

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA

**ESCUELA DE INGENIERÍA DE
TELECOMUNICACIÓN Y ELECTRÓNICA**



PROYECTO FIN DE CARRERA

**IMPLEMENTACIÓN DE UN SENSOR INALÁMBRICO DE
PROPÓSITO GENERAL CON BAJO CONSUMO**

Titulación: **Sistemas Electrónicos**
Tutores : **Dr. D. Roberto Esper-Chaín Falcón**
Dr. D. Félix Bernardo Tobajas Guerrero
Autor : **Carlos M. Fernández Reyes**
Fecha : **Mayo 2010**

UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA

**ESCUELA DE INGENIERÍA DE
TELECOMUNICACIÓN Y ELECTRÓNICA**



PROYECTO FIN DE CARRERA

**IMPLEMENTACIÓN DE UN SENSOR INALÁMBRICO DE
PROPÓSITO GENERAL CON BAJO CONSUMO**

Presidente:

Secretario:

Vocal:

Tutores:

Autor:

NOTA:

Titulación: **Sistemas Electrónicos**

Tutores : **Dr. D. Roberto Esper-Chaín Falcón**
Dr. D. Félix Bernardo Tobajas Guerrero

Autor : **Carlos M. Fernández Reyes**

Fecha : **Mayo 2010**

A mi madre

Índice general

1. Introducción	25
1.1 Introducción	25
1.2 Objetivos	27
1.3 Peticionario	29
1.4 Estructura de la memoria	29
2. Tecnología inalámbrica	31
2.1 Introducción	31
2.2 Un poco de historia	32
2.3 Redes inalámbricas	33
2.3.1 Categorías	34
2.3.2 Aplicaciones	36
2.3.3 Estándares de comunicaciones inalámbricas	37
2.4 Redes de sensores inalámbricas	40
2.4.1 Definición	40
2.4.2 Historia	41
2.4.3 Aplicaciones	42
2.4.4 Características	44
2.4.5 Estándares y Tecnologías	45
3. ZigBee	49
3.1 ¿Qué es ZigBee?	49
3.2 Características	50
3.3 Comparación con otros estándares	51
3.4 Tipos de dispositivos	53
3.5 Topologías de red	55
3.5.1 Topología en estrella	56
3.5.2 Topología en árbol	56
3.5.3 Topología en malla (<i>mesh</i>)	57
3.6 Arquitectura	58
3.7 IEEE 802.15.4	60
3.7.1 Capa física	60
3.7.1.1 Frecuencia de operación y velocidad de transmisión	61
3.7.1.2 Modulación	63
3.7.1.3 Técnica de ensanchado de espectro DSSS	65
3.7.2 Capa MAC	67
3.7.2.1 Estructura de la trama MAC	68
3.7.2.2 Acceso al Canal	72
3.7.2.3 Estructura de la <i>Superframe</i>	73
3.7.2.4 Algoritmo CSMA-CA	74
4. Descripción de los Componentes	77
4.1 El procesador ZigBee Ember EM260	78
4.1.1 Descripción general	78
4.1.2 Características generales	81
4.1.3 Características de radiofrecuencia del EM260	82
4.1.4 Descripción de los pines del EM260	83

4.1.5	Diagrama funcional del EM260	85
4.1.6	Protocolo SPI del EM260	86
4.1.6.1	Temporización	89
4.1.6.2	Formato de las tramas	90
4.1.7	EZSP (<i>EmberZNet Serial Protocol</i>)	92
4.1.7.1	Estructura de las tramas EZSP	92
4.1.7.2	Configuración de la pila ZigBee	94
4.1.7.3	Definición de tipos de variable	96
4.1.7.4	Comandos EZSP	98
4.2	El microcontrolador Atmel ATmega168	102
4.2.1	Características generales	103
4.2.2	Descripción de los pines del ATmega168	105
4.2.3	Arquitectura AVR	107
4.2.4	SPI (<i>Serial Peripheral Interface</i>)	109
4.2.4.1	Registros de la interfaz SPI	112
5.	Diseño hardware del nodo sensor inalámbrico	115
5.1	Descripción general	115
5.2	Bloque de alimentación	121
5.2.1	Alimentación externa	122
5.2.1.1	Regulador 3.3V	123
5.2.2	Alimentación con pila	125
5.2.3	Desacoplo de masa y alimentación	126
5.2.3.1	Desacoplo de alimentación	127
5.2.3.2	Desacoplo entre masas	129
5.2.3.3	Desacoplo de AVCC y VCC3V3	130
5.3	Bloque de radiofrecuencia	131
5.4	Bloque del dispositivo EM260	136
5.4.1	Regulación del dispositivo EM260	136
5.4.2	Reloj del dispositivo EM260	138
5.4.3	Conector de programación del EM260	140
5.4.4	Indicador de actividad en TX/RX y señal TX_ACTIVE	142
5.4.5	Conexión SPI	143
5.4.5.1	nHOST_INT y nWAKE	144
5.5	Bloque del dispositivo ATmega168	145
5.5.1	Reloj del ATmega168	145
5.5.2	Conector de programación del ATmega168	147
5.5.3	Circuito de reset	149
5.6	Conector de expansión	151
5.7	Bloque Sensor	153
5.7.1	Sensor de temperatura	154
5.7.2	Sensor de presión	157
5.7.3	Sensor de movimiento (PIR)	161
5.7.4	Diodos LEDs, interruptor y reset	165
6.	Diseño y montaje de las placas de circuito impreso	169
6.1	Descripción general	170
6.2	Captura del esquemático	171
6.2.1	Librería de símbolos	171
6.2.2	Librería de <i>footprints</i>	174
6.2.3	Enlace entre librerías	178
6.3	Características tecnológicas de las PCBs	182
6.3.1	Módulo de comunicaciones inalámbricas	182
6.3.2	Módulo sensor	185

6.4	Diseño de las PCBs	186
6.4.1	Módulo de comunicaciones inalámbricas	186
6.4.1.1	Reglas de diseño	187
6.4.1.2	Consideraciones especiales sobre la antena	188
6.4.1.3	Distribución de componentes por zonas	189
6.4.1.4	Ruteado de las pistas	191
6.4.1.5	Distribución del plano de masa	193
6.4.1.6	Distribución del plano de alimentación	195
6.4.1.7	Diseño final	198
6.4.2	Módulo sensor	200
6.4.2.1	Reglas de diseño	201
6.4.2.2	Distribución de componentes	201
6.4.2.3	Ruteado de las pistas	203
6.4.2.4	Diseño final	204
6.5	Montaje de las PCBs	206
7.	Desarrollo del firmware	209
7.1	Descripción general	209
7.2	Programación de los dispositivos	210
7.3	Medida del consumo de corriente	216
7.4	Test de alcance	219
7.4.1	Escenario A	221
7.4.2	Escenario B	222
7.4.3	Escenario C	223
8.	Conclusiones y líneas futuras	227
8.1	Conclusiones	227
8.2	Líneas Futuras	229
	Bibliografía	231
	Presupuesto	237
	Baremos utilizados	237
	Cálculo del presupuesto	238
P.1.	Coste de debidos a los recursos humanos	238
P.2.	Coste de amortización de los recursos hardware	239
P.3.	Coste de amortización de los recursos software	240
P.4.	Coste de fabricación de prototipos	241
P.5.	Coste de la redacción del proyecto	241
P.6.	Coste por derechos de visado	242
P.7.	Coste de tramitación y envío	243
P.8.	Gastos generales	243
P.9.	Presupuesto total del proyecto	244
	Anexo A. Ficheros Gerber	247
	Anexo B. Listado de Componente	251
	Anexo C. Guía de fabricación	255
	Anexo D. Código fuente	281
	Anexo F. Contenido del CD del PFC	313

Índice de figuras

Figura 1.1. Ejemplo de red de sensores inalámbrica con topología en malla	26
Figura 1.2. Diagrama de bloques simplificado del sistema completo.....	28
Figura 2.1. Relación entre área de cobertura y tecnologías.....	36
Figura 3.1. Logotipo de ZigBee Alliance.....	50
Figura 3.2. Estructura de red ZigBee	55
Figura 3.3. Topología de red en estrella.....	56
Figura 3.4. Topología de red en árbol	57
Figura 3.5. Topología de red en malla	58
Figura 3.6. Capas de la arquitectura ZigBee	58
Figura 3.7. Canales disponibles en la capa física del 802.15.4.1	62
Figura 3.8. Modulación PSK.....	63
Figura 3.9. Constelación de la modulación BPSK	63
Figura 3.10. Constelación de la modulación QPSK (2 bits por símbolo)	64
Figura 3.11. Codificación de Barker	66
Figura 3.12. Efecto en el espectro al desensanchar.....	67
Figura 3.13. Trama de datos de la subcapa MAC	69
Figura 3.14. Trama <i>acknowledgment</i> de la subcapa MAC.....	70
Figura 3.15 Trama de comando de la subcapa MAC	70
Figura 3.16. Trama beacon de la subcapa MAC	71
Figura 3.17. Trama PPDU.....	72
Figura 3.18. Estructura de una <i>Superframe</i>	73
Figura 3.19. Ventana de contención y tiempo de espera.....	75
Figura 3.20. Diagrama de flujo del algoritmo CSMA-CA.....	76
Figura 4.1. Arquitectura formada por la conexión del EM260 y un microcontrolador.....	79
Figura 4.2. Procesador ZigBee EM260	80
Figura 4.3. Esquemático de aplicación típica del EM260	81
Figura 4.4. Pines del dispositivo EM260	83
Figura 4.5. Diagrama de bloques simplificado del EM260.....	85
Figura 4.6. Cronograma de transferencia SPI en Modo 0 y Modo 3.....	87
Figura 4.7. Diagrama de una transacción SPI básica	87
Figura 4.8. Secuencia de activación del dispositivo EM260.....	89
Figura 4.9. Cronograma de tiempos críticos del protocolo SPI del EM260.....	89

Figura 4.10. Formato de la trama SPI	90
Figura 4.11. Cronograma del comando <i>Callback</i>	92
Figura 4.12. Estructura de la trama EZSP	93
Figura 4.13. Diagrama de bloques del dispositivo ATmega168	103
Figura 4.14. Microcontrolador ATmega168	105
Figura 4.15. Pines del dispositivo ATmega168.....	105
Figura 4.16. Arquitectura AVR.....	109
Figura 4.17. Interconexión lógica del interfaz SPI.....	110
Figura 4.18. Interconexión mediante SPI del los dispositivos ATmega168 y EM260.....	111
Figura 4.19. Organización del registro SPCR	112
Figura 4.20. Organización del registro SPSR.....	113
Figura 4.21. Organización del registro SPDR	114
Figura 5.1. Diagrama de bloques funcionales del módulo	116
Figura 5.2. Esquemático completo del módulo de comunicaciones inalámbricas desarrollado.....	118
Figura 5.3. Esquemático del bloque de alimentación.....	122
Figura 5.4. Asignación de los pines del TPS77633D.....	123
Figura 5.5. Esquema del circuito regulador de tensión	124
Figura 5.6. Pila de litio modelo CR2032.....	126
Figura 5.7. Condensadores de desacoplo del módulo	128
Figura 5.8. Desacoplo entre las distintas masas del módulo	129
Figura 5.9. Desacoplo entre AVCC y VCC3V3.....	130
Figura 5.10. Desacoplo del pin AREF del ATmega168.....	130
Figura 5.11. Esquema del bloque de radiofrecuencia.....	131
Figura 5.12. Asignación de pines del balún HHM-1521	132
Figura 5.13. Esquema del circuito balún.....	133
Figura 5.14. Esquema del filtro de armónicos.....	133
Figura 5.15. Esquema del circuito del conector externo y la antena	134
Figura 5.16. Asignación de pines de la antena integrada	135
Figura 5.17. Esquema de las diferentes alimentaciones del EM260	137
Figura 5.18. Esquema de filtrado entre VCC1V8_D y VCC1V8	137
Figura 5.19. Esquema de filtrado entre VCC1V8 y 1.8V _VCO	138
Figura 5.20. Esquema de filtrado entre VCC1V8 y VCC1V8_RF	138
Figura 5.21. Diagrama de pines del ABM3B-24.000MHZ.....	139
Figura 5.22. Esquema del circuito oscilador del dispositivo EM260.....	139
Figura 5.23. Conexión del conector de programación del EM260.....	140
Figura 5.24. Dispositivo de programación InSight USB Link	141

Figura 5.25. Esquema de pull-down y pull-up añadidos a la interfaz SIF	142
Figura 5.26. Indicador de actividad del circuito de radiofrecuencia	142
Figura 5.27. Esquema del acondicionamiento de TX_ACTIVE	143
Figura 5.28. Conexión del EM260 y el ATmega168 mediante SPI	144
Figura 5.29. Conexión de las señales nWAKE y nHOST_INT	144
Figura 5.30. Diagrama de pines del ABMM-8.000MHZ.....	146
Figura 5.31. Esquema del circuito oscilador del ATmega168	146
Figura 5.32. Capacidad de carga del circuito oscilador del ATmega168.....	147
Figura 5.33. Conexión del conector de programación del ATmega168.....	147
Figura 5.34. Dispositivo de programación <i>AVRISP mkII In-system Programmer</i>	148
Figura 5.35. Esquema del conector para el protocolo EZSP.....	149
Figura 5.36. Esquema del circuito de reset del ATmega168.....	150
Figura 5.37. Esquema del circuito de reset del EM260.....	150
Figura 5.38. Esquema del conector de expansión	151
Figura 5.39. Esquema de los pull-up de PD2 y PD3	152
Figura 5.40. Esquema del bloque sensor.....	153
Figura 5.41. Pinout de sensor de temperatura TMP121	155
Figura 5.42. Esquema del sensor de temperatura TMP121 en el bloque sensor	156
Figura 5.43. Diagrama de bloques del sensor de presión MP3H6115A6U.....	158
Figura 5.44. Función de transferencia del MP3H6115A.....	158
Figura 5.45. Banda de error de temperatura.....	159
Figura 5.46. Relación entre la tensión de salida y presión en el MP3H6115A.....	159
Figura 5.47. Esquema del sensor de presión MP3H6115A en el bloque sensor	160
Figura 5.48. Representación del sensor de movimiento PIR (vista inferior)	162
Figura 5.49. Asignación de los pines del convertidor REG710-5/250.....	163
Figura 5.50. Esquema del circuito de acondicionamiento para el sensor de movimiento PIR	164
Figura 5.51. Fotografía del sensor de movimiento PIR	165
Figura 5.52. Esquema de los diodos LEDs del bloque sensor	166
Figura 5.53. Esquema del interruptor del bloque sensor.....	166
Figura 5.54. Esquema del pulsador de reset del bloque sensor	167
Figura 6.1. Proyecto de diseño de una PCB en el DXP 2004	170
Figura 6.2. Resistencia de la librería estándar de la herramienta DXP 2004	172
Figura 6.3. Condensador de la librería estándar de la herramienta DXP 2004	172
Figura 6.4. Símbolo del dispositivo EM260.....	173
Figura 6.5. Símbolo del microcontrolador ATmega168	174
Figura 6.6. Descripción de medidas del encapsulado QFN40 (EM260).....	175

Figura 6.7. <i>Footprint</i> C1005-0402 de la librería <i>miscellaneous devices</i>	176
Figura 6.8. Dimensiones del encapsulado del conector Molex 54722-0168-C	177
Figura 6.9. <i>Footprint</i> de la antena ANT-2.45-CHP de Linx Technologies Inc.	177
Figura 6.10. <i>Footprint</i> de micro-conector coaxial MM8430-2600 de Murata	178
Figura 6.11. Enlace entre un <i>footprint</i> y un símbolo	179
Figura 6.12. Esquemático completo del módulo de comunicaciones inalámbrico	179
Figura 6.13. Esquemático completo del módulo sensor	181
Figura 6.14. Características físicas y distribución de capas de la PCB del módulo de comunicaciones inalámbrico	183
Figura 6.15. Ancho mínimo de pista	183
Figura 6.16. Separación mínima entre pistas	184
Figura 6.17. Diámetro mínimo de vias	184
Figura 6.18. Ancho de pistas <i>microstrip</i>	185
Figura 6.19. Características físicas y distribución de capas de la PCB del módulo sensor	185
Figura 6.20. Antena empleada en el PFC	188
Figura 6.21. Ubicación de la antena y los planos de masa	189
Figura 6.22. Distribución del diseño en la PCB del módulo de comunicaciones inalámbricas	190
Figura 6.23. Diseño de <i>Top Layer</i> del módulo de comunicaciones inalámbricas	192
Figura 6.24. Diseño de <i>Bottom Layer</i> del módulo de comunicaciones inalámbricas	192
Figura 6.25. Sección RF_GND del plano de masa	194
Figura 6.26. Sección AGND del plano de masa	194
Figura 6.27. Sección GND del plano de masa	195
Figura 6.28. Sección VCC1V8_RF del plano de alimentación	196
Figura 6.29. Sección VCC1V8_D del plano de alimentación	197
Figura 6.30. Sección VCC1V8 del plano de alimentación	197
Figura 6.31. Sección VCC3V3 del plano de alimentación	198
Figura 6.32. Fichero de salida <i>gerber</i> del módulo de comunicaciones inalámbricas diseñado	199
Figura 6.33. Capa <i>Top Layer</i> de la PCB fabricada	199
Figura 6.34. Capa <i>Bottom Layer</i> de la PCB fabricada	200
Figura 6.35. Conexión PCB del módulo de comunicaciones inalámbricas y el módulo sensor	202
Figura 6.36. Distribución de componentes de la PCB del módulo sensor (<i>Top Layer</i>)	203
Figura 6.37. Distribución de componentes de la PCB del módulo sensor (<i>Bottom Layer</i>)	203
Figura 6.38. Diseño de la <i>Top Layer</i> del módulo sensor	204
Figura 6.39. Diseño de la <i>Bottom Layer</i> del módulo sensor	204
Figura 6.40. Fichero de salida <i>gerber</i> del módulo sensor diseñado	205
Figura 6.41. Capa <i>Top Layer</i> de la PCB fabricada del módulo sensor	205
Figura 6.42. Capa <i>Bottom Layer</i> de la PCB fabricada del módulo sensor	205

Figura 6.43. Aspecto de la cara superior de la PCB finalizada	206
Figura 6.44. Aspecto de la cara inferior de la PCB finalizada	207
Figura 6.45. Aspecto de la cara superior de la PCB del módulo sensor finalizado.....	207
Figura 6.46. Aspecto de la cara inferior de la PCB del módulo sensor finalizado.....	207
Figura 6.47. Nodo sensor inalámbrico terminado	208
Figura 7.1. Red de nodos implementada en la aplicación	210
Figura 7.2. Asignación de señales al puerto B del dispositivo ATmega168	211
Figura 7.3. Asignación de señales al puerto C del dispositivo ATmega168	211
Figura 7.4. Asignación de señales al puerto D del dispositivo ATmega168.....	211
Figura 7.5. Flujograma de la aplicación desarrollada	213
Figura 7.6. Medida de consumo del nodo sensor inalámbrico	217
Figura 7.7. Consumo del nodo sensor inalámbrico en modo reposo.....	217
Figura 7.8. Consumo del nodo sensor inalámbrico en modo transmisión.....	218
Figura 7.9. Consumo del nodo sensor inalámbrico en modo lectura sensores.....	218
Figura 7.10. Antena omnidireccional externa auxiliar	220
Figura 7.11. Alcance máximo en espacio abierto sin emplear antena externa.....	221
Figura 7.12. Alcance máximo en espacio abierto empleando antena externa.....	222
Figura 7.13. Alcance obtenido en el interior del Pabellón A sin antena externa.....	222
Figura 7.14. Alcance obtenido en el interior del Pabellón A con antena externa	223
Figura 7.15. Alcance obtenido en el interior del Pabellón C sin antena externa.....	224
Figura 7.16. Alcance obtenido en el interior del Pabellón C con antena externa.....	224

Índice de tablas

Tabla 3.1. Comparativa entre las principales tecnologías inalámbricas	52
Tabla 3.2. Características de modulación de la capa física	62
Tabla 3.3. Tipos de comandos de la subcapa MAC	71
Tabla 4.1. Características principales del Ember EM260	82
Tabla 4.2. Características de radiofrecuencia del Ember EM260	83
Tabla 4.3. Descripción de los pines del dispositivo EM260	84
Tabla 4.4. Parametros temporales del protocolo SPI del dispositivo EM260	90
Tabla 4.5. Comandos y respuestas SPI del dispositivo EM260	91
Tabla 4.6. Significado del byte de control de trama.....	93
Tabla 4.7. Significado de los bits de los modos de bajo consumo <i>sleepMode</i>	94
Tabla 4.8. Significado del bit <i>overflow</i> de la trama de respuesta	94
Tabla 4.9. Significado del bit <i>truncated</i> de la trama de respuesta.....	94
Tabla 4.10. Valores de configuración de la pila del EM260	96
Tabla 4.11. Tipos de datos básicos empleados en protocolo EZSP en el PFC.....	96
Tabla 4.12. Tipos de variables implementadas en el protocolo EZSP en el PFC	97
Tabla 4.13. Estructura de datos implementada en el protocolo EZSP en el PFC.....	97
Tabla 4.14. Descripción de los pines del dispositivo ATmega168	106
Tabla 4.15. Señales implicadas en el interfaz SPI.....	110
Tabla 4.16. Relación entre la señal SCK y la frecuencia del reloj	113
Tabla 5.1. Tensión de alimentación de los diferentes bloques del módulo	121
Tabla 5.2. Descripción de los pines del TPS77633D	124
Tabla 5.3. Características del balún HHM-1521	132
Tabla 5.4. Características de la antena ANT-2.45-CHP empleada	135
Tabla 5.5. Descripción de los pines de la antena integrada.....	135
Tabla 5.6. Señales del conector de programación del EM260	141
Tabla 5.7. Descripción de señales de la interfaz SPI.....	143
Tabla 5.8. Señales del conector de programación del ATmega168	148
Tabla 5.9. Señales adicionales para el protocolo EZSP	149
Tabla 5.10. Señales del conector de expansión.....	152
Tabla 5.11. Principales características del sensor TMP121	154
Tabla 5.12. Descripciones de los pines del TMP121	155

Tabla 5.13. Registro de temperatura del TMP121	155
Tabla 5.14. Principales características del sensor MP3H6115A6U	157
Tabla 5.15. Descripciones de los pines del MP3H6115A6U	158
Tabla 5.16. Principales características del sensor de movimiento PIR	161
Tabla 5.17. Descripciones de los pines del sensor de movimiento PIR	162
Tabla 5.18. Descripciones de los pines del convertidor REG710NA-5/250	163
Tabla 6.1. Listado parcial de componentes	181
Tabla 6.2. Especificaciones de fabricación de la PCB del módulo sensor	186
Tabla 6.3. Reglas básicas de diseño empleadas en el módulo de comunicaciones inalámbricas...	187
Tabla 6.4. Parámetros establecidos en el asistente de PCB para el módulo sensor.....	200
Tabla 6.5. Reglas básicas de diseño empleadas en el módulo sensor	201
Tabla 7.1. Señalización de los diodos LEDs del firmware desarrollado	215
Tabla 7.2. Consumo del nodo sensor inalámbrico	217
Tabla 7.3. Consumo reducido del nodo sensor inalámbrico	219
Tabla 7.4. Alcance máximo del nodo sensor inalámbrico.....	225

Acrónimos

ACK	Acknowledgement
ALU	Arithmetic Logic Unit
AODV	Ad Hoc On-Demand Vector Routing
API	Application Programming Interface
APS	Application Layer
ASK	Amplitude Shift Keying
AVR	Advanced Virtual RISC
AWACS	Airborne Warning and Control System
BPSK	Binary Phase Shift Keying
CAP	Contention Access Period
CCA	Clear Channel Assessment
CDMA	Code Division Multiple Access
CFP	Contention Free Period
CISC	Complex Instruction Set Computer
CLR	Clear Register
COIT	Colegio Oficial Ingenieros de Telecomunicación
COITT	Colegio Oficial de Ingenieros Técnicos de Telecomunicación
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CS	Chip Select
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance
DARPA	Defense Advanced Research Projects Agency
DECT	Digital Enhanced Cordless Telephone
DIFS	Distributed Inter Frame Space
DSN	Distributed Sensor Networks
DSSS	Direct Sequence Spread Spectrum
ED	Energy Detection
EDGE	Enhanced Data Rates for GSM Evolution
EEPROM	Electrically-Erasable Programmable Read-Only Memory
EITE	Escuela de Ingeniería de Telecomunicación y Electrónica
EMI	Electromagnetic Interference
ESR	Equivalent Series Resistance

ETSI	European Telecommunications Standards Institute
EZSP	EmberZNet Serial Protocol
FCC	Federal Communication Commission
FCS	Frame Check Sequence
FFD	Full Function Device
FHSS	Frecuency Hopping Spread Spectrum
GNSS	Global Navigation Satellite System
GPRS	General Packet Radio Service
GSM	Groupe Spécial Mobile
GTS	Guaranteed Time Slot
HART	Highway Addressable Remote Transducer Protocol
HF	High Frecuency
I²C	Inter-Integrated Circuit
ID	Identifier
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IFS	Inter Frame Spacing
IR	Infrared
IrDA	Infrared Data Association
ISA	Setting the Standards for Automation
ISM	Industrial, Scientific and Medical
ISO	International Organization for Standardization
ISP	In System Programming
ITU	International Telecommunication Union
IUMA	Instituto Universitario de Microelectrónica Aplicada
JTAG	Joint Test Action Group
JTC1	Joint Technical Committee 1 (ISO/IEC)
LAN	Local Area Network
LED	Light Emitting Diode
LF	Low Frequency
LMDS	Local Multipoint Distribution Service
LQI	Link Quality Indicator
LR-WPAN	Low-Rate Wireless Personal Area Network
LoWPAN	Low power Wireless Personal Area Networks
LSB	Least Significant Bit

MAC	Medium Access Control
MF	Medium Frequency
MFR	MAC Footer
MHR	MAC Header
MISO	Master In Slave Out
MIT	Massachusetts Institute of Technology
MLF	Micro Lead Frame
MOSFET	Metal-Oxide Semiconductor Field-Effect Transistor
MOSI	Master Out Slave In
MPDU	MAC Protocol Data Unit
MSB	Most Significant Bit
MSDU	MAC Service Data Unit
NORAD	North American Aerospace Defense Command
NWK	Network Layer
O-QPSK	Offset Quadrature Phase Shift Keying
OSI	Open System Interconnection
P2P	Peer to Peer
PC	Personal Computer
PCB	Printed Circuit Board
PDA	Personal Digital Assistant
PDIP	Plastic dual in-line package
PG	Power-Good
PHY	Physical Layer
PIR	Passive Infra-Red
PPDU	Phy Protocol Data Unit
PPM	Parts Per Million
PPS	Polyphenylene Sulfide
PSK	Phase Shift Keying
PTI	Packet Trace Interface
QFN	Quad Flat No leads
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying
RAM	Random-Access Memory
RCR	Research and Development Center for Radio Systems
RF	Radio frequency
RFID	Reduced-Functionality Device
RFI	Radio Frequency Interference

RFID	Radio Frequency Identification
RISC	Reduced Instruction Set Computer
SCK	Serial Clock
SER	Set Register
SIF	Serial Interface
SIFS	Short Inter Frame Space
SMD	Surface Mount Device
SMS	Short Message Service
SOSUS	Sound Surveillance System
SPCR	SPI control register
SPDR	SPI Data Register
SPI	Serial Peripheral Interface
SPIE	SPI Interrupt Enable
SPIF	Serial Peripheral Interface Flag
SRAM	Static Random Access Memory
SS	Slave Select
SSO	Simultaneous Switching Outputs
SWAP	Shared Wireless Access Protocol
TCP-IP	Transmission Control Protocol-Internet Protocol
TIA	Telecommunications Industry Association
TQFP	Thin Quad Flat Pack
TTR	Telecommunications Technology Research
UART	Universal Asynchronous Receiver Transmitter
ULPGC	Universidad de Las Palmas de Gran Canaria
USART	Universal Synchronous Asynchronous Receiver Transmitter
USB	Universal Serial Bus
UHF	Ultra High frequency
UMTS	Universal Mobile Telecommunications System
VCO	Voltage-Controlled Oscillator
VHF	Very High Frequency
VLF	Very Low Frequency
WiMax	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Network
WMAN	Wireless Metropolitan Area Network
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network
WWAN	Wireless Wide Area Network

ZC	ZigBee Coordinator
ZDO	ZigBee Device Objects
ZED	ZigBee End Device
ZR	ZigBee Router

Capítulo 1

Introducción

1.1 Introducción

En la actualidad, el empleo de redes inalámbricas es muy popular en nuestro entorno, sobre todo en dispositivos móviles como cámaras de fotos, reproductores de música, vídeo y radio, teléfonos móviles, ordenadores portátiles, agendas electrónicas, etc. El uso de la tecnología inalámbrica ha tomado gran importancia, ya que facilita la operación de los dispositivos móviles y ordenadores que no pueden estar en el mismo lugar, o bien que se encuentran separados a una distancia física. De esta manera permite la creación de redes estableciendo una conexión sin necesidad de cables, desapareciendo ciertas limitaciones existentes en las redes cableadas.

Desde que hace años la tecnología inalámbrica comenzara a emplearse en pequeñas redes de oficinas WLAN o en la transmisión de archivos en dispositivos móviles, su evolución ha sido constante, logrando una elevada integración en la sociedad. El crecimiento y la integración adquirida por las redes inalámbricas en los últimos años se basan principalmente en sus características, ofreciendo ventajas que las redes cableadas no proporcionan, tales como movilidad, flexibilidad, coste reducido o escalabilidad, entre otras.

El auge de las comunicaciones inalámbricas en los últimos años ha propiciado su empleo en entornos cada vez más diversos. Así, en campos como el industrial o el doméstico (domótica) no ha pasado desapercibido el auge experimentado, aunque su trascendencia en la sociedad pueda parecer menor que en entornos como el de servicios o el de entretenimiento, donde tecnologías como Bluetooth [9] o Wi-Fi [3] gozan de mucha popularidad y se mencionan de manera cotidiana, existiendo otros no tan conocidos pero también importantes como HART (*High Addressable Remote Transducer protocol*) [26], 6LOWPAN (*IPv6 Over Low power WPAN*) [28] o ZigBee [10], más orientados a la domótica y a entornos industriales.

Los recientes avances en sistemas micro-electro-mecánicos (MEMS), comunicaciones inalámbricas y electrónica digital, han permitido el desarrollo de nodos sensores multifuncionales, de bajo coste, baja potencia, ligeros y portables. Las redes de estos sensores multifuncionales, también llamados motas, se denomina WSN (*Wireless Sensor Network*), o red de sensores inalámbricos. Por lo general, cada nodo de la red consta de un dispositivo con un microcontrolador, sensores, transmisor/receptor, una fuente de energía (habitualmente una batería), y un transceptor encargado de la comunicación. Por otra parte, un sensor es capaz de procesar una limitada cantidad de datos, si bien al coordinar la información entre un importante número de nodos, éstos tienen la habilidad de medir un determinado medio físico con gran detalle. Estas redes tienen gran potencial y existen multitud de sectores en los que su aplicación provoca gran interés, como los campos de medicina, domótica, militar, monitorización de desastres naturales, etc. [63] [64] [65] [66]. En la Figura 1.1 se muestra una red de sensores con topología de malla, muy usada en redes de sensores inalámbricos (WSN), ya que permite a los nodos más alejados del nodo central hacer llegar su mensaje mediante el uso de los demás nodos de la red como repetidores.

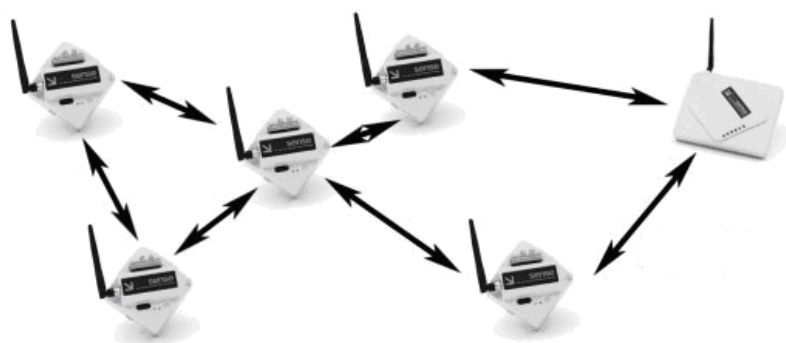


Figura 1.1. Ejemplo de red de sensores inalámbrica con topología en malla

Dado el potencial y al amplio espectro de aplicaciones que soportan las redes de sensores inalámbricos, en los últimos años el desarrollo de dispositivos, protocolos y arquitecturas para el diseño de este tipo de redes se ha convertido en un campo de investigación muy importante [13]. El desarrollo de estas tecnologías ha venido de la mano de nuevos estándares inalámbricos de comunicación, como el mencionado ZigBee, que han propiciado la aparición de nuevos paradigmas como la computación ubícu [13].

El presente Proyecto Fin de Carrera se centra en el estándar de comunicaciones inalámbricas ZigBee, el cual está basado en el estándar IEEE 802.15.4 [25] de redes inalámbricas de área personal (*Wireless Personal Area Network*, WPAN) y tiene como objetivo las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de sus baterías. Además de estas características, ZigBee proporciona la capacidad de una implementación sencilla, empleando un número de componentes reducido, lo que facilita el desarrollo, integración y fabricación de los dispositivos implementados. Así, ZigBee se presenta como un estándar global idóneo tanto para la domótica como para las redes de sensores inalámbricos.

1.2 Objetivos

El principal objetivo de este Proyecto Fin de Carrera es el diseño, desarrollo, implementación y fabricación de un nodo sensor inalámbrico de bajo consumo y propósito general, basado en el estándar de comunicaciones inalámbricas ZigBee, capaz de comunicarse y enviar los datos obtenidos a través de sensores, por radiofrecuencia, a diferentes estructuras de red de propósito general, como una red de sensores, una red de control de determinadas funciones, o un dispositivo domótico.

Para el diseño del nodo inicialmente se llevará a cabo un estudio teórico del estándar de comunicaciones empleado, ZigBee, así como del dispositivo que lo implementará, el co-procesador de red Ember EM260 [38], con el fin de comprender los parámetros más importantes que se deben tener en cuenta a la hora del diseño. También se realizará el estudio teórico del microcontrolador seleccionado para formar parte del diseño, el microcontrolador ATmega168 de Atmel [44], que facilitará su implementación en el posterior diseño.

Una vez completado el estudio teórico se procede a la elaboración del diseño. Para la elaboración del mismo se realizará el esquemático completo del circuito realizado. Posteriormente se pasa a la elaboración y fabricación de las PCBs, creando las librerías de componentes necesarias, así como

ubicar los componentes y realizar el ruteado de las pistas del mismo. Una vez finalizadas las PCBs y realizadas las verificaciones oportunas, se procede al montaje de las placas de circuito impreso. El presente Proyecto Fin de Carrera concluye con la implementación de una aplicación práctica, donde el nodo sensor inalámbrico diseñado obtiene información del medio mediante sensores, para luego ser enviada a un nodo central (coordinador).

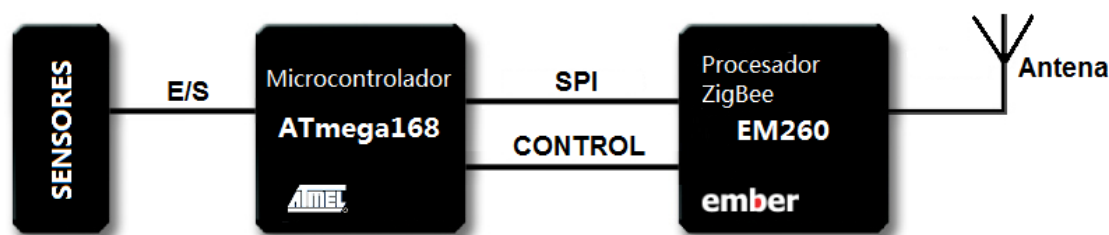


Figura 1.2. Diagrama de bloques simplificado del sistema completo

En la Figura 1.2 se representa el diagrama de bloques simplificado del nodo sensor inalámbrico a desarrollar, compuesto principalmente por el co-procesador de comunicaciones EM260, el microcontrolador ATmega168, y los sensores incorporados. El dispositivo EM260 se encarga de implementar la parte de radiofrecuencia y la pila del protocolo ZigBee, mientras que el microcontrolador ATmega168 realiza el procesamiento de los datos recibidos por los sensores a través de los puertos de E/S y se encarga de la gestión de la red mediante el interfaz SPI (*Serial Peripheral Interface*) y las señales de control conectadas al dispositivo EM260.

El nodo desarrollado se compondrá de dos módulos. Por un lado, un módulo de comunicaciones inalámbrico, donde se implementan el co-procesador ZigBee, el dispositivo ATmega168, y la circuitería necesaria de radiofrecuencia. Este módulo funcionará como plataforma de comunicaciones inalámbrica para el segundo módulo, denominado módulo sensor, el cual contiene los diferentes sensores empleados en el presente Proyecto Fin de Carrera, así como la circuitería para el acondicionamiento de sus señales. Cada módulo es implementado en una placa de circuito impreso diferente y su conexión se realizará mediante un conector de expansión, que se incorporará a los módulos.

Así pues, el objetivo de este Proyecto Fin de Carrera es demostrar la funcionalidad del sistema ZigBee en el nodo implementado, para lo cual se completará una comunicación entre dispositivos que permita el desarrollo de múltiples aplicaciones con gran flexibilidad. Además, se busca determinar y evaluar las capacidades funcionales del co-procesador de red EM260, incluyendo también en ellas el consumo de potencia y su alcance aproximado bajo ciertas condiciones.

1.3 Peticionario

Actúa como petionario de este Proyecto Fin de Carrera, la Escuela de Ingeniería de Telecomunicación y Electrónica (EITE) de la Universidad de Las Palmas de Gran Canaria (ULPGC), como requisito para la obtención del título académico de Ingeniero Técnico de Telecomunicación en Sistemas Electrónicos.

1.4 Estructura de la memoria

La memoria del presente Proyecto Fin de Carrera se desarrollará en ocho capítulos, acompañados de una lista de Acrónimos, situada al comienzo del documento, una lista de Referencias Bibliográficas, el Presupuesto y cuatro Anexos, localizados al final de la memoria. A continuación se proporciona un breve resumen de cada capítulo.

