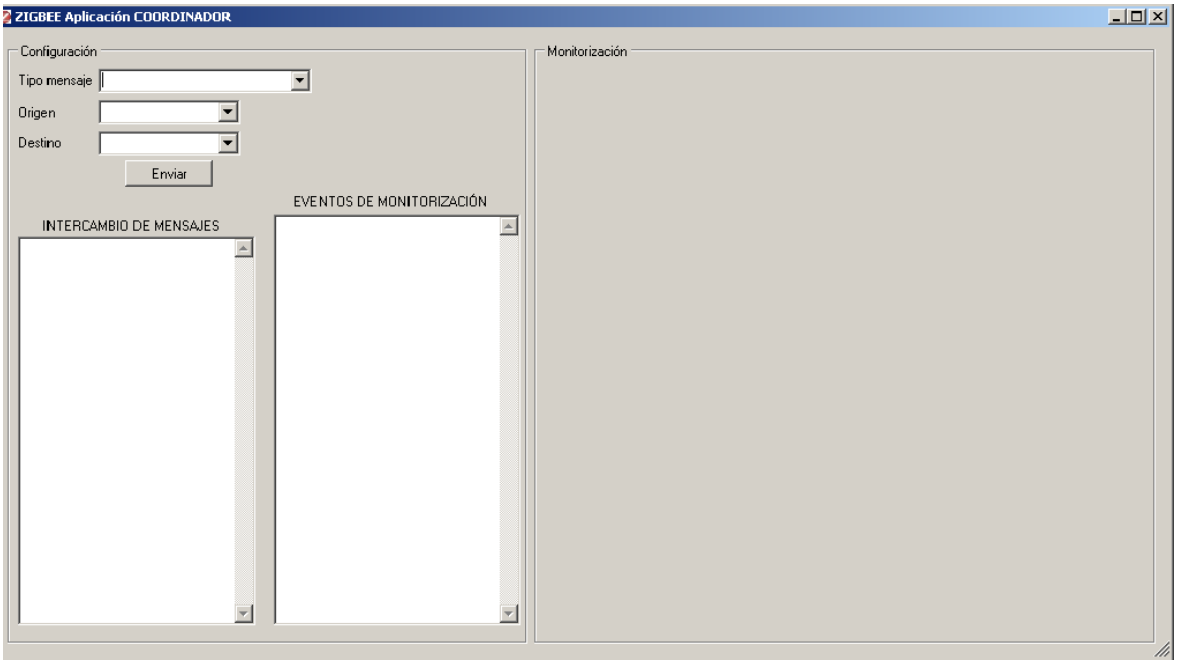


Figura A.2: Interfaz para dispositivo Coordinador.

Como se puede observar, la zona derecha del interfaz está reservada para la monitorización del sistema. A medida que se vayan activando y asociando a la red nuevos nodos, irán apareciendo en esta zona representaciones de cada dispositivo conectado, de diferentes colores según el tipo de dispositivo asociado (véase tabla A.1), junto con su dirección corta identificativa. Igualmente la disposición de los dispositivos en la pantalla mostrará la topología de red existente, tal y como se muestra en los ejemplos de la figura A.3.

Color	Tipo de dispositivo
ROJO	Coordinador
AZUL	*Router*
VERDE	*End-device*

Tabla A.1: Código de colores identificador del tipo de dispositivo.