El primer capítulo presenta una breve introducción a las redes de comunicación inalámbricas y la función que éstas desarrollan en la sociedad actual.

En el segundo capítulo se describirán las tecnologías inalámbricas existentes y los tipos de redes que se pueden encontrar. También se incluye una descripción completa de lo que son las redes de sensores, indicando sus principales características y las posibles aplicaciones que éstas tienen actualmente.

En el tercer capítulo se realiza un estudio detallado del estándar de comunicaciones inalámbricas ZigBee, donde se exponen sus principales características, los tipos de dispositivos existentes, las topologías de red permitidas, etc. Además se describe la capa física y *MAC* que emplea el estándar, así como la modulación permitida para la transmisión de datos o el modo de direccionamiento de paquetes.

El cuarto capítulo describe el co-procesador EM260 y el microcontrolador ATmega168, los dos componentes más significativos empleados en este Proyecto Fin de Carrera. Se realiza la descripción de las características más importantes que ofrece cada dispositivo, haciendo especial hincapié en las funciones utilizadas en el presente Proyecto Fin de Carrera, como los protocolos de comunicaciones empleados por los dispositivos, SPI y EZSP, explicando con detalle su modo de funcionamiento.

En el quinto capítulo se explican todos aquellos aspectos relacionados con el diseño hardware del nodo sensor inalámbrico, describiendo cada bloque funcional, así como el proceso de diseño seguido en cada uno para su correcto funcionamiento. También se explica con detenimiento la función de cada uno de los componentes utilizados en el diseño, especificando las tensiones de alimentación empleadas, o los puertos de entrada y salida usados.

El sexto capítulo contiene una explicación detallada de los pasos seguidos en el diseño de las placas de circuito impreso, así como el montaje de las mismas.

En el séptimo capítulo se realiza la descripción a nivel funcional del *firmware* desarrollado para verificar la funcionalidad del nodo sensor inalámbrico implementado en el presente Proyecto Fin de Carrera. Además, se explica detalladamente cada una de las partes del programa ejecutado por el microcontrolador y se exponen los resultados obtenidos en la realización de las medidas de las principales prestaciones del nodo sensor inalámbrico desarrollado, incluyendo su alcance y su consumo.

Por último, en el octavo capítulo se resumen las principales conclusiones a las que se ha llegado tras la realización del presente PFC y se proponen algunas líneas a seguir en futuros desarrollos.

Capítulo 2

Tecnología inalámbrica

En este capítulo se describen brevemente los principios básicos de las la comunicaciones inalámbricas y se explica el concepto de redes inalámbricas, definiendo los tipos que existen y las ventajas y desventajas que presenta su uso. Para finalizar, se abordan en detalle las redes de sensores inalámbricos, mencionando los aspectos más relevantes que ofrecen actualmente, además de una breve reseña histórica.

2.1 Introducción

Se entiende por comunicación inalámbrica la transferencia de información a distancia sin la utilización de conductores eléctricos o cables. Las distancias pueden ser cortas (unos pocos metros como los mandos de los receptores de televisión) o muy largas (miles o incluso millones de kilómetros para el caso de las comunicaciones por radio). La comunicación inalámbrica se considera generalmente una rama de las telecomunicaciones, pudiéndose establecer la siguiente clasificación:

- Comunicación por radiofrecuencia.
- Comunicación por microondas.
- Comunicación por infrarrojos de corto alcance (IR).

Así, existen situaciones en las que el uso de la tecnología inalámbrica se encuentra justificada, como pueden ser:

- Para abarcar grandes distancias, donde no es viable el cableado típico.
- Para evitar obstáculos, tales como las estructuras físicas, EMI, o RFI.
- Con el fin de proporcionar un enlace de comunicaciones de seguridad en caso de fallo de la red normal.
- Para conectar portátiles o estaciones de trabajo temporales.
- Evitar situaciones en las que el cableado resulta difícil (por su emplazamiento) o económicamente inviable.
- Para conectar remotamente a usuarios móviles o a redes.

2.2 Un poco de historia

Los servicios inalámbricos representan una evolución de la tecnología y una nueva era en las telecomunicaciones, si bien estos servicios no son nuevos, habiendo sido utilizados durante años. Los comienzos de estos servicios inalámbricos datan del siglo XIX, cuando Marconi 'el padre de la radio' dejó su huella en el mundo de la tecnología inalámbrica. Marconi comenzó a experimentar con las ondas de radio (ondas hertzianas) en 1894 su objetivo era producir y detectar ondas de radio sobre largas distancias. En 1896 fructificaron las investigaciones de Marconi y obtuvo una patente, para luego fundar la *Wireless Telegraph & Signal Co. Ltd*, la primera fábrica de radios del mundo [1]. En 1901 se recibieron las primeras señales del otro lado del Atlántico, y ya en 1905 se envió la primera señal inalámbrica usando Código Morse.

La tecnología inalámbrica fue evolucionando en gran medida gracias al ejército de los Estados Unidos, que la utilizó como una herramienta útil en su desarrollo militar. El ejército utilizó esta tecnología para la transmisión de datos con un complejo sistema de encriptado, lo cual hacía casi imposible que personal no autorizado pudiera acceder a esta información. Este tipo de tecnología se introdujo por primera vez durante la Segunda Guerra Mundial, cuando el ejército comenzó a enviar planes entre batallones sobre las líneas enemigas, o cuando la armada daba instrucciones a su flota desde la costa.

La tecnología inalámbrica demostró ser un valioso medio de comunicación. Así, muchas empresas y universidades pensaron que podían ampliar el alcance de sus redes cableadas de área local (LAN) mediante el uso de redes LAN inalámbricas. La primera LAN inalámbrica apareció en 1971, y fue desarrollada por la Universidad de Hawaii, denominada ALOHANET [2]. Esta primera red basada

en *packet radio*, permitía a los ordenadores de siete campus repartidos en cuatro islas que se comunicaran con una transmisión vía radio a un ordenador central ubicado en Oahu. La arquitectura de red utilizaba una topología en estrella con un ordenador central (*hub*, en el centro de la estrella). Dos ordenadores podían establecer un enlace bi-direccional de comunicación entre ellos, pasando por el *hub* central. ALOHANET incorporó el primer conjunto de protocolos para el acceso y encaminamiento de paquetes al canal en los sistemas de radio. Aun hoy en día se siguen usando en las transmisiones inalámbricas muchos de estos principios básicos. De esta manera, la tecnología inalámbrica comenzó su viaje hacia lo que se conoce hoy en día, adentrándose en el interior de cada casa, aula o empresa de todo el mundo.

2.3 Redes inalámbricas

Las redes inalámbricas imitan a las redes cableadas, pero se comunican a través de radio frecuencia (RF) en lugar de utilizar las transmisiones convencionales de las redes cableadas. Comúnmente se le asocia con una red de telecomunicaciones, que están generalmente aplicadas a sistemas con algún tipo de transmisión de información remota, utilizando ondas electromagnéticas, como las ondas de radio. La transmisión y recepción de estas ondas se realiza a través de antenas.

Las redes inalámbricas presentan una serie de ventajas y desventajas con respecto a las redes cableadas. Entre las ventajas podemos citar las siguientes propiedades:

- Escalabilidad

Los sistemas inalámbricos ofrecen una gran facilidad para expandir la red después de su instalación inicial, además de poder ser configurados en una variedad de topologías con el fin de satisfacer las necesidades de las instalaciones y aplicaciones específicas.

- Flexibilidad

Al no estar los dispositivos unidos por cables, la flexibilidad de cada nodo es mucho mayor, pudiendo emplearse redes inalámbricas en lugares temporales, para cubrir zonas de difícil acceso al cable, enlazar edificios distantes, etc.

- Coste

Diseñar o instalar una red cableada puede llegar a alcanzar un elevado coste; las redes inalámbricas permiten reducir estos gastos. En muchos casos también se consigue una

disminución en el coste de mantenimiento de la red. Los beneficios a largo plazo son superiores en ambientes dinámicos que requieren acciones y desplazamientos frecuentes.

Como desventajas más significativas de las redes inalámbricas se pueden citar las siguientes:

- Interferencia

En este caso es posible que varias redes inalámbricas interfieran entre sí, pero en la práctica, las interferencias no son usuales, debido a que no todas las redes comparten las mismas bandas de frecuencia, y los niveles de potencia y modulación del espectro suelen ser diferentes. Además, los últimos estándares están consiguiendo alta inmunidad frente a interferencias utilizando técnicas de modulación más avanzadas [25].

- Seguridad

En principio, cualquier dispositivo puede ingresar en el rango de comunicación de una red inalámbrica. Este campo está bastante criticado en casi todos los estándares actuales, que, según dicen, no se deben utilizar en entornos críticos en los cuales un “robo” de datos pueda resultar peligroso, si bien los estándares cada vez son más sofisticados y tienen más sistemas de seguridad, siendo menos vulnerables.

2.3.1 Categorías

Las redes inalámbricas pueden clasificarse en cuatro categorías:

- WPAN (*Wireless Personal Area Network*)

Las WPAN son un tipo de red inalámbrica que interconecta los dispositivos dentro de un área relativamente pequeña, que normalmente comprende algunas decenas de metros. Este tipo de red se usa generalmente para conectar dispositivos periféricos (por ejemplo, impresoras, teléfonos móviles y electrodomésticos) o un asistente personal digital (PDA) a un ordenador sin cables. Se usan varios tipos de tecnologías, como pueden ser *HomeRF* (estándar para conectar todos los teléfonos móviles de la casa y los ordenadores mediante un dispositivo central); Bluetooth (protocolo que sigue la especificación IEEE 802.15.1); ZigBee (basado en la especificación IEEE 802.15.4); o RFID (sistema remoto de almacenamiento y recuperación de datos con el propósito de transmitir la identidad de un objeto mediante ondas de radio, similar a un número de serie único).

- WLAN (*Wireless Local Area Network*)

Las WLAN son un sistema de comunicación de datos inalámbrico flexible, muy utilizado como alternativa a las redes LAN cableadas o como extensión de éstas, cubriendo un área equivalente a la red local de una empresa, con un alcance aproximado de cien metros. Permite que los terminales que se encuentran dentro del área de cobertura puedan conectarse entre sí. Existen varios tipos de tecnologías, si bien la más utilizada es Wi-Fi [3] (o IEEE 802.11 [4]).

- WMAN (*Wireless Metropolitan Area Network*)

Las redes inalámbricas de área metropolitana (WMAN) están diseñadas para cubrir áreas geográficas más amplias que una LAN, que van desde varios bloques de edificios a ciudades enteras, pudiéndose extender hasta 50 kilómetros. La tecnología habitualmente utilizada es WiMax (*Worldwide Interoperability for Microwave Access*), basada en el estándar IEEE 802.16 [5], aunque también existen otros sistemas de comunicación basados en este estándar para redes WMAN, como el LMDS (*Local Multipoint Distribution Service*).

- WWAN (*Wireless Wide Area Network*)

Las redes inalámbricas de área extensa (WWAN) tienen el alcance más amplio de todas las redes inalámbricas. Son capaces de cubrir distancias comprendidas entre unos 100 km y 1000 km, dando el servicio a un país o a un continente. Como ejemplos se pueden encontrar los teléfonos móviles, que están conectados a una red inalámbrica de área extensa. Entre las principales tecnologías de área extensa se encuentra GSM [6] (*Global System for Mobile Communications*), GPRS (*General Packet Radio Service*) o UMTS (*Universal Mobile Telecommunications System*).

En la Figura 2.1 se puede observar la relación que existe entre los tipos de redes en función de su área de cobertura y las tecnologías más usuales asociadas a cada categoría.

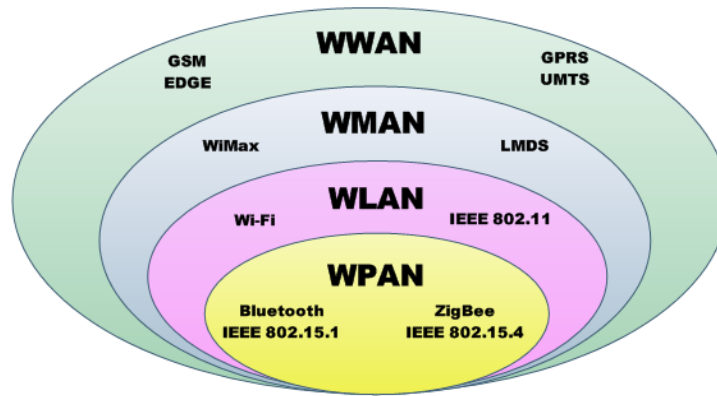


Figura 2.1. Relación entre área de cobertura y tecnologías

2.3.2 Aplicaciones

Según como sea el medio de transmisión (ondas de radio, microondas terrestres, microondas por satélite, o infrarrojos) la red inalámbrica tendrá unas características que resultarán idóneas dependiendo de la aplicación.

Así, en el rango de frecuencias de las ondas de radio se encuentran bandas muy importantes [7] con aplicaciones inalámbricas, como son VLF (*Very Low Frequency*) o LF (*Low Frequency*), que se utilizan para control aeronáutico, navegación, información y servicios meteorológicos. También se encuentran las bandas MF (*Medium Frequency*) y HF (*High Frequency*) que son reservadas para uso de las fuerzas de seguridad y de defensa, además de para las transmisiones de onda corta de emisoras de radio internacionales y estaciones de radioaficionados. No se pueden olvidar otras bandas tan conocidas como VHF (*Very High Frequency*) y UHF (*Ultra High Frequency*) que ofrecen la radio y la televisión que se utiliza a diario.

A partir de las microondas terrestres, existen diferentes aplicaciones basadas en protocolos como Bluetooth, ZigBee o Wi-Fi, que se emplean para interconectar ordenadores portátiles, PDAs, teléfonos u otros aparatos. También se utilizan las microondas para comunicaciones con radares (detección de velocidad u otras características de objetos remotos) y para la televisión digital terrestre. Las microondas por satélite se usan además para la transmisión telefónica a larga distancia y en redes privadas.

Los infrarrojos tienen aplicaciones en la comunicación a corta distancia de los ordenadores con sus periféricos. También se utilizan para mandos a distancia, ya que así no interfieren con otras señales electromagnéticas, como por ejemplo la señal de televisión. Uno de los estándares más usados en

estas comunicaciones es el IrDA (*Infrared Data Association*). Otros usos que tienen los infrarrojos son técnicas como la termografía, la cual permite determinar la temperatura de objetos a distancia.

2.3.3 Estándares de comunicaciones inalámbricas

Los estándares son un conjunto de normas y recomendaciones técnicas elaboradas con el fin de garantizar la interoperabilidad entre elementos construidos independientemente. La Organización Internacional para la Estandarización (ISO) [8], uno de los principales organismos internacionales desarrolladores de estándares, los define como "acuerdos documentados que contienen especificaciones técnicas u otros criterios precisos para ser usados consistentemente como reglas, guías o definiciones de características para asegurar que los materiales, productos, procesos y servicios cumplan con su propósito".

Los principales organismos en el desarrollo de estándares son:

- ITU: International Telecommunication Union.
- JTC1: Joint Technical Committee 1 (ISO/IEC).
- ISO: International Organization for Standardization.
- IEC: International Electrotechnical Commission.
- ETSI: European Telecommunications Standards Institute (Europa).
- T1: Committee T1 (EE.UU.).
- TIA: Telecommunications Industry Association (EE.UU.).
- IEEE: Institute of Electrical and Electronics Engineers (EE.UU.).
- TTR: Telecommunications Technology Research (Japón).
- RCR: Research and Development Center for Radio Systems (Japón).

Algunos de los estándares más importantes que se han desarrollado en relación a la tecnología inalámbrica son:

- Bluetooth

El estándar Bluetooth [9] es una tecnología de ondas de radio de corto alcance (2.4 GHz de frecuencia) cuyo objetivo es simplificar las comunicaciones entre dispositivos informáticos, como ordenadores portátiles, teléfonos móviles, dispositivos de PDAs, etc. A su vez, estos pueden conectarse a Internet. También pretende simplificar la sincronización de datos entre los dispositivos y otros ordenadores.

- ZigBee

ZigBee [10] es un estándar de comunicaciones inalámbricas diseñado por *ZigBee Alliance*. Es un conjunto estandarizado de soluciones que pueden ser implementadas por cualquier fabricante. ZigBee está basado en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (WPAN) y tiene como objetivo las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de sus baterías.

- RFID Contactless tags

RFID [11] responde a las siglas *de Radio Frequency IDentification*, en español Identificación por radiofrecuencia. Consiste en un método de almacenamiento y recuperación de datos remoto que usa etiquetas (*RFID tag* en inglés). Dichas etiquetas o *tags* son de tamaño reducido y pueden ser adheridas a una mercancía, producto, animal o persona.

- Wi-Fi

Wi-Fi [3] constituye un conjunto de estándares para redes inalámbricas basados en las especificaciones IEEE 802.11 [4]. Fue creado para ser utilizado en redes locales inalámbricas, si bien es frecuente que en la actualidad también se utilice para acceder a Internet. El protocolo IEEE 802.11 o Wi-Fi es un estándar de protocolo de comunicaciones de IEEE que define el uso de los dos niveles más bajos de la arquitectura OSI (capas física y de enlace de datos), especificando sus normas de funcionamiento en una WLAN. La norma IEEE.802.11 fue diseñada para sustituir a las capas Física y MAC de la norma 802.3 [12] (Ethernet).

- GSM

GSM [6] (*Global System for Mobile Communications*) es un sistema telefónico digital difundido en Europa usado especialmente en telefonía móvil para la comunicación mediante teléfonos móviles que incorporan tecnología digital. Por ser digital, cualquier cliente de GSM puede conectarse a través de su teléfono con su ordenador y puede enviar y recibir mensajes por e-mail, faxes, navegar por Internet, acceder de forma segura a la red informática de una compañía (LAN/Intranet), así como utilizar otras funciones digitales de transmisión de datos, incluyendo el Servicio de Mensajes Cortos (SMS) o mensajes de texto.

- GPRS

El estándar GPRS (*General Packet Radio Service*) es una evolución de GSM, extendiendo la arquitectura del estándar GSM para permitir la transferencia de datos con una tasa teórica de

171,2 Kbits/s (hasta 114 Kbits/s en la práctica), muy por encima de los 13 Kbit/s del estándar GSM.

- EDGE

El estándar EDGE (*Enhanced Data Rates for GSM Evolution*) representa otra evolución del estándar GSM que modifica el tipo de modulación. Al igual que el estándar GPRS, EDGE está pensado para ser una transición hacia la tercera generación de telefonía móvil (3G). También se utiliza el término 2.75G para describir el estándar EDGE.

- GPS

GPS [35] constituye un sistema global de navegación por satélite (GNSS) que permite localizar con precisión un dispositivo GPS en cualquier lugar del mundo. El sistema fue desarrollado y actualmente es operado por el Departamento de Defensa de los EE.UU. Está compuesto por veinticuatro satélites (21 operativos y 3 de respaldo) que están en órbita a unos 20.200 km de la Tierra con trayectorias sincronizadas para cubrir toda la superficie terrestre.

- UMTS

UMTS (Universal Mobile Telecommunications System) es el estándar que se emplea en la denominada tercera generación de telefonía móvil, permitiendo disponer de banda ancha en telefonía móvil y transmitir un volumen de datos importante a través de la red. Con la tercera generación es posible establecer videoconferencias, descargar vídeos, intercambiar postales electrónicas, paseos 'virtuales' por casas en venta, etc., todo desde el móvil.

- WIMAX

WiMax (del inglés *Worldwide Interoperability for Microwave Access*) es un estándar de transmisión inalámbrica basado en las especificaciones IEEE 802.16 [5] y diseñado para ser utilizado en el área metropolitana (MAN), proporcionando accesos concurrentes en áreas de hasta 48 kilómetros y a velocidades de hasta 70 Mbps, utilizando tecnología portátil LMDS [36].

De todos los estándares introducidos, el presente Proyecto Fin de Carrera se centra en el estándar ZigBee (IEEE 802.15.4), que se describirá más detalladamente en el siguiente capítulo.

2.4 Redes de sensores inalámbricas

Hoy en día los sensores pueden encontrarse en gran número de sistemas y dispositivos electrónicos. Una gran parte de estos sensores adolecen de la capacidad de procesar y analizar los datos que detectan, limitándose a funcionar como un transductor que realiza la medición de una o más variables del entorno y envía dicha información a un procesador central. Sin embargo, se está introduciendo gradualmente una nueva generación de sensores, dotados de inteligencia propia, capaces de organizarse autónomamente y de interconectarse de forma inalámbrica con otros dispositivos semejantes. Surgen así las llamadas redes de sensores inalámbricos (*Wireless Sensor Networks*, WSN) [14].

El desarrollo de protocolos y arquitecturas para el diseño de redes de sensores inalámbricos se ha convertido en un campo de investigación muy importante en los últimos años [13]. El desarrollo de estas tecnologías ha venido de la mano de nuevos estándares inalámbricos de comunicación, que han permitido la aparición de nuevos paradigmas, como la computación ubícuca [13]. Dentro de este nuevo escenario se prevé un futuro en el que minúsculos sensores monitoricen continuamente el medio y reporten su información a los nodos próximos y a una estación base central, formando una arquitectura computacional en *grid*.

En los próximos apartados se desarrollará el concepto de redes de sensores, definiéndola y describiendo sus principales características. También se explicará brevemente su historia, finalizando con una revisión de sus principales campos de aplicación, además de los diferentes estándares y tecnologías que existen en WSN.

2.4.1 Definición

Una red de sensores inalámbricos consiste en un sistema complejo de distribución de nodos autónomos dotados de sensores capaces de monitorizar condiciones medioambientales (tales como temperatura, humedad, movimiento, presión etc.), procesar datos, almacenarlos e intercambiar información entre ellos. Por lo general se busca un consumo de potencia limitado para ayudar a su portabilidad. Para ello, cada nodo de la red inalámbrica está equipado, típicamente, con un transmisor u otro dispositivo de comunicación inalámbrico, un pequeño microcontrolador, uno o varios sensores, y una fuente de alimentación (por lo general una batería) [15], [16]. Estos dispositivos se encuentran dispersos de manera *ad-hoc* en una determinada área a monitorizar. En redes de comunicación, dicha expresión hace referencia a una red en la que no hay nodo central,

sino que todos los nodos están en igualdad de condiciones. El despliegue de este tipo de redes de sensores distribuidos ofrece varias ventajas frente a sistemas centralizados, como son su gran robustez, diversidad, y escalabilidad, así como su capacidad de auto-organización o mayor densidad espacial de medida, entre otras.

Otro aspecto importante a considerar es el de la seguridad (tanto desde el punto de vista de autorización como de protección de la información transmitida y/o almacenada, p.e., en el caso de datos biométricos de pacientes).

2.4.2 Historia

Como sucede con muchas tecnologías, el desarrollo de las redes de sensores tiene su origen en la investigación para aplicaciones militares. La primera de estas redes conocidas fue desarrollada por Estados Unidos durante la Guerra Fría y se trataba de una red de sensores acústicos desplegados en el fondo del mar en la línea que va desde Canadá al Reino Unido, pasando por Islandia, lo que permitió estimar las posiciones de los submarinos soviéticos por técnicas de triangulación. El nombre de esta red era SOSUS (*SOund SURveillance System*) [17] [18]. Una vez finalizada la Guerra Fría, la marina de los EE.UU. permitió a científicos civiles utilizar el sistema SOSUS para realizar investigaciones básicas. Por otro lado, durante la Guerra Fría, EE.UU. también desarrolló una red de radares aéreos, a modo de sensores, que han ido evolucionando hasta lo que hoy en día se conoce como aviones de control y vigilancia aérea o AWACS (*Airborne WArning and Control System*) [19]. Otro ejemplo de radares distribuidos es el sistema NORAD (*North American Aerospace Defense Command*) [20], creado por Estados Unidos y Canadá durante la década de los 50 como respuesta a la amenaza de ataque de bombarderos soviéticos. Para el sistema NORAD se construyeron 58 estaciones de radar a lo largo del paralelo 69, lo que daba una alerta 3 horas antes de que un ataque llegara a núcleos de población de países amigos. Estas primeras redes de sensores adoptaban generalmente una estructura de procesamiento jerárquico, donde el procesado se producía en niveles consecutivos antes de que la información sobre los eventos de interés llegase al usuario. Los nodos eran grandes estaciones distantes espacialmente y su comunicación tenía lugar a través de una infraestructura cableada.

El mayor impulso a la investigación de redes de sensores se le atribuye al programa DSN (*Distributed Sensor Networks*) de la agencia militar de investigación avanzada de Estados Unidos (*Defense Advanced Research Projects Agency*, DARPA) en la década de los 80 [21]. El programa DSN postulaba la existencia de multitud de nodos sensores de bajo coste, distribuidos espacialmente, colaborando unos con otros pero funcionando de forma autónoma, con el fin de que

la información fuera encaminada al nodo que hiciese mejor uso de esa información. Para llevarlo a cabo, el DSN pretendía determinar si la reciente aparición del protocolo TCP-IP de la red Arpanet [22] era extensible a las redes de sensores. Aunque la idea original de los investigadores se basaba en redes compuestas por un gran número de pequeños sensores, la tecnología para pequeños sensores todavía no estaba suficientemente madura. El programa DSN centró finalmente las investigaciones en la computación distribuida, el procesamiento de señales y en el *tracking* [37]. El programa recibió colaboraciones de otras instituciones, así como de investigadores de la universidad Carnegie Mellon [23] o del Instituto Tecnológico de Massachussets (MIT) [24].

En las décadas de los 80 y los 90, basándose en los resultados de las investigaciones del programa DSN, se desarrollaron importantes redes de sensores con propósitos militares, acaparando una gran importancia en las aplicaciones militares de detección y rastreo debido a su amplio rango de detección y su pequeño tiempo de respuesta. Estas características de las WSN proporcionaban ventajas sobre otros tipos de redes.

Los avances electrónicos y en el ámbito de la comunicación que tuvieron lugar a finales de los años 90 y principios del siglo XXI dieron como resultado una nueva generación tecnológica en las redes de sensores, adquiriendo una mayor relevancia en el ámbito civil debido a mejoras significativas, como la disminución de su tamaño y el incremento en su autonomía.

Actualmente continúa la evolución de las redes de sensores gracias en gran parte al desarrollo de la tecnología microelectrónica y a la comunicación inalámbrica, utilizando circuitos integrados más complejos y con mayor integración de sensores de todo tipo.

2.4.3 Aplicaciones

Las aplicaciones de las redes de sensores inalámbricos pueden ser muchas y muy variadas, pudiendo soportar un amplio espectro de aplicaciones, si bien suelen incluir algún tipo de supervisión, seguimiento y control. Pueden encontrarse aplicaciones específicas para las redes de sensores inalámbricos, como pueden ser: supervisión de una zona, seguimiento de un objeto, control de un reactor nuclear, detección de incendios, vigilancia del tráfico, etc. En una aplicación típica, los sensores de una WSN están dispersos en la región donde se pretende recoger los datos y son transmitidos mediante el dispositivo de comunicación del nodo.

Tradicionalmente, las WSNs se han utilizado en el contexto de aplicaciones de alto coste, tales como: sistemas de detección de radiación y amenazas nucleares; sensores de armas para buques;

reconocimiento y vigilancia en el campo de batalla; sistemas de inteligencia, comunicaciones, control, orientación y mando militar; aplicaciones biomédicas; detección de hábitat; y monitorización de zonas sísmicas. Recientemente, el interés se ha extendido a las redes de sensores biológicas y químicas para aplicaciones de seguridad nacional. Además, su evolución ha conseguido que el interés se extienda a las aplicaciones de consumo.

Las redes de sensores tienen una amplia variedad de aplicaciones, entre las que se pueden destacar algunas categorías con un gran potencial de crecimiento en los próximos años:

- Control de edificios comerciales.
- Sensores inalámbricos medioambientales (tierra, aire, mar) y agrícolas.
- Domótica, incluyendo las alarmas (por ejemplo, un sensor de alarma que activa la llamada a una empresa de seguridad).
- Aplicaciones de seguridad nacional: sensores inalámbricos químicos, biológicos, radiológicos y nucleares (sensores para agentes de tóxicos químicos, explosivos y biológicos).
- Vigilancia y control industrial.
- Procesos de control en cadenas de montaje, etc.
- Automatizado inalámbrico de lectura y gestión de carga.
- Asistencia militar o civil.
- Control de inventario.
- Control médico.
- Control de tráfico.

En los últimos años han surgido múltiples iniciativas y proyectos de investigación de enorme interés y aplicabilidad práctica. Un ejemplo de ello se encuentra en las aplicaciones agrícolas, un área donde la rapidez de la implantación de este tipo de redes está favoreciendo una reducción del consumo de agua y pesticidas, así como la preservación del entorno. Además, la miniaturización de los dispositivos en los últimos años ha favorecido la aparición de nuevas tendencias en este sector, como la llamada agricultura de precisión, encargada de las aplicaciones relativas a la gestión de cultivos, árboles, flores, etc. Otro área que se ha visto beneficiada por las redes de sensores y con un gran potencial, es el de las aplicaciones sociales y sanitarias, tanto en cuidados médicos en hospitales con monitorización en tiempo real, como en el control y seguimiento de personas mayores en sus domicilios, permitiendo mejorar aspectos como la calidad de vida de las personas. Además de las mencionadas, existe muchas aplicaciones relacionadas con la naturaleza que están en desarrollo, debido al aumento de espacios verdes y parques naturales y la necesidad de un mantenimiento y cuidado constante, resultando bastante complejo en zonas

especiales por las características de las mismas, ya sea por su bastas áreas o la dificultad del acceso, de este modo las redes de sensores resultan de gran ayuda para controlar diversos parámetro ecológicos, además en muchos casos resulta menos intrusivo que otros métodos, los cuales necesitan la circulación de personas y vehículos cerca de las especies de vegetales y animales que se quieren preservar.

2.4.4 Características

Aunque las redes de sensores inalámbricos tienen algunas similitudes con las redes tradicionales y las redes *ad-hoc*, una red de sensores inalámbricos debe tener unas características únicas para cumplir las necesidades de las aplicaciones comentadas. La necesidad de estas características conlleva una interesante combinación de cuestiones técnicas, que no se encuentran en otras redes inalámbricas.

Las principales características de las redes de sensores son:

- *Escalabilidad*: puesto que las aplicaciones van creciendo con el tiempo y el despliegue de la red es progresivo, es necesario que la solución elegida para la red permita su crecimiento. No sólo es necesario que la red funcione correctamente con el número de nodos con que inicialmente se cuente, sino que también debe permitir aumentar ese número sin que las prestaciones de la red se degraden drásticamente.
- *Consumo energético*: es uno de los aspectos más importantes en las redes de sensores. El nodo sensor inalámbrico, siendo un dispositivo microelectrónico, sólo puede estar equipado con una fuente energética limitada. En los escenarios de algunas aplicaciones, la recarga de los recursos energéticos puede ser imposible, por lo que cuanto más se reduzca el consumo de un nodo, mayor será el tiempo durante el cual pueda operar y, por tanto, mayor tiempo de vida tendrá la red.
- *Bajo coste*: las redes de sensores pueden estar formadas por un gran número de nodos, por lo que el coste de un nodo individual es muy importante para justificar el coste completo de la red. Si el coste de la red es mayor que el despliegue de sensores tradicionales, la red sensora no está justificada desde el punto de vista económico. Para minimizar el coste de los componentes se intentan fabricar grandes cantidades, facilitando la estandarización un alto volumen de producción, y en consecuencia, minimizando los costes.

- *Topología de red dinámica*: la posición en la que se coloca cada nodo puede ser arbitraria y normalmente es desconocida por los otros nodos. La localización no tiene porqué estar diseñada ni preestablecida, lo que permitiría un despliegue aleatorio en terrenos inaccesibles u operaciones de alivio en desastres. Por otro lado, los algoritmos y protocolos de red deberán poder organizarse automáticamente.
- *Dispositivos con capacidad limitada*: los dispositivos, como contrapartida a su bajo consumo de potencia, coste y pequeño tamaño, disponen de recursos limitados a nivel hardware y software.
- *Tolerancia a fallos*: algunos nodos sensores pueden fallar o bloquearse debido a la falta de energía, o recibir daños físicos o interferencias medioambientales. En cualquier caso, se deben minimizar las consecuencias de ese fallo evitando comprometer el funcionamiento global de la red de sensores.
- *Comunicación multi-hop*: los nodos sensores usan comunicación por difusión, y debido a que están densamente desplegados, los vecinos están muy cerca unos de otros, por lo que la comunicación *multi-hop* (salto múltiple de uno a otro) proporciona un menor consumo de potencia que la comunicación *single-hop* (salto simple). Además, los niveles de transmisión de potencia se mantienen muy bajos y existen menos problemas de propagación en comunicaciones inalámbricas de larga distancia.

2.4.5 Estándares y Tecnologías

Los estándares inalámbricos más extendidos hoy en día son IEEE 802.15.4 [25], Bluetooth [9] e IEEE 802.11 [4]. Cada uno de ellos nació con un objetivo distinto y por lo tanto tiene un rango de aplicación y unas características diferentes.

En los estándares recogidos bajo la norma 802.11 se busca la interconexión de computadores y periféricos con unas necesidades de ancho de banda elevadas. La norma fue definida en 1997 por la IEEE y establece los dos niveles inferiores del modelo OSI (capas física y de enlace de datos) para las conexiones de las redes de área local inalámbricas (WLAN). El estándar original especificaba la operación a 1 y 2 Mbps usando tres tecnologías diferentes, *Frequency Hopping Spread Spectrum* (FHSS), *Direct Sequence Spread Spectrum* (DSSS) e Infrarojos (IR). Actualmente puede operar a 54Mbps, en el caso particular del 802.11g. La principal ventaja que proporcionan los estándares recogido en la norma 802.11, aparte de las altas velocidades de transferencia, es la capacidad de

suministrar cobertura en un gran rango de distancia (capaz de alcanzar los 100 metros). Por otra parte, la desventaja fundamental de estas redes existe en el campo de la seguridad. Existen algunos programas capaces de capturar paquetes enviados y calcular la contraseña de la red, y de esta forma acceder a ella.

El caso de Bluetooth, norma que define un estándar global de comunicación inalámbrica orientado hacia la implementación de redes de área personal (PAN) con dispositivos como PDA, teléfonos móviles y otros pequeños dispositivos que quieran transmitir información (voz y datos) en un rango muy corto (máximo 10m). Las nuevas especificaciones Bluetooth pueden alcanzar velocidad de transmisión de hasta 3Mbps. En 1998 las empresas Ericsson, IBM, Intel, Nokia y Toshiba trabajaron para el establecimiento de la norma que luego fue acogida por IEEE en su norma 802.15. El nombre procede del rey vikingo Harald Blatan, conocido por los ingleses como Bluetooth, que reinó en Dinamarca y Noruega en la segunda mitad del siglo X y fue uno de los hombres más poderosos de Europa.

El problema que presentan estos dos estándares para convertirse en una tecnología aplicable a nivel de redes de sensores, es la complejidad del protocolo y la rigidez de la topología de la red, así como el alto consumo de potencia de los transceptores. Por su parte, el estándar IEEE 802.15.4 está encaminado hacia el desarrollo de redes inalámbricas de baja velocidad, bajo coste y bajo consumo de potencia (LR-WPAN). Este estándar está orientado a aplicaciones donde la velocidad de transferencia no es muy alta, permitiendo que los nodos de la red puedan ser alimentados mediante baterías y operar sin ser recargados durante años. Es por lo tanto el estándar que mejor se adapta a los requisitos impuestos para el desarrollo de redes de sensores. En el siguiente capítulo se analizan más en detalle las características de estos estándares, confeccionando una tabla comparativa que los resume.

Ya en 1998, un consorcio formado por más de 100 empresas, entre las que se encontraban Siemens, Motorola, Philips, etc., impulsó el estándar *HomeRF*, una especificación de redes inalámbricas para que dispositivos del hogar pudiesen conectarse entre sí. La idea de este estándar se basa en el teléfono inalámbrico digital mejorado (*Digital Enhanced Cordless Telephone*, DECT) que es un equivalente al estándar de los teléfonos móviles GSM. Transporta voz y datos por separado, al contrario que protocolos como Wi-Fi, que transporta la voz como una forma de datos. La especificación que desarrolló el grupo de trabajo *HomeRF* se denominó SWAP (*Shared Wireless Access Protocol*) y era abierta a la industria, permitiendo a los PCs, periféricos, teléfonos inalámbricos y otros dispositivos de consumo compartir y comunicar voz y datos, en y alrededor de la casa, sin la complicación y el gasto de nuevos cableados, en la banda de ISM de 2.4GHz. El grupo que desarrollaba el estándar *HomeRF* se disolvió en Enero de 2003, después de que se

popularizara el uso de las redes Wi-Fi (802.11b) y Microsoft incluyera soporte para Bluetooth en sus sistemas operativos.

Actualmente hay varias iniciativas en curso que buscan normalizar las redes WSN, algunas de ellas en fase de desarrollo. Una de las más conocidas es la norma ZigBee [10], que es una especificación inalámbrica de baja potencia, bajo coste y baja velocidad de transferencia de datos, destinada a electrodomésticos, juguetes, aplicaciones industriales y otras similares. Otra importante iniciativa es el estándar *WirelessHART* [26], publicado en septiembre de 2007, que se basa directamente en el protocolo *HART*¹, pero es independiente de la ruta de transferencia física. El protocolo *WirelessHART* usa una red de comunicaciones en malla para aplicaciones de automatización. Además de tener capacidad inalámbrica, mantiene la compatibilidad con los dispositivos HART existentes, a nivel de comandos y herramientas. Utiliza una banda de 2,4 GHz como medio de transferencia para varias tecnologías de radio, incluidos WLAN, Bluetooth y ZigBee. Con el fin de evitar colisiones en la banda de frecuencia de 2,4 GHz, *WirelessHART* lleva a cabo una búsqueda especial de los canales no utilizados en esta banda y comprueba las interferencias mutuas de las tecnologías de radio.

Una tercera iniciativa en marcha es la norma ISA-SP100 [27]. El comité ISA100 pertenece a la sociedad internacional ISA (*Setting the Standards for Automation*), el comité fue formado en el año 2005 para establecer un conjunto de normas que definan los procedimientos para la aplicación de sistemas inalámbricos en los campos de la automatización y el control. En lugar de normalizar todos los elementos del sistema, ISA-SP100 especifica sólo los niveles superiores de la pila, con varias implementaciones posibles a nivel inferior. 6LoWPAN [28] es otro intento de estandarización en el ámbito de las redes LoWPAN (*Low power Wireless Personal Area Networks*), llevado a cabo por el grupo de trabajo 6LoWPAN del IETF, cuyo objetivo principal es intentar proporcionar los mecanismos necesarios para que una red 802.15.4 pueda ser vista como un enlace IPv6. Para ello requiere de implementaciones en aspectos como la fragmentación de los paquetes, para que puedan ser encapsulados en tramas 802.15.4, el encaminamiento a través de la red de sensores, etc. Estas iniciativas se basan en el mismo estándar inalámbrico, IEEE 802.15.4, aunque en algunos casos hacen uso de diferentes estándares.

¹ El protocolo de comunicaciones HART (*Highway Addressable Remote Transducer Protocol*) es una implementación del bus de campo, un protocolo digital de automatización industrial. Este protocolo fue desarrollado por Rosemount Inc. para instrumentos de campo inteligentes, que viene del protocolo abierto HART desarrollado en 1986. Desde entonces, las capacidades del protocolo han tenido mejoras con sucesivas revisiones a la especificación.

Capítulo 3

ZigBee

3.1 ¿Qué es ZigBee?

ZigBee es un estándar de comunicaciones inalámbricas en el que se especifica un conjunto de protocolos de alto nivel. Así, ZigBee está basado en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (*Wireless Personal Area Network*, WPAN) en lo referente a la capa de acceso al medio (MAC) y la capa física (PHY), mientras que la alianza ZigBee establece la capa NWK (*Network Layer*) y la capa APS (*Application Layer*) para completar lo que se denomina como pila ZigBee.



El objetivo del estándar ZigBee es proporcionar una solución a las aplicaciones que requieren comunicaciones seguras con una baja tasa de envío de datos y una gran vida útil de sus baterías.

Por otro lado, ZigBee cuenta con la *ZigBee Alliance* que es una alianza, sin ánimo de lucro, de más de 100 empresas, que trabajan juntas con el objetivo de promover el desarrollo e implantación de un estándar inalámbrico fiable de bajo coste y bajo consumo de energía, para permitir así la interoperabilidad entre dispositivos fabricados por compañías diferentes [68]. La alianza ZigBee está regida por un conjunto de empresas promotoras, que componen el consejo de administración. Las 16 empresas que lo forman son Ember, ST Microelectronics, Philips, Schneider Electric,

Siemens, Itron, Mitsubishi Electric, Samsung, Honeywell, Texas Instruments, Freescale, Huawei, Reliant Energy, Motorola, Landis+Gyr y Tendril. Las actividades de la alianza se llevan a cabo a través de grupos de trabajo dedicados a un campo específico de la tecnología, entre los cuales hay un grupo para redes, un grupo para la seguridad, un grupo para los perfiles de las aplicaciones, etc. Estas actividades se llevan a cabo de forma abierta, permitiendo descargarse el estándar con los resultados, independientemente de que se sea miembro o no de la alianza. En la Figura 3.1 se muestra el logotipo de *ZigBee Alliance*.



Figura 3.1. Logotipo de ZigBee Alliance

3.2 Características

Las características más destacables y con las que ZigBee marca diferencias en relación con otras tecnologías inalámbricas son entre otras, su bajo consumo (los dispositivos pueden llegar a funcionar varios años alimentados con pilas), su fácil integración (consiguiendo bajos costes), y la posibilidad de usar una topología de malla (lo que permite a la red recuperarse autónomamente de problemas en la comunicación, aumentando así su confiabilidad) [31].

Otras características destacables de ZigBee son las siguientes:

- Velocidades de transmisión comprendidas entre 20 Kbps y 250 Kbps.
- Los rangos de cobertura son de 10 m a 75 m (aunque con amplificadores de potencia esta cobertura puede aumentarse considerablemente).
- Opera en las bandas libres ISM (*Industrial, Scientific & Medical*) de 2.4GHz (en todo el mundo), 868 MHz (Europa) y 915 MHz (Estados Unidos).

- Capacidad para soportar redes con alta densidad de nodos. Las redes disponen de un identificador único, lo que permite que coexistan diferentes redes en un mismo canal. Teóricamente pueden existir hasta 2^{14} (≈ 16000) redes diferentes en un mismo canal y cada red puede estar constituida por hasta 2^{16} ($\approx 65\ 000$) nodos [30].
- Acceso de canal mediante CSMA-CA (*Carrier sense multiple access with collision avoidance*).
- Es un protocolo de comunicación *multi-hop*, es decir, que se puede establecer comunicación entre dos nodos aún cuando éstos se encuentren fuera del rango de transmisión, siempre y cuando existan otros nodos intermedios que los interconecten, incrementándose de esta manera el área de cobertura de la red.
- A pesar de coexistir en la misma frecuencia con otro tipo de redes como Wi-Fi o Bluetooth, su operación no se ve afectada debido a su baja tasa de transmisión y a características propias del estándar IEEE 802.15.4 [29].
- Soporta diferentes tipos de topología, como estrella, punto a punto, o árbol, así como la topología malla (*mesh*).

3.3 Comparación con otros estándares

La diferencia entre ZigBee y otros estándares disponibles en el mercado radica fundamentalmente en que los estándares existentes están orientados a altas y medias tasas de transmisión, haciendo uso de un gran ancho de banda, con el inconveniente de la repercusión que esto tiene en el consumo de energía. En aplicaciones que no necesiten una elevada tasa de transferencia, el uso de este ancho de banda se vuelve innecesario. ZigBee está enfocado principalmente a aplicaciones que requieran baja tasa de transferencia y bajo consumo, cubriendo un vacío existente entre el resto de tecnologías inalámbricas. Además, cuenta con una conectividad más sofisticada que los anteriores sistemas inalámbricos, permitiendo una mayor fiabilidad y seguridad. Todo esto hace de ZigBee un estándar idóneo para un amplio mercado de aplicaciones en el ámbito industrial o de la domótica, además de muy adecuado para la mayoría de redes de sensores inalámbricos, donde características como la del consumo en reposo, o el número de dispositivos que puede albergar la red, son de capital importancia.

En la Tabla 3.1 se presenta una comparativa entre ZigBee y otros estándares disponibles en el mercado capaces de formar redes inalámbricas de área personal y que comparten la banda de 2,4 GHz, como son Bluetooth o Wi-Fi.

	Wi-fi	Bluetooth	ZigBee
Bandas de Frecuencias	2.4GHz	2.4GHz	2.4GHz
Tamaño de Pila	~ 1Mb	~ 1Mb	~ 20kb
Tasa de Transferencia	48Mbps	1Mbps	250kbps (2.4GHz) 40kbps (915MHz) 20kbps (868MHz)
Números de Canales	11 - - 14	79	16 (2.4GHz) 10 (915MHz) 1 (868MHz)
Tipos de Datos	Digital	Digital, Audio	Digital (Texto)
Rango de Nodos Internos	100m	10m - 100m	10m - 100m
Números de Dispositivos	32	8	255 / 65535
Requisitos de Alimentación	Media - Horas de Batería	Media - Días de Batería	Muy Baja - Años de Batería
Arquitecturas	Estrella	Estrella	Estrella, Árbol, Punto a Punto y Malla
Aplicaciones apropiadas	Conexiones a Internet en Edificios	PCs y Teléfonos	Monitorización, Control y Localización
Consumo de Potencia	400ma transmitiendo, 20ma en reposo	40ma transmitiendo, 0.2ma en reposo	30ma transmitiendo, 3µa en reposo
Precio	Costoso	Accesible	Bajo
Complejidad	Complejo	Complejo	Simple

Tabla 3.1. Comparativa entre las principales tecnologías inalámbricas

Aunque Bluetooth y ZigBee tienen mucho en común, se pueden observar diferencias notorias, como su tasa de transmisión, muy superior en el caso de Bluetooth (1 Mbps) o el número de dispositivos que puede formar una red, donde Bluetooth presenta una importante deficiencia respecto a ZigBee, ya que el protocolo sólo contempla redes con un máximo de 8 dispositivos, mientras que ZigBee puede llegar a millares de dispositivos. Otra diferencia destacable es el consumo, donde los estándares considerados mantienen en transmisión unos consumos similares, si bien en reposo, el consumo por parte de Bluetooth es casi 70 veces mayor que el consumo registrado en ZigBee. Esto es debido a que los dispositivos en redes Bluetooth deben proporcionar

información a la red periódicamente para mantener la sincronización. Estas diferencias son comprensibles, ya que Bluetooth está orientado a las comunicaciones entre dispositivos electrónicos y accesorios, como puede ser el caso de un ordenador y una impresora, o de comunicación entre teléfonos móviles.

Por otro lado se encuentran las redes Wi-Fi, basadas en las especificaciones IEEE 802.11, con la ventaja de que pueden transmitir una gran cantidad de datos, alcanzando una velocidad de 54 Mbps (aunque se está desarrollando un nuevo estándar que podría alcanzar los 108 Mbps). En su contra, presenta un consumo de corriente muy alto, no sólo en transmisión sino también en reposo, debido a que los dispositivos de estas redes deben permanecer activos casi de forma permanente. Esta desventaja hace inviable su uso en sistemas portátiles que requieran bajo consumo, pero sí muy adecuado para conexiones a Internet en redes de ordenadores con infraestructura fija o redes de ordenadores portátiles.

3.4 Tipos de dispositivos

En una red ZigBee se pueden identificar tres tipos diferentes de dispositivos, según su función en la red:

- *Coordinador ZigBee (ZigBee Coordinator, ZC)*: es el tipo de dispositivo más completo. Existe un coordinador por cada red, siendo el nodo que la inicializa, además de administrar los nodos, encargándose de controlar red y las rutas que deben seguir éstos para conectarse entre ellos, y almacenar información.
- *Router ZigBee (ZigBee Router, ZR)*: es un nodo que crea y mantiene información sobre la red con el fin de determinar la mejor ruta para interconectar nodos y encaminar los paquetes con información. Un dispositivo *router* ZigBee debe unirse a una red ZigBee antes de poder actuar como tal y retransmitir los paquetes de otros dispositivos *routers* ZigBee o de los dispositivos finales. El *router* ZigBee también ofrece un nivel de aplicación para la ejecución de código de usuario.
- *Dispositivo final (ZigBee End Device, ZED)*: actúa como nodo extremo de una red y posee la funcionalidad necesaria para comunicarse con su nodo padre (el coordinador o un *router* ZigBee), pero no puede transmitir información destinada a otros dispositivos. De esta forma, este tipo de nodos puede estar inactivo la mayor parte del tiempo, al no tener que realizar

funciones de encaminamiento, aumentando así la vida media de su batería. Un ZED tiene requisitos de memoria menores que un ZR o un ZC y es por tanto significativamente más barato.

Alternativamente se puede plantear una segunda clasificación para incurrir en el menor coste posible en los dispositivos. Esta clasificación está descrita por el estándar IEEE 802.15.4 y define dos tipos de dispositivos físicos:

- FFD (*Full Function Device*): también conocido como *nodo activo*. Son dispositivos capaces de organizar y coordinar el acceso al medio de otros dispositivos de la misma red gracias a la memoria adicional y a la capacidad de computar, pudiendo funcionar como coordinador o router o bien ser usado en dispositivos de red que actúen de interfaz con los usuarios. Suele requerir un consumo de energía superior a otros nodos, por lo que generalmente se conecta a la red eléctrica.
- RFD (*Reduced Functionality Device*): también conocido como *nodo pasivo*. Tiene capacidad y funcionalidad limitadas (especificada en el estándar) con el objetivo de lograr un bajo coste, bajo consumo de energía y una gran simplicidad. Sólo pueden asociarse a un FFD a la vez, y son generalmente alimentados mediante batería. Básicamente, son los sensores/actuadores de la red.

En la Figura 3.2 se muestra una red ZigBee, con los diferentes tipos de dispositivos que existen y la forma en que se relacionan entre sí.

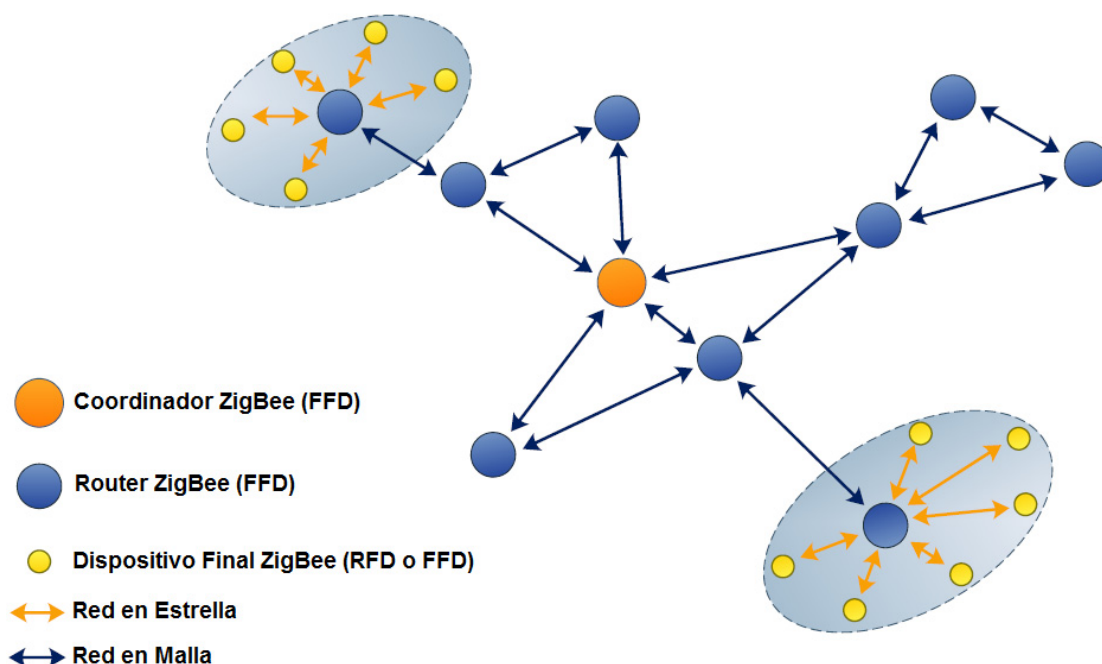


Figura 3.2. Estructura de red ZigBee

Una red ZigBee requiere de al menos un FFD que actuará como coordinador de la red, pudiendo ser los dispositivos terminales FFD o RFD, aunque son más recomendables los RFD con el fin de reducir el coste del sistema. Estos dispositivos pueden buscar redes disponibles, transferir datos en el momento en que su aplicación lo requiera, determinar cuándo existen datos pendientes, requerir datos del coordinador de red y permanecer inactivos por largos períodos de tiempo para reducir el consumo. Un RDF puede interactuar sólo con un FFD el cual posee suficientes recursos de sistema como para realizar funciones de encaminamiento dentro de la red.

3.5 Topologías de red

Las especificaciones del estándar ZigBee permiten tres topologías de red diferentes que pueden ser implementadas dependiendo de la aplicación; estrella, árbol y malla. En las tres topologías existe un nodo de red que asume el papel de coordinador central encargado de centralizar la adquisición y las rutas de comunicación entre dispositivos. En los siguientes apartados se describen estas tres topologías.

3.5.1 Topología en estrella

La comunicación en esta topología es centralizada y consta de un coordinador y uno o más dispositivos finales. En una red en estrella, todos los dispositivos finales se comunican únicamente con el coordinador. Si un dispositivo final desea comunicarse con otros dispositivos debe enviar su información al coordinador PAN, el cual transferirá esta información al dispositivo correspondiente. De este modo, el coordinador de la red actuaría como encaminador pasando por el todos los paquetes. Entre sus ventajas más destacables es que proporciona la posibilidad de desconectar nodos finales de la red sin que ésta se vea afectada, y como desventaja, que los dispositivos finales (nodos) sólo tendrán conexión a la red si están en el área de cobertura de su coordinador PAN. Las aplicaciones más comunes que utilizan este tipo de topología son la conexión entre un ordenador personal y dispositivos periféricos, domótica o en juguetes. En la Figura 3.3 se muestra la estructura de una topología en estrella.

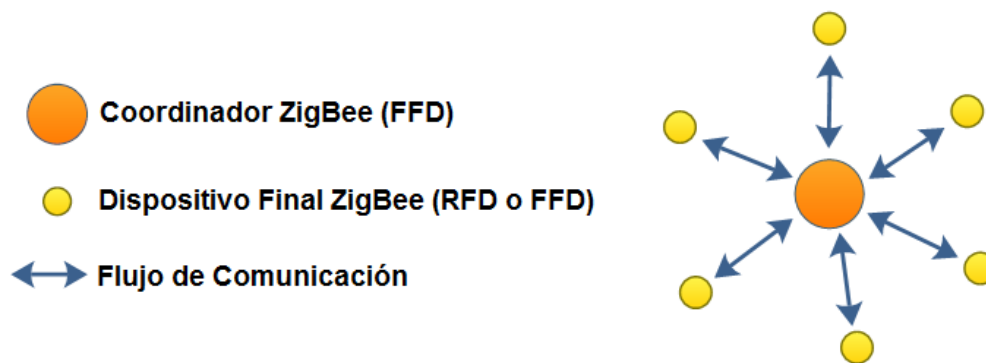


Figura 3.3. Topología de red en estrella

3.5.2 Topología en árbol

En esta configuración el coordinador representa la raíz de árbol y los dispositivos finales pueden unirse unos con otros mediante el coordinador o los diferentes nodos enrutadores. Mediante el uso de *routers* se obtienen dos ventajas. Una es el incremento del número de nodos que puedan formar parte de la red. La otra es la ampliación física del rango de alcance de la red. Con el ingreso de un *router*, un dispositivo final no necesita estar dentro de un radio determinado del coordinador. Todos los mensajes en la topología en árbol se encaminarán a lo largo del árbol. Como desventaja, en el caso de que un *router* se desconecte o deje de operar, los dispositivos finales (nodos) que estén conectados a la red mediante este elemento, también dejarán de funcionar. En la Figura 3.4 se muestra la estructura de una topología en árbol.

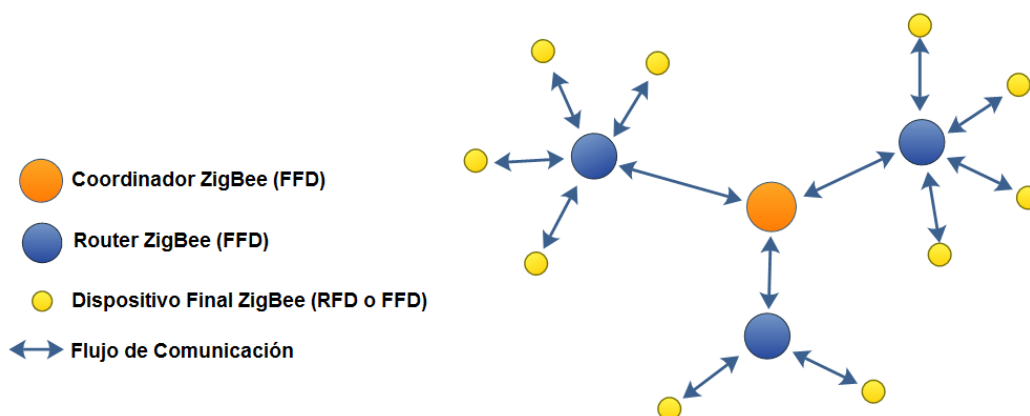


Figura 3.4. Topología de red en árbol

3.5.3 Topología en malla (*mesh*)

La topología más interesante (y una de los motivos del éxito de ZigBee) es la topología en malla. Esta topología permite que cualquier dispositivo pueda comunicarse con cualquier otro mientras ambos estén en el mismo área de cobertura o utilizando otros nodos para llegar al destino. Al uso de otros nodos como repetidores para enviar un mensaje de un dispositivo a otro de la red se le conoce como *multi-hop*. De esta forma, si en un momento dado un nodo de la ruta falla y se cae, la comunicación entre todos los demás nodos puede continuar debido a que es posible rehacer todas las rutas (la gestión de las rutas es tarea del coordinador). De esta forma se consiguen altos niveles de confiabilidad y escalabilidad al disponer de más de una ruta a través de la red.

Es posible encontrar aplicaciones industriales que obtienen beneficios operando con esta topología de red, tales como control y vigilancia, seguimiento de inventarios, agricultura inteligente, seguridad, etc. Una red en malla puede ser *ad-hoc*, auto-organizable y auto-reparable. En la Figura 3.5 se muestra la estructura de una topología en malla.

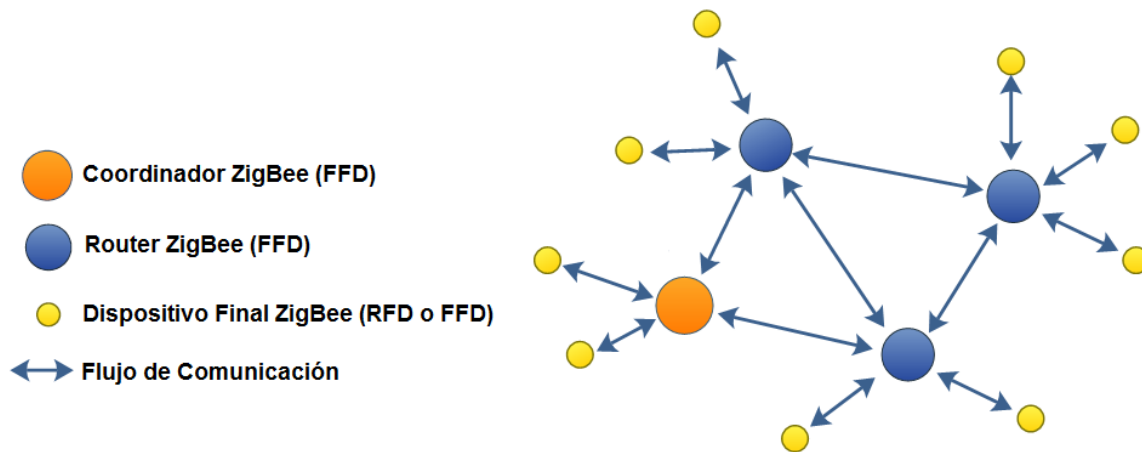


Figura 3.5. Topología de red en malla

3.6 Arquitectura

La arquitectura ZigBee, de manera similar al modelo OSI, está constituida por diferentes capas haciendo uso del modelo de referencia simplificado de cinco capas independientes unas de otras. Las capas inferiores de la arquitectura ZigBee están basadas en el estándar 802.15.4, que define las especificaciones de la capa física (*PHYsical layer*, PHY) y la subcapa de control de acceso al medio (*Medium Access Control*, MAC) para redes inalámbricas de área personal de baja transferencia de datos (LR-WPAN), redes destinadas a aplicaciones de un bajo consumo de energía y costes reducidos. La capa de red (NWK) y la subcapa de aplicación (APS) son especificadas por *ZigBee Alliance*. Por último, la capa de aplicación completará la arquitectura con la aplicación desarrollada por el usuario o el fabricante, en su caso. En la Figura 3.6 se muestran gráficamente las diferentes capas que conforman la arquitectura ZigBee.

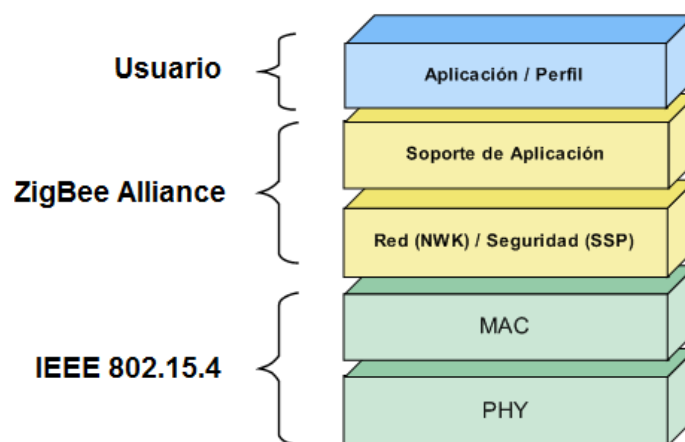


Figura 3.6. Capas de la arquitectura ZigBee

La capa física (PHY), que en conjunto con la capa de acceso al medio (MAC), brinda los servicios de transmisión de datos *punto a punto*, opera sobre las bandas ISM de uso no regulado, donde se definen hasta 16 canales en el rango de 2.4 GHz, cada una de ellas con un ancho de banda de 5 MHz. Se utilizan radios con un espectro de dispersión de secuencia directa, lográndose tasas de transmisión en el aire de hasta 250 Kbps en rangos que oscilan entre los 10 y 75 m, los cuales dependen bastante del entorno.

La capa de red (NWK) tiene como objetivo principal permitir el correcto uso del subnivel MAC y ofrecer una interfaz adecuada para su uso por parte de la capa de aplicación. En esta capa se brindan los métodos necesarios para: iniciar la red, unirse a la red, encaminar paquetes dirigidos a otros nodos en la red, proporcionar los medios para garantizar la entrega del paquete al destinatario final, filtrar paquetes recibidos, cifrarlos, y autentificarlos. Se debe tener en cuenta que el algoritmo de encaminamiento que se usa es el de malla, el cual se basa en el protocolo *Ad Hoc On-Demand Vector Routing* (AODV) [32]. Cuando esta capa se encuentra cumpliendo la función de unir o separar dispositivos a través del controlador de red, implementa mecanismos de seguridad, y encamina tramas a sus respectivos destinos. Además, la capa de red del controlador de red es responsable de crear una nueva red y asignar direcciones a los dispositivos de la misma. Es en esta capa donde se implementan las distintas topologías de red que ZigBee soporta (árbol, estrella y malla).

La siguiente capa es la de soporte a la aplicación, que es la responsable de mantener la función que cada nodo desempeña en la red, filtrar paquetes a nivel de aplicación, mantener la relación de grupos y dispositivos con los que la aplicación interactúa, y simplificar el envío de datos a los diferentes nodos de la red. La capa de Red y de soporte a la aplicación son definidas por *ZigBee Alliance*.

En el nivel conceptual más alto se encuentra la capa de aplicación, que no es otra cosa que la aplicación propiamente dicha, y de la que se encargan, por lo general, los fabricantes. Es en esta capa donde se encuentran los dispositivos ZDO (*ZigBee Device Objects*), encargados de definir el papel del dispositivo en la red (si opera como coordinador, *router* o dispositivo final). La subcapa APS y los objetos de aplicación son definidos por cada uno de los fabricantes.

Cada capa se comunica con sus capas subyacentes a través de una interfaz de datos y otra de control. Las capas superiores solicitan servicios a las capas inferiores, y éstas reportan sus resultados a las capas superiores. Además de las capas mencionadas, en la arquitectura ZigBee se

integran otro par de módulos: el *módulo de seguridad*, encargado de proveer los servicios para cifrar y autenticar los paquetes, y el *módulo de administración* del dispositivo ZigBee, encargado de administrar los recursos de red del dispositivo local, además de proporcionar a la aplicación funciones de administración remota de red.

3.7 IEEE 802.15.4

Como ya se comentó anteriormente, ZigBee está basado en el estándar 802.15.4 [25]. Este estándar define las especificaciones de la capa física y la subcapa de control de acceso al medio para redes inalámbricas de área personal de baja transmisión de datos. IEEE 802.15.4 fue creado por el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) y nace de la necesidad de dar solución a determinadas demandas en aplicaciones industriales y del hogar, con el fin de interconectar dispositivos electrónicos a través de la comunicación por radiofrecuencia, definiendo una red inalámbrica de área personal (PAN) de baja transferencia de datos (LR-WPAN, *Low-Rate Wireless Personal Area Network*), la cual sea económica, sencilla, con una larga duración de las baterías, y sin necesidad de transmitir grandes tasas de datos entre los sistemas de intercomunicación. En los siguientes apartados se comentan sus características más relevantes, además de proporcionar una breve descripción del funcionamiento de las capas que define.

3.7.1 Capa física

La capa física es la responsable de la transmisión y la recepción de datos en un canal de radio y de acuerdo con las técnicas de modulación y *spreading*. El estándar IEEE 802.15.4 define las funciones y las técnicas utilizadas para transmitir la información a través del medio de la capa física. Esta capa es responsable de las siguientes tareas:

- Activación y desactivación del radio transceptor: el radio transceptor puede operar en tres estados diferentes: transmitiendo, recibiendo o en modo sleeping.
- Detección de energía (ED): es una estimación de la potencia de la señal recibida dentro del ancho de banda del canal, siendo este valor analizado con respecto a un valor umbral predeterminado (umbral ED). Esta medida es usada como parte del algoritmo de selección de canal, y por CCA (*Clear Channel Assessment*) para determinar si el canal está libre u ocupado.

- Indicador de la calidad del enlace (*Link quality indicación*, LQI) para paquetes recibidos: determina la calidad de recepción de la información. Para la medición puede utilizar: una relación de señal-ruido, el detector de energía (ED), o una combinación de ambas.
- Asignación de canales (Clear Channel Assessment, CCA): mediante el CCA, la capa física proporciona el estado de los canales, es decir, si están libres u ocupados. El CCA tiene tres modos para la detección del estado del canal:
 - *Modo 1*: obtendrá el resultado de la ocupación del canal si la energía detectada se encuentra por encima del valor umbral ED.
 - *Modo 2*: se basa en la comprobación de la ocupación del canal utilizando las técnicas de modulación y *spreading* establecidas por el IEEE 802.15.4.
 - *Modo 3*: este modo es una combinación de los dos anteriores, detectando una señal con las técnicas de modulación y *spreading* establecidas por IEEE 802.15.4 y comprobando posteriormente si está por encima o por debajo del valor umbral ED.
- Selección de la frecuencia del canal: la capa física se encargará de sintonizar la frecuencia dentro de los 27 canales que define el estándar IEEE 802.15.4.
- Transmisión y recepción de datos: utilizando las técnicas y especificaciones vistas para la capa física, ésta debe realizar la transmisión y recepción de los datos de una manera fiable y segura.

3.7.1.1 Frecuencia de operación y velocidad de transmisión

En la capa física, el estándar IEEE 802.15.4 define 27 canales de comunicación en tres rangos de frecuencia distintos dentro de la banda ISM: 16 Canales en la banda de 2.4GHz, 10 Canales en la banda de 915 MHz, y 1 Canal en la banda 868 MHz, como se muestra en la Figura 3.7.

A diferencia del canal en la banda 868 MHz, que se usa exclusivamente en Europa a una tasa de datos de 20 kbps, y de la banda de los 915 MHz, usada en EEUU a una tasa de 40 kbps, la banda de 2.4 GHz está disponible en todo el mundo y opera a la velocidad nominal de 250 kbps.

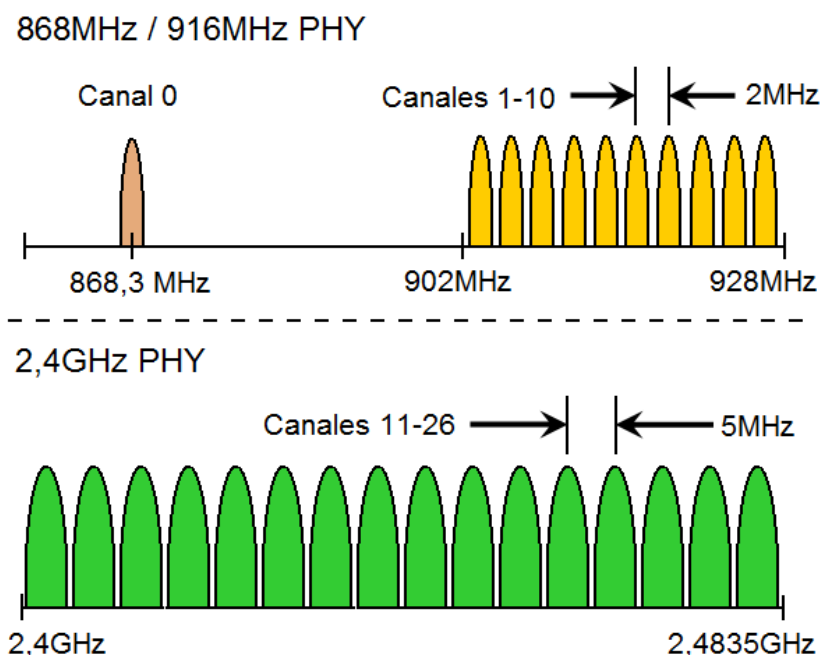


Figura 3.7. Canales disponibles en la capa física del 802.15.4.1

El estándar utiliza las modulaciones BPSK (Binary Phase Shift Keying) y O-QPSK (Offset Quadrature Phase Shift Keying). La primera es usada en el canal 868 MHz con una tasa de chip de 300 por segundo y en la banda de los 915 MHz con una tasa de chip de 600 por segundo, mientras que la segunda modulación se emplea en la banda de 2450 MHz. Opcionalmente se pueden usar también las modulaciones ASK (Amplitude Shift Keying) y O-QPSK para las bandas 868/915 MHz. En la Tabla 3.2 se muestra un resumen de las características de la capa física.

Física (MHz)	Banda de frecuencia (MHz)	Velocidad de Chip (kchip/s)	Modulación	Velocidad de bit (kb/s)	Velocidad de símbolo (ksímbolos/s)	Símbolos
868/915	868 - 868.6	300	BPSK	20	20	Binario
	902 - 928	600	BPSK	40	40	Binario
868/915 (Opcional)	868 - 868.6	400	ASK	250	12.5	20-bits PSSS
	902 - 928	1600	ASK	250	50	5-bits PSSS
868/915 (Opcional)	868 - 868.6	400	O-QPSK	100	25	16-ary Ortogonales
	902 - 928	1000	O-QPSK	250	62.5	16-ary Ortogonales
2450	2400 - 2483.5	2000	O-QPSK	250	62.5	16-ary Ortogonales

Tabla 3.2. Características de modulación de la capa física

3.7.1.2 Modulación

El estándar 802.15.4 emplea dos tipos de modulación diferentes, dependiendo de la frecuencia utilizada. Así, si se emplean las frecuencias de 915 MHz y 868 MHz, la señal es modulada con BPSK, si bien opcionalmente también se puede utilizar ASK, mientras que a la frecuencia 2.4 GHz se emplea una técnica de modulación O-QPSK. Además el estándar IEEE 802.15.4 utiliza la técnica DSSS (*Direct Sequence Spread Spectrum*) para transmitir la información a través del medio.

Las dos modulaciones usadas son del tipo PSK (*Phase Shift Keying*), es decir, modulaciones digitales que se basan en ir cambiando la fase de la señal en función de los valores de los bits de información. En la Figura 3.8 se puede observar un ejemplo de modulación PSK.

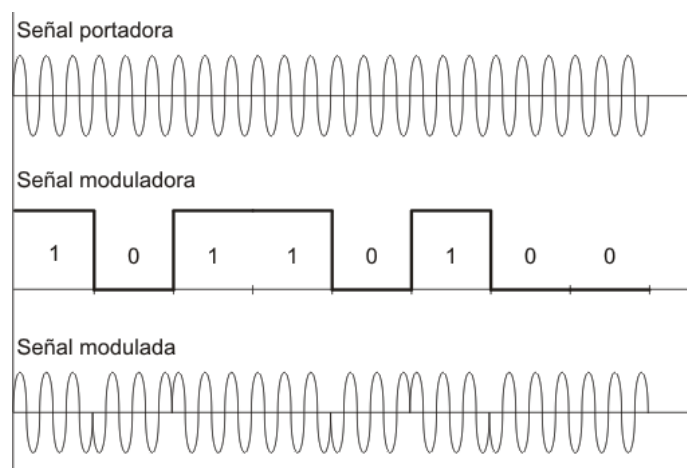


Figura 3.8. Modulación PSK

En la modulación BPSK se definen dos fases separadas 180° , donde una indica el estado bajo “0” y la otra indica el estado alto “1”. En la Figura 3.9 se puede observar la constelación de esta modulación.

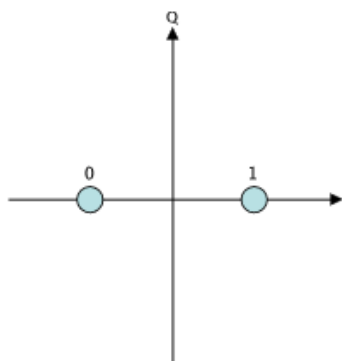


Figura 3.9. Constelación de la modulación BPSK

Cuanto mayor sea la separación entre los dos estados, más fácil es para el receptor distinguir entre ellos la información válida en presencia de ruido e interferencias. En la modulación BPSK se utiliza 1 bit por símbolo.

La modulación O-QPSK consiste en una modulación QPSK pero introduciendo un retardo entre las componentes de fase y cuadratura, con lo que en la modulación QPSK la codificación se realiza por cambio de fase en cuadratura. La señal de información se divide en 4 fases posibles y a cada una de estas fases se le asigna un estado representado por 2 bits. Típicamente, las cuatro fases elegidas para QPSK son ± 45 grados y ± 135 grados. En la Figura 3.10 se puede ver la constelación de la modulación QPSK y los 2 bits asignados a cada una de los estados.

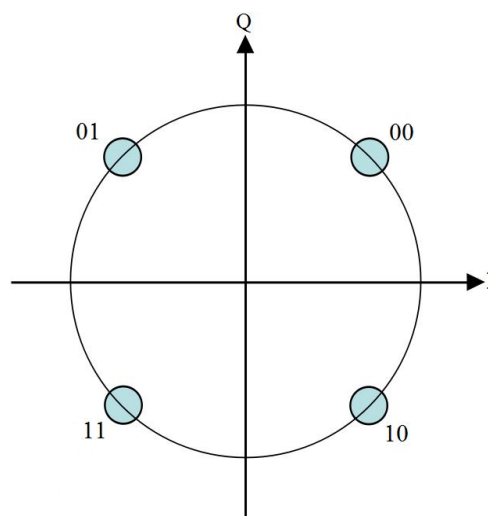


Figura 3.10. Constelación de la modulación QPSK (2 bits por símbolo)

La modulación O-QPSK ofrece cierta ventaja sobre la QPSK, ya que en la QPSK real, la envolvente no es constante debido a la presencia de filtros que le afecta, fundamentalmente, en los cambios de fase. El efecto es una interferencia inter-simbólica y un aumento del ancho de banda, mientras que con la modulación O-QPSK se obtiene una disminución de su interferencia inter-simbólica y de su ancho de banda.