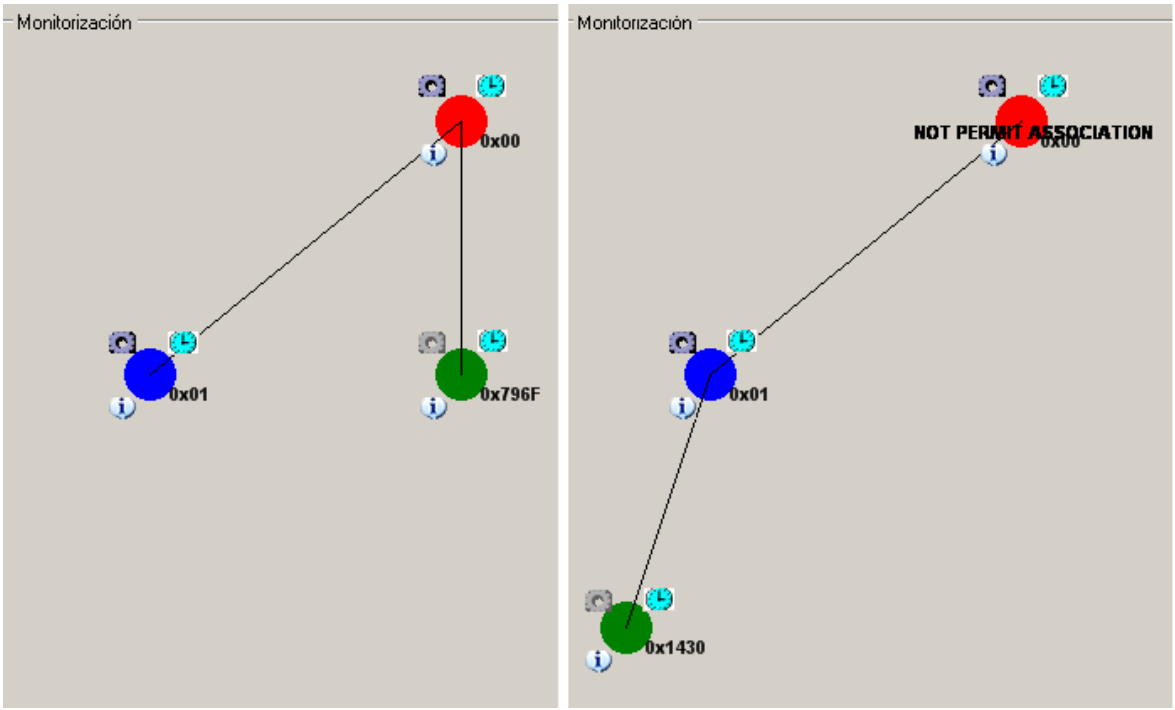


Figura A.3: Monitorización de diversas topologías de red.

Junto a la representación de cada dispositivo aparecen tres iconos:  ,  y  . Manteniendo el cursor sobre el icono  se obtiene información general del dispositivo y de la red a la que pertenece. Véase el ejemplo ilustrado en la figura A.4, donde se muestra:

- Tipo de dispositivo.
- Dirección corta (*Short address*).
- PANID.
- Canal lógico en el que se está operando.
- Modo de funcionamiento de la red (balizado o no balizado)
- Número máximo de hijos que el nodo soporta.
- Número máximo de routers que el nodo soporta.
- Profundidad máxima permitida.

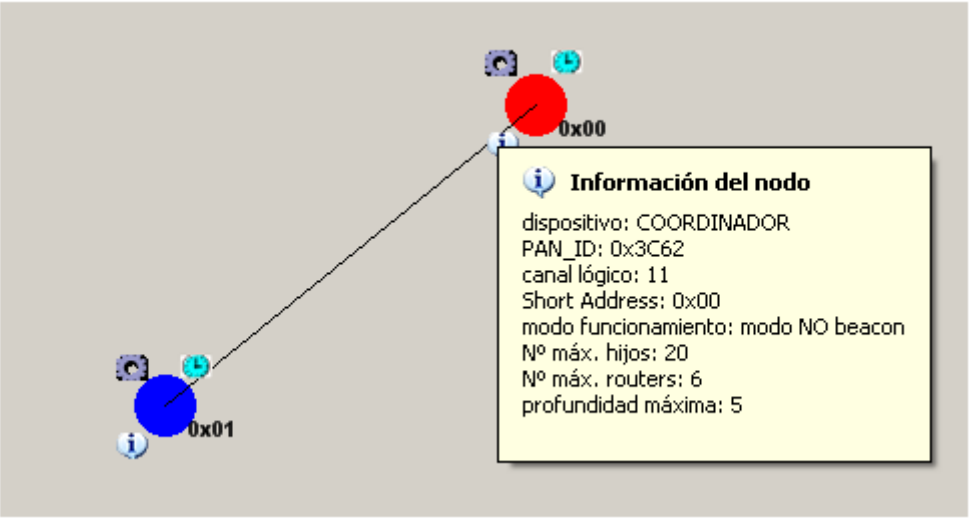


Figura A.4: Ventana emergente.

Pulsando el icono  (sólo activo si el nodo es un coordinador o un *router*, ya que no se va a configurar nada en el caso de los dispositivos finales, en cuyo caso el icono aparece sombreado) se tendrá acceso a un menú de configuración. Tal y como se representa en la figura A.5.