En términos de eficiencia (energía requerida por *bit*), la modulación ortogonal mejora su funcionamiento en 2 dB con respecto a BPSK. La modulación QPSK, comparada con BPSK, ofrece el doble de bits de información por portadora de cambio de fase, lo cual hace que tenga el doble de eficiencia en ancho de banda y que sea más adecuada que la BPSK para transmitir datos a mayor velocidad. Sin embargo, en términos de sensibilidad de recepción, a las frecuencias 868MHz y 915 MHz se obtienen mejoras de 6-8 dB debido a que presentan velocidades de transmisión menores.

3.7.1.3 Técnica de ensanchado de espectro DSSS

El espectro ensanchado es una técnica por la cual la señal transmitida se ensancha a lo largo de un amplio ancho de banda, mucho más amplio que el mínimo ancho de banda requerido para transmitir la información que se desea enviar. No se puede decir que las comunicaciones mediante espectro ensanchado sean medios eficientes de utilización del ancho de banda. Sin embargo, presentan un rendimiento elevado al combinarlas con otros sistemas existentes. La señal de espectro ensanchado, una vez ensanchada, puede coexistir con señales en banda estrecha, ya que sólo les aportan un pequeño incremento en el ruido. En lo que se refiere al receptor de espectro ensanchado, éste no se ve afectado por las señales de banda estrecha, ya que está escuchando un ancho de banda mucho más amplio gracias a una secuencia de código preestablecido.

Como se comentó anteriormente, esta técnica no hace el uso más eficiente posible del ancho de banda, pero a cambio ofrece ciertas ventajas que en la actualidad han cobrado mucha importancia, entre las que destacan las siguientes:

- Baja probabilidad de ser interceptada debido al ensanchamiento del espectro, lo que dificulta la captación de las señales transmitidas por parte de un receptor ajeno a la comunicación.
- Alta inmunidad frente a interferencias intencionadas (más conocidas con el nombre de *jamming*).
- Alta inmunidad frente a interferencias de señales multitrayecto y al uso de un mismo canal por dos o más usuarios.
- Posibilidad de acceso múltiple aleatorio (CDMA), con lo cual es posible tener varios usuarios cursando comunicaciones independientes, utilizando el mismo canal.
- Privacidad en las comunicaciones.

Existen varias técnicas de modulación de espectro ensanchado que se diferencian en el modo de ensanchamiento del espectro y en el tipo de modulación que emplean. En el caso particular de este PFC, el estándar IEEE 802.15.4 emplea la técnica de espectro ensanchado por secuencia directa (*Direct Sequence Spread Spectrum*, DSSS). Esta técnica genera un patrón de bits redundante para cada uno de los bits que componen la señal. A cada bit del patrón, se le denomina *chip* y cuanto mayor es la cantidad de chips, mayor será su resistencia a la interferencia. La secuencia de bits utilizada para modular los bits se conoce como secuencia de Barker (también llamado código de dispersión o *PseudoNoise*) [33]. Es una secuencia rápida diseñada para que aparezca

aproximadamente la misma cantidad de 1 que de 0. En la Figura 3.11 se observa cómo se utiliza la secuencia de Barker para codificar la señal original a transmitir y cómo afecta a su espectro.

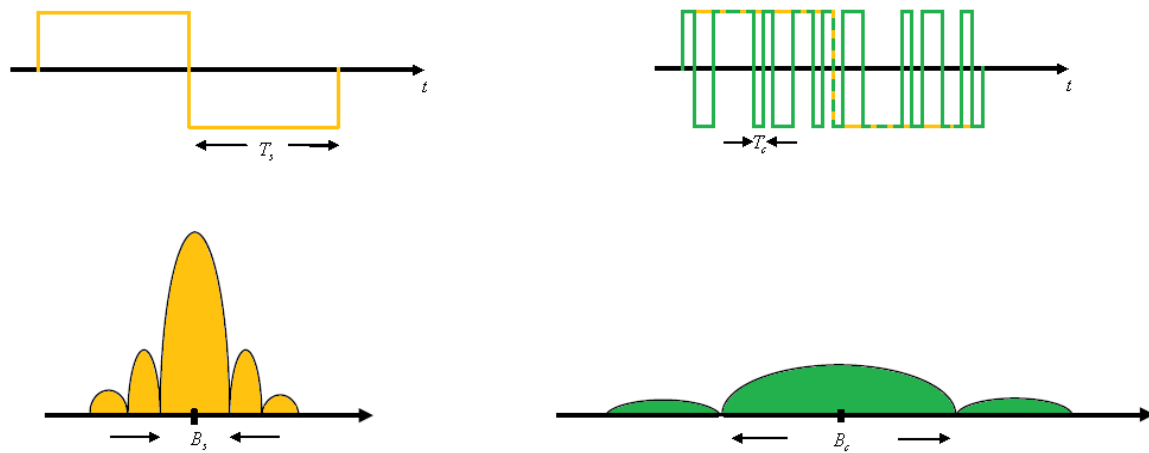


Figura 3.11. Codificación de Barker

Únicamente aquellos receptores a los que el emisor haya enviado previamente la secuencia patrón de bits utilizada en el modulador, podrán recomponer la señal original. Además, al sustituir cada bit de datos a transmitir por la secuencia patrón equivalente, aunque parte de la señal de transmisión se vea afectada por interferencias, el receptor aún puede reconstruir fácilmente la información a partir de la señal recibida. Esta técnica aumenta la sensibilidad de recepción en varios dBm.

Las señales de espectro ensanchado se demodulan en dos fases; en una primera fase se elimina la modulación en espectro, mientras que en un segundo paso se demodula la señal. Al proceso de desensanchado de una señal se le conoce como correlación. Este proceso se consigue mediante la sincronización adecuada de la secuencia pseudoaleatoria ensanchadora entre el transmisor y receptor. La sincronización es el aspecto más difícil que tiene que resolver el receptor. Al asignar una secuencia pseudoaleatoria dada a un determinado receptor, la información se le debe direccionar de forma distinta con respecto a los otros receptores a los que se les ha asignado una secuencia diferente. Las secuencias también pueden escogerse para minimizar las interferencias entre grupos de receptores al elegir los que tengan una correlación cruzada baja [34]. En la Figura 3.12 se representa cómo se minimizan las interferencias al desensanchar la señal.

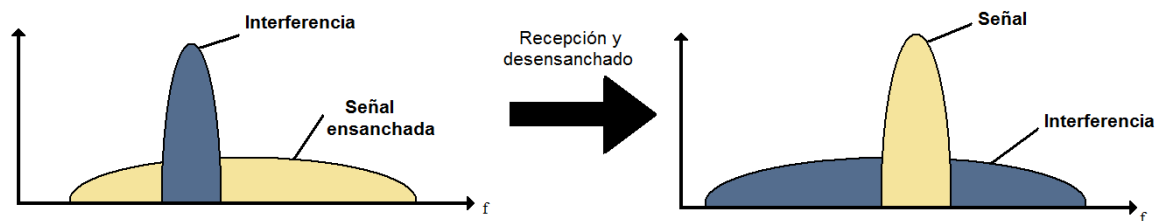


Figura 3.12. Efecto en el espectro al desensanchar

En la actualidad esta técnica de modulación es muy usada, ya que el aumento de redes y sistemas inalámbricos hace que aspectos como la inmunidad frente a interferencias o la confidencialidad de las comunicaciones hayan cobrado mucha importancia, aún no haciendo un uso óptimo del ancho de banda.

3.7.2 Capa MAC

La subcapa MAC, también definida por el estándar IEEE 802.15.4, se encarga de gestionar los accesos físicos al canal de radio, proveyendo una interfaz entre la capa física y las capas superiores, y siendo responsable de las siguientes tareas:

- Asociación/disociación de la PAN: cuando un dispositivo necesita unirse a una PAN, primero debe asociarse a un coordinador para que éste le brinde información sobre las características de funcionamiento de la PAN, y a su vez sea reconocido por otros dispositivos como miembro de la red. De manera similar, cuando un dispositivo ya no requiere transmitir o recibir datos de otros dispositivos de la red, se puede disociar del coordinador.
- Reconocimientos de entrega de trama (ACK, *Acknowledgement*): mediante los ACK se puede garantizar la entrega de información de un dispositivo a otro, ya que ofrece un mecanismo que permite confirmar que un mensaje ha sido recibido correctamente.
- Uso del mecanismo CSMA-CA para acceder al canal: la subcapa MAC ofrece mecanismos que facilitan el acceso al medio físico. Para evitar problemas de colisiones de información que puedan ocasionar pérdida de datos, el estándar IEEE 802.15.4 usa el algoritmo CSMA-CA (*Carrier Sense, Multiple Access/Collision Avoidance*), que verifica la disponibilidad del canal antes de transmitir, evitando así colisiones con otros transmisores y garantizando confiabilidad y seguridad.

- Validación de trama: con el fin de detectar bits erróneos y proteger cada trama, la subcapa MAC aplica un mecanismo de comprobación de trama, denominado FCS (*Check Sequence*). El FCS es generado a partir de un código estándar de control cíclico redundante del ITU-T (*Internacional Telecommunication Union*) de 16-bits.
- Control de garantía de ranuras de tiempo (GTS, *Guaranteed Time Slot*): GTS permite garantizar la consistencia de los datos transmitidos, así como controlar que los dispositivos transmitan cada cierto periodo. Dependiendo de la configuración de la PAN, estos tiempos o ranuras de tiempos serán asignados sólo por el coordinador de la PAN y deberán utilizarse únicamente para las comunicaciones entre el coordinador de la PAN y los dispositivos, lo cual evita que las señales transmitidas por dos o más dispositivos sufran colisiones debido a un desorden en el control de la transmisión.
- Control de guías (*Beacon*) y exploración del canal: el estándar IEEE 802.15.4 ofrece la posibilidad de localizar a los dispositivos miembros de una red a través de guías denominadas *beacon* o a través de exploraciones del canal físico. Las balizas se utilizan además para sincronizar los dispositivos o identificar las PAN. Cabe señalar que las exploraciones del canal también se usan para encontrar dispositivos que por alguna circunstancia han quedado disociados de cualquier coordinador en calidad de huérfanos.

Las especificaciones de la subcapa MAC definidas en el estándar IEEE 802.15.4 se caracterizan por su baja complejidad en comparación con otros estándares, como puede ser el estándar 802.15.1 (Bluetooth), cuyo administrador de servicios MAC tiene alrededor de 131 primitivas en 32 eventos, cantidad muy superior a las 26 primitivas con las que cuenta el estándar IEEE 802.15.4 en su MAC. De esta manera se consigue una MAC muy simple, haciéndola muy versátil. Otra característica es su robustez, ya que emplea diversos mecanismos, como son las tramas de reconocimiento ACK, la verificación de datos, o el algoritmo CSMA-CA, para así garantizar las transmisiones de datos.

3.7.2.1 Estructura de la trama MAC

El formato general de las tramas MAC está diseñado de modo que refleje la simplicidad y flexibilidad del protocolo, facilitando la adaptación a las necesidades de las diferentes aplicaciones con diversas topologías de red.

A la trama de la subcapa MAC se le denomina unidad de datos del protocolo MAC (MPDU, *MAC Protocol Data Unit*) y consta de tres partes:

- Cabecera de la MAC (MAC Header, MHR).
- Unidad de servicio de datos MAC (MAC Service Data Unit, MSDU).
- Pie de la MAC (MAC Footer, MFR).

A su vez, el estándar IEEE 802.15.4 define cuatro tipos de tramas MAC:

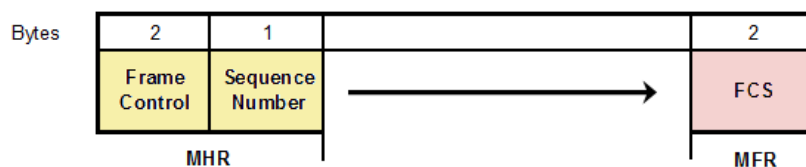
- *Data Frame*: la subcapa MAC utiliza la trama de datos para la transferencia de información y su formato se muestra en la Figura 3.13.

Bytes	2	1	4-20				Variable	2
	Frame Control	Sequence Number	Destination PAN Identifier	Destination Address	Source PAN Identifier	Sources Address	Payload	FCS
			Address Info					
			MHR				MSDU	MFR

Figura 3.13. Trama de datos de la subcapa MAC

La trama de datos está formada por los siguientes campos:

- *Frame Control (MHR)*: Contiene información que define el tipo de trama que se está transmitiendo.
 - *Sequence Number (MHR)*: Especifica el identificador de secuencia de una trama, siendo único para cada trama.
 - *Destination PAN Identifier (MHR)*: Especifica el identificador único de la PAN a la que va dirigida la trama.
 - *Destination Address (MHR)*: Especifica la dirección del dispositivo a la que va dirigida la trama.
 - *Source PAN Identifier (MHR)*: Es el identificador único de la PAN que origina la trama.
 - *Sources Address (MHR)*: Especifica la dirección del dispositivo que origina la trama.
 - *Payload (MSDU)*: Es de longitud variable; y contiene información según el tipo de trama. Contienen información proveniente de las capas superiores.
 - *FCS (MFR)*: Este campo contiene el CRC (*Cyclic Redundancy Check*).
- *Acknowledgment Frame*: la trama de reconocimiento, o también llamada de acuse de recibo, es enviada por la subcapa MAC para confirmar la recepción satisfactoria. El formato de esta trama se muestra en la Figura 3.14.

Figura 3.14. Trama *acknowledgment* de la subcapa MAC

La trama de reconocimiento (ACK) está formada por los siguientes campos:

- *Frame Control (MHR)*: Contiene información que define el tipo de trama que se está transmitiendo.
- *Sequence Number (MHR)*: Contiene el valor del número de secuencia de la trama que ha sido recibida por el dispositivo.
- *FCS (MFR)*: Este campo contiene el CRC (*Cyclic Redundancy Check*).

La trama ACK no necesita ningún tipo de campo de direccionamiento. Esto es debido a que el estándar define el envío del ACK como un evento síncrono, de manera que un emisor envía una trama de datos a un destino determinado y automáticamente permanece a la escucha para la recepción de un ACK con el número de secuencia de esa trama determinada. De esta manera, el ACK se difunde a todos los nodos, pero sólo lo procesará el remitente de la trama de datos.

- *MAC Command Frame*: la trama de comandos MAC es generada por la subcapa MAC y se encarga de gestionar el control de entidad MAC. El formato de esta trama se muestra en la Figura 3.15.

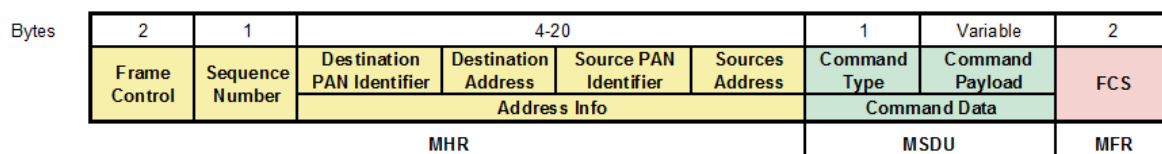


Figura 3.15 Trama de comando de la subcapa MAC

La cabecera (*MAC Header*) y el pie (*MAC footer*) de la MAC son los mismos presentados en el formato de la trama de datos. Los campos relevantes de la trama de comando MAC se encuentran en la Unidad de Servicio de Datos MAC (*MAC Service Data Unit, MSDU*) y está formada por dos campos:

- *Command Type (MSDU)*: Este campo indica el tipo de comando que se ha generado, mostrándose en la Tabla 3.3 los tipos existentes.
- *Command payload (MSDU)*: Contiene información específica del tipo de comando en uso, siendo su longitud variable en función del comando utilizado.

Identificador de comando	Tipo de Comando
1	Solicitud de Asociación
2	Respuesta de Asociación
3	Notificación de Disociación
4	Petición de datos
5	Conflicto en la notificación del Identificador de Red de PAN ID
6	Notificación de huérfano
7	Petición de trama Beacon
8	Realineación del coordinador
9-255	Reservado

Tabla 3.3. Tipos de comandos de la subcapa MAC

- *Beacon Frame*: la trama *beacon* es habilitada por los nodos coordinadores y su objetivo es localizar a los dispositivos ZigBee y unirlos a la red, además de informar a los dispositivos asociados a él, de la existencia de datos cuya transmisión esté pendiente, para que en un momento apropiado puedan reclamarlos, si es que les pertenecen. El formato de una *trama beacon* se muestra en la Figura 3.16.

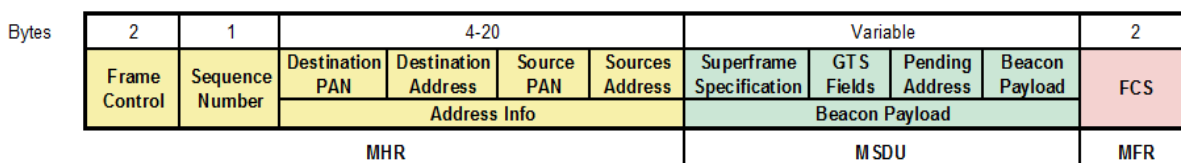


Figura 3.16. Trama beacon de la subcapa MAC

En la unidad de servicio de datos MAC (*MAC Service Data Unit, MSDU*) se encuentran los siguientes campos:

- *Superframe Specification (MSDU)*: Este campo especifica parámetros de la *superframe*².

² Trama especial empleada en el modo de acceso al medio CSMA/CA ranurado, permitiendo el uso de slots GTS y diseñado para aplicaciones en las que se deba garantizar una determinada latencia.

- *GTS Fields (MSDU)*: Contiene información acerca de los GTSS asignados por el nodo coordinador.
- *Pending Address (MSDU)*: Contiene información relativa a los dispositivos que tienen datos pendientes en el coordinador.
- *Beacon Payload (MSDU)*: Contiene información proveniente de las capas superiores.

La trama MAC se ensambla posteriormente en el paquete de la capa física, formando la Unidad de Datos del Protocolo Phy (PPDU, *Phy Protocol Data Unit*) representada en la Figura 3.17.

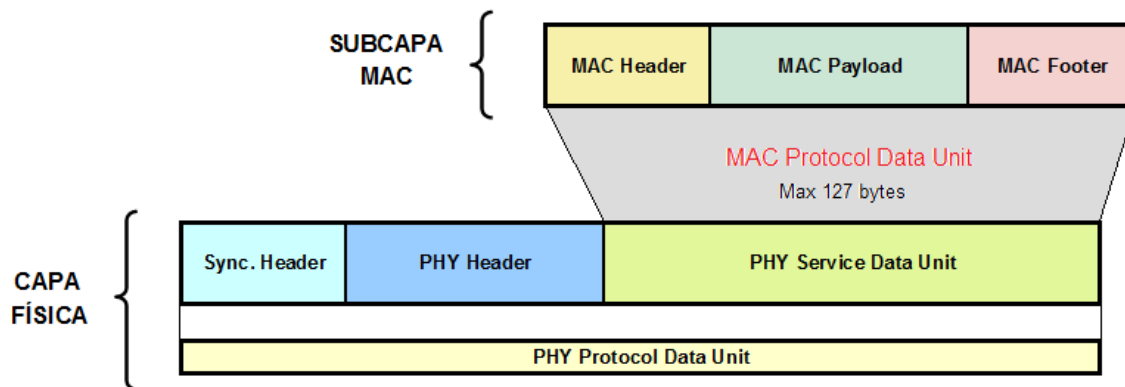


Figura 3.17. Trama PPDU

3.7.2.2 Acceso al Canal

En el estándar IEEE 802.15.4 se indican dos mecanismos para acceder al medio:

- El CSMA-CA no ranurado, usado en el modo de operación *beacon-no habilitado*
- El CSMA-CA ranurado, usado en el modo de operación *beacon-habilitado*

El modo *beacon-no habilitado* se define para las redes *peer to peer* (P2P) donde existe comunicación entre todos los nodos sin la necesidad de intervención de un coordinador que los sincronice. El acceso al canal en este modo se lleva a cabo por medio de un mecanismo CSMA-CA no ranurado, en el que cada dispositivo transmite en el momento que es necesario sin esperar la pauta (guía) de un PAN coordinador.

El modo *beacon-habilitado* está definido para redes en las que existe uno o varios coordinadores en la red, encargados de transmitir *beacons* cada cierto tiempo para sincronizar los dispositivos que se encuentran dentro de su red, permitiendo en caso de necesidad, una latencia mínima para aquellos

dispositivos que necesiten tener este parámetro garantizado. Con esta finalidad se emplea una estructura especial denominada *superframe*, la cual se implementa mediante una portadora dividida en intervalos de tiempo ofreciendo una porción en la cual se compita para acceder al medio y otra en la que las transmisiones sean dedicadas.

3.7.2.3 Estructura de la *Superframe*

El coordinador de una PAN puede limitar opcionalmente su tiempo de canal usando una estructura de *superframe*. La *superframe* o supertrama está limitada por dos tramas *beacon* y opcionalmente puede tener un período activo y uno inactivo como muestra la Figura 3.18.

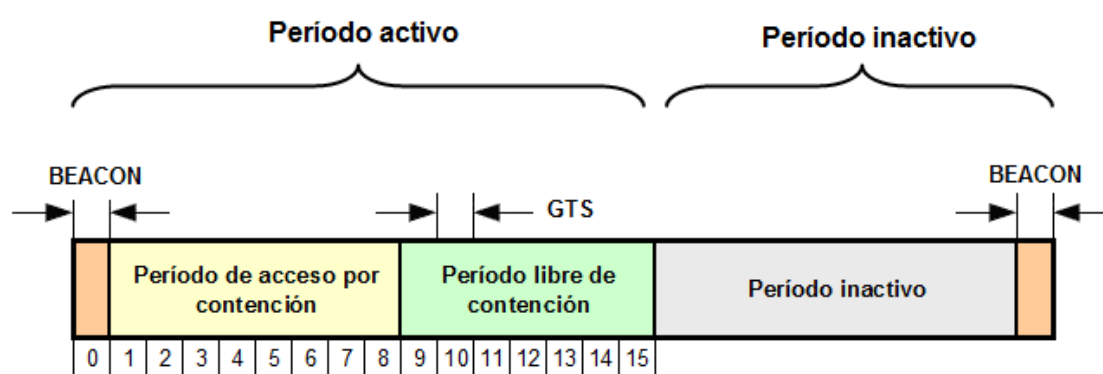


Figura 3.18. Estructura de una *Superframe*

La duración del tiempo del periodo activo puede estar comprendida entre 15 ms y 245s (con 2,45 GHz), e independientemente de la duración de cada *superframe*, siempre está dividida en 16 slots, de igual duración. La parte activa de la *superframe* está compuesta por:

- *Beacon*.
- Periodo de acceso por contención (CAP, *Contention Access Period*).
- Periodo libre de contención (CFP, *Contention Free Period*).

La trama *beacon* es transmitida sin necesidad de CSMA y ocupará el slot 0. Además contiene información referente al formato de la *superframe*. El CAP comenzará inmediatamente después de la trama *beacon* y terminará antes del inicio del CFP (si éste existe). Durante el CAP, todas las tramas se transmitirán utilizando el mecanismo CSMA-CA (en su versión ranurada) para acceder al canal, y sólo en este periodo se podrán transmitir las tramas de comandos de la MAC. Un dispositivo que transmita dentro del CAP se asegurará de que su transacción dure un periodo

completo IFS³ (es decir, incluyendo la recepción de cualquier reconocimiento) antes del final del CAP. Si esto no es posible, y ya se ha llegado al final del CAP, debe postergar su transmisión hasta el CAP del próximo *superframe*.

El CFP consiste en ranuras de tiempo garantizado (GTS, *Guaranteed Time Slot*) que son asignadas por el coordinador PAN para aplicaciones que requieren baja latencia, un ancho de banda específico y QoS (*Quality of Service*). El coordinador PAN puede asignar hasta un máximo de 7 GTS, pudiendo ocupar cada GTS sólo un slot. Los GTS se usan únicamente para comunicaciones entre el coordinador PAN y un dispositivo. El CFP, si está presente, se extenderá hasta el final de la parte activa de la *superframe* y ninguna de las transmisiones dentro del CFP utilizará el mecanismo CSMA-CA para tener acceso al canal. Un dispositivo que transmite en el CFP se asegurará de que sus transmisiones sean de un periodo completo IFS antes del final de su GTS.

En ambas configuraciones (solamente CAP o CAP/CFP), la *superframe* puede tener un periodo inactivo durante el cual el coordinador PAN no interactúa con los dispositivos y entra en un modo de bajo consumo de energía (*sleep*). En estos periodos de inactividad los dispositivos ahorran energía, lo que permite extender su tiempo de vida.

3.7.2.4 Algoritmo CSMA-CA

El estándar IEEE 802.14.5 utiliza el algoritmo CSMA-CA para la transmisión de las tramas de datos o de comando de la MAC. La finalidad es evitar colisiones, para lo cual utiliza un mecanismo de prevención de colisiones basado en mensajes recíprocos de acuse de recibo que el transmisor y receptor intercambian.

De acuerdo al algoritmo de acceso CSMA-CA, una estación que está preparada para enviar una trama debe escuchar la red antes de hacerlo. Si la red está ocupada, la transmisión se pospone. Sin embargo, si el medio permanece libre durante un cierto período de tiempo, llamado DIFS (*Distributed Inter Frame Space*), la estación transmisora realiza un intento de envío. El tiempo que tarda en transmitir la trama después del DIFS se denomina *ventana de contención*. La *ventana de contención* está ranurada usando unidades del tiempo llamados periodos *backoff*, que dependen de la capa física y de si están habilitados los *beacon*. Si estuviese la red ocupada, volvería a esperar que el medio estuviese libre y después del DIFS, la estación esperaría un numero de *backoff* aleatorio, antes de volver a intentar transmitir.

³ El período IFS (Inter Frame Spacing) define el tiempo que separa dos tramas consecutivas, debido a que la subcapa MAC necesita una cantidad finita de tiempo para procesar los datos recibidos por la PHY.

Una vez que la estación transmisora ha terminado de transferir su trama, espera una confirmación (ACK) de la estación receptora que permite confirmar que la recepción ha sido satisfactoria y que la trama recibida no contiene ningún error. La trama ACK debe ser enviada en un intervalo de tiempo denominado *SIFS* (*Short Inter Frame Space*, menor que el DIFS) después de que haya sido recibida la trama. Pasado ese tiempo, si no se ha recibido la trama ACK, la estación emisora debe considerar que ha habido un error en la transmisión o que se ha producido una colisión.

En la Figura 3.19 se muestra la ventana de contención y los tiempos de espera en la transmisión de una trama mediante CSMA-CA.

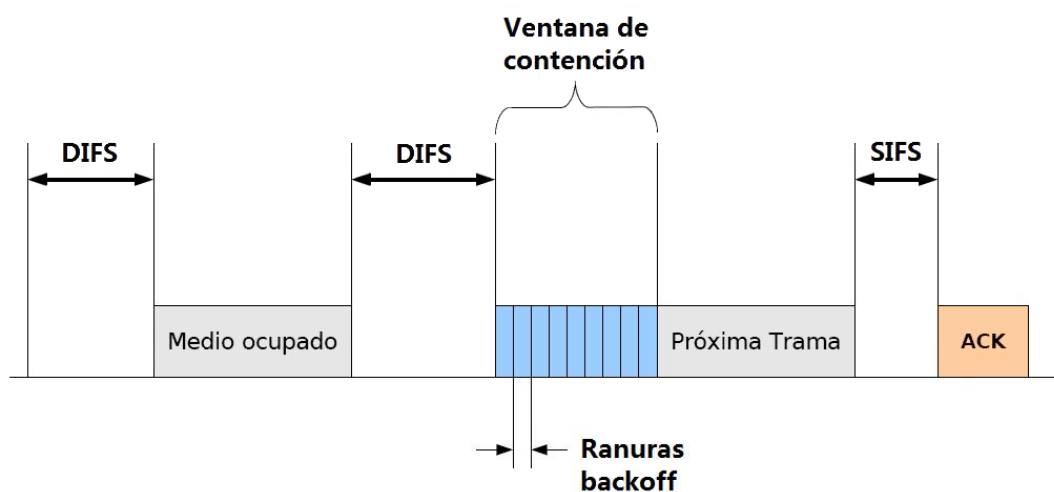


Figura 3.19. Ventana de contención y tiempo de espera

Si los *beacon* están habilitados en la PAN, la subcapa empleará la versión ranurada del algoritmo CSMA-CA, mientras que si los *beacon* no están habilitados, se empleará la versión no ranurada del algoritmo.

En el CSMA-CA ranurado, los límites del periodo *backoff* de cada dispositivo en la PAN se alinearán con los límites de la ranura de la *superframe* del coordinador de la PAN; es decir, el comienzo del primer periodo de *backoff* de cada dispositivo se alinea con el comienzo de la transmisión del *beacon*. En el CSMA-CA ranurado, la subcapa MAC se asegurará de que la PHY comienza todas sus transmisiones en el límite de un periodo *backoff*.

En el CSMA-CA no ranurado, los periodos *backoff* de un dispositivo no se relacionan en tiempo con los periodos *backoff* de ningún otro dispositivo en la PAN.

En la Figura 3.20 se muestra el algoritmo utilizado por el CSMA-CA en forma de diagrama de flujo.

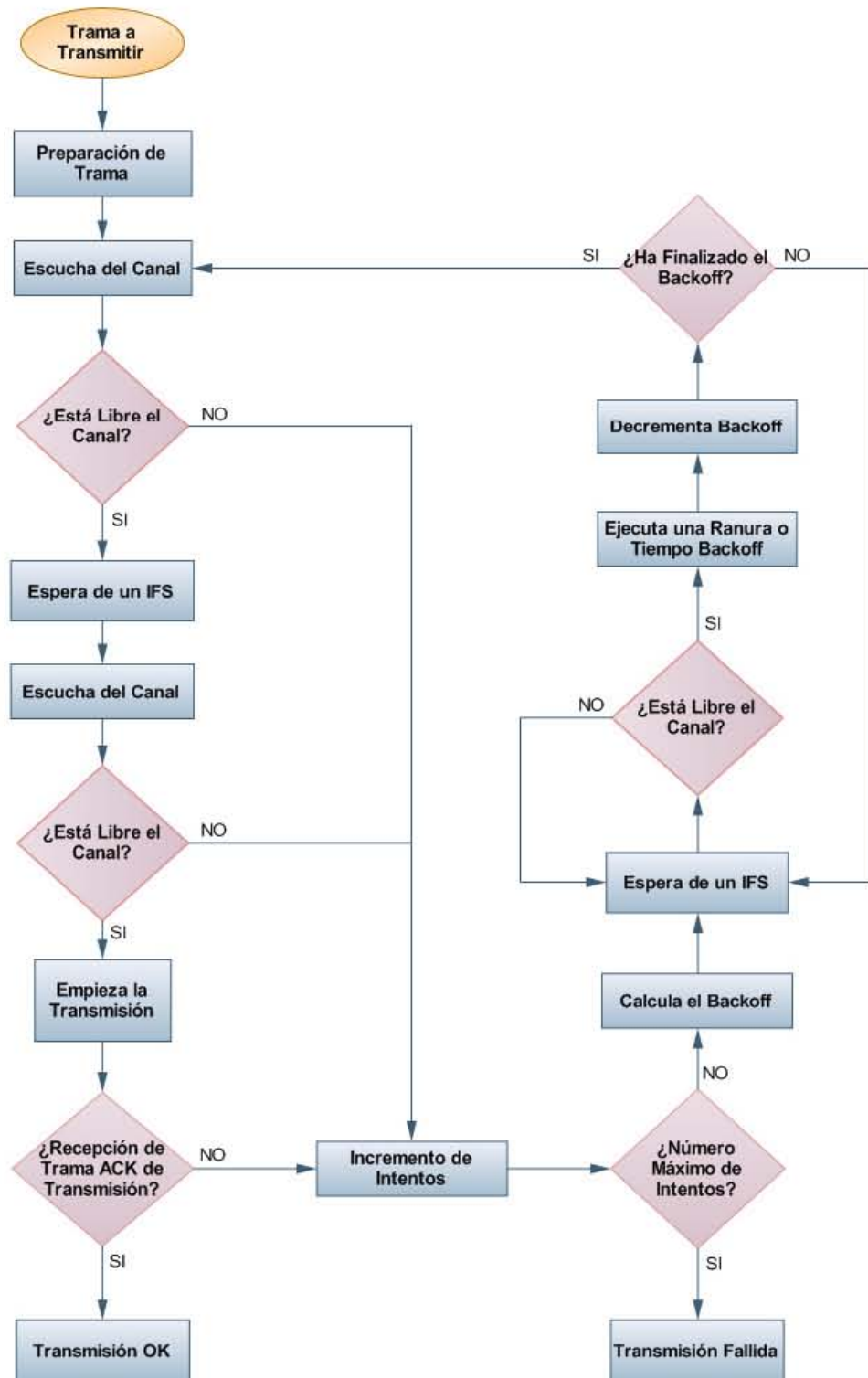


Figura 3.20. Diagrama de flujo del algoritmo CSMA-CA

Capítulo 4

Descripción de los Componentes

En este capítulo se realiza una descripción detallada de los dos componentes más significativos que se ha utilizado en el diseño del nodo desarrollado en el presente PFC, el co-procesador ZigBee Ember EM260 y el microcontrolador Atmel ATmega168.

En la primera parte del capítulo se realiza la descripción general del dispositivo EM260, para pasar a detallar sus características, tanto generales como en radiofrecuencia; a continuación se describen los pines del integrado EM260 y se explica su diagrama funcional. Esta primera parte del capítulo finaliza con el análisis del protocolo SPI (*Serial Peripheal Interface*) y del EZSP (*EmberZNet Serial Protocol*) del co-procesador EM260.

La segunda parte del capítulo se centra en el otro componente principal del diseño, el microcontrolador de ATMEL, ATmega168. Así, se comienza describiendo sus características generales y sus pines, además de incluir una descripción de la arquitectura AVR que lleva implementada, para acabar con la descripción funcional del bloque específico que se emplea en este Proyecto Fin de Carrera, como es la interfaz de comunicación SPI del ATmega168.

4.1 El procesador ZigBee Ember EM260

Para el diseño del módulo ZigBee se ha usado el co-procesador de comunicaciones EM260 fabricado por la empresa de semiconductores EMBER [38]. El circuito integrado EM260 facilitará el desarrollo del estándar de comunicaciones inalámbricas ZigBee en el módulo de comunicaciones inalámbrico objeto de este PFC, ya que ofrece una solución muy flexible en el diseño de dispositivos compatibles con el protocolo ZigBee.

El co-procesador de comunicaciones EM260 ofrece todas las funciones de red ZigBee ya desarrolladas e incluidas internamente con hardware específico mediante máscaras y accesibles mediante una interfaz serie, lo que permite implementar las prestaciones de red ZigBee utilizando cualquier microcontrolador, en el caso particular de este PFC, el ATmega168.

4.1.1 Descripción general

El chip EM260 de Ember [38] es un co-procesador de redes ZigBee que combina un transceptor de radio que opera en la banda de 2.4 GHz (de acuerdo al estándar IEEE 802.15.4) con un microcontrolador de 16-bits basado en la arquitectura XAP2b [39], muy apropiado para su uso con ZigBee, gracias a que combina un rendimiento muy optimizado con un bajo consumo, además de contar con memoria Flash y RAM, todo esto con un alto nivel de integración.

El microcontrolador se encarga de la ejecución de la pila de protocolos incluidos en el EM260. Esta pila, desarrollada por Ember y conocida como EmberZNet [40], implementa el protocolo ZigBee y proporciona una solución directa para aplicaciones ZigBee. El acceso al API de EmberZnet se realiza a través de un puerto SPI o UART, protocolos que integra el dispositivo EM260, lo cual permite el desarrollo de plataformas sobre un servidor que se conecte al dispositivo EM260 como si éste fuese un periférico, mediante una interfaz serie. En la Figura 4.1 se puede observar el diagrama del EM260 conectado a un microcontrolador mediante SPI, así como la arquitectura implementada por cada dispositivo.

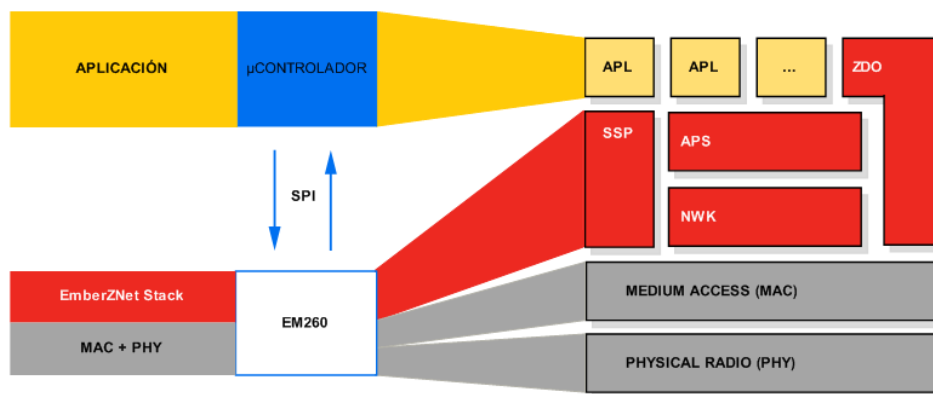


Figura 4.1. Arquitectura formada por la conexión del EM260 y un microcontrolador

El transceptor funciona con una eficiente arquitectura que consigue superar en más de 15 dB el rango dinámico impuesto por las condiciones del estándar 802.15.4. El filtro de canal de recepción integrado permite la co-existencia con otros estándares de comunicación en la banda de los 2,4 GHz, como pueden ser Bluetooth o 802.11g. El dispositivo EM260 requiere de pocos componentes externos gracias a que lleva integrado un regulador de tensión, un VCO (*Voltage-Controlled Oscillator*), un *loop filter* (filtro de bucle) y un amplificador de potencia. Además se provee de una función adicional que permite incrementar el rango dinámico del transceptor en 3dB.

La implementación de la pila de red se realiza en la memoria Flash y RAM que contiene el circuito integrado; el EM260 presenta un eficaz algoritmo para la gestión de memorias, minimizando el uso de la memoria Flash para optimizar su vida útil y permitiendo a las aplicaciones la configuración de la pila y de los *tokens*.

Para cumplir las restricciones temporales impuestas por el estándar IEEE 802.15.4 [25] y ZigBee [10], el dispositivo EM260 integra algunas funciones MAC dentro del hardware. De esta manera, mediante hardware específico, se encarga del manejo automático de funciones como la transmisión y recepción de las señales ACK, los retardos de *backoff*, las asignaciones de los canales libres para la transmisión, así como el filtrado automático de paquetes recibidos. El uso del hardware específico para estas funciones, no sólo consigue cumplir los requerimientos temporales, sino que también libera al procesador del circuito integrado para que pueda realizar otras tareas. Además el EM260 permite cierto nivel de depuración de la MAC mediante una interfaz de traza de paquetes (PTI, *Packet Trace Interface*).

El EM260 dispone de cuatro modos de funcionamiento, entre ellos, dos de bajo consumo: *Deep Sleep* y *Power Down*, que proporcionan al integrado consumos menores de 1 μ A, permitiendo que las baterías de los productos desarrollados alcancen una larga duración.

También es destacable el potente módulo SIF no intrusivo que proporciona el EM260, el cual implementa una interfaz serie síncrona para la depuración y programación del *firmware*, de manera que pueda ser actualizado en caso de aparecer futuras versiones.

Entre las posibles aplicaciones del dispositivo EM260 destacan las siguientes:

- Automatización y control de edificios y hogares.
- Seguimiento y control de inventarios.
- Controles para el entretenimiento en hogares.

En la Figura 4.2 se muestra una fotografía del procesador EM260 de Ember, con un encapsulado de montaje superficial QFN (*Quad Flat No leads*), de 40 pines, y una dimensión de 6 x 6 mm con una distancia entre pin es de 0.50 mm.



Figura 4.2. Procesador ZigBee EM260

La Figura 4.3 muestra el esquemático de una aplicación típica del dispositivo EM260 usando el protocolo SPI, si bien es necesario indicar que en esta figura no se contemplan todos los condensadores de desacoplo que requiere este circuito.

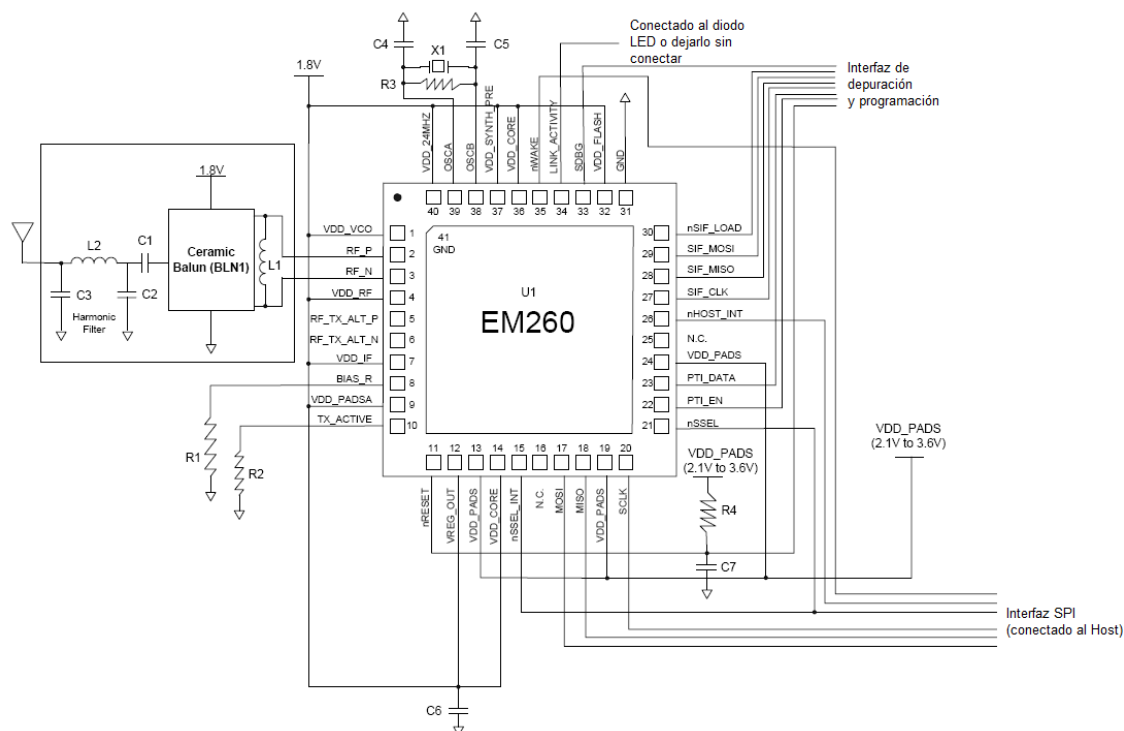


Figura 4.3. Esquemático de aplicación típica del EM260

Como se puede observar en el esquemático representado en Figura 4.3 el dispositivo EM260 ofrece un alto nivel de integración en chip, necesitando una cantidad mínima de componentes externos. Únicamente con la implementación externa del circuito adaptador de radiofrecuencia, el circuito oscilador, y la polarización de algunas ramas del dispositivo, se puede disponer de un diseño listo para conectar al servidor.

4.1.2 Características generales

En la Tabla 4.1 se pueden observar las principales características del circuito integrado EM260 [43].

Características principales del EM260	
- Tres modo de bajo consumo: <div> <div>IDLE (modo de reposo)</div> <div>Deep Sleep</div> <div>Power Down</div> </div> <div> <div>1 μA max</div> <div>1 μA max</div> </div>	
- Consumo en recepción	28 mA - Modo Normal; 34 mA - Modo Boost
- Consumo en transmisión	28 mA - Modo Normal; 30 mA - Modo Boost
- Frecuencia del cristal	24 MHz
- Rango de temperatura de operación	- 40° C hasta 85° C
- Rango de voltaje de operación	2.1 V a 3.6 V
- Encapsulado	40 pin QFN (6 mm x 6 mm)
- Regulador de voltaje integrado de 1.8 V	
- Interfaz serie SPI/UART que permite la conexión a un microcontrolador (Host) para la comunicación con la API EmberZNet	
- Implementación de la capa física y Mac del estándar IEEE 802.15.4	
- Circuitería de "Watchdog Timer" y "Reset-on-power" integrada	
- Acelerador de encriptación AES (Advanced Encryption Standard) integrado	

Tabla 4.1. Características principales del Ember EM260

4.1.3 Características de radiofrecuencia del EM260

En la Tabla 4.2 se muestran las características de radiofrecuencia más importantes del dispositivo EM260 [43].

Características RF del EM260	
- Velocidad de transmisión	250 kbps
- Rango de frecuencia	2400 MHz a 2500 MHz
- Sensibilidad de recepción:	
Modo normal	- 99 dBm
Modo boost	- 100 dBm
- Rechazo del canal adyacente (Definido por el estándar IEEE 802.15.4 para una señal de - 82 dBm)	>35 dB
- Rechazo de 802.11g (En un rango de +12/-13 MHz)	>35 dB
- Potencia de transmisión:	
Modo normal (Min/Max)	-32 dBm / +2.5 dBm
Modo boost (Min/Max)	-32 dBm / +4.5 dBm
- Puerto RF de transmisión secundario para aplicaciones que requieran mayor alcance y necesiten un amplificador de potencia externo	

Tabla 4.2. Características de radiofrecuencia del Ember EM260

4.1.4 Descripción de los pines del EM260

En la Figura 4.4 se muestra la asignación de los pines del dispositivo EM260 con el protocolo SPI, mientras que en la Tabla 4.3 se listan los pines, realizando una breve descripción de cada uno de ellos [43].

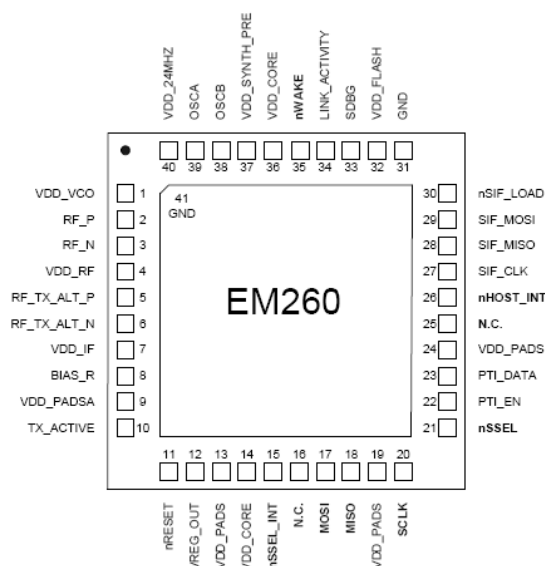


Figura 4.4. Pines del dispositivo EM260

Pin	Señal	Orientación	Descripción
1	VDD_VCO	Alimentación	Tensión de alimentación VCO, 1.8V
2	RF_P	I/O	Entrada-receptor/salida-transmisor diferencial (con RF_N)
3	RF_N	I/O	Entrada-receptor/salida-transmisor diferencial (con RF_P)
4	VDD_RF	Alimentación	Tensión de alimentación RF (LNA y PA), 1.8V
5	RF_TX_ALT_P	O	Salida-transmisor diferencial (con RF_TX_ALT_N), opcional
6	RF_TX_ALT_N	O	Salida-transmisor diferencial (con RF_TX_ALT_P), opcional
7	VDD_IF	Alimentación	Tensión de alimentación IF (filtros y mezcladores del ember), 1.8V
8	BIAS_R	I	Regulación de polarización (mediante resistencia)
9	VDD_PADSA	Alimentación	Pad de alimentación analógico, 1.8V
10	TX_ACTIVE	O	Control de nivel lógico para un conmutador externo de transmisión/recepción.
11	nRESET	I	Reset del chip, activo a nivel bajo.
12	VREG_OUT	Alimentación	Salida del regulador interno del ember, 1.8V
13	VDD_PADS	Alimentación	Pad de alimentación, 2.1 - 3.6V
14	VDD_CORE	Alimentación	Alimentación digital del núcleo, 1.8V
15	nSSEL_INT	I	Interrupción de selección de dispositivo esclavo del bus SPI
16	N.C.	I	Cuando se utiliza el interface SPI, esta señal queda desconectada
17	MOSI	I	Señal de datos del SPI, salida del dispositivo maestro al dispositivo esclavo
18	MISO	O	Señal de datos del SPI, entrada del dispositivo maestro al dispositivo esclavo
19	VDD_PADS	Alimentación	Pad de alimentación, 2.1 - 3.6V
20	SCLK	I	Reloj del bus SPI (del Host al EM260)
21	nSSEL	I	Seleccionar el dispositivo esclavo del bus SPI (del host al EM260)
22	PTI_EN	O	Señal Frame del interface PTI (Packet Trace Interface).
23	PTI_DATA	O	Señal Datos del interface PTI (Packet Trace Interface).
24	VDD_PADS	Alimentación	Pad de alimentación, 2.1 - 3.6V
25	N.C.	I	Cuando se utiliza el interface SPI, esta señal queda desconectada
26	nHOST_INT	O	Señal de interrupción al host del bus SPI, (del EM260 al host)
27	SIF_CLK	I	Reloj del interface de programación y depuración SIF
28	SIF_MISO	O	MISO del interface de programación y depuración SIF
29	SIF_MOSI	I	MOSI del interface de programación y depuración SIF
30	nSIF_LOAD	I/O	Señal de strobe del interface de programación y depuración SIF
31	GND	Masa	Masa
32	VDD_FLASH	Alimentación	Alimentación de la memoria flash, 1.8V
33	SDBG	O	Señal de depuración
34	LINK_ACTIVTY	O	Señal de actividad y enlace
35	nWAKE	I	Señal de interrupción para despertar al EM260 del bus SPI (del Host al EM260)
36	VDD_CORE	Alimentación	Alimentación digital del núcleo, 1.8V
37	VDD_SYNTH_PRE	Alimentación	Alimentación del sintetizador y el preescalador, 1.8V
38	OSCB	I/O	Cristal oscilador de 24Mhz
39	OSCA	I/O	Cristal oscilador de 24Mhz
40	VDD_24MHZ	Alimentación	Alimentación de oscilador de alta frecuencia, 1.8V

Tabla 4.3. Descripción de los pines del dispositivo EM260

4.1.5 Diagrama funcional del EM260

En la Figura 4.5 se puede observar el diagrama de bloques que conforma el circuito integrado EM260 de Ember, además de los diferentes bloques funcionales que componen su estructura interna.

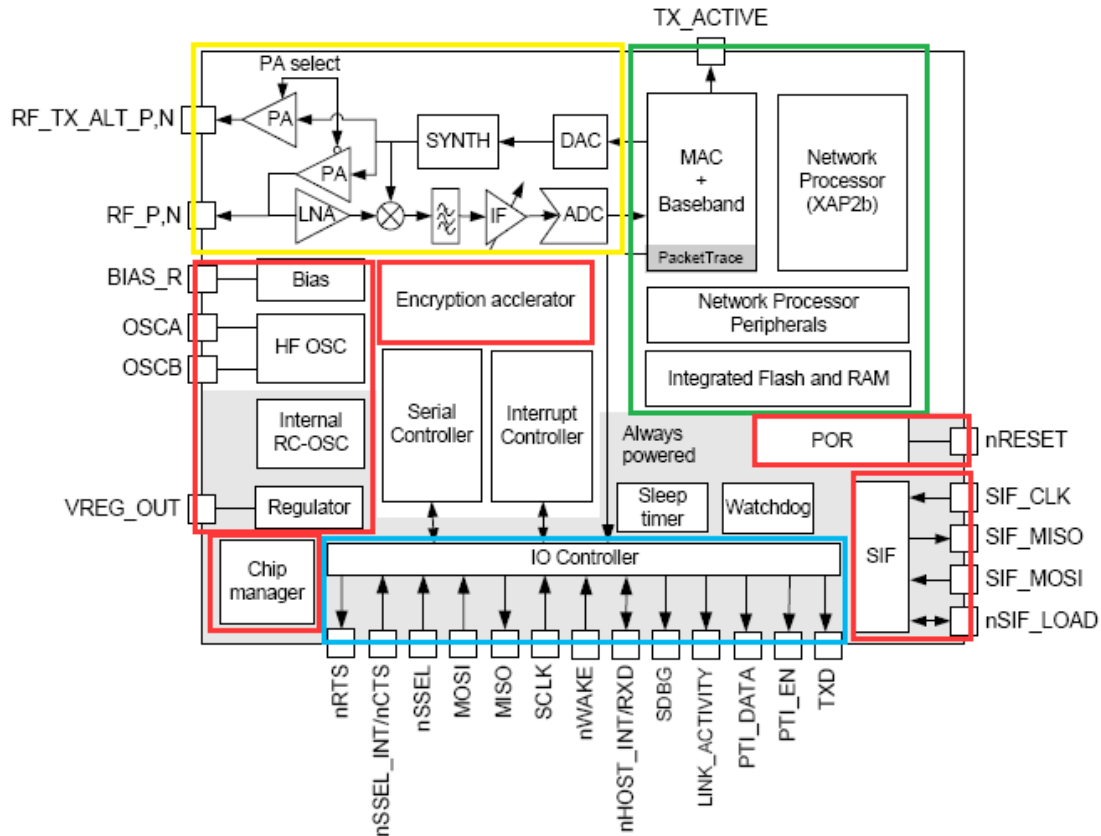


Figura 4.5. Diagrama de bloques simplificado del EM260

El cuadro de color amarillo de la Figura 4.5 se corresponde con el bloque de transmisión y recepción de radiofrecuencia, con entrada diferencial, a fin minimizar las interferencias por ruido, este bloque cuenta con una arquitectura optimizada para coexistir con otros dispositivos que estén en la banda de los 2,4 GHz. La parte de recepción del bloque implementa un receptor superheterodino que consta de un amplificador de bajo ruido seguido de un mezclador; posteriormente se encuentra el filtro y el amplificador de frecuencias intermedias, antes del convertidor analógico/digital, que es la última etapa de la demodulación en banda base de la señal entrante. La parte de transmisión cuenta con un eficiente diseño de modulación que consta de un convertidor digital/analógico al que le entra la señal en banda base, seguido de un sintetizador y un amplificador de potencia, este último encargado de entregar a la salida la señal amplificada cumpliendo con las especificaciones del dispositivo.

El bloque que define el rectángulo de color verde es el encargado de procesar y filtrar los paquetes basados en el estándar IEEE 802.15.4, así como ejecutar la API y manejar las diferentes funciones para el correcto funcionamiento del dispositivo; este bloque está compuesto por el procesador XAP2b, un modulo MAC que implementa a nivel de hardware las funciones más críticas de la MAC, y las memorias Flash y RAM integradas.

En el bloque resaltado con un rectángulo de color azul se encuentra el hardware encargado de conectar el dispositivo con el *host* mediante el protocolo SPI, realizando de esta manera la comunicación del *host* con los módulos que manejan la API. Además, este bloque se encarga de manejar diversas señales, como pueden ser indicadores de actividad, interrupciones, interfaz PTI, etc.

Los módulos restantes, señalados con rectángulos de color rojo tienen diversas funciones, como son la de regular el voltaje, establecer la frecuencia de oscilación, manejar los sistemas de bajo consumo, la programación y depuración del dispositivo EM260 (mediante el módulo con interfaz SIF), etc.

4.1.6 Protocolo SPI del EM260

El dispositivo Ember EM260 utiliza el bus SPI como estándar para la comunicación con el *host*, en este caso el ATmega168. Para gestionar de manera correcta la comunicación SPI con el dispositivo EM260, el fabricante proporciona ciertas recomendaciones:

- El EM260 se maneja como si se tratase de un periférico hardware más.
- El EM260 es el dispositivo *slave* (esclavo) y todas las transacciones son iniciadas por el *host* (master).
- El EM260 soporta una velocidad de datos razonablemente alta (hasta 5 MHz).

Entre las características que ofrece la interfaz SPI del dispositivo EM260, destacan las siguientes:

- El reloj de la interfaz SPI permite trabajar a una frecuencia máxima de 5 MHz.
- El dispositivo EM260 puede soportar la interfaz SPI en dos modos: esclavo, modo 0, en el que la señal de reloj se encuentra inactiva a nivel bajo, y modo 3, donde la señal de reloj se encuentra inactiva a nivel alto; en ambos modos se muestrea por flanco de subida, como se representa en la Figura 4.6.

- La transacción SPI es *half-duplex*, lo que significa que el sentido del intercambio es bidireccional, pero sólo se podrán transmitir o recibir datos en un mismo instante de tiempo.
- Además de las señales usuales de la comunicación SPI (clock, MISO, MOSI, Chip Select) el dispositivo EM260 cuenta con dos señales adicionales denominadas nHOST_INT y nWAKE; la primera sirve como señal de interrupción al *host*, mientras que la señal nWAKE se encarga de activar al EM260 en caso de que se encuentre en algún modo de bajo consumo.

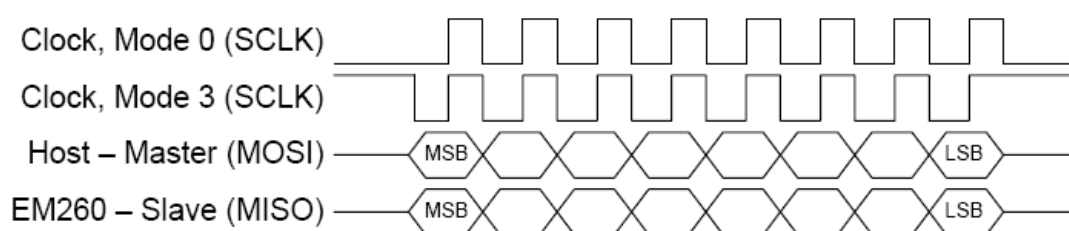


Figura 4.6. Cronograma de transferencia SPI en Modo 0 y Modo 3

La comunicación básica entre el *host* y el dispositivo EM260 mediante interfaz SPI se puede dividir en tres secciones: *Comando*, *Espera* y *Respuesta*. En la Figura 4.7 se muestra el diagrama de tiempos de una transacción SPI con las tres secciones que lo forman.

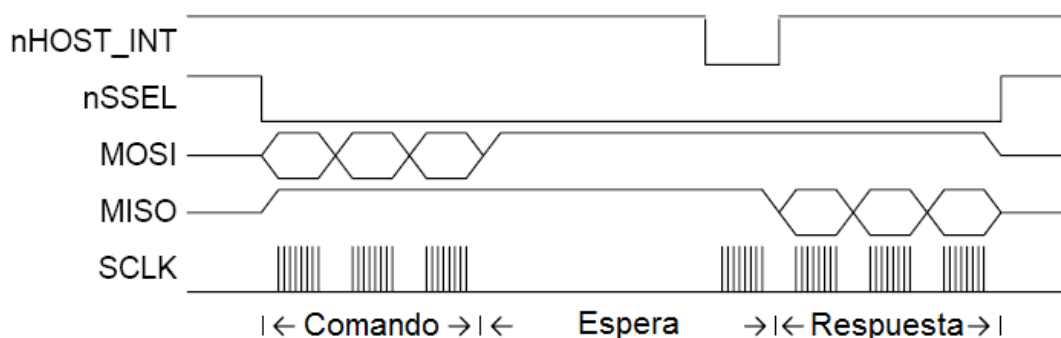


Figura 4.7. Diagrama de una transacción SPI básica

En la sección *Comando*, el *host* (el dispositivo ATmega168 en el caso particular de este PFC) comienza activando la señal Chip Select (poniendo a nivel bajo nSSEL) y luego envía un comando al EM260. El tamaño del comando puede variar entre 2 y 136 bytes, y no debe empezar con 0xFF. Durante esta sección, el dispositivo EM260 responderá sólo con 0xFF, mientras que el *host* deberá ignorar los datos que puedan transmitirse por la señal MISO. Una vez finalizada la transmisión del

comando, se pasará a la sección de *Espera*. Esta sección durará el tiempo necesario para que el dispositivo EM260 pueda procesar el comando o realizar las operaciones pertinentes. El tiempo de espera puede superar los 200 ms, debido a que algunos comandos necesitan bastante tiempo para procesar la respuesta. El dispositivo EM260 utiliza la señal nHOST_INT como interrupción cuando está lista la respuesta, de forma que el *host* puede realizar otras tareas y no estar ocupado haciendo un *polling* en espera de la respuesta. De esta manera, el dispositivo EM260 proporciona dos métodos para determinar el instante en el que está lista una respuesta:

- Comprobar durante la sección de espera si el dispositivo EM260 transmite un valor diferente a 0xFF; en el momento que sea diferente, se pasará a la sección de *Respuesta*. Para comprobarlo, se debe enviar desde el *host* 0xFF para generar la señal de reloj.
- Esperar a que la señal nHOST_INT se active (se activa cuando cambia a nivel bajo), lo cual significará que la respuesta está lista para ser leída por el *host*.

Una vez que se conoce que la respuesta está lista, se pasará a la sección de *Respuesta*. El formato de la trama que se va a recibir es igual al visto en la sección *Comando*. El tamaño de la trama respuesta puede variar entre 2 y 136 bytes, y no empezará con 0xFF. Dependiendo de la respuesta, el primer o el segundo byte recibido determina el tamaño de la trama que se espera, lo que permite conocer exactamente el número de bytes que enviará el dispositivo EM260, y de esta manera generar los ciclos de reloj necesarios (transfiriendo desde el *host* los códigos 0xFF mediante SPI) para recibir todos los bytes de la *Respuesta*. Una vez recibidos los bytes de toda la trama, el *host* desactiva la señal Chip Select (nSSEL). La señal Chip Select debe estar desactivada (poniéndolo a nivel alto) durante un tiempo no inferior a 1 ms, que es el tiempo mínimo necesario para que el dispositivo EM260 esté listo para procesar un nuevo comando, a este tiempo se le denomina *espaciado entre comandos*.

Cuando el dispositivo EM260 está en modo *sleep*, es necesario activarlo antes de enviarle algún comando. En la Figura 4.8 se muestra la secuencia necesaria para que el dispositivo EM260 se active y esté preparado para procesar cualquier comando si se encuentra en modo *sleep*. Los pasos necesarios para activarlo son los siguientes:

- 1. El *host* activa la señal nWAKE.
- 2. El dispositivo EM260 sale del modo *sleep* por la interrupción nWAKE.
- 3. El dispositivo EM260 realiza todas las operaciones necesarias para estar en disposición de recibir comandos.
- 4. El dispositivo EM260 activa la señal nHOST_INT dentro de los 10 ms de la activación de la señal nWAKE. Si el dispositivo EM260 no activa la señal nHOST_INT dentro de

esos 10 ms, el *host* puede considerar que no hay respuesta por parte del dispositivo EM260 y realiza un reset.

- 5. Al activarse la señal *nHOST_INT*, el dispositivo EM260 indica que puede aceptar un comando mediante SPI, en este caso el *host* no necesitará mantener la señal Chip Select a nivel alto 1 ms para el espacio entre comandos, como normalmente sucede.
- 6. El *host* desactiva la señal *nWAKE* después de detectar la activación de la señal *nHOST_INT*.
- 7. El dispositivo EM260 desactivará la señal *nHOST_INT* en menos de 25 μ s tras la desactivación de la señal *nWAKE*.
- 8. Después de los 25 μ s, cualquier cambio en la señal *nHOST_INT* se interpreta como una operación normal (significa que existe una respuesta esperando).

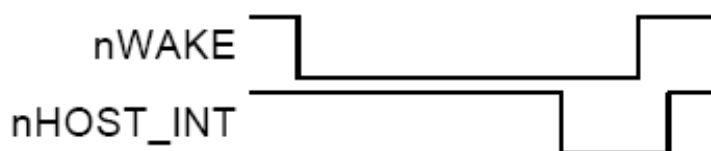


Figura 4.8. Secuencia de activación del dispositivo EM260

4.1.6.1 Temporización

En la Figura 4.9 se muestra un cronograma con todos los parámetros críticos del protocolo SPI. Estas temporizaciones de los parámetros son debidas a las operaciones internas del dispositivo EM260 y el funcionamiento del *host* debe estar dentro de estas restricciones temporales para asegurar el correcto funcionamiento de la comunicación mediante el protocolo SPI.

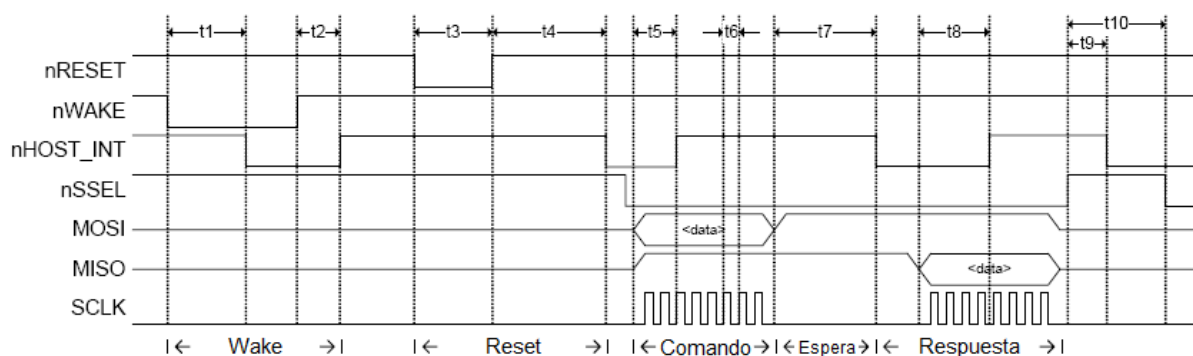


Figura 4.9. Cronograma de tiempos críticos del protocolo SPI del EM260

En la Tabla 4.4 se puede observar una lista con los valores de los parámetros que se muestran en la Figura 4.9.

Parámetro	Descripción	Mín.	Típ.	Máx.	Unidad
t1 (a)	Tráfico para despertar al EM260 (estando despierto).		133	150	μs
t1 (b)	Tráfico para despertar al EM260 (estando dormido).		7.3	10	ms
t2	Finalización del tráfico para despertar.	1.1	1.2	25	μs
t3	Ancho del Reset.	8			μs
t4 (a)	Tiempo de arranque (entrando en la aplicación).		250	1500	ms
t4 (b)	Tiempo de arranque (entrando en el bootloader).		2.5	7.5	s
t5	Desactivación del nHOST_INT después del Comando.	13	3.5	75	μs
t6	Velocidad del reloj.	200			μs
t7	Sección de Espera.	25	755	200000	μs
t8	Desactivación del nHOST_INT después de la Respuesta.	20	130	800	μs
t9	Activación del nHOST_INT después de una transacción	25	70	800	μs
t10	Espaciado entre comandos.	1			ms

Tabla 4.4. Parámetros temporales del protocolo SPI del dispositivo EM260

4.1.6.2 Formato de las tramas

El formato de trama empleado es el mismo, tanto para los comandos como para las respuestas, y su tamaño nunca debe exceder de 136 bytes. En la Figura 4.10 se muestra el formato de la trama del protocolo SPI.

1 byte	1 byte	Longitud variable	1 byte
SPI Byte	Length or Error	Payload Frame	Frame Terminator

Figura 4.10. Formato de la trama SPI

Todas las tramas deben comenzar con el *SPI Byte*, que identifica de qué comando se trata. Como mínimo, cada trama contará con dos bytes, que son *SPI Byte* y *Frame Terminator*. El *Length Byte* sólo se incluirá si existe información en el *Payload Frame*. En el caso de que exista el *Length byte* sólo define el tamaño de *Payload Frame* y no el de toda la trama. Por lo tanto, si la trama tiene *Payload Frame*, el valor de *Length byte* estará entre 2 y 133, y el tamaño de la trama total estará comprendido entre 5 y 136 bytes. *SPI Byte* puede contener un valor que indica si existe *Payload Frame* o no; si existe *Payload Frame*, se esperará un valor especificado en el campo *Length*. *Byte Error* es usado por las respuestas de errores, suministrando información adicional sobre el error. Se ubica en el espacio que normalmente se usa para *Length Byte*.

Payload Frame contiene los datos necesarios para realizar funciones EmberZNet utilizando el protocolo EZSP (*EmberZNet Serial Protocol*) que están definidas por el dispositivo EM260 y serán explicadas en posteriores apartados.

Frame Terminator es un byte especial de control usado para señalar el final de la trama. El byte *Frame Terminator* es definido con un 0xA7 y se añade al final de cualquier trama, con la intención de proporcionar al protocolo SPI un byte conocido para detectar tramas corruptas. Por ejemplo, en el caso de que haya un reset durante la sección de respuesta y el *host* siga enviando 0xFF para generar el reloj y leer el número de bytes esperados, cuando termine e intente comprobar el valor 0xA7 al final de la respuesta, verá que el valor no coincide, y dependiendo de si ha leído un 0x00 o un 0xFF determinará qué ha sucedido. Así, en el caso de leer 0x00 sabrá que se ha producido un reset, y en el caso de leer 0xFF, que la respuesta es corrupta y deberá desecharla.

En la Tabla 4.5 se pueden observar los posibles comandos que existen del *SPI Byte* y el valor de su respuesta, con una breve interpretación de cada una.

Valor del Comando	Comando	Valor de la Respuesta	Respuesta
Cualquiera	Cualquiera	0x00	Reset del EM260. Esta respuesta indica que se ha producido un reset en el EM260, sólo aparece después de un reset.
Cualquiera	Cualquiera	0x01	Payload desbordado. Esta respuesta indica que el payload de la trama sobrepasa el límite permitido.
Cualquiera	Cualquiera	0x02	Transacción abortada. Esta respuesta indica que la transacción no se ha completada y ha sido abortada.
Cualquiera	Cualquiera	0x03	Falta <i>Frame Terminator</i> . Esta respuesta indica que no se ha recibido el <i>Frame Terminator</i> .
Cualquiera	Cualquiera	0x04	Comando SPI no permitido. Esta respuesta indica que el comando SPI recibido no es válido.
0x00 - 0x0F	Reservado	Ninguno	Ninguno
0x0A	Versión del protocolo SPI	0x81 - 0xBF	El bit[7] y el bit[6] tienen siempre un valor fijo, '1' y '0' respectivamente. Los bits [5:0] indican un valor entre 1 y 63.
0x0B	Estado del SPI	0xC0 - 0xC1	El bit[7] y el bit[6] están siempre a '1', los demás a '0' excepto el bit[0], que indicara el estado del SPI, un '1' si esta OK o un '0' en caso contrario.
0x0C - 0xFC	Reservado	Ninguno	Ninguno
0xFD	Trama Bootloader	0xFD	Trama Bootloader.
0xFE	Trama EZSP	0xFE	Trama EZSP.
0xFF	Inválido	0xFF	Inválido

Tabla 4.5. Comandos y respuestas SPI del dispositivo EM260

4.1.7 EZSP (EmberZNet Serial Protocol)

Para facilitar al diseñador el uso de la pila *API del EmberZNet* [40], el fabricante del dispositivo EM260, Ember, ha desarrollado un protocolo serie llamado *EZSP (EmberZNet Serial Protocol)* [41], el cual se encarga de comunicar el dispositivo EM260 con el *host* mediante la interfaz SPI o UART, dependiendo del *firmware* que se haya programado en el dispositivo EM260.

El protocolo EZSP está formado por un conjunto de comandos que llevan asociada una respuesta. Cada comando es interpretado por el dispositivo EM260 para llevar a cabo una tarea específica y la respuesta proporcionará información sobre el estado del dispositivo, así como información de sobre la ejecución de la tarea asignada en el comando. El funcionamiento del protocolo consiste en realizar la comunicación del *host* con el dispositivo EM260 mediante la transacción de dos mensajes. En primer lugar el *host* inicia la transacción enviando un comando; una vez recibido por el EM260, éste enviará la correspondiente respuesta al *host*. En los casos en que el dispositivo EM260 necesite conectarse asincrónicamente con el *host*, se lo indicará usando la señal de interrupción (*nHOST_INT*); a continuación, el *host* enviará el comando *callback*. Una vez recibido este comando, el dispositivo EM260 le enviará el mensaje correspondiente. Las tramas del protocolo EZSP siempre están precedidas del byte 0xFE, que corresponde al campo *SPI Byte* de la trama, e indica que ésta contiene un mensaje EZSP en su contenido.

En la Figura 4.11 se muestra un cronograma de un ejemplo de funcionamiento del protocolo EZSP, donde se puede observar una transacción simple de dos mensajes con comunicación asíncrona mediante el comando *callback*.

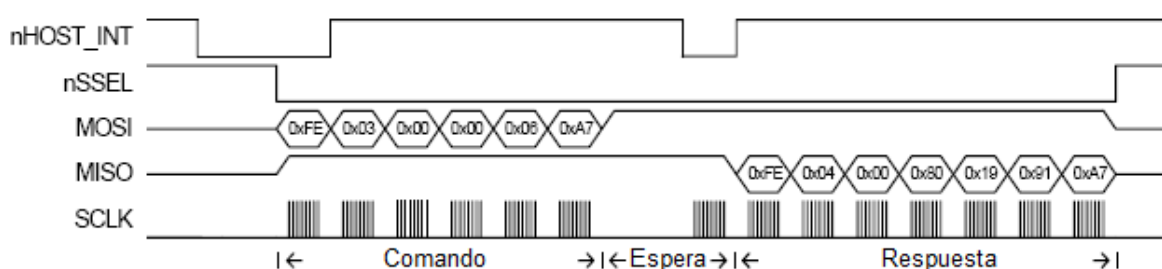


Figura 4.11. Cronograma del comando *Callback*

4.1.7.1 Estructura de las tramas EZSP

El formato de todas las tramas EZSP comienza siempre con los mismos tres campos: secuencia, control de trama e ID de trama. El resto de campos de cada trama dependerá del ID de

trama. La mayoría de las tramas tienen una longitud fija, excepto algunas, como por ejemplo las que contienen mensajes con texto, que tienen longitud variable. En la Figura 4.12 se representa la estructura de una trama EZSP.

1 byte	1 byte	1 byte	Longitud variable
Secuencia (Sequence)	Control de Trama (Frame Control)	ID de Trama (Frame ID)	Parámetros (Parameters)

Figura 4.12. Estructura de la trama EZSP

La secuencia es el primer byte de todas las tramas EZSP, cuyo valor debe ser incrementado por el *host* cada vez que se envíe un comando al dispositivo EM260; a su vez, el dispositivo EM260 usará el número de secuencia recibido cuando envíe la respuesta correspondiente. El segundo byte de todas las tramas EZSP es el byte de control de trama, que indica si la trama es de comando o de respuesta; en el caso de que sea un comando, también contiene información sobre los modos de consumo, y si es una respuesta, contiene información sobre el estado del EM260. En la Tabla 4.6 se describe el significado de los comandos y de las respuestas de los bits que forman el byte de control de trama. En la Tabla 4.7 se muestran los posibles valores de los bits de *sleepMode*, que identificarán los diferentes modos de bajo consumo. Por otro lado la Tabla 4.8 y la Tabla 4.9 describen los estados de los bits *overflow* y *truncated*, respectivamente. El tercer byte es el ID de trama, encargado de identificar el tipo de mensaje EZSP que es enviado o recibido, y que tiene asociados los parámetros que le siguen, cuya longitud es variable y dependerán del ID de trama. El ID de trama y sus parámetros serán descritos en el apartado 4.1.7.4.

Bit	Comando	Respuesta
7 (MSB)	0	1
6	0 (Reservado)	0 (Reservado)
5	0 (Reservado)	0 (Reservado)
4	0 (Reservado)	0 (Reservado)
3	0 (Reservado)	0 (Reservado)
2	0 (Reservado)	0 (Reservado)
1	sleepMode [1]	truncated
0 (LSB)	sleepMode [0]	overflow

Tabla 4.6. Significado del byte de control de trama

sleepMode[1]	sleepMode[0]	Descripción
1	1	Reservado
1	0	Power down
0	1	Deep sleep
0	0	Idle

Tabla 4.7. Significado de los bits de los modos de bajo consumo *sleepMode*

overflow	Descripción
1	La memoria del EM260 se ha desbordado durante la respuesta.
0	No hay falta de memoria durante la respuesta.

Tabla 4.8. Significado del bit *overflow* de la trama de respuesta

truncated	Descripción
1	El EM260 ha truncado la respuesta actual para evitar que se exceda la longitud máxima de trama EZSP
0	La respuesta actual no fue truncada.