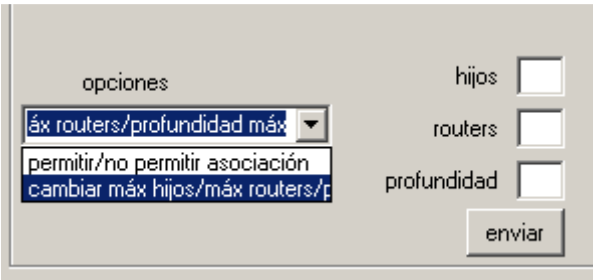


Figura A.5: Menú de opciones.

Por último, pulsando el icono  se accederá al menú de configuración de la periodicidad de los mensajes de actualización de la monitorización. La figura A.6 muestra dicho menú.

Figura A.6: Menú de configuración de actualización de la monitorización.

- Permitir/No permitir nuevas asociaciones al nodo que se está configurando. En el caso de no permitir nuevas asociaciones aparecerá un mensaje de texto atravesando la representación visual del nodo modificado, como se muestra en la figura siguiente.

Figura A.7: Dispositivo que no permite más asociaciones.

Además se mostrará en la ventana de EVENTOS DE MONITORIZACIÓN la situación actual cada vez que ésta se modifique.

- Configuración del número máximo de hijos, número máximo de *routers* y profundidad máxima permitida para el nodo en cuestión.

La modificación de todos estos parámetros permitirá crear la topología de red deseada además de poder observar el comportamiento de la red según sus posibles combinaciones.

En la parte superior izquierda de la interfaz se encuentra la zona de configuración de mensajes (ver figura A.8), desde aquí se podrá enviar órdenes para el envío de mensajes de encendido de LED, oscilación del *buzzer*, configuración de mensajes periódicos y mensajes de texto.

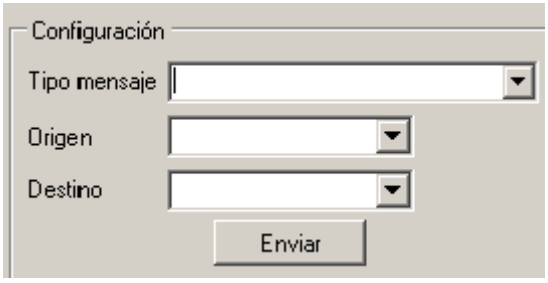


Figura A.8: Configuración de mensajes.

Pulsando en el primer desplegable se accede a la selección del tipo de mensaje (ver figura A.9), el segundo desplegable ofrece las posibles direcciones origen del mensaje y el tercer desplegable los posibles destinos. Estas direcciones permiten el envío de mensajes *unicast* o *broadcast* (figura A.10).

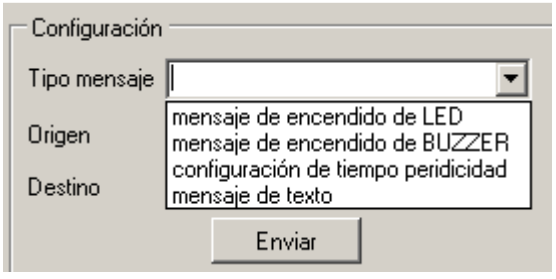


Figura A.9: Selección de tipo de mensaje

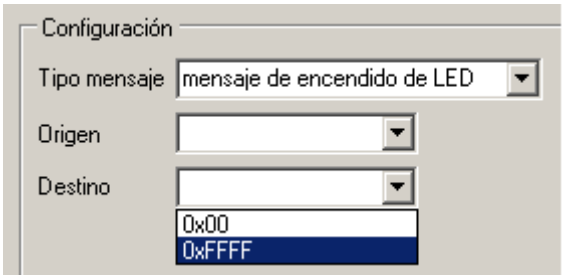


Figura A.10: Configuración direcciones origen y destino

En el caso de seleccionar mensaje periódico o mensaje de texto, se tendrá acceso a un campo donde seleccionar el tiempo de periodicidad en el primero o el mensaje de texto a enviar en el segundo (Figuras A.11 y A.12):

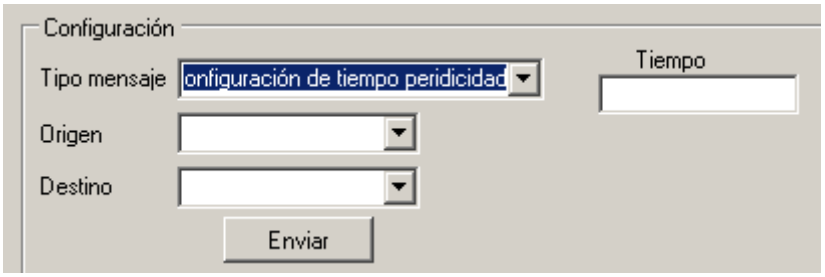


Figura A.11: Configuración mensaje periódico.