Tabla 4.9. Significado del bit *truncated* de la trama de respuesta

4.1.7.2 Configuración de la pila ZigBee

Cada vez que se produzca un reset o se inicie por primera vez el dispositivo EM260, el *host* debe ponerse de acuerdo con el protocolo de comunicación que se va a emplear, antes de empezar cualquier comunicación. Para ponerse de acuerdo y asegurar que el dispositivo EM260 y el *host* utilizan el mismo protocolo, el *host* debe enviar el comando ‘version’, que permitirá al *host* determinar qué versión del EZSP a emplear; la respuesta a este comando proporcionará información sobre el *firmware* que se está ejecutando en el integrado EM260.

La pila del dispositivo EM260 tiene un conjunto de valores de configuración que afectan a su comportamiento. Una vez conocido el protocolo de comunicación a utilizar, se deben configurar estos valores para especificar las características deseadas en el nodo. En cualquier momento es posible leer estos valores usando el comando *getConfigurationValue*, o modificarlos utilizando el comando *setConfigurationValue*. En la Tabla 4.10 se muestra el conjunto de valores de configuración de la pila, con los valores máximos, mínimos y por defecto que se pueden asignar, así como el uso de RAM necesario, su ID y una breve descripción.

ID	Valor	Mín.	Def.	Máx.	Unidad	RAM	Descripción
0x01	EZSP_CONFIG_PACKET_BUFFER_COUNT	5	24		paquetes	39	Número de paquetes en memoria disponible para la pila.
0x02	EZSP_CONFIG_NEIGHBOR_TABLE_SIZE	8	16	16	nodos vecinos	18	Número máximo de nodos ruteados que la pila puede mantener en seguimiento.
0x03	EZSP_CONFIG_APS_UNICAST_MESSAGE_COUNT	0	10		mensajes	6	Número máximo de mensajes retransmitidos de la APS que la pila puede transmitir en cualquier momento.
0x04	EZSP_CONFIG_BINDING_TABLE_SIZE	0	0	32	entradas	2	Número máximo de enlaces no-volátiles permitidos por la pila.
0x05	EZSP_CONFIG_ADDRESS_TABLE_SIZE	0	8		entradas	12	Número máximo de direcciones EUI64 asociadas a la red que la pila puede mantener.
0x06	EZSP_CONFIG_MULTICAST_TABLE_SIZE	0	8		entradas	4	Número máximo de grupos multicast que el nodo puede ser miembro.
0x07	EZSP_CONFIG_ROUTE_TABLE_SIZE	0	16		entradas	6	Número máximo de destinos a los que un nodo puede enrutar los mensajes. Incluye tantos los mensajes originados por el nodo como los retransmitidos.
0x08	EZSP_CONFIG_DISCOVERY_TABLE_SIZE	0	8		entradas	10	Número de rutas descubiertas simultáneas que el nodo puede soportar.
0x09	EZSP_CONFIG_BROADCAST_ALARM_DATA_SIZE	0	0	16	bytes	1	Tamaño del buffer de alarma broadcast.
0x0A	EZSP_CONFIG_UNICAST_ALARM_DATA_SIZE (A)	0	0	16	bytes	(B)	Tamaño del buffer de alarma unicast asignada para los dispositivos finales (nodos hijos).
0x0C	EZSP_CONFIG_STACK_PROFILE	0	0			0	Especifica el perfil de la pila.
0x0D	EZSP_CONFIG_SECURITY_LEVEL	0	5	5		0	Nivel de seguridad usado por la capa MAC y la capa de RED. Soporta valores entre 0 (sin seguridad) y 5 (payload encriptado y autenticación con cuatro bytes de MIC).
0x10	EZSP_CONFIG_MAX_HOPS	0	10		saltos (hops)	0	Número máximo de saltos que puede realizar un mensaje.
0x11	EZSP_CONFIG_MAX_END_DEVICE_CHILDREN (B)	0	6	32	nodos hijos	9 + (A)	Número máximo de dispositivos finales que el coordinador o router pueden soportar.
0x12	EZSP_CONFIG_INDIRECT_TRANSMISSION_TIMEOUT	0	3000	30000	ms.	0	Tiempo máximo que la MAC puede retener un mensaje para la transmisión indirecta a un nodo hijo.
0x13	EZSP_CONFIG_END_DEVICE_POLL_TIMEOUT	0	5	255	minutos	0	Tiempo máximo que un dispositivo final puede permanecer sin retransmitir ningún dato al dispositivo padre. Si se cumple este tiempo el dispositivo padre eliminará el dispositivo final de su tabla de enrutamiento.
0x14	EZSP_CONFIG_MOBILE_NODE_POLL_TIMEOUT	0	20		250 ms.	0	Tiempo máximo que un nodo móvil puede permanecer sin retransmitir ningún dato al dispositivo padre. Si se cumple este tiempo el dispositivo padre eliminará el dispositivo final de su tabla de enrutamiento.
0x15	EZSP_CONFIG_RESERVED_MOBILE_CHILD_ENTRIES	0	0	(B)	entradas	0	Número de entradas en la tabla de nodos hijos reservado para nodos móviles.
0x16	EZSP_CONFIG_HOST_RAM	0	0	255	bytes	1	RAM disponible para el uso del <i>host</i> .

0x17	EZSP_CONFIG_TX_POWER_MODE	0	0	3		0	Especifica el Modo de transmisión a usar. Puede ser modo normal, modo boost (pudiendo transmitir a 5dB de potencia), salida de transmisión alternativa y combinar el modo boost con la salida de transmisión alternativa.
0x18	EZSP_CONFIG_DISABLE_RELAY	0	0	1		0	0: Permite al nodo retransmitir los mensajes. 1: Evita al nodo la retransmisión de los mensajes.
0x1A	EZSP_CONFIG_SOURCE_ROUTE_TABLE_SIZE	0	0		entradas	4	Tamaño de la tabla del router origen.
0x19	EZSP_CONFIG_TRUST_CENTER_ADDRESS_CACHE_SIZE	0	0		entradas	12	Número máximo de direcciones EUI64 asociadas a la red que el Trust Center (centro de confianza) puede mantener.

Tabla 4.10. Valores de configuración de la pila del EM260

4.1.7.3 Definición de tipos de variable

Entre los campos de la estructura de las tramas EZSP se encuentra el campo de parámetros, que depende del ID de la trama, como se comentó en el apartado 4.1.7.1. Estos parámetros están formados por tipos de variables, que indican las especificaciones de los comandos EZSP. Aunque existen diversos tipos de variable y estructuras en el protocolo EZSP, en la Tabla 4.11, Tabla 4.12 y Tabla 4.13 se nombran las que han sido necesario utilizar en el desarrollo de este Proyecto Fin de Carrera. La totalidad de las variables, así como de su estructura, se pueden consultar en la documentación que proporciona Ember sobre del dispositivo EM260 [38].

TIPOS BÁSICOS	
Tipo	Descripción
int8u	Variable entera sin signo de 8 bits. Rango de 0 a 255.
int16u	Variable entera sin signo de 16 bits. Rango de 0 a 65535.

Tabla 4.11. Tipos de datos básicos empleados en protocolo EZSP en el PFC

TIPOS DE VARIABLES		
Tipo	Formato	Descripción
Boolean	int8u	Expresa un valor de verdadero o falso.
EzspConfigId	int8u	Identificador de un valor de configuración.
EmberConfigPowerTxMode	int8u	Valores de EZSP_CONFIG_TX_POWER.
EzspStatus	int8u	Valores de estado utilizadas por EZSP.
EmberStatus	int8u	Valor de retorno para funciones de la pila.
EmberNodeType	int8u	Identificador del tipo de nodo.
EmberNetworkStatus	int8u	Posibles estados de unión a la red de un nodo.
EmberIncommingMessageType	int8u	Tipo de mensaje entrante.
EmberOutgoingMessageType	int8u	Tipo de mensaje saliente.
EmberNodeId	int8u	Dirección de red en ZigBee de 16bits.
EmberApsOption	int16u	Opciones permitidas en el envío de un mensaje.
EmberPanId	int8u	802.15.4 PAN ID. Identificador de red.
EmberEUI64	int8u[8]	Identificador de dispositivo de red de 64bits.

Tabla 4.12. Tipos de variables implementadas en el protocolo EZSP en el PFC

DEFINICIÓN DE ESTRUCTURAS DE VARIABLES		
Tipo	Campo	Descripción
EmberNetworkParameters		Parámetros de la red ZigBee.
	int8u[8] extendPanId	Identificador extendido de la red.
	int16u panId	Identificador reducido de la red.
	int8u radioTxPower	Nivel de transmisión, en dBm.
	int8u radioChannel	Canal de radio a utilizar, de 11 a 26.
EmberApsFrame		Parámetros de la trama APS de ZigBee.
	int16u profileId	ID de perfil de aplicación.
	int16u clusterId	ID del clúster para el mensaje.
	int8u sourceEndPoint	Endpoint de origen.
	int8u destEndPoint	Endpoint de destino.
	EmberApsOption options	Máscara de opciones.
	int16u groupId	ID del grupo de destino (Multicast).
	int8u sequence	Número de secuencia del mensaje.

Tabla 4.13. Estructura de datos implementada en el protocolo EZSP en el PFC

4.1.7.4 Comandos EZSP

El protocolo EZSP cuenta con una gran cantidad de comandos que son soportados por las tramas EZSP. Estos comandos son identificados por el dispositivo EM260 mediante su ID y llevan asociados unos parámetros de comando, así como unos parámetros de respuesta. Estos comandos están organizados en varios grupos dependiendo del tipo. Así, existen comandos de configuración, comandos de utilidades, comandos de red, comandos de seguridad, comandos de mensaje, etc.

En las siguientes tablas se muestran los comandos utilizados en este PFC, describiendo en ellas cada comando, especificando su ID correspondiente, y describiendo la función que realiza, así como los parámetros de comando y de respuesta que llevan asociados. La totalidad de comandos, incluida su descripción, se pueden consultar en la documentación que proporciona Ember sobre el dispositivo EM260 [38].

Nombre: version		ID: 0x00
Descripción: Este comando permite que el <i>Host</i> y el EM260 determinen la versión del protocolo EZSP que se va a utilizar. La respuesta indica información acerca del protocolo EZSP y el <i>firmware</i> que el EM260 está utilizando.		
Parámetros del comando:		
int8u desiredProtocolVersion	Versión del protocolo EZSP que el <i>Host</i> desea utilizar para comunicarse.	
Parámetros de respuesta:		
int8u ProtocolVersion	Versión del protocolo EZSP que el EM260 está usando. Si el EM260 no soporta la versión deseada por el <i>Host</i> , se usará la mayor versión soportada por el EM260.	
int8u stackType	Tipo de pila implementada en el EM260. Los comandos EZSP disponibles y los parámetros de configuración dependen del tipo de pila implementada en el EM260. Para una topología de red de tipo malla este parámetro es 2.	
int16u stackVersion	El número de versión de la pila.	

Nombre: reset	ID: 0x08
Descripción: Este comando permite al <i>Host</i> reiniciar por software el dispositivo EM260.	
Parámetros del comando: ninguno	
Parámetros de respuesta: ninguno	

Nombre: setConfigurationValue	ID: 0x53
Descripción: Este comando permite modificar los valores de la pila ZigBee establecidos por defecto. Una vez que el estado de la red ha cambiado, no se podrá modificar ninguno de los valores de la pila. Si se intentara modificar cuando la red se encuentra EMBER_NETWORK_UP, la API daría como respuesta el comando EZSP_ERROR_INVALID_CALL.	
Parámetros del comando:	
EzspConfigId configId	Establece el valor de la pila que se desea modificar.
int16u value	El nuevo valor de la configuración.
Parámetros de respuesta:	
EzspStatus status	<p>Las posibles respuestas al comando son:</p> <p>EZSP_SUCCESS, indica que el valor de configuración ha sido modificado correctamente;</p> <p>EZSP_ERROR_OUT_OF_MEMORY, indica que el nuevo valor de configuración excede la memoria disponible;</p> <p>EZSP_ERROR_INVALID_VALUE, indica que el valor establecido no se encuentra en el rango permitido;</p> <p>EZSP_ERROR_INVALID_ID, indica que el parámetro que se desea modificar no es reconocido;</p> <p>EZSP_ERROR_INVALID_CALL, indica que no está permitido modificar el parámetro solicitado.</p>

Nombre: networkState	ID: 0x18
Descripción: Este comando devuelve un valor que indica el estado del nodo: si se está uniendo a la red, si está unido a la red, o si está abandonando la red.	
Parámetros del comando: Ninguno	
Parámetros de respuesta: <div> <div>EmberNetworkStatus status</div> <div>Variable de estado de la red, que indica el estado del nodo correspondiente.</div> </div>	

Nombre: joinNetwork	ID: 0x1F
Descripción: Este comando hace que el dispositivo ZigBee se una a una red con los parámetros indicados en el comando. La unión a la red puede llevar varios segundos hasta que se hace efectiva, de manera que no es posible enviar comandos hasta que el comando callback stackStatusHandler informe que el proceso de unión ha finalizado.	
Parámetros del comando: <div> <div>EmberNodeType nodeType</div> <div>Este campo indica el papel que va a desarrollar el nodo en la red. En este caso no puede ser coordinador, ya que éste es el único que puede formar la red. Establece el valor de la pila que se desea modificar. Puede ser nodo final, denominado END_DEVICE o router, denominado ROUTER.</div> </div> <div> <div>EmberNetworkParameters parameters</div> <div>Especificación de los parámetros de la red a la cual el dispositivo se intenta unir.</div> </div>	
Parámetros de respuesta: <div> <div>EmberStatus status</div> <div>Mensaje de estado indicando si la unión del dispositivo a la red se inició, o por el contrario se produjo algún fallo.</div> </div>	

Nombre: sendUnicast	ID: 0x34												
Descripción: Este comando envía de un mensaje unicast por el dispositivo. El mensaje llegará a su destino sólo si existe una ruta conocida por el nodo. Existen varias opciones que se encargarán de descubrir las rutas adecuadas para el envío del mensaje.													
Parámetros del comando: <table> <tr> <td data-bbox="252 622 794 656">EmberOutgoingMessageType type</td><td data-bbox="810 622 1455 987">Este campo especifica el tipo de direccionamiento que se desea para el envío del mensaje. En este caso puede ser EMBER_OUTGOING_DIRECT, el mensaje se envía directamente a un EmberNodeId; EMBER_OUTGOING_VIA_ADDRESS_TABLE, el mensaje se envía empleando una entrada de la tabla de direcciones; EMBER_OUTGOING_VIA_BINDING, el mensaje se envía empleando una entrada de la tabla asociada.</td></tr> <tr> <td data-bbox="252 1010 794 1043">EmberNodeId indexOrDestination</td><td data-bbox="810 1010 1455 1133">Este campo depende del tipo de direccionamiento empleado, pudiendo ser: un EmberNodeId, un índice de la tabla de direcciones, o un índice de la tabla asociada.</td></tr> <tr> <td data-bbox="252 1178 794 1211">EmberApsFrame apsFrame</td><td data-bbox="810 1178 1455 1256">Este campo añade los parámetros de la trama APS ZigBee al mensaje.</td></tr> <tr> <td data-bbox="252 1290 794 1323">int8u messageTag</td><td data-bbox="810 1290 1455 1447">Este campo establece un valor para identificar al mensaje, el valor es seleccionado por el <i>host</i>. Este identificador se emplea en la respuesta <i>ezspMessageSentHandler</i> para referirse a un mensaje en concreto.</td></tr> <tr> <td data-bbox="252 1480 794 1514">int8u messageLength</td><td data-bbox="810 1480 1455 1514">Tamaño del contenido del mensaje en bytes.</td></tr> <tr> <td data-bbox="252 1547 794 1581">int8u[] messageContents</td><td data-bbox="810 1547 1455 1581">Contenido del mensaje.</td></tr> </table>		EmberOutgoingMessageType type	Este campo especifica el tipo de direccionamiento que se desea para el envío del mensaje. En este caso puede ser EMBER_OUTGOING_DIRECT, el mensaje se envía directamente a un EmberNodeId; EMBER_OUTGOING_VIA_ADDRESS_TABLE, el mensaje se envía empleando una entrada de la tabla de direcciones; EMBER_OUTGOING_VIA_BINDING, el mensaje se envía empleando una entrada de la tabla asociada.	EmberNodeId indexOrDestination	Este campo depende del tipo de direccionamiento empleado, pudiendo ser: un EmberNodeId, un índice de la tabla de direcciones, o un índice de la tabla asociada.	EmberApsFrame apsFrame	Este campo añade los parámetros de la trama APS ZigBee al mensaje.	int8u messageTag	Este campo establece un valor para identificar al mensaje, el valor es seleccionado por el <i>host</i> . Este identificador se emplea en la respuesta <i>ezspMessageSentHandler</i> para referirse a un mensaje en concreto.	int8u messageLength	Tamaño del contenido del mensaje en bytes.	int8u[] messageContents	Contenido del mensaje.
EmberOutgoingMessageType type	Este campo especifica el tipo de direccionamiento que se desea para el envío del mensaje. En este caso puede ser EMBER_OUTGOING_DIRECT, el mensaje se envía directamente a un EmberNodeId; EMBER_OUTGOING_VIA_ADDRESS_TABLE, el mensaje se envía empleando una entrada de la tabla de direcciones; EMBER_OUTGOING_VIA_BINDING, el mensaje se envía empleando una entrada de la tabla asociada.												
EmberNodeId indexOrDestination	Este campo depende del tipo de direccionamiento empleado, pudiendo ser: un EmberNodeId, un índice de la tabla de direcciones, o un índice de la tabla asociada.												
EmberApsFrame apsFrame	Este campo añade los parámetros de la trama APS ZigBee al mensaje.												
int8u messageTag	Este campo establece un valor para identificar al mensaje, el valor es seleccionado por el <i>host</i> . Este identificador se emplea en la respuesta <i>ezspMessageSentHandler</i> para referirse a un mensaje en concreto.												
int8u messageLength	Tamaño del contenido del mensaje en bytes.												
int8u[] messageContents	Contenido del mensaje.												
Parámetros de respuesta: <table> <tr> <td data-bbox="252 1704 794 1738">EmberStatus status</td><td data-bbox="810 1704 1455 1827">Mensaje de estado indicando si el comando se ha procesado correctamente o por el contrario se produjo algún fallo.</td></tr> </table>		EmberStatus status	Mensaje de estado indicando si el comando se ha procesado correctamente o por el contrario se produjo algún fallo.										
EmberStatus status	Mensaje de estado indicando si el comando se ha procesado correctamente o por el contrario se produjo algún fallo.												

Nombre: callback	ID: 0x06
Descripción: Este comando permite al dispositivo EM260 responder en caso de que tenga una respuesta pendiente de enviar.	
Parámetros del comando: ninguno	
Parámetros de respuesta: La respuesta a este comando puede ser cualquiera de las respuestas de comandos existentes.	

Nombre: leaveNetwork	ID: 0x20
Descripción: Este comando realiza una desconexión de la red ZigBee a la que esté conectado el dispositivo. Este comando provoca un callback <i>stackStatusHandler</i> indicando al <i>Host</i> que se ha desconectado con éxito de la red. Una vez desconectado el dispositivo de la red empleando este comando, la radio no se volverá a usar hasta que reciba un comando <i>joinNetwork</i> o <i>formNetwork</i> .	
Parámetros del comando: Ninguno	
Parámetros de respuesta: <div> <div>EmberStatus status</div> <div>Mensaje de estado indicando si la desconexión de la red se hizo efectiva o por el contrario se produjo algún error y la razón.</div> </div>	

4.2 El microcontrolador Atmel ATmega168

El dispositivo Atmega168 [42] es un microcontrolador CMOS de 8-bits de bajo consumo fabricado por Atmel Corporation, basado en las mejoras AVR de la arquitectura RISC (*Reduced Instruction Set Computer*). Consigue una gran optimización en su velocidad de procesamiento frente a su consumo, a partir de la ejecución de la mayoría de instrucciones en un solo ciclo de reloj, logrando con ello un rendimiento aproximado de 1 MIPS por MHz.

Su núcleo AVR combina una amplia lista de instrucciones con 32 registros de propósito general, los cuales están directamente conectados a la Unidad Aritmético Lógica (ALU), lo que permite el acceso a dos registros independientes en una sola instrucción ejecutada en un ciclo de reloj. Esta arquitectura permite obtener un código más eficiente alcanzando un rendimiento hasta diez veces mayor que los microcontroladores convencionales con arquitectura CISC.

4.2.1 Características generales

El dispositivo ATmega168 incluye una memoria Flash de 16 Kbytes, una memoria EEPROM de 512 bytes, 1024 bytes de memoria SRAM, 23 líneas de E/S de propósito general, 32 registros de propósito general, tres temporizadores/contadores con modo de comparación entre ellos, interrupciones internas y externas, un puerto serie programable (USART), interfaz serie de dos hilos (I²C), un puerto serie SPI, un convertidor analógico/digital de 8 canales con 10 bits de resolución, y un *watchdog* programable con oscilador propio.

En la Figura 4.13 se muestra el diagrama de bloques del dispositivo ATmega168; entre los bloques se pueden diferenciar las distintas funciones que incluye el microcontrolador antes comentadas (Memorias, SPI, USART, A/D Conv., etc.).

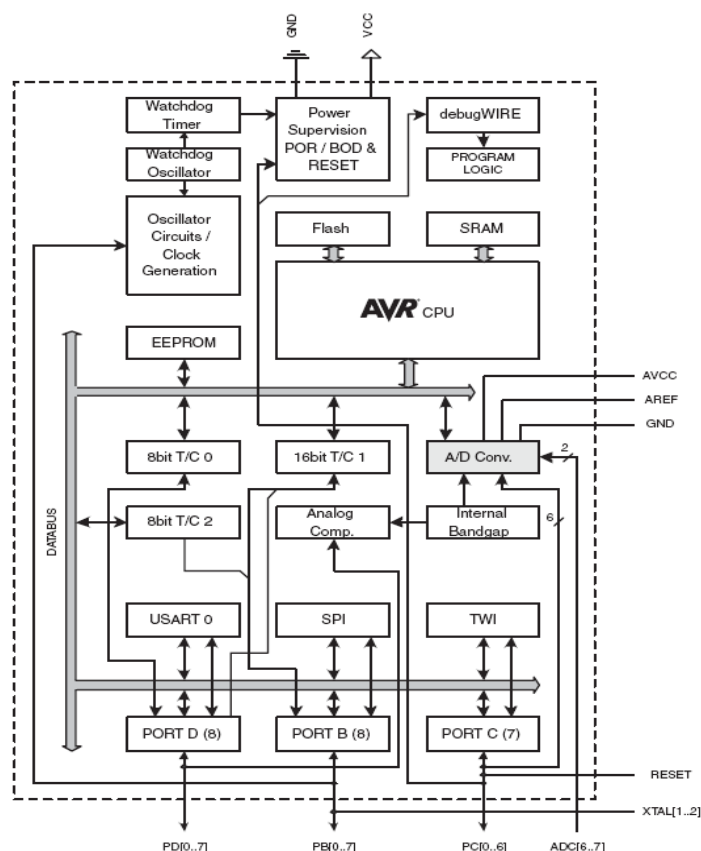


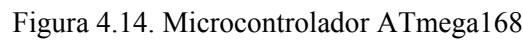
Figura 4.13. Diagrama de bloques del dispositivo ATmega168

Otro aspecto importante del microcontrolador es su consumo, ofreciendo 5 modos de bajo consumo que pueden ser seleccionados mediante software. El modo *Idle* mantiene detenida la CPU mientras continúan funcionando la memoria SRAM, los temporizadores/contadores, la USART, la interfaz serie de dos hilos (I²C), el puerto SPI y las interrupciones. El modo *Power-down* almacena el contenido de los registros y detiene todo el hardware del microcontrolador hasta que sea reactivado mediante una interrupción o un *hardware reset*. En el modo *Power-save* el dispositivo ATmega168 mantiene el temporizador asíncrono funcionando, mientras el resto del microcontrolador permanece inactivo, lo cual permite al usuario tener una base de tiempo activa con un consumo mínimo. El modo *ADC Noise Reduction* desconecta la CPU y todos los módulos de entrada/salida excepto el temporizador asíncrono y el convertidor analógico/digital, consiguiendo minimizar el ruido por conmutación durante la conversión. En el modo *Standby*, el oscilador sigue funcionando, mientras el resto del dispositivo está en modo inactivo, permitiendo combinar un arranque más rápido con un bajo consumo.

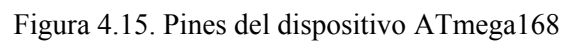
El microcontrolador ha sido fabricado utilizando la tecnología de memoria no volátil de alta densidad de Atmel. La memoria Flash que ofrece este microcontrolador es programable mediante ISP (*In System Programming*), con opciones de lectura mientras se está escribiendo a través del interfaz serie ISP, que da la opción de reprogramarse en la propia placa en la cual debe funcionar mediante el empleo de un programador convencional, o bien mediante un proceso de autoprogramación que se incluye en el núcleo de los microcontroladores AVR (*AVR core*).

Atmel distribuye para sus microcontroladores, entre ellos el dispositivo ATmega168, un entorno de desarrollo integrado, denominado *AVR Studio 4*. Esta herramienta es muy completa e incluye compilador de C, simulador, depurador y soporta ensamblador; además, cuenta con la posibilidad de adquirir kits de evaluación a un coste razonable. Finalmente, es de destacar que los microcontroladores de Atmel son compatibles con muchas herramientas de programación de código.

En la Figura 4.14 se puede observar el microcontrolador empleado en este PFC, un ATmega168 de Atmel, con un encapsulado de montaje superficial MLF (*Micro Lead Frame*), que consta de 32 pines, con una dimensión de 5 x 5 mm, y una distancia entre pin y pin de 0.50 mm.



En la Figura 4.15 se muestra la asignación de los 32 pines del ATmega168, mientras que en la Tabla 4.14 se describen las funciones de cada uno de ellos [44].



Pin	Señal	Función
1	PD3 (PCINT19/OC2B/INT1)	Puede realizar varias funciones, puerto de E/S digital, Output Compare o interrupción externa.
2	PD4 (PCINT20/XCK/T0)	Puede realizar varias funciones, Timer/Contador, puerto de E/S digital (pudiendo ser seleccionado como interrupción) o reloj externo de la USART.
3	GND	Masa
4	VCC	Tensión de alimentación digital.
5	GND	Masa
6	VCC	Tensión de alimentación digital.
7	PB6 (PCINT6/XTAL1/TOSC1)	Puede realizar varias funciones, entrada del reloj (oscilador o crystal) o cualquier fuente de reloj externa, si se usa el oscilador RC interno, puede funcionar como timer/contador asíncrono o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
8	PB7 (PCINT7/XTAL2/TOSC2)	Puede realizar varias funciones, entrada del reloj (oscilador o crystal) o cualquier fuente de reloj externa, si se usa el oscilador RC interno, puede funcionar como timer/contador asíncrono o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
9	PD5 (PCINT21/OC0B/T1)	Puede realizar varias funciones, timer/contador, Output Compare o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
10	PD6 (PCINT22/OC0A/AIN0)	Puede realizar varias funciones, comparador analógico, Output Compare o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
11	PD7 (PCINT23/AIN1)	Puede funcionar como comparador analógico o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
12	PB0 (PCINT0/CLKO/ICP1)	Puede realizar varias funciones, salida de reloj, Input Compare o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
13	PB1 (PCINT1/OC1A)	Puede realizar varias funciones, Output Compare o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
14	PB2 (PCINT2/SS/OC1B)	Puede realizar varias funciones, Output Compare, puerto de E/S digital (pudiendo ser seleccionado como interrupción) o cuando se utilice el bus SPI, como selección de esclavo (SS).
15	PB3 (PCINT3/OC2A/MOSI)	Puede realizar varias funciones, Output Compare, puerto de E/S digital (pudiendo ser seleccionado como interrupción) o salida de datos del bus SPI en modo maestro (MOSI).
16	PB4 (PCINT4/MISO)	Puede realizar varias funciones, puerto de E/S digital (pudiendo ser seleccionado como interrupción) o entrada de datos del bus SPI en modo maestro (MISO).
17	PB5 (PCINT5/SCK)	Puede funcionar como puerto de E/S digital (pudiendo ser seleccionado como interrupción) o salida del reloj cuando se utilice el bus SPI en modo maestro.
18	AVCC	Tensión de alimentación del convertidor analógico/digital.
19	ADC6	Su función es la de convertidor analógico/digital.
20	AREF	Tensión de referencia analógica para el convertidor analógico/digital.
21	GND	Masa del conversor (Masa analógica).
22	ADC7	Su función es la de convertidor analógico/digital.
23	PC0 (ADC0/PCINT8)	Su función es la de convertidor analógico/digital o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
24	PC1 (ADC1/PCINT9)	Su función es la de convertidor analógico/digital o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
25	PC2 (ADC2/PCINT10)	Su función es la de convertidor analógico/digital o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
26	PC3 (ADC3/PCINT11)	Su función es la de convertidor analógico/digital o puerto de E/S digital (pudiendo ser seleccionado como interrupción).
27	PC4 (ADC4/SDA/PCINT12)	Puede realizar varias funciones, convertidor analógico/digital, puerto de E/S digital (pudiendo ser seleccionado como interrupción) o señal de datos del bus I2C (SDA).
28	PC5 (ADC5/SCL/PCINT13)	Puede realizar varias funciones, convertidor analógico/digital, puerto de E/S digital (pudiendo ser seleccionado como interrupción) o reloj del bus I2C (SCL).
29	PC6 (RESET/PCINT14)	Su función es la de puerto de E/S digital (pudiendo ser seleccionado como interrupción) o RESET.
30	PD0 (RXD/PCINT16)	Su función es la de puerto de E/S digital (pudiendo ser seleccionado como interrupción) o entrada de datos (RXD) para la USART.
31	PD1 (TXD/PCINT17)	Su función es la de puerto de E/S digital (pudiendo ser seleccionado como interrupción) o salida de datos (TXD) para la USART.
32	PD2 (INT0/PCINT18)	Su función es la de puerto de E/S digital o interrupción externa.
33	GND	Masa. Está masa se refiere a la del bottom pad del integrado.

Tabla 4.14. Descripción de los pines del dispositivo ATmega168

4.2.3 Arquitectura AVR

La arquitectura AVR está basada en la filosofía de diseño de la arquitectura RISC para microcontroladores de 8 bits, que fue concebida por dos estudiantes del *Norwegian Institute of Technology* en 1996, y posteriormente refinada y desarrollada por Atmel Norway, la empresa subsidiaria de Atmel, fundada por los dos diseñadores del chip.

Los AVR utilizan una arquitectura Harvard modificada, con el bus de datos y el bus de memoria separados. Sin embargo, existen una serie de instrucciones especiales que permiten el acceso a la memoria de programa durante la ejecución del mismo.

Los microcontroladores AVR disponen de un *pipeline* sencillo de dos etapas (cargar y ejecutar), de forma que mientras una instrucción se ejecuta, la siguiente instrucción está lista para ser ejecutada en la memoria de programa, lo que permite ejecutar la mayoría de instrucciones en un único ciclo de reloj.

La ALU ofrece un alto rendimiento operando con los 32 registros de propósito general con los que tiene conexión directa, debido a que las operaciones aritméticas entre registros de propósito general o entre registros y un inmediato, son ejecutadas dentro de un ciclo de reloj. Esto permite que al ejecutarse una operación típica en la ALU, los dos operandos estén a la salida del vector de registros y el resultado quede almacenado al final del vector de registros, en un solo ciclo de reloj. Las operaciones de la ALU se dividen en tres categorías: aritméticas, lógicas y operaciones a nivel de bit. Después de realizar una operación aritmética, el registro de estado actualiza la información del resultado de la operación.

El AVR fue diseñado desde un comienzo para la ejecución eficiente de código C compilado. Como este lenguaje utiliza punteros para el manejo de variables en memoria, 6 de los 32 registros se pueden usar como registros apuntadores de direccionamiento indirecto a 16 bits para datos almacenados en memoria, siendo X, Y y Z los registros de 16 bits. Este es un compromiso que se realiza en arquitecturas de ocho bits, ya que su tamaño de palabra nativo de 8 bits (256 posiciones accedidas) es limitado para poder realizar el direccionamiento. Por otro lado, hacer que todo el banco superior de 16 registros de 8 bits tenga un comportamiento alterno, como un banco de 8 registros de 16 bits, complicaría mucho el diseño, violando la premisa original de simplicidad.

La arquitectura AVR tiene dos espacios principales de memoria, la *Memoria de Datos*, que es almacenada en la EEPROM, y la *Memoria de Programa*, que se encuentra en la memoria Flash. Además posee memoria SRAM, encontrándose todas ellas integradas en el chip. De este modo, se elimina la necesidad de emplear memorias externas.

La arquitectura AVR está diseñada para soportar frecuencias de reloj de hasta 20MHz, pudiendo llegar algunos dispositivos hasta los 32MHz. Sin embargo, las aplicaciones que requieran un bajo consumo de potencia necesitarán frecuencias de reloj menores. Algunas series de microcontroladores AVR integran un oscilador en chip, como en el caso del ATmega168, eliminándose así la necesidad de utilizar una fuente de reloj externa, o un circuito resonante. Con el fin de incrementar las posibilidades del sistema, también se incluye un *prescaler* para la señal de reloj, de forma que es posible dividir la frecuencia de reloj principal por un valor máximo de 256. El valor del *prescaler* puede ser configurado en el momento de la programación o durante la ejecución de la aplicación, aportando máxima flexibilidad.

Durante las interrupciones o llamadas a subrutinas, la dirección del *Contador de Programa* se almacena en la pila. La pila se localiza en la SRAM. Al inicio de un programa, el usuario deberá inicializar el SP en la rutina de reset. El espacio de memoria de I/O contiene 64 direcciones para las funciones periféricas del CPU, como Registros de Control, SPI, y otras funciones de I/O.

El repertorio de instrucciones de los AVR es más regular que la de la mayoría de los microcontroladores de 8 bits. Sin embargo, no es completamente ortogonal, es decir:

- Los registros punteros X, Y y Z tienen capacidades de direccionamiento diferentes del resto de registros.
- Los registros del 0 al 15 tienen diferentes capacidades de direccionamiento que los registros del 16 al 31.
- Los registros de I/O del 0 al 31 tienen distintas capacidades de direccionamiento que los registros del 32 al 63.
- La instrucción CLR (*Clear Register*) afecta los *flag* del registro de estado, mientras que la instrucción SER (*SEt Register*) no lo hace, a pesar de ser instrucciones complementarias.

En la Figura 4.16 se muestra el diagrama de bloques del núcleo del microcontrolador, que se corresponde con el bloque ‘AVR CPU’ de la Figura 4.13.

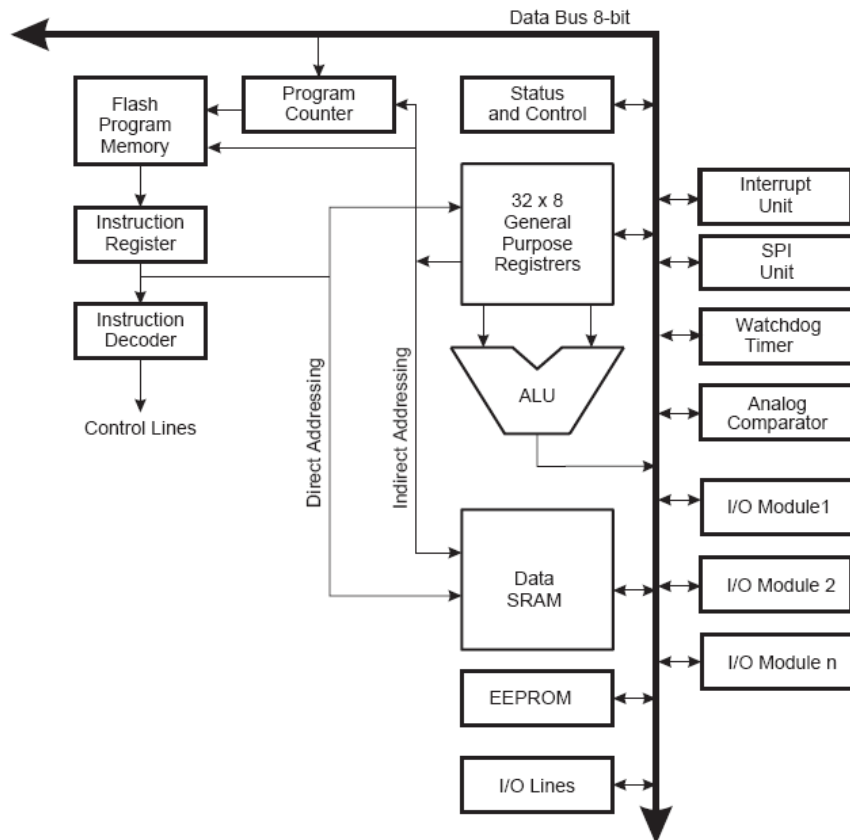


Figura 4.16. Arquitectura AVR

4.2.4 SPI (*Serial Peripheral Interface*)

El puerto SPI permite una comunicación síncrona de alta velocidad entre el dispositivo ATmega168 y los periféricos conectados a este sistema de comunicación. Las principales características son:

- Comunicación *Full-duplex* a través de tres hilos.
- Operación modo maestro o modo esclavo.
- Puede comenzar la transferencia por el bit menos significativo (LSB) o por el bit más significativo (MSB).
- Siete velocidades de transmisión configurables.
- *Flag* de fin de transmisión.
- *Flag* de protección de colisiones en la escritura.
- Modo SPI de doble velocidad (CK/2) como Maestro.

El interfaz SPI del ATmega168 está compuesto por 4 señales: una señal de reloj (SCK), una señal de datos entrantes (MISO), una señal de datos salientes (MOSI) y la señal de Chip Select (SS). En la Tabla 4.15 se muestra una descripción de cada señal.