El valor introducido en el campo *Tiempo* se traducirá en el intervalo en segundos entre dos mensajes periódicos consecutivos. Un valor de 0 indicará al dispositivo direccionado en el campo *Origen* que no debe enviar mensajes periódicos.

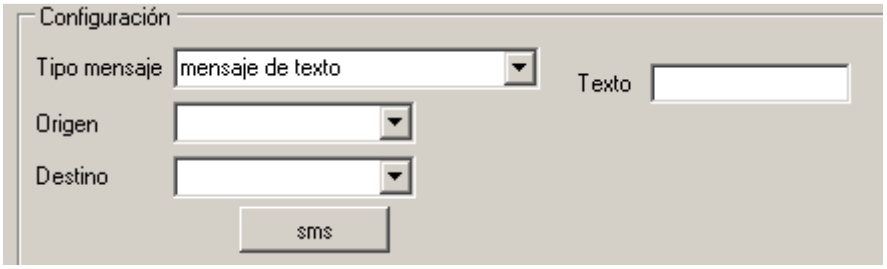


Figura A.12: Configuración mensaje de texto.

La plataforma no permite introducir en el campo *Texto* más de 6 caracteres, ya que es éste el tamaño máximo de visualización que permite el *display* del LCD.

Bajo esta zona de configuración se encuentra la zona de visualización de mensajes y eventos.

En la ventana de EVENTOS DE MONITORIZACIÓN de la interfaz general se muestran los acontecimientos que se producen en la red a nivel general, como la asociación de un nuevo dispositivo, la modificación en el permiso de asociación de alguno de los

nodos o el cambio de "padre" de un dispositivo debido a la pérdida de comunicación con su "padre" original. (Véase la figura A.13).

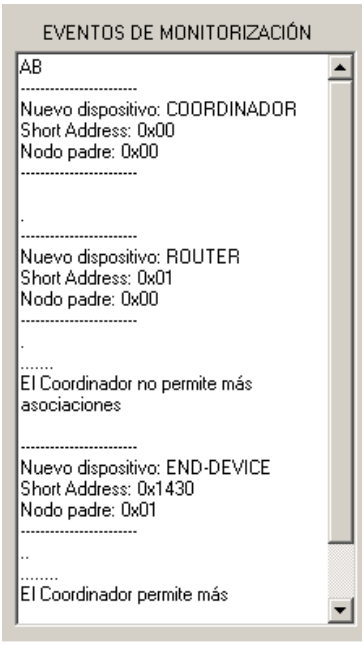


Figura A.13: Ventana de EVENTOS DE MONITORIZACIÓN.

En la ventana anexa, la ventana de INTERCAMBIO DE MENSAJES se muestran a tiempo real cada uno de los mensajes de comandos que intercambia el dispositivo conectado al puerto serie con los demás dispositivos pertenecientes a la red. Aquí se visualizarán los mensajes de comandos tanto entrantes como salientes al dispositivo, junto con la información más relevante que aporta dicho comando, como direcciones, permisos, condiciones, etc. En esta ventana también se muestra la transmisión y recepción de mensajes de datos con información sobre el tipo de datos, dirección de procedencia y la calidad del enlace (LQI) por donde se ha transmitido.

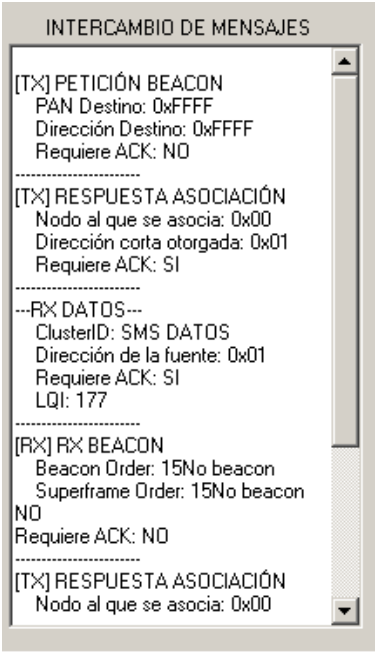


Figura A.14: Ventana de INTERCAMBIO DE MENSAJES.

Si el dispositivo conectado es un ***router*** o un **dispositivo final** se debe acceder desde la pantalla principal a la interfaz apropiada pulsando el botón ROUTER/END-DEVICE. Aparecerá entonces la ventana de aplicación para estos tipos de dispositivos (figura A.15). Una ventana mucho más simple que se debe utilizar conjuntamente con la monitorización de la aplicación del coordinador.

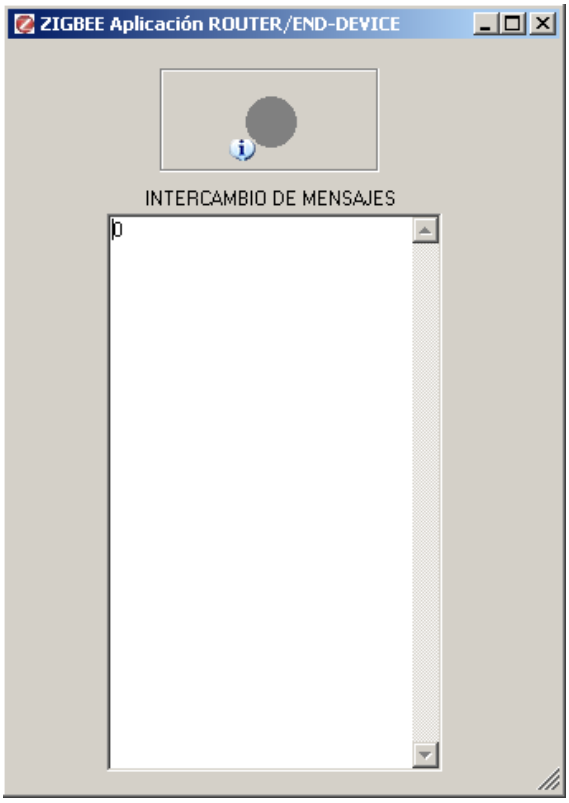


Figura A.15: Interfaz para dispositivo *router* y *end device*.

Esta interfaz consta de dos partes (tal y como se ilustra en la figura A.16), en la parte superior se muestra la representación del dispositivo conectado por puerto serie a la plataforma, junto a dicha representación se muestra la dirección corta identificativa del nodo en la red y el icono . Manteniendo el cursor sobre dicho icono aparecerá toda la información disponible sobre el nodo en cuestión y la red. En la parte inferior se encuentra la ventana de INTERCAMBIO DE MENSAJES donde se van mostrando en tiempo real los mensajes de comandos enviados y recibidos por el nodo conectado.

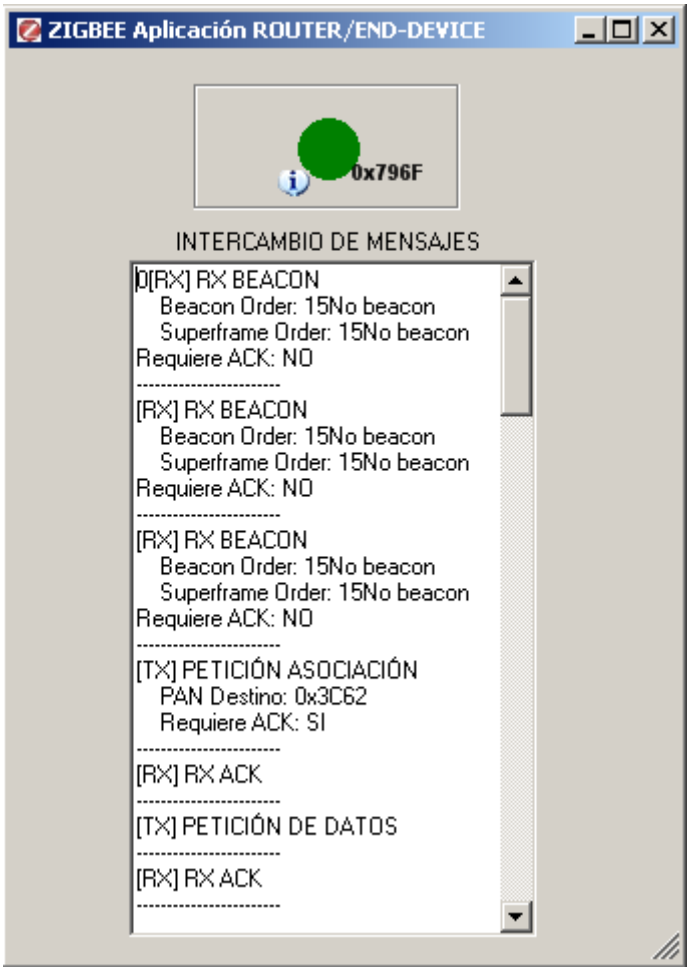


Figura A.16: captura de un instante de la aplicación con un dispositivo *end device* conectado.