Señal	Descripción
MOSI	La señal MOSI (<i>Master Output Slave Input</i>) se encarga de llevar los datos del maestro al esclavo, si el microcontrolador es maestro esta señal será de salida y si es esclavo será de entrada.
MISO	La señal MISO (<i>Master Input Slave Output</i>) se encarga de llevar los datos del esclavo al maestro, si el microcontrolador es maestro esta señal será de entrada y si es esclavo será de salida.
SCK	La señal SCK (<i>Signal Clock</i>) se encarga de transmitir el reloj para la sincronización de la transferencia, es generada por el maestro.
SS	La señal SS (<i>Slave Select</i>) se encarga de seleccionar el dispositivo esclavo, si el microcontrolador es maestro podrá ser configurada de entrada o de salida y la activación de un esclavo será mediante software, si el microcontrolador es esclavo será de entrada y será activa a nivel bajo.

Tabla 4.15. Señales implicadas en el interfaz SPI

En la Figura 4.17 se muestra la interconexión del interfaz SPI entre un Maestro y un Esclavo. El sistema SPI se puede ver como 2 registros de desplazamiento en serie de 8 bits interconectados y gobernados por un mismo reloj. El dispositivo Maestro inicia la comunicación cuando activa (a nivel bajo) la señal SS del dispositivo esclavo, con el cual desea comunicarse; en ese momento, el dispositivo Maestro y Esclavo preparan el dato que va a ser enviado en su respectivo registro de desplazamiento, generando el dispositivo maestro la señal de reloj (SCK) necesaria para el intercambio de datos. Los datos siempre se transferirán del Maestro al Esclavo por la línea MOSI, y del Esclavo al Maestro por la línea MISO. Después de cada paquete de datos, el Maestro se sincronizará con el Esclavo desactivando (a nivel alto) la señal SS.

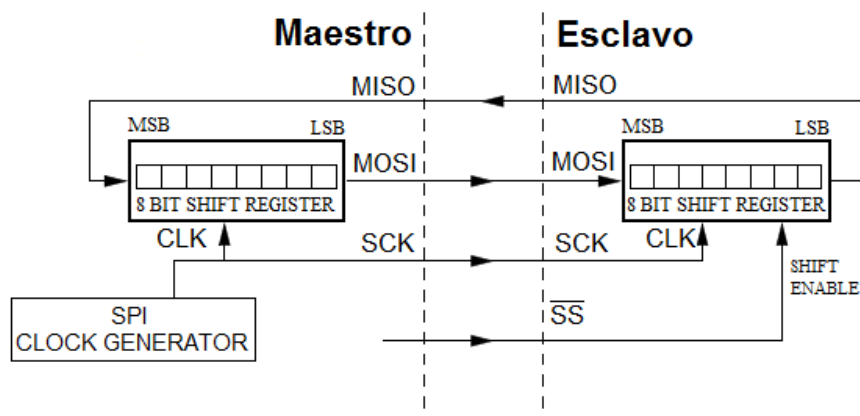


Figura 4.17. Interconexión lógica del interfaz SPI

Cuando el dispositivo ATmega168 está configurado en modo Maestro, el control de la señal SS del interfaz SPI no es automático y debe ser activado por software antes de comenzar la comunicación por el usuario. Una vez activada la señal SS, escribiendo un byte en el *SPI Data Register* (SPDR) comenzará a generarse el reloj del SPI, y se transferirán los 8 bits dentro de dispositivo Esclavo. Después de transferirse el byte, el generador de reloj SPI parará y se activará el flag de final de transmisión (SPIF). Si el bit de activación de interrupción SPI (SPIE) del registro de control SPI (SPCR) está a nivel alto, se solicita la interrupción. El dispositivo Maestro puede continuar y desplazar el siguiente byte escribiéndolo en el registro SPDR, o finalizar la transmisión desactivando (a nivel alto) la señal SS. El último byte que entró quedará almacenado en el registro del *Buffer* para su posterior uso.

Si el dispositivo ATmega168 está configurado en modo Esclavo, la interfaz SPI se mantendrá desactivado, con la señal MISO en triestado, mientras la señal SS esté desactivada (a nivel alto). En este estado, se puede actualizar el contenido del SPI Data Register (SPDR) por software, pero los datos no serán desplazados por la señal MISO aunque el dispositivo Maestro esté generando la señal de reloj (SCK), hasta que se active la señal SS (a nivel bajo). Cuando se desplace completamente un byte, se activará el *flag* de fin de transmisión (SPIF a nivel alto). Si el bit de activación de interrupción SPI (SPIE) del registro de control SPI (SPCR) está a nivel alto, la interrupción es solicitada. El dispositivo Esclavo puede preparar nuevos datos dentro del SPDR para ser enviados antes de leer los datos recibidos. El último byte que entró quedará almacenado en el registro del *Buffer* para su uso posterior.

En este PFC, la comunicación entre los dispositivos ATmega168 y EM260 se realizará mediante la interfaz SPI, de forma que el *host* será el ATmega168 y funcionará como dispositivo Maestro, y el EM260 será el dispositivo Esclavo. En la Figura 4.18 se muestra cómo quedaría la conexión de los dos dispositivos mediante la interfaz SPI.

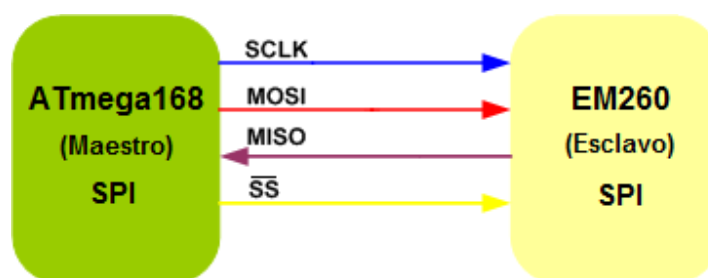


Figura 4.18. Interconexión mediante SPI de los dispositivos ATmega168 y EM260

Gracias a la conexión SPI, los dispositivos ATmega168 y EM260 se encontrarán conectados y podrán realizar la transmisión de datos entre ellos, permitiendo al ATmega168 comunicarse con el EM260 mediante EZSP. De este modo, se podrá crear y gestionar la red ZigBee a partir de la programación del microcontrolador ATmega168.

4.2.4.1 Registros de la interfaz SPI

La interfaz SPI del dispositivo ATmega168 cuenta con tres registros, SPCR, SPSR y SPDR, que proporcionan las funciones de control, estado y almacenamiento de datos, respetivamente. A continuación se realiza una descripción de cómo están organizados cada uno de estos registros.

• SPCR – SPI Control Register (Registro de control SPI)

El registro SPCR (*SPI Control Register*) permite configurar los distintos aspectos del periférico, como son las interrupciones, la habilitación, el modo, Master/Slave o la frecuencia para el SCK. En la Figura 4.19 se puede observar cómo se organiza este registro.

Bit	7	6	5	4	3	2	1	0	
0x2C (0x4C)	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Valor inicial	0	0	0	0	0	0	0	0	

Figura 4.19. Organización del registro SPCR

- Bit 7 (SPIE): Habilitación de interrupción SPI. Este bit es el encargado de habilitar la interrupción SPI si el SPIF del registro SPSR se pone a '1' y el bit de interrupciones del microcontrolador está habilitado (SREG a '1').
- Bit 6 (SPE): Habilitación de SPI. Cuando en el bit SPE se tiene un '1', el puerto SPI está habilitado. Este bit debe estar a '1' si se desea realizar cualquier operación en el puerto.
- Bit 5 (DORD): Orden de los datos. Cuando el bit DORD está a '1', el primer bit de datos que se transmite es el LSB, si está a '0', el primer bit de datos transmitido será el MSB.
- Bit 4 (MSTR): Selección del modo Maestro/Esclavo. Con el bit MSTR a '1' se activa el modo SPI Maestro, y el modo Esclavo cuando esté a '0'

- Bit 3 (CPOL): Polaridad del reloj. Si el bit CPOL está a '1', la señal de reloj SCK se mantendrá a nivel alto cuando esté en *idle*; si CPOL está a '0' se mantendrá a nivel bajo.
- Bit 2 (CPHA): Fase del reloj. El bit CPHA determina cuándo se muestrea el dato; si CPHA está a '1', se muestrea en el último flanco del pulso de reloj, y si está a '0' en el primero. La configuración del bit CPHA, combinado con CPOL, indicará si el dato es muestreado con el flanco de subida del reloj o con el franco de bajada.
- Bit 1, 0 (SPR1, SPR0): Velocidad del reloj SPI. Estos dos bits controlan la velocidad de la señal de reloj SCK cuando el microcontrolador se configura como Maestro. Cuando el microcontrolador se configura como Esclavo, SPR1 y SPR0 no tienen efecto alguno. En la Tabla 4.16 se muestra la relación que existe entre la frecuencia del reloj SCK y la frecuencia de reloj del oscilador f_{osc} respecto a los bits que controlan la velocidad (SPI2X, SPR1 y SPR0).

SPI2X	SPR1	SPR2	Frecuencia SCK
0	0	0	$f_{osc}/4$
0	0	1	$f_{osc}/16$
0	1	0	$f_{osc}/64$
0	1	1	$f_{osc}/128$
1	0	0	$f_{osc}/2$
1	0	1	$f_{osc}/8$
1	1	0	$f_{osc}/32$
1	1	1	$f_{osc}/64$

Tabla 4.16. Relación entre la señal SCK y la frecuencia del reloj

• *SPSR – SPI Status Register (Registro de estado SPI)*

El registro de estado informa mediante los distintos Flag/bits el estado de la interfaz SPI. En la Figura 4.20 se observa cómo se organiza este registro.

Bit	7	6	5	4	3	2	1	0	
0x2D (0x4D)	SPIF	WCOL	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	SPI2X	SPSR
Read/Write	R	R	R	R	R	R	R	R/W	
Valor inicial	0	0	0	0	0	0	0	0	

Figura 4.20. Organización del registro SPSR

- Bit 7 (SPIF): *Flag* de interrupción SPI. Cuando se completa la transferencia serie, el *flag* SPIF se pone a '1', la interrupción SPI de transferencia completada se genera siempre que el bit SPIE del registro SPCR está a '1' y las interrupciones generales están habilitadas. El *flag* SPIF se pone a '0' por hardware, una vez que ejecute el vector de interrupción correspondiente, o accediendo al registro SPDR después de haber leído el SPIF a '1'.
- Bit 6 (WCOL): *Flag* de colisión de escritura. El *flag* WCOL se pone a '1' si se escribe en el registro SPDR durante la transferencia de un dato. El *flag* WCOL se pone a '0' accediendo al registro SPDR después de haber leído el WCOL a '1'.
- Bit 5..1 (Res): Bits reservados. Estos bits son reservados en el dispositivo ATmega168 y al leerlos siempre se obtendrá '0'.
- Bit 0 (SPI2X): Bit de doble velocidad SPI. Cuando el bit SPI2X es puesto a '1', se duplicará la velocidad de la interfaz SPI (frecuencia de SCK) si el microcontrolador está configurado como Maestro. Cuando el ATmega168 está configurado como Esclavo, la frecuencia máxima de trabajo que se puede garantizar es de $f_{osc}/4$.

• *SPDR – SPI Data Register (Registro de datos del SPI)*

El registro SPDR es el registro de lectura y escritura donde se encuentran los datos transferidos. La transmisión o recepción de un byte, sólo se inicia al realizar una escritura en el registro, cuando el microcontrolador esté configurado como Maestro. En la Figura 4.21 se puede observar cómo se organiza este registro.

Bit	7	6	5	4	3	2	1	0	
0x2E (0x4E)	MSB							LSB	SPDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Valor inicial	X	X	X	X	X	X	X	X	Undefined

Figura 4.21. Organización del registro SPDR

Capítulo 5

Diseño hardware del nodo sensor inalámbrico

Una vez descritos en el capítulo anterior de forma detallada los componentes más significativos que se han empleado para llevar a cabo el presente PFC, en este capítulo se explica el proceso que se ha seguido para el desarrollo del nodo sensor inalámbrico, además de detallar la función de las partes que conforman el diseño propuesto. El capítulo comienza con una descripción general del hardware del nodo sensor inalámbrico, comentando cada uno de los bloques funcionales que componen el diseño. En los apartados siguientes se profundiza con mayor detalle en cada uno de ellos, describiendo los aspectos más relevantes en cada caso, como conexionado, modo de operación, descripción de los pines de E/S, etc. Así, abordando de forma separada cada bloque funcional del esquemático propuesto, se facilita el proceso de diseño y se consigue una mayor comprensión del funcionamiento del módulo.

5.1 Descripción general

Una vez estudiado el protocolo de comunicación inalámbrica ZigBee, y tras realizar una descripción de los componentes principales en los que se basa la realización de este PFC, se puede comenzar el diseño del nodo a desarrollar, que consiste en un módulo de comunicaciones

inalámbricas que funciona como plataforma sobre la cual se pueden implantar diferentes sensores. En el caso de este PFC se ha desarrollado otro módulo que contiene los sensores y el hardware necesario para completar el módulo y demostrar la funcionalidad del dispositivo. Este módulo sensor se conecta al módulo de comunicaciones inalámbricas mediante el conector de expansión, el cual provee de las señales necesarias para su correcto funcionamiento.

El módulo de comunicaciones está diseñado alrededor del co-procesador de comunicaciones EM260 y del microcontrolador ATmega168; el primero se encarga de implementar la parte de radiofrecuencia y la pila del protocolo ZigBee, mientras que el segundo realizará el procesamiento de los datos recibidos por el sensor y la gestión de la red. Para conseguir la funcionalidad completa del nodo, será necesario agregar varios bloques funcionales más, como son el bloque de alimentación, el bloque sensor, una parte RF externa, y el bloque de expansión. En la Figura 5.1 se presenta el diagrama de bloques funcionales del módulo de comunicaciones inalámbricas desarrollado.

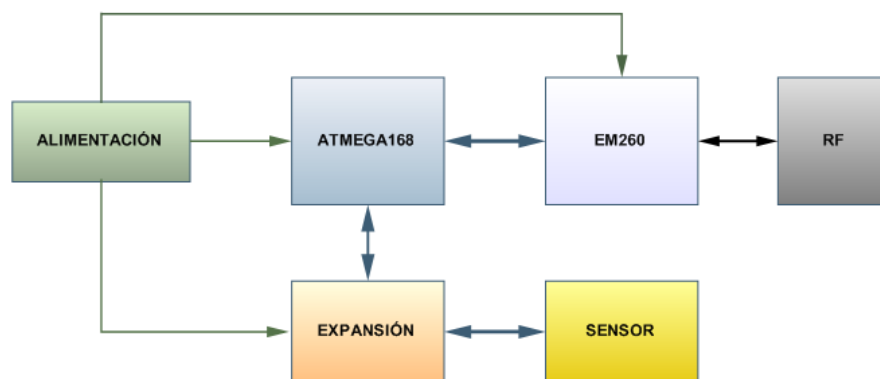


Figura 5.1. Diagrama de bloques funcionales del módulo

- El *bloque de alimentación* es el encargado de suministrar a los otros bloques la tensión necesaria para su correcto funcionamiento. Este bloque puede funcionar mediante una pila de botón (de 3V y 20mm) o una fuente de alimentación externa, de forma que en el caso de usar una fuente externa, la tensión es regulada mediante el regulador de tensión incorporado al bloque de alimentación.
- El *bloque de radiofrecuencia* (RF) está compuesto por un balún, el cual convierte las señales RF de transmisión y recepción balanceadas provenientes del dispositivo EM260, a una sola salida no balanceada, además de realizar una adaptación de impedancias. Estas dos funciones del balún son necesarias para realizar la conexión de la antena con la parte

de radiofrecuencia del dispositivo EM260. En el bloque RF también se encuentra un filtro de armónicos, que es responsable de atenuar los armónicos de segundo orden existentes. El bloque se completa con una antena en chip y un conector coaxial tipo SWD, que va a permitir al módulo operar mediante la conexión de una antena externa.

- El *bloque EM260*, formado por el dispositivo del mismo nombre, ofrece todas las funciones de red ZigBee ya desarrolladas, integrando un radio-transceptor compatible con el estándar IEEE 802.15.4, un procesador de red y memoria para ejecutar el protocolo completo de la pila de red ZigBee. Estas funciones son accesibles mediante la interfaz serie SPI y dos señales adicionales, permitiendo implementar la prestaciones de red ZigBee con el microcontrolador usado en el presente PFC, el ATmega168.
- El *bloque ATmega168* va a permitir procesar toda la información proveniente del bloque sensor y del bloque de expansión para ser enviada de forma inalámbrica con la ayuda del dispositivo EM260, mediante el protocolo ZigBee. Además se encarga de gestionar la red junto con éste, con el que está conectado mediante el protocolo SPI, y dos señales adicionales, como se comentó anteriormente.
- Con el *bloque de expansión* se pretende facilitar la conexión de cualquier tipo de sensor y ampliar las posibilidades del módulo, pudiendo conectarse con otros dispositivos o añadir hardware complementario que implemente otras funciones en futuras aplicaciones prácticas. El bloque de expansión está compuesto por un conector externo que está conectado a los pines del ATmega168, los cuales permiten que la expansión se realice mediante varias interfaces de comunicación, como pueden ser SPI, USART, I²C, etc. También ofrece la posibilidad de usar varias interrupciones, convertidores A/D, puertos E/S, etc.
- El *bloque sensor* implementa los tres tipos de sensores que se emplean en el presente PFC; un sensor de temperatura, un sensor de presión y un sensor de movimiento PIR. En el bloque sensor se obtienen las variables que se desean medir o controlar por los sensores para ser transmitidas mediante el conector de expansión al microcontrolador ATmega168. Además de los tres sensores empleados y el hardware necesario para acondicionar ciertas señales de los sensores, el bloque sensor se completa con 2 diodos LEDs (rojo y verde), un interruptor y un pulsador encargado de inicializar el módulo.
- Además de lo comentado en cada bloque, se ha hecho necesaria la inclusión de varios conectores. Entre ellos, un conector específico para la programación del *firmware* ZigBee

en el dispositivo EM260. Este conector se compone de 10 pines, que se conecta a las líneas de la interfaz de depuración y programación del EM260 (SIF); mediante esta interfaz se programa la memoria del microcontrolador que lleva integrado el dispositivo EM260, pudiendo actualizar el *firmware* siempre que sea necesario. El módulo cuenta con otro conector más, que es el encargado de la programación del ATmega168; este conector permite reprogramar el microcontrolador mediante una interfaz SPI sin necesidad de extraerlo del circuito, ya que para la programación de la memoria flash el microcontrolador dispone de la tecnología On-chip ISP (*In-System Programming*). El conector está compuesto por 6 pines, aunque el conector ha sido ampliado añadiendo 3 pines adicionales que van a permitir la comunicación mediante el protocolo EZSP al dispositivo EM260, permitiendo enviar comandos y recibir las respuestas directamente del dispositivo EM260, sin pasar por el microcontrolador, facilitando con ello la depuración del dispositivo EM260.

En la Figura 5. se puede observar el esquemático completo del circuito desarrollado en este Proyecto Fin de Carrera, mostrándose las diferentes regiones con el fin de delimitar los distintos bloques funcionales que constituyen el diseño, exceptuando el bloque sensor que se ha elaborado en una nueva placa de circuito impreso y su diseño se encuentra descrito en el apartado 5.7.

Implementación de un sensor inalámbrico de propósito general con bajo consumo

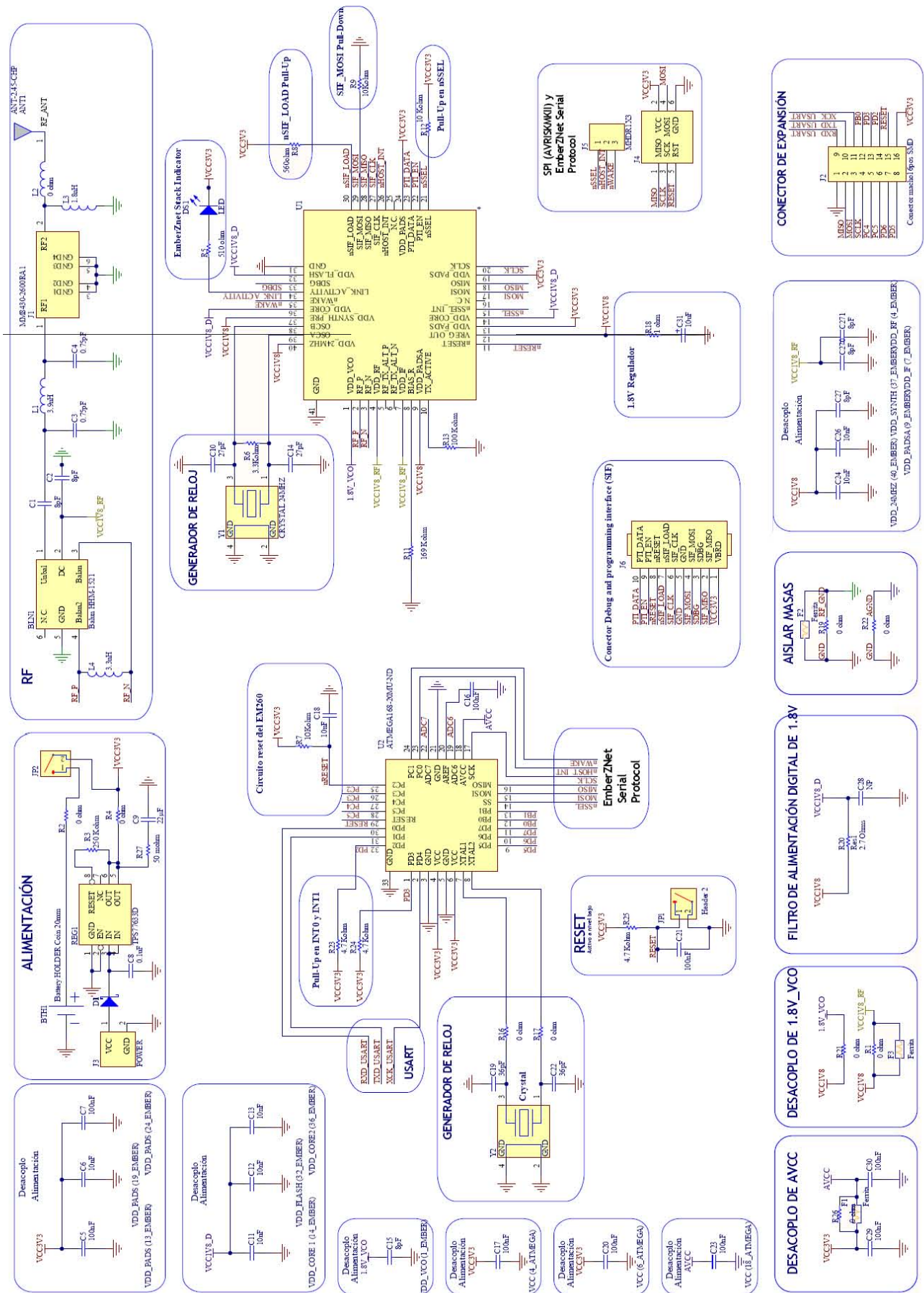


Figura 5.2. Esquemático completo del módulo de comunicaciones inalámbricas desarrollado

Durante la implementación de cada bloque se han tenido que tomar ciertas decisiones en base a unos criterios de diseño básicos. Así, los principales criterios que se han seguido en el desarrollo de este módulo son los siguientes:

- *Bajo consumo:* El consumo de potencia es uno de los principales aspectos que se han tenido en cuenta en el diseño. Debido a la posibilidad del nodo de funcionar con una pila de botón, es de vital importancia minimizar su consumo para conseguir maximizar la vida de la pila. Además como ya se comentó, el protocolo ZigBee y a su vez el co-procesador EM260 se caracteriza por su bajo consumo de potencia. Además, el ATmega168 también ofrece muy buenas prestaciones en consumo de potencia. Para seguir manteniendo unos bajos consumos de potencia en el nodo no se deben obviar los demás bloques, intentando optimizar el consumo de todos ellos dentro de sus posibilidades.
- *Solución completa:* Este criterio establece la necesidad de que el módulo incluya todo lo necesario para que funcione sin ningún componente externo. Es importante en el diseño la integración de todos los componentes en la PCB, facilitándose el trabajo en el momento de hacer uso del módulo, o, en el caso de que se anexe a un sistema más complejo, no precise de requisitos difíciles de obtener. Además, una solución completa influirá en el siguiente criterio, sobre todo en la fabricación, donde la integración de los componentes puede reducir el tamaño del módulo.
- *Tamaño reducido:* Tratándose en este PFC de implementar un sensor inalámbrico de propósito general, el módulo debe ser lo más pequeño posible para que sea capaz de funcionar bajo una amplia variedad de aplicaciones. Al ser un dispositivo inalámbrico, cuenta con la ventaja de ser portátil, siendo esta portabilidad una de las principales características de las redes de sensores inalámbricas, la cual depende mucho del tamaño, ya que hay aplicaciones, por ejemplo de localización o seguimiento, que precisan de módulos lo más reducidos posibles. Además, un tamaño reducido conlleva otros beneficios en el módulo, como pueden ser una mayor facilidad para ser implantado en otros sistemas, o el ahorro en costes de fabricación que ofrece, debido a un menor tamaño de la PCB.

A continuación, se describe con más detalle la implementación de los bloques funcionales que componen el módulo desarrollado en este PFC.

5.2 Bloque de alimentación

A la hora de alimentar el módulo se ha de tener en cuenta la tensión de alimentación necesaria para cada bloque por separado. Como se muestra en la Tabla 5.1, los componentes de cada bloque establecen las tensiones de alimentación de los mismos, la cual debe ser suministrada para conseguir el funcionamiento adecuado de cada bloque.

Bloque	Principal componente	Tensión de alimentación (V)
ZigBee	EM260	2.1 ~ 3.6
Microcontrolador	ATmega168	2.7 ~ 5.5
Expansión	-	Sin requerimientos
Sensor	TMP121AIDBVT	2.7 ~ 5.5
	MP3H6115A6U	2.7 ~ 3.3
	SENSOR MOTION PIR ⁴	2.7 ~ 5.5 ⁴

Tabla 5.1. Tensión de alimentación de los diferentes bloques del módulo

El diseño realizado ofrece dos posibilidades diferentes para alimentar al circuito; la primera de ellas es mediante alimentación externa a través de un conector, mientras que la segunda se basa en emplear una pila de litio de tipo botón de 3V.

De la Tabla 5.1 se deduce el rango de valores de alimentación en los cuales el módulo puede funcionar, debiendo estar comprendido éste, entre 2.7V y 3.6V. En el caso de hacer uso de alimentación externa mediante conector, el módulo cuenta con un regulador de tensión para garantizar una mayor estabilidad, ya que le protegerá de la posible variación de diferencia de potencial, así como del ruido o de las descargas eléctricas que pueda provocar la fuente de alimentación externa que se emplee. En el caso de hacer uso de la alimentación mediante pila, no se emplea el regulador de tensión, debido a que la variación de tensión suministrada por la pila y el ruido que puede presentar es despreciable, además de que la tensión de la pila está dentro del rango de alimentación de los diferentes bloques y el regulador de tensión haría disminuir esta tensión y

⁴ El sensor motion PIR empleado (555-28027) necesite una tensión de alimentación de 3.3V a 5V, lo que ha echo necesario el uso de un convertidor *Buck-boost* que ofrece una tensión de salida de 5V, ante una entrada comprendida entre 2.7V y 5.5V, como se muestra en la Tabla 5.1, de este modo mantenemos los rangos de valores adecuados con el resto de bloques y la posibilidad de emplear pila de botón. En el apartado del bloque sensor (apartado 5.7.3) se explica más en detalle la implementación de esta parte del diseño.

aumentaría el consumo del módulo. En la Figura 5.3 se puede observar el esquemático completo del bloque de alimentación del módulo.

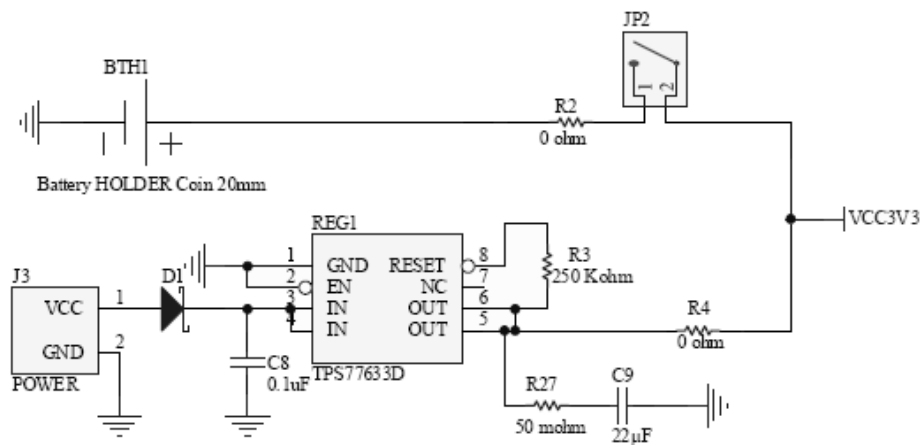


Figura 5.3. Esquemático del bloque de alimentación

A continuación se presentará una descripción más detallada de cada una de las opciones de alimentación soportadas por el módulo desarrollado en el presente PFC: alimentación externa o alimentación con pila. Además, se describirán las técnicas empleadas para desacoplar las diferentes masas y alimentaciones que comprende el módulo.

5.2.1 Alimentación externa

La opción de alimentación externa provee al circuito de dos grandes pads a los que se puede conectar una fuente de tensión. Se asume que dicha tensión no ha sido regulada y que proviene de una fuente de alimentación ajena al circuito. La conexión se puede realizar mediante conector de tipo cocodrilo o conector simple, en el caso de soldar los dos pines. En la parte inferior de la Figura 5.3 se observa la configuración del circuito con alimentación externa, donde tras los pads de conexión a la fuente externa (J3) se encuentra un diodo Schottky (D1), cuya función principal es la de evitar realimentaciones inversas no deseadas, asegurando que sólo circule corriente en el sentido previsto en el diseño, y no en sentido inverso. El diodo Schottky empleado es el ZLLS2000TA [45] del fabricante Zetex, que se caracteriza principalmente por presentar una caída de tensión baja, concretamente en torno a los 0.3V, haciendo, por tanto, pequeña la pérdida de tensión en esta parte del bloque de alimentación. A continuación, conectado en serie al diodo Schottky se localiza el regulador de tensión integrado, que será el encargado de regular y estabilizar la tensión de la fuente externa a la tensión deseada para el nodo sensor inalámbrico, en el caso particular de este PFC, 3.3V. El regulador utilizado es el TPS77633D [46], que se describirá

detalladamente en el siguiente apartado. Para terminar la rama de alimentación externa se ha añadido una resistencia de 0Ω (R4), que va a permitir deshabilitar o habilitar vía hardware esta forma de alimentación, dependiendo de la aplicación final del nodo sensor inalámbrico.

5.2.1.1 Regulador 3.3V

Como ya se comentó anteriormente, cuando se utiliza la configuración de alimentación externa, se hace necesario regular y estabilizar la tensión proporcionada por ésta, con el fin de obtener la tensión de trabajo idónea en el módulo. En este PFC se ha elegido la tensión de trabajo de 3.3V en el caso de operar con la configuración de fuente externa. Para obtener esta tensión de 3.3V estabilizada se ha empleado el regulador integrado antes comentado, TPS77633D [46], fabricado por Texas Instruments. Además, es recomendable que la fuente externa entregue a la entrada del regulador, como mínimo, 500mV más de la tensión de trabajo requerida (3.3V), debido a la caída de tensión producida por la estructura del circuito interno del regulador, entre la entrada y la salida.

Es recomendable que la fuente de alimentación externa incluya previamente un limitador de corriente, como máximo de 500mA, con el fin de evitar la destrucción del circuito en caso de producirse un cortocircuito accidental.

En la Figura 5.4 se pueden observar los pines de E/S del regulador integrado TPS77633D empleado en el módulo, describiendo en la Tabla 5.2 la función de cada pin.

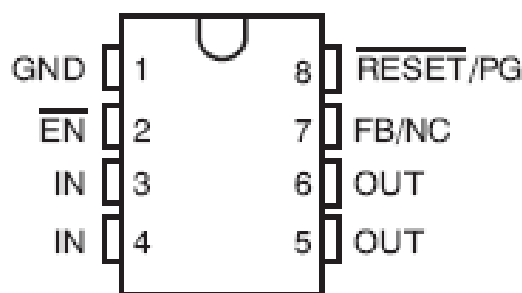


Figura 5.4. Asignación de los pines del TPS77633D

Pin	Señal	Descripción
1	GND	Masa
2	EN	Señal de habilitación, activa a nivel bajo. Nos permite habilitar o deshabilitar el funcionamiento del regulador.
3	IN	Tensión de entrada del regulador. Debe ir cableada con el otro pin de entrada del regulador (pin 4).
4	IN	Tensión de entrada del regulador. Debe ir cableada con el otro pin de entrada del regulador (pin 3).
5	OUT	Tensión de salida regulada y estabilizada. Debe ir cableada con el otro pin de entrada del regulador (pin 6).
6	OUT	Tensión de salida regulada y estabilizada. Debe ir cableada con el otro pin de entrada del regulador (pin 5).
7	FB/NC	Señal de <i>FeedBack</i> , En este caso se dejará sin conectar, sólo se conecta cuando el regulador funcione en modo ajustable. En este PFC la tensión necesaria es fija, 3.3V.
8	RESET/PG	Señal de salida (<i>Power-Good</i>) que indica si la tensión de salida regulada se encuentra dentro de unos márgenes óptimos.

Tabla 5.2. Descripción de los pines del TPS77633D

Además de las señales de entrada, salida, masa, y la de ajuste, que típicamente proporcionan los reguladores, el regulador TPS77633D también provee una señal indicadora de su buen funcionamiento. Es la señal denominada RESET/PG (*Power-Good*). Para ello, un comparador interno provoca un nivel lógico bajo en la salida de este pin cuando la tensión de salida del regulador disminuye hasta encontrarse en un valor comprendido entre el 92% y el 98% de su valor nominal. Se emplea en caso de que se desee incluir un indicador de batería baja o para la activación automática de alimentación auxiliar externa.

En la Figura 5.5 se muestra el esquema del regulador TPS77633D incorporado en el módulo, con todas sus señales acondicionadas siguiendo las especificaciones de su hoja de características.

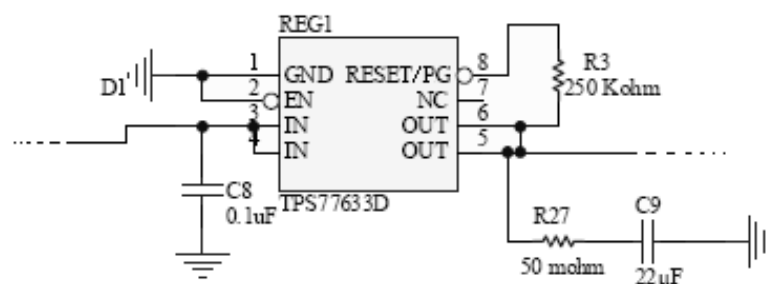


Figura 5.5. Esquema del circuito regulador de tensión

Como se puede observar, y siguiendo las indicaciones del *datasheet*, proporcionado por el fabricante, las dos entradas (IN) están cableadas entre sí, además de añadirse el condensador de 0.1uF (C8) entre la entrada y masa. Si bien este condensador no es obligatorio, resulta recomendable, debido a que permite mejorar la respuesta de carga transitoria, así como lograr un mayor rechazo al ruido de alimentación de entrada del regulador.

La señal enabled (EN) está conectada a la señal GND del regulador, y ésta a su vez está conectada a la masa del circuito, como se comentó anteriormente en la descripción del *pinout*, la señal EN es activa a nivel bajo, lo que quiere decir que el regulador de tensión siempre está activo. Por otro lado, la señal FB se ha dejado sin conectar, ya que en este caso el regulador busca una tensión regulada fija (3.3V) y sólo se usa en el caso de que sea ajustable. Siguiendo las recomendaciones del *datasheet* se ha dejado sin conectar.

Las salidas del regulador (OUT) (pin 5 y pin 6), también han sido conectadas entre sí, al igual que las señales de entrada. Además se ha conectado la señal RESET/PG a la salida mediante una resistencia de 250K Ω (R3), aunque el *datasheet* señala que también se puede dejar sin conectar en caso que no requiera de esta función.

Siguiendo las especificaciones, a la salida del regulador de tensión se requiere un condensador, que se conecta entre la salida y masa (C9), siendo encargado de estabilizar el control de bucle interno de regulador. El valor mínimo recomendable es de 10uF y su ESR (resistencia serie equivalente) deberá estar comprendida entre 50m Ω y 1.5 Ω . En este caso se ha seleccionado un condensador de 22uF con bajo ESR y se ha colocado en serie una resistencia de 50m Ω , la cual va a garantizar el mínimo ESR exigido en el *datasheet* del regulador.

La salida del regulador, con una tensión de 3.3V, será el nodo de alimentación principal, proporcionando la tensión de alimentación a todo el hardware del circuito.

5.2.2 Alimentación con pila

La segunda alternativa de alimentación que ofrece el módulo desarrollado en este PFC es mediante el uso de una pila. Se ha decidido utilizar una pila de botón de 3V de tensión y 20mm de tamaño, debido a que proporciona una tensión suficiente para alimentar a todos los bloques del módulo sin necesidad de regulador, puesto que los 3V están comprendidos entre los 2.7V de tensión mínima y los 3.6V de tensión máxima necesarios para la alimentación, como se pudo observar en la Tabla 5.1. Además, cuenta con un tamaño reducido al ser una pila de botón de 20

mm. De esta manera, la opción de emplear pila, permite optimizar el consumo y el tamaño del diseño final, permitiendo al nodo aumentar su portabilidad.

La pila recomendada para un óptimo funcionamiento es la pila de litio modelo CR2032 fabricada por Panasonic, ya que ofrece unas buenas prestaciones, entre ellas su capacidad nominal, que está en torno a los 225mAh con el voltaje nominal antes comentado de 3V, permitiendo una larga autonomía en nodos con bajo consumo. En la Figura 5.6 se muestra una fotografía de la pila CR2032 empleada en el módulo.