A.3. La placa experimental.

A continuación se exponen los aspectos fundamentales de funcionamiento, configurados en este proyecto, para la placa experimental. También se detalla el significado de todos aquellos iconos que se pueden observar en el *display* LCD y que indican algún suceso o estado en el dispositivo.

Alimentación

Ya se mencionó en el capítulo 3 que la alimentación de un dispositivo puede configurarse para que sea local utilizando pilas AAA o bien externa a través del FET. La situación más común será siempre el uso de pilas ya que no todos los dispositivos estarán en todo momento cerca de una fuente de alimentación de la que alimentarse.

Por tanto el encendido y apagado de los dispositivos se realizará a través del *jumper* que controla el circuito de alimentación. Con el *jumper* conectado entre los terminales etiquetados con la palabra BATT se proporcionará alimentación y el dispositivo se encenderá y sin *jumper* entre dichos terminales el dispositivo permanecerá apagado. (Véase la figura A.17).

Figura A.17: *Jumper* de alimentación.

Los demás *jumpers* utilizados en la placa experimental tienen funciones de proporcionar alimentación a los distintos periféricos que se utilizan y de activar sus funcionalidades por lo que se recomienda leer detenidamente el capítulo 3 de este proyecto para entender la función de cada uno de ellos antes de decidir modificar la posición de alguno.

Botones

Se ha configurado el uso de los botones como forma alternativa a la plataforma para el envío de mensajes. Así la pulsación del botón etiquetado con S1 provoca un envío *broadcast* de un mensaje para la activación del *buzzer* y la pulsación del botón etiquetado con S2 provoca el envío *broadcast* de un mensaje de encendido de LED a todos los dispositivos de la red.

Figura A.18: Botones de la placa experimental.

LCD

En el LCD aparecen una serie de iconos y mensajes para indicar tanto el estado y configuración del dispositivo al cual monitoriza como la aparición de algún evento en la red en el que el dispositivo ha tenido algún papel.

Los posibles mensajes a aparecer en la zona de texto (ver figura A.19) del LCD se detallan en la siguiente tabla:

Mensaje	Texto LCD
Petición de *Beacon*	BEAC PT
Transmisión de *Beacon*	BEAC TX
Recepción de *Beacon*	BEAC RX
Petición de Asociación	ASOC PT
Respuesta de Asociación	ASOC RS
Notificación de Orfandad	ORF NOT
Mensaje de *Buzzer*	BUZZER
Mensaje de LED	LED
Mensaje periódico	PERIOD
Mensaje de texto	"texto recibido"

Tabla A.2: Mensajes en la zona de texto del LCD.

Los iconos representados en la pantalla del LCD (ilustrados en la figura A.18) y su significado se explican a continuación:

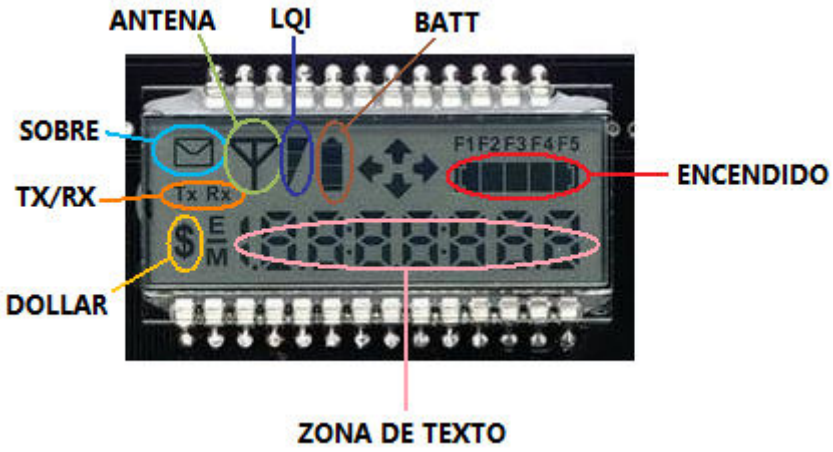


Figura A.19: LCD de la placa experimental.

- **DOLLAR:** Indica que el dispositivo está configurado como coordinador de la red.
- **TX/RX:** Se activará el símbolo TX o RX según el dispositivo transmita o reciba un mensaje vía OTA.
- **SOBRE:** Utilizado para indicar la recepción de un mensaje.
- **ANTENA:** Se encenderá cuando el dispositivo esté conectado a una red.

- **LQI:** Indica la calidad con la que se ha recibido el último mensaje.

- **BATT:** Muestra el estado actual de las baterías con las que se está proporcionando alimentación.

- **ENCENDIDO:** Este símbolo se activa cuando un dispositivo se enciende. De esta forma se puede observar si un dispositivo está encendido pero no puede acceder a la red por algún motivo.

- **ZONA DE TEXTO:** En esta zona del LCD se visualizan los mensajes descritos en la tabla A.2.

Referencias

[1] L. Jin-Shyan, S. Yu-Wei, S. Chung-Chou, "A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi" *The 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON)*, Taipei (Taiwan), Noviembre, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://w3.nctu.edu.tw/users/u8812812/WWW/jslee_ieeeIECON07.pdf*

[2] Memsen Corporation, "A Technology Comparison Adopting Ultra-Wideband for Memsen´s file sharing and wireless marketing platform". Documento en formato pdf accesible por Internet en la dirección: *http://wireless.fcc.gov/outreach/2004broadbandforum/comments/ultrawideband.pdf*

[3] "TI Low Power Wireless, ZigBee Technical Overview", Texas Instruments, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ml/swrp086a/swrp086a.pdf*

[4] Sinem Coleri Ergen, "ZigBee/IEEE 802.15.4 Sumary", Septiembre, 2004. Documento en formato pdf accesible por Internet en la dirección: *www.sinemergen.com/zigbee.pdf*

[5] IEEE Computer Society, "Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specification for Low-Rate Wireless Personal Area Networks (LR-WPANs)", Septiembre, 2006. Documento en formato pdf accesible en la dirección: *http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf*

[6] A. Koubâa, A. Cunha, M.Alves, "A Time Division Beacon Scheduling Mechanism for IEEE 802.15.4/ZigBee Cluster-Tree Wireless Sensor Networks". Documento en formato pdf accesible por Internet en la dirección: *http://www.dei.isep.ipp.pt/~akoubaa/publications/ECRTS2007.pdf*

[7] A. Cuevas, R. Cuevas, M. Urueña, D. Larrabeiti, "A Proposal for ZigBee Clusters Interconnection based on ZigBee Extension Devices". Documento en formato pdf accesible por Internet en la dirección: *http://www.it.uc3m.es/~acuevasr/publicaciones/WSAN07.pdf*

[8] J. Martín Moreno, D. Ruiz Fernández, "Informe Técnico: Protocolo ZigBee (IEEE 802.15.4), Junio, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://rua.ua.es/dspace/bitstream/10045/1109/7/Informe_ZigBee.pdf*

[9] J.C. Cano, J.M. Cano, E. González, C. Calafate, P. Manzoni, "Evaluation of the Energetic Impact of Bluetooth Low-Power Modes for Ubiquitous Computing Applications", 3rd ACM* International Workshop on Performance Evaluation of Wireless Ad-Hoc, Sensor and Ubiquitous Networks, Octubre, 2006.

[10] J. Puiggros, A. Müller, A. González, "Redes de Sensores Inalámbricas. Protocolo 802.15.4 y ZigBee". Sociedad Agrícola Ojos Buenos Ltda, Abril, 2005. Documento en formato ppt accesible por Internet en la dirección: *http://alumnos.elo.utfsm.cl/.../05-04-18v.1-RSI-802-15-4.**ZigBee**.ppt*

[11] A. Koubaa, M. Alves, E. Tovar, "IEEE 802.15.4 for Wireless Sensor Networks: A Technical Overview", TR-050702, *Technical Report*, Julio, 2005. Documento en formato pdf accesible por Internet en la dirección: *http://www.hurray.isep.ipp.pt/asp/*

[12] ZigBee Alliance, "ZigBee Specification", Enero 2008.

[13] "Z-Stack User´s Guide For MSP430 Experimenter´s Board",F8W-2007-0009, versión 1.4.3, Texas Instruments, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[14] "Z-Stack Sample Application For ATC4618", F8W-2007-0015, versión 1.0, Texas Instruments, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[15] "Z-Stack Compile Options",F8W-2007-0018, versión 1.0, Texas Instruments, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[16] "Application Notes: Create New Application For The EXP4618",F8W-2007-0019, versión 1.1, Texas Instruments, 2007. Documentos en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[17] "802.15.4 MAC Application Programming Interface", F8W-2005-1503, versión 1.1, Texas Instruments, Marzo, 2007. Documento en formato pdf accesible por Internet en la dirección *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[18] "HAL Drivers Application Programming Interface", F8W-2005-1504, versión 1.2, Texas Instruments, 2005. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[19] "Z-Stack OS Abstraction Layer Application Programming Interface", F8W-2003-0002, versión 1.5, Texas Instruments, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[20] "Z-Stack/Z-Tool Serial Port Interface", F8W-2003-0001, versión 1.11, Texas Instruments, 2006. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[21] "Z-Stack Application Programming Interface", F8W-2006-0021, versión 1.2, Texas Instruments, Abril, 2007. Documento en formato pdf accesible por Internet en la dirección*:* *http://focus .ti.com/docs/toolsw/folders/print/z-stack.html*

[22] "Z-Stack Developer´s Guide", F8W-2006-0022, versión 1.1, Texas Instruments, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[23] "IAR Embedded Workbench for TI MSP430", *http://iar.com/website1/1.0.1.0/220/1/*

[24] "MSP430x4xx Family User´s Guide", SLAU056I, Texas Instruments, Septiembre, 2009. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ug/slau056i/slau056i.pdf*

[25] "MSP430FG4618/F2213 Experimenter´s Board User´s Guide", SLAU213A, Texas Instruments, Octubre, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ug/slau213a/slau213a.pdf*

[26] "MSP430xG461x Mixed Signal Microcontroller", SLAS508H, Texas Instruments, Abril, 2006, revisado en Diciembre, 2009. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ds/symlink/msp430fg4618.pdf*

[27] "CC2520 Datasheet", SWRS068, Texas Instruments, Diciembre, 2007. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ds/symlink/cc2520.pdf*

[28] "CC2520-CC2591EMK Quick Start Guide", SWRU172A, Texas Instruments, Agosto, 2008. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ml/swru172a/swru172a.pdf*

[29] "SmartRF05 Evaluation Board User´s Guide", SWRU210, Texas Instruments, Marzo, 2009. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ug/swru210/swru210.pdf*

[30] "SmartRF™ Packet Sniffer User Manual Rev. 1.9", SWRU187, Texas Instruments, Febrero, 2009. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ug/swru187a/swru187a.*pdf

[31] "MSP430 IAR EMBEDDED WORKBENCH™ Tutorials", IAR Systems, Junio, 2000. Documento en formato pdf accesible por Internet en la dirección: *http://www.eecs.berkeley.edu/boser/courses/40/labs/docs/IAR%20tutorial.pdf*

[32] "2.4 GHz IEEE 802.15.4/ZigBee-ready RF Transceiver" CC2420 Data Sheet. SWRS041B, Texas Instruments. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ds/symlink/cc2420.pdf*

[33] "CC2420 MSP430 ZigBee Development Kit Quick Start Guide", SWRU118, Texas Instruments. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ug/swru118/swru118.pdf*

[34] "Quick Start Instructions CC2420DK Development Kit", SWRU044, Texas Instruments. Documento en formato pdf accesible por Internet en la dirección: *http://focus.ti.com/lit/ug/swru044/swru044.pdf*

[35] URL de Bluetooth SIG. *http://www.bluetooth.com*

[36] URL de Bluetooth Low *Energy Technology*. *http://www.bluetooth.com/Bluetooth/Products/Low_Energy.htm*

[37] URL de Texas Instruments: *http://www.ti.com/*

[38] URL de Texas Instruments sobre TIMAC: *http://focus.ti.com/docs/toolsw/folders/print/timac.html*

[39] URL de descarga de la aplicación Xvi32: *http://www.chmaas.handshake.de/delphi/freeware/xvi32/xvi32.htm*

[40] URL de descarga de *Microsoft Visual Basic 2008 Express Edition*: *http://www.microsoft.com/express/Downloads/#2008-Visual-Basic*

[41] E. Garay Gaitán "Visual Basic 2008", Nicaragua. Documento en formato pdf accesible por Internet en la dirección: *http://www.vacationinnicaragua.com/microsoft/visual-basic/visual-basic-2008.pdf*

[42] URL de tutorial online para Microsoft Visual Basic 2008 Express Edition: *http://msdn.microsoft.com/es-es/library/aa187916.aspx*

[43] URL de Texas Instruments sobre Z-Stack: *http://focus.ti.com/docs/toolsw/folders/print/z-stack.html*

[44] URL de Z-Wave Alliance: *http://www.z-wavealliance.org*