Figura 5.6. Pila de litio modelo CR2032

En la parte superior de la Figura 5.3 se muestra el esquema de alimentación mediante pila, en el que se puede observar que la configuración es muy simple, ya que consta de una resistencia de 0Ω en serie con el terminal positivo de la pila, que va a permitir habilitar la posibilidad de usar esta alimentación mediante pila por hardware (si la resistencia es soldada). Además, la configuración incluye unos pines, que mediante un *jumper*, permiten tener conectada o interrumpir la alimentación de una manera simple, sin necesidad de estar retirando la pila para interrumpir la alimentación. El esquema termina en el nodo de alimentación principal para la alimentación del módulo. En esta configuración se ha excluido el regulador de tensión debido a que la pila proporciona una tensión ya regulada y estabilizada, además de evitar la caída de tensión que podrían sufrir los 3V nominales que proveen la pila por el paso del regulador; igualmente se ha suprimido el diodo, ya que si hay caída de tensión en los 3V, se aproximaría al margen mínimo de alimentación necesario en el nodo sensor inalámbrico.

5.2.3 Desacoplo de masa y alimentación

Como se ha visto, el módulo diseñado incluye una serie de componentes como el dispositivo EM260, o el ATmega168, que precisan de una adecuada alimentación para su correcto

funcionamiento, además de tener en cuenta que las señales que se generan o se reciben se referirán a un nivel de potencial determinado, que es masa, es decir 0 V. En consecuencia, la alimentación y la masa se deben distribuir por todo el circuito para llegar a los componentes que la necesiten, con el objetivo de mantener su integridad, es decir, que las alimentaciones conserven su valor de continua sin presencia de señales ajenas, y que el valor de tensión de todos los puntos de las líneas de masa sea 0V. Esto último es especialmente importante por cuanto todos los valores se refieren a él y, en consecuencia, cualquier cambio en el nivel de referencia afectará a todas las señales del módulo, y en particular al receptor, ya que es muy sensible al ruido. Por lo tanto, es necesario que la alimentación sea lo más estable posible con la intención de que el módulo diseñado presente las mejores prestaciones posibles en cuanto a estabilidad y nivel de ruido.

Para evitar que tanto las alimentaciones como las masas posean ruido, acoplamientos parásitos o componentes de rizado en alta frecuencia, en el módulo diseñado se han empleado diferentes tipos de desacoplo, que se describen a continuación.

5.2.3.1 Desacoplo de alimentación

Es común que la misma tensión de alimentación se aplique a varios pads del circuito integrado o a pads de diferentes integrados. Así, cualquier pad que pueda generar ruido o produzca variaciones, repercute en todas las demás alimentaciones. De este modo, las diferentes tensiones aplicadas a los pads se deben mantener lo más estables posibles frente a las variaciones que se puedan producir, como la presencia de ruido en las líneas de alimentación. Para minimizar estos efectos se han empleado condensadores de desacoplo de alimentación, situados entre los pads de alimentación y masa, con vistas a mantener esos puntos de tensión constantes y ayudar a proporcionar los picos de consumo de los pads que requieran un consumo pulsante. Los condensadores de desacoplo deben ser cerámicos y estar ubicados lo más cerca posible de los pads de alimentación.

Además del suministro de picos de intensidad, que los circuitos integrados precisan en las conmutaciones para mantener su tensión constante, los condensadores de desacoplo producen más efectos, como pueden ser:

- La reducción de las perturbaciones de alta frecuencia que pudieran llegar a los circuitos integrados, debido a su configuración filtro paso-bajo.

- El desacoplo de los efectos capacitivos propios de los circuitos integrados respecto de las autoinducciones que presentan las líneas de alimentación, evitando las oscilaciones de alta frecuencia que se podrían producir por acoplo LC.
- La división en dos que realiza del bucle que conforman las líneas de alimentación (ya que el condensador actúa como cortocircuito para frecuencias altas) y consigue que el bucle de alimentación en la parte del circuito integrado sea de área mínima (por hallarse el condensador muy próximo a los pads de los integrados).

En la Figura 5.7 se muestra el esquema de todos los condensadores de desacoplo que se han empleado en el módulo desarrollado.

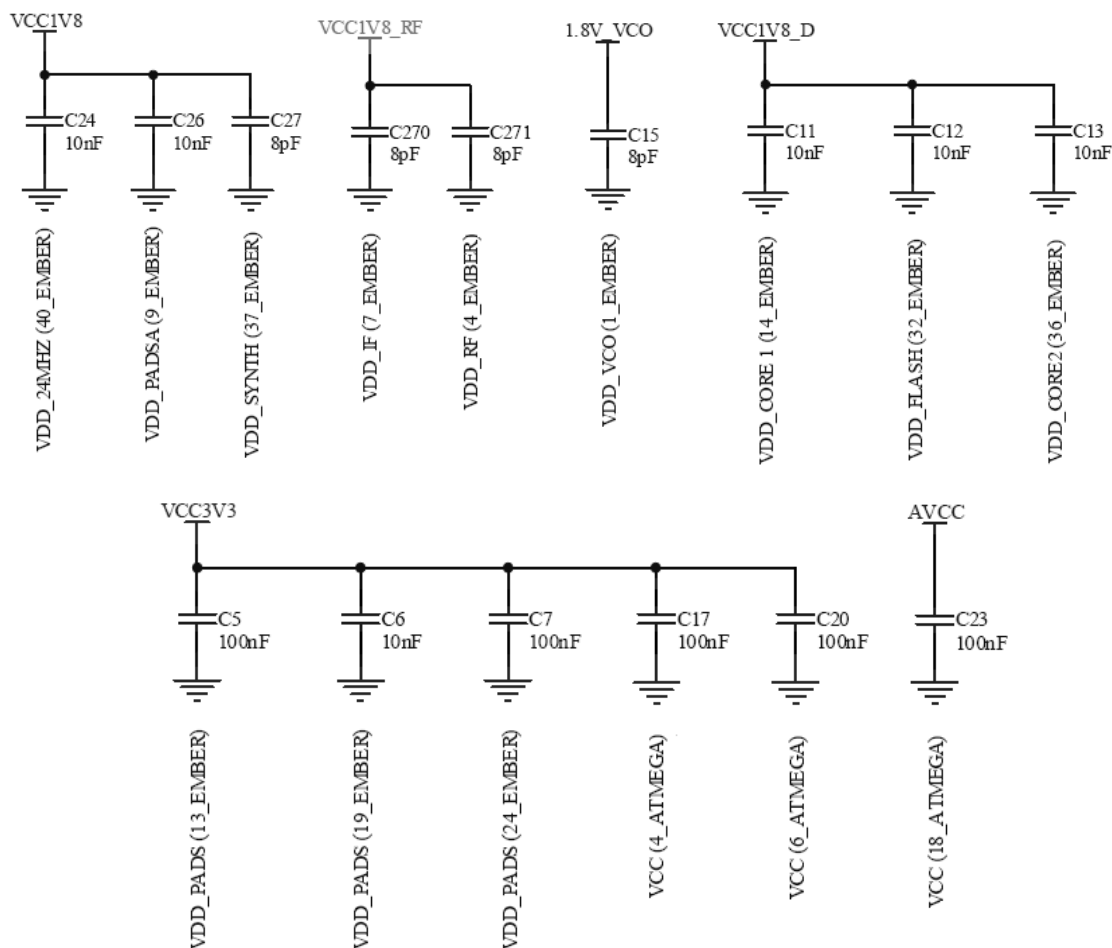


Figura 5.7. Condensadores de desacoplo del módulo

Como se puede observar en cada una de las alimentaciones, se han dispuesto tantos condensadores de desacoplo como pads hay conectados a cada alimentación, indicándose en forma de títulos

verticales, el nombre, numeración y componente (estos dos últimos entre paréntesis) del pad al que va conectado (lo más cerca posible) cada condensador.

5.2.3.2 Desacoplo entre masas

El ruido en el plano de masa es un problema más grave que el ruido en el plano de alimentación. El ruido en el plano de masa es producido por los transitorios de alimentación y por las corrientes de retorno SSO (*Simultaneous Switching Outputs*). Como se presentó en el apartado anterior, los transitorios de la alimentación pueden ser controlados con el uso adecuado de condensadores de desacoplo, pero los ruidos de masa no pueden ser desacoplados. Debido a esto, lo que se hace es minimizar el ruido, para lo cual se separan los diferentes tipos de masas, de manera que se puedan desacoplar los planos de masas e impedir así que las masas más ruidosas se acoplen a otras menos ruidosas. Dicho desacoplo se realiza mediante el uso de ferritas o, en su defecto, resistencias de 0Ω (debido a su efecto inductivo a altas frecuencias) conectadas en serie entre los distintos planos de masa.

En la Figura 5.8 se muestra el esquema del desacoplo entre las masas realizado en el diseño del módulo desarrollado en este PFC.

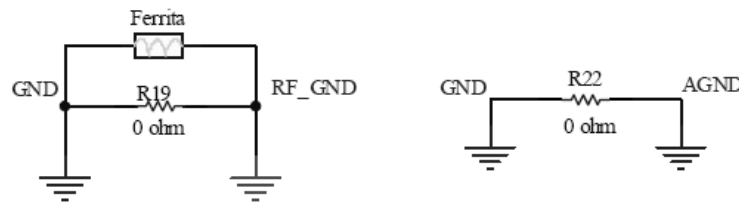


Figura 5.8. Desacoplo entre las distintas masas del módulo

Las nomenclaturas de las distintas masas definidas en el presente diseño son las siguientes:

- GND: masa digital del módulo, al que se conectan todas las tierras de origen digital, tanto del dispositivo EM260 como del ATmega168.
- RF_GND: masa de radiofrecuencia, a las que se conectan las masas provenientes de los componentes de radiofrecuencia, como son el balún, el filtro, el conector SWD, y las masas del dispositivo EM260 que corresponden a la parte de radiofrecuencia.
- AGND: masa analógica, donde se conecta la masa correspondiente al convertidor analógico/digital.

Como se puede observar, el desacoplo de la masa de radiofrecuencia permite la posibilidad de elegir entre desacoplar mediante resistencia de 0Ω , o bien mediante ferrita, debido a que la masa RF_GND es la más sensible al ruido.

5.2.3.3 Desacoplo de AVCC y VCC3V3

Además del desacoplo de masa y alimentación realizados, también se han desacoplado las tensiones del bloque convertidor analógico/digital del ATmega168 (AVCC) y la tensión de referencia (VCC3V3). El desacoplo se ha realizado, tanto entre los planos de alimentación (AVCC y VCC3V3), como en los pads, evitando el ruido que pueda producirse en el pad AVCC del ATmega168. El desacoplo entre los planos se realiza mediante una ferrita o, en su defecto, mediante una resistencia de 0Ω , y mientras que en el desacoplo del pad AVCC del ATmega168 se emplea un condensador de 100nF , que debe estar ubicado muy cerca del pad. En la Figura 5.9 se muestra el esquema de los diferentes desacoplos entre VCC3V3 y AVCC.

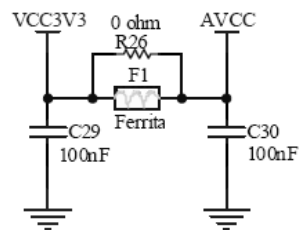


Figura 5.9. Desacoplo entre AVCC y VCC3V3

Como se muestra en la Figura 5.10, también se ha empleado un condensador de desacoplo en el pin 20 del dispositivo ATmega168. Esta señal es la encargada de establecer una tensión de referencia para el margen de conversión del convertidor analógico/digital, evitando así que se pueda acoplar ruido por el pad y de lugar a conversiones erróneas.

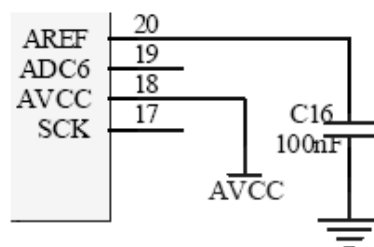


Figura 5.10. Desacoplo del pin AREF del ATmega168

5.3 Bloque de radiofrecuencia

Para realizar la comunicación inalámbrica, el módulo necesita transmitir y recibir información mediante un enlace de radiofrecuencia (RF) en la banda de 2.4 GHz. El uso del co-procesador EM260 facilita esta labor, ya que realiza la gestión del protocolo ZigBee recibiendo y enviando las tramas mediante el transceptor implementado. Aunque el chip posee gran parte del hardware de radiofrecuencia integrado, como amplificadores, filtros, modulador, etc., existe una parte de RF que no se encuentra contenida en el integrado y que necesita ser añadida mediante circuitería externa para conseguir que la transmisión y recepción de la información se lleve a cabo de la forma más eficiente posible. En este caso, la circuitería que hay que agregar consta de un circuito de balún, un filtro RF, una antena, y un conector de antena externa.

En la Figura 5.11 se puede observar el esquema del bloque de radiofrecuencia añadido al dispositivo EM260.

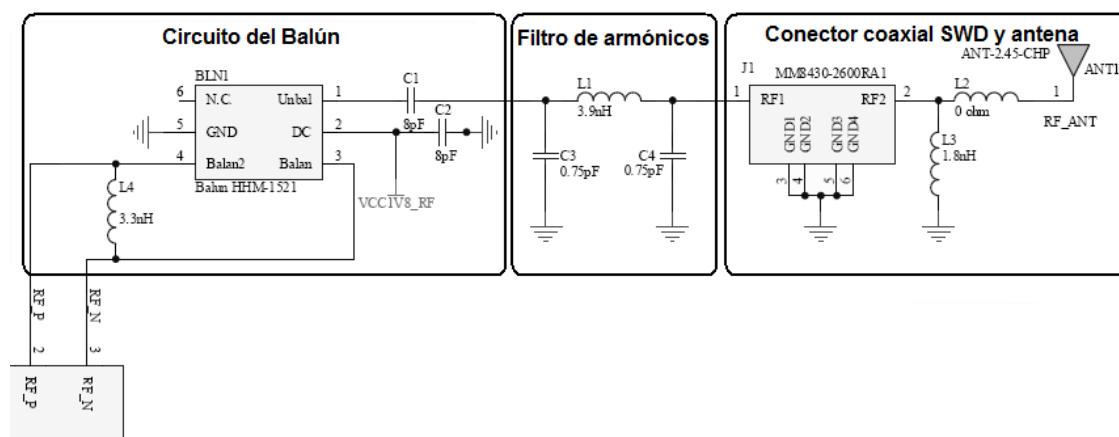


Figura 5.11. Esquema del bloque de radiofrecuencia

La conexión del bloque de radiofrecuencia se realiza mediante los pines RF_P y RF_N del dispositivo EM260, que como se vio en el capítulo 4 son la entrada/salida diferencial de la parte de radiofrecuencia del integrado. Como se muestra en el esquema de la Figura 5.11, los pines RF_P y RF_N van conectados directamente a la entrada/salida diferencial del circuito balún, el cual realiza una adaptación de impedancias de 200Ω a 50Ω y convierte la señal balanceada del dispositivo EM260 a no balanceada.

Ember, el fabricante del dispositivo EM260 establece dos condiciones básicas para la selección del balún a utilizar: primero debe permitir una alimentación de 1.8V DC para el amplificador que se encuentra dentro del dispositivo EM260, y segundo, debe realizar una conversión de impedancia de 50Ω a 200Ω con baja pérdida. Teniendo en cuenta estas condiciones, el balún empleado en el diseño es el HHM-1521 [47], cuyo uso está indicado para aplicaciones que utilicen la banda de 2.4GHz, como Bluetooth, IEEE 802.11b/g, etc. En la Tabla 5.3 se pueden observar sus principales características.

Características del balún HHM-1521
Impedancia del puerto balanceado: 200Ω
Impedancia del puerto no balanceado: 50Ω
Rango de frecuencias de funcionamiento: 2400 a 2500MHz
Pérdida de inserción máxima: 1 dB

Tabla 5.3. Características del balún HHM-1521

A continuación, en la Figura 5.12 se muestra el diagrama de pines de E/S del balún HHM-1521.

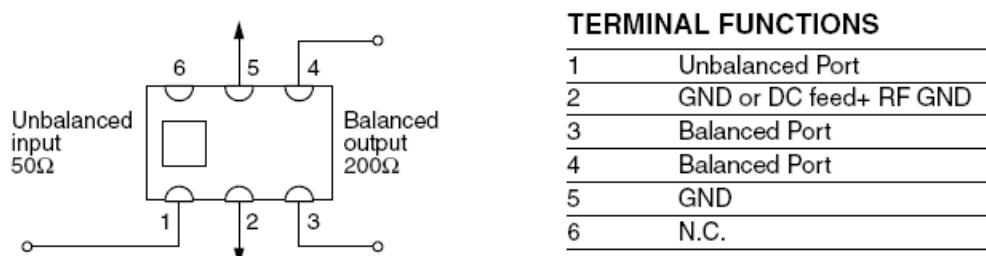


Figura 5.12. Asignación de pines del balún HHM-1521

En la Figura 5.13 se muestra el diagrama del conexionado del balún. Como antes se comentó, se puede observar que los pines RF_P (pin 2) y RF_N (pin 3) del EM260 están conectados a los terminales 3 y 4 del balún, además de que entre dichos pines se ha conectado una bobina de $3.3nH$, siguiendo la recomendación de diseño que indica el fabricante del dispositivo EM260. La bobina permite adaptar los dos pines de la señal diferencial de radiofrecuencia del EM260 con el balún, con el fin de maximizar la potencia de transmisión al puerto no balanceado del balún. Por otro lado, la masa del balún (GND), que se encuentra en el pin 5, se ha conectado a la masa de radiofrecuencia, denotada por RF_GND. En lo que se refiere al terminal *DC Feed* (pin número 2

del balún), se ha alimentado con la tensión de 1.8V para radiofrecuencia, denotada por VCC1V8_RF, al cual se le ha añadido un filtro RC adicional para mejorar la inmunidad al ruido, formado por un condensador de 8pF (C2) y una resistencia de 0Ω (R1). Por último, el puerto no balanceado del balún (pin 1) tiene conectado un condensador (C1) que evita el paso de cualquier corriente continua, conectándose seguidamente al filtro de armónicos.

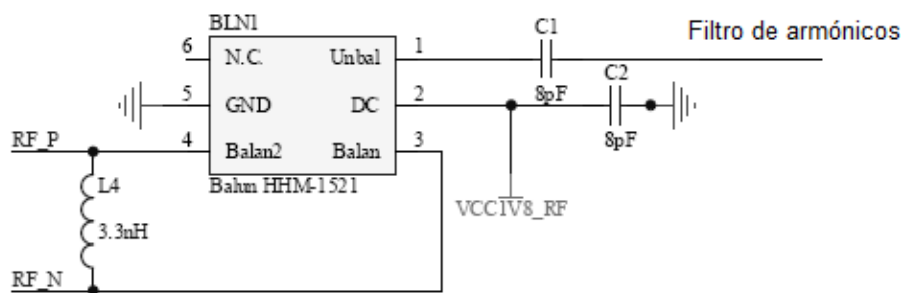


Figura 5.13. Esquema del circuito balún

El filtro de armónicos es otra parte del bloque de radiofrecuencia, cuyo objetivo principal de este filtro es atenuar lo máximo posible el segundo y tercer armónico, generado por el amplificador que lleva integrado el dispositivo EM260. De esta manera se cumplen las especificaciones impuestas por el FCC y el ETSI, que se encuentran descritas en el estándar IEEE 802.15.4, en cuanto al margen que debe existir entre la portadora y el 2º y 3º armónico del circuito transmisor. El filtro implementado se corresponde con un filtro Chebyshev paso banda, con un rizado máximo en la banda de paso de 0.1dB y como se puede observar en la Figura 5.14, está compuesto por dos condensadores de 0.75pF (C3 y C4) y una bobina de 3.9nH (L1). Los valores y el esquema de implementación se han obtenido de las recomendaciones indicadas en el diseño de referencia del dispositivo EM260.

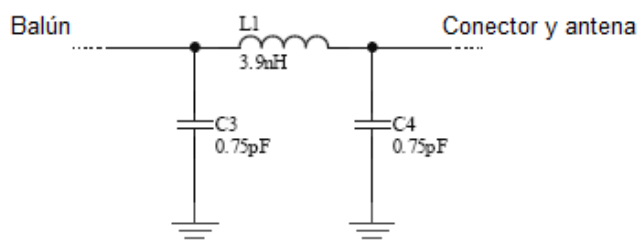


Figura 5.14. Esquema del filtro de armónicos

La última parte del bloque de radiofrecuencia está formada por un conector coaxial tipo SWD y una antena en chip con un pequeño circuito adaptador, como se muestra en la Figura 5.15. El conector coaxial tipo SWD empleado es el modelo MM8430-2600RA1 [48], fabricado por Murata, que está especialmente diseñado para aplicaciones de hasta 6GHz e impedancia características de 50Ω , entre ellas telefonía móvil, redes W-LAN y otras redes inalámbricas como ZigBee. Entre sus características se puede destacar el conmutador que lleva integrado en el enchufe, que ofrece la posibilidad de emplear una antena externa, la cual provoca una desconexión automática del resto del circuito que se encuentra a continuación del conector, o en el caso de no conectar ninguna antena, el circuito que se encuentra a continuación del conector funcionará normalmente, ya sea una antena u otro dispositivo. También es destacable su tamaño, con un perfil muy bajo, midiendo $3 \times 3 \times 1.75$ mm. (LxWxH). El conector consta de 6 pines: una entrada de radiofrecuencia, una salida de radiofrecuencia y 4 puntos a masa (que se conectan a la masa de radiofrecuencia, en este caso denotada por RF_GND).

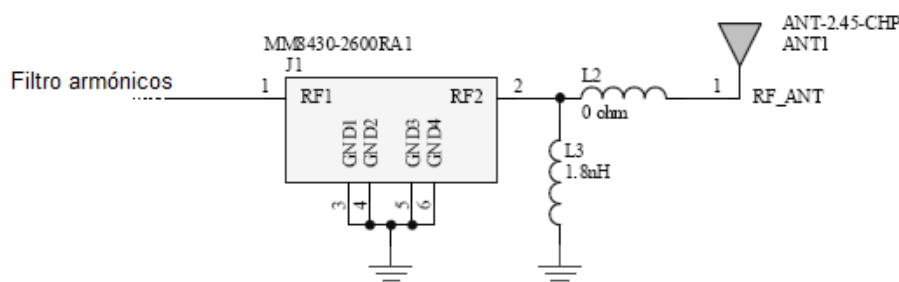


Figura 5.15. Esquema del circuito del conector externo y la antena

Como se observa en la Figura 5.15, entre el conector coaxial y la antena se encuentran L3 y L2. Estas dos bobinas son opcionales y se emplean únicamente si se decide hacer uso de la antena recomendada por el diseño de referencia del dispositivo EM260. La inclusión de esta red de adaptación es indicada por el fabricante de las antenas como medida para eliminar la parte reactiva de la impedancia y mejorar así la adaptación de la antena con los 50Ω . En el caso de este PFC no se requiere esta red de adaptación, ya que no se emplea la antena recomendada por el diseño de referencia del EM260. De esta manera, en el espacio de L3 no se incluirá ningún componente y en L2 se colocará una resistencia de 0Ω .

En la parte final del bloque de radiofrecuencia se encuentra la antena de montaje superficial, cuyo modelo es el ANT-2.45-CHP [49] fabricada por Linx Technologies. Se trata de una antena ultra

compacta de montaje superficial diseñada para operar en la banda de los 2.45GHz. En la Tabla 5.4 se muestran sus principales características.

Características de la antena ANT-2.45-CHP
Frecuencia central de trabajo: 2.45GHz
Ancho de banda de operación: 180MHz
Patrón de radiación: Omnidireccional
Ganancia máxima: +0.5dBi
Impedancia característica: 50Ω
Polarización: Lineal
VSWR ⁵ (máximo) < 2.0

Tabla 5.4. Características de la antena ANT-2.45-CHP empleada

El diagrama de pines de E/S de la antena de montaje superficial, y su descripción, se muestran en la Figura 5.16 y en la Tabla 5.5, respectivamente.



Figura 5.16. Asignación de pines de la antena integrada

Pin	Descripción
1	Entrada/Salida de la antena. Este pin debe ir conectado transmisor/receptor.
2	Terminación de la antena. Este pin se emplea únicamente como soporte físico de la antena, no debe ir conectado a ninguna señal eléctrica.

Tabla 5.5. Descripción de los pines de la antena integrada

⁵ VSWR (Voltage Standing Wave Ratio)

5.4 Bloque del dispositivo EM260

Además de proveer de la alimentación adecuada, el dispositivo EM260 necesita del acondicionamiento de determinadas señales para que todos los bloques que forman el dispositivo EM260 funcionen adecuadamente. En este apartado se describe el diseño de hardware necesario entorno al dispositivo EM260 para obtener la funcionalidad completa del bloque.

5.4.1 Regulación del dispositivo EM260

El diseño del módulo utiliza el co-procesador EM260 de Ember. Para su funcionamiento es necesario aplicarle una tensión de alimentación comprendida entre 2.1V y 3.6V (se aplica en los pines 13,19 y 24, denotados por VDD_PADS, como se describe en el capítulo anterior). Sin embargo, no todos los bloques funcionales del EM260 funcionan con esos valores de alimentación.

Existen diversos bloques funcionales en el integrado EM260 que necesitan alimentarse con 1.8V, como son: el bloque RF, el bloque del VCO, el bloque *prescaler*, el de la memoria flash, etc. Para todos estos bloques, el EM260 cuenta con un regulador integrado que convierte la tensión de alimentación a los 1.8V necesarios. La salida de la tensión regulada 1.8V se obtiene por el pin 12 del dispositivo EM260, denotada como VREG_OUT. Aunque la regulación se realiza internamente, el fabricante recomienda en su hoja de características que se empleen una serie de resistencias y condensadores en determinados terminales con el fin de proporcionar una tensión lo más estable posible, además de actuar como filtro entre los diferentes bloques funcionales. Por esa razón se emplean diferentes denominaciones a las distintas alimentaciones de los bloques funcionales, como son: VCC1V8_D para la alimentación digital del núcleo y la memoria flash, VCC1V8_RF para la alimentación de los filtros, mezcladores y el resto de partes de radiofrecuencia RF, 1.8V_VCO para la alimentación del VCO, y VCC1V8 para el resto de bloques funcionales del dispositivo EM260. En la Figura 5.17 se puede observar el esquemático del dispositivo EM260 con las distintas denominaciones de las alimentaciones existentes en su *pinout*.

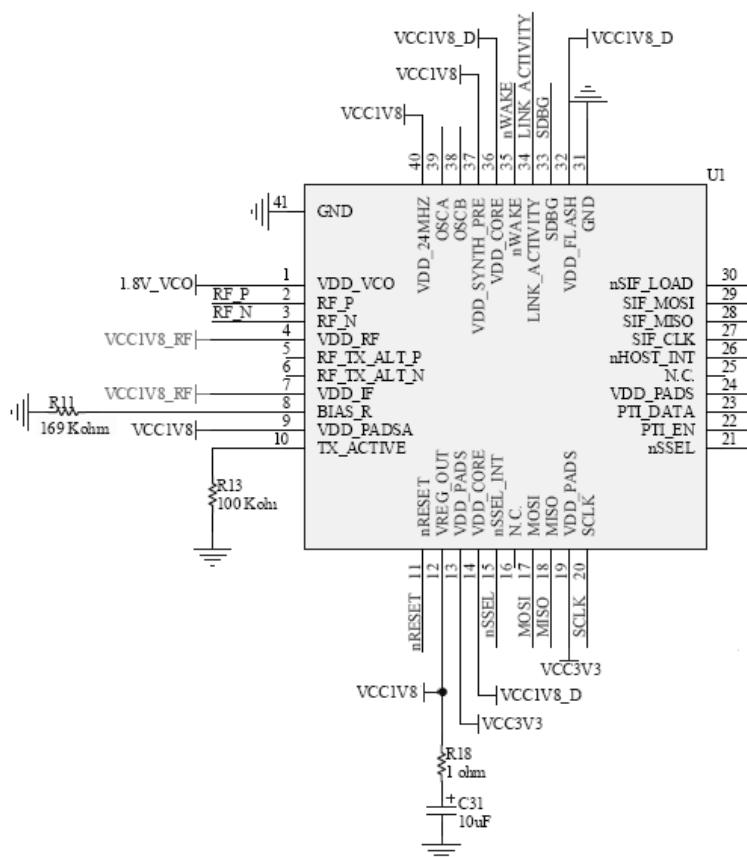


Figura 5.17. Esquema de las diferentes alimentaciones del EM260

En el esquema de la Figura 5.17 se puede observar que a la salida del regulador integrado (el pin 12) se ha empleado un condensador de tántalo de 10uF para mantener el regulador estabilizado, además de añadirse una resistencia de 1Ω en serie con el condensador, con la que se obtiene el mínimo ESR exigido, de manera que se cumplan las recomendaciones propuestas por el fabricante.

Las distintas tensiones de alimentación deben ser previamente filtradas, de manera que se minimice el ruido que puedan presentar los diferentes bloques alimentados. La tensión de alimentación digital, VCC1V8_D es filtrada mediante una resistencia de 2.7Ω, como se muestra en el esquema de la Figura 5.18. La resistencia, además de actuar como filtro, permite incrementar la sensibilidad en la Rx (Recepción), debido a la reducción de tensión que se logra.

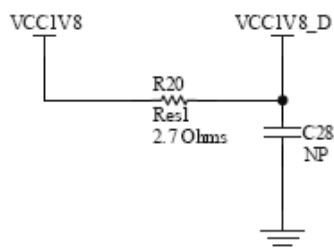


Figura 5.18. Esquema de filtrado entre VCC1V8_D y VCC1V8

El filtrado de las restantes alimentaciones, 1.8V_VCO y VCC1V8_RF, se realiza mediante una resistencia de 0Ω , como se muestra en el esquema de la Figura 5.19 y la Figura 5.20, respectivamente. En el caso de la alimentación VCC1V8_RF también se da la posibilidad de realizar el filtrado mediante ferrita, ya que presenta una mejor respuesta en alta frecuencia.

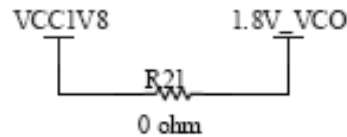


Figura 5.19. Esquema de filtrado entre VCC1V8 y 1.8V_VCO

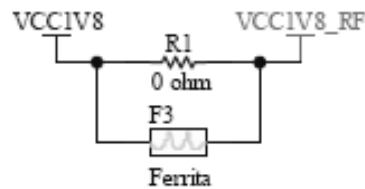


Figura 5.20. Esquema de filtrado entre VCC1V8 y VCC1V8_RF

Para concluir el apartado de regulación, hay que señalar que Ember, el fabricante del dispositivo EM260, indica la necesidad de emplear una resistencia de $169K\Omega$ entre el pin 8 (BIAS_R) del EM260 y GND, como se puede observar en la Figura 5.17. Esta resistencia sirve para establecer el punto de trabajo del regulador de tensión interno, que es el encargado de proporcionar la tensión de 1.8V a los distintos bloques internos del dispositivo. La resistencia debe ser precisa con una tolerancia del 1%.

5.4.2 Reloj del dispositivo EM260

En este apartado se aborda el acondicionamiento de las señales de reloj empleadas en cada dispositivo, tanto en el ATmega168, como en el dispositivo EM260.

Para que el dispositivo EM260 funcione de una manera correcta, es necesario disponer de una señal de reloj estable, utilizándose para ello el cristal cerámico de montaje superficial a 24MHz, cuyo modelo es el ABM3B-24.000MHZ [50] fabricado por Abracon Corporation. Este cristal es el recomendado por el fabricante del dispositivo EM260 en su *datasheet* y en los diseños de

referencia. El *datasheet* del EM260 exige ciertos parámetros, como su tolerancia a frecuencia, que está en torno entorno a los ± 40 ppm, y su máximo ESR, que no debe superar los 60Ω . El modelo seleccionado cumple con estos parámetros, ya que posee una tolerancia en frecuencia de ± 10 ppm a una temperatura de 25°C y su ESR no supera los 50Ω . En la Figura 5.21 se muestra el diagrama de pines de E/S del cristal ABM3B-24.000MHZ.

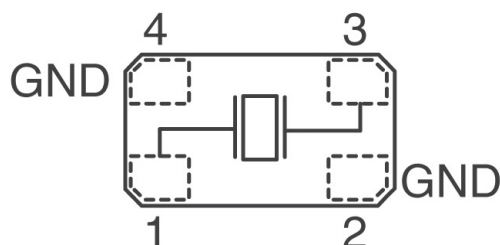


Figura 5.21. Diagrama de pines del ABM3B-24.000MHZ

Como se puede observar en el diagrama de pines, el cristal se encuentra conectado entre las patillas 1 y 3 del encapsulado, mientras que las patillas 2 y 4 deben conectarse a la masa del circuito. Siguiendo las recomendaciones del dispositivo EM260, es necesario conectar a masa mediante unos condensadores los pines a los que está conectado el cristal (patillas 1 y 3). Los condensadores han de ser de igual valor, siendo dicho valor de 27pF . También indica que es conveniente conectar una resistencia de $3.3\text{K}\Omega$ entre los nodos del cristal. En la Figura 5.22 se muestra el conexionado del circuito oscilador generador de pulsos de reloj del dispositivo EM260.

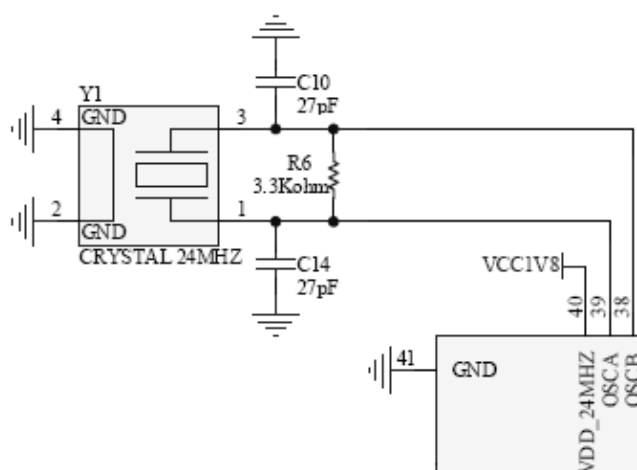


Figura 5.22. Esquema del circuito oscilador del dispositivo EM260

5.4.3 Conector de programación del EM260

El dispositivo EM260 debe ser programado con el *firmware* que proporciona Ember, el cual contiene implementada la pila ZigBee que necesita el módulo para poder desempeñar sus funciones, así como el protocolo EZSP (*EmberZNet Serial Protocol*) [41], necesario para enviar comandos y configurar el módulo mediante la comunicación con el ATmega168.

La inclusión del conector de programación externo como medio de programación de la memoria flash del EM260 ha sido obligatoria, ya que inicialmente la memoria flash no viene programada con el *firmware*. Además servirá para programar futuras actualizaciones del *firmware* que pueda desarrollar Ember y realizar la depuración de las aplicaciones del EM260.

En la Figura 5.23 se muestra el esquema del conexionado que presenta el conector de programación EM260.

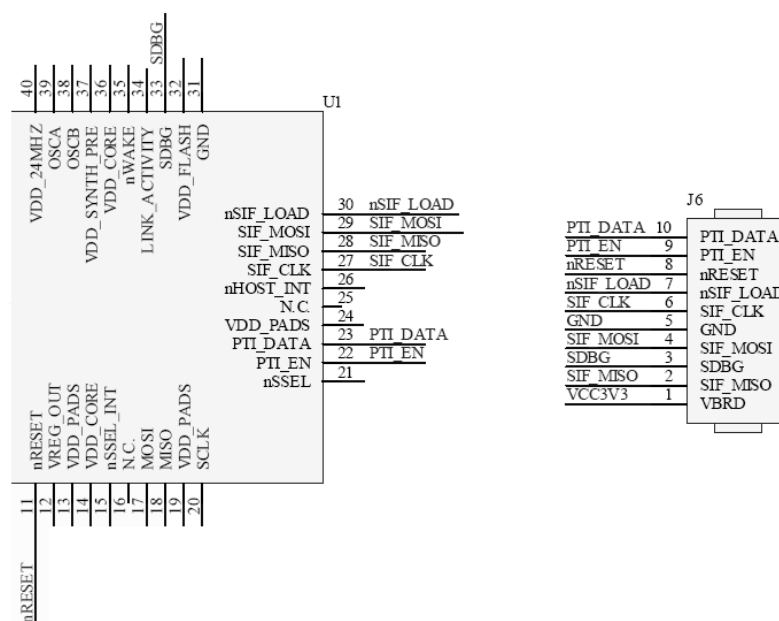


Figura 5.23. Conexionado del conector de programación del EM260

La programación del *firmware* se realiza mediante una interfaz serie síncrona, llamada SIF, cuyo funcionamiento se basa en comandos/respuestas, de forma similar a la interfaz JTAG. Es necesario un dispositivo de programación de memoria flash, denominado *InSight USB Link* [51], el cual permite programar el *firmware*, ya que realiza la conexión entre un PC (vía USB) y el conector de programación EM260. El dispositivo de programación *InSight USB Link* es el que se muestra en la Figura 5.24.



Figura 5.24. Dispositivo de programación InSight USB Link

En la Tabla 5.6 se indican las señales que forman parte de los 10 pines que posee el conector de programación del dispositivo EM260, además de realizar una breve descripción de cada señal.

CONECTOR DE PROGRAMACIÓN DEL EM260		
Pin	Señal	Descripción
1	VBRD	Tensión de alimentación (entre 2.1 y 3.6V).
2	SIF_MISO	Señal MISO (<i>Master Input Slave Output</i>) de la interfaz de programación SIF.
3	SDBG	Señal empleada en el modo depuración.
4	SIF_MOSI	Señal MOSI (<i>Master Output Slave Input</i>) de la interfaz de programación SIF.
5	GND	Masa.
6	SIF_CLK	Señal de reloj de la interfaz de programación SIF.
7	nSIF_LOAD	Señal de carga de datos de la interfaz de programación SIF.
8	nRESET	Reset del EM260, se encarga de inicializar el dispositivo.
9	PTI_EN	Señal de <i>frame</i> en la traza de paquetes.
10	PTI_DATA	Señal de datos en la traza de paquetes.

Tabla 5.6. Señales del conector de programación del EM260

Por otro lado, el fabricante del dispositivo EM260 recomienda que a la señal SIF_MOSI se le agregue una resistencia de *pull-down* con el fin de garantizar el nivel lógico adecuado cuando el EM260 se encuentre en modo *deep sleep*. A la señal nSIF_LOAD también se le debe añadir una resistencia, pero en este caso la configuración que requiere es de *pull-up*, con la que se consigue mejorar la inmunidad al ruido de la interfaz SIF.

Las resistencias de *pull-down* y *pull-up* añadidas al diseño se pueden observar en el esquema de la Figura 5.25.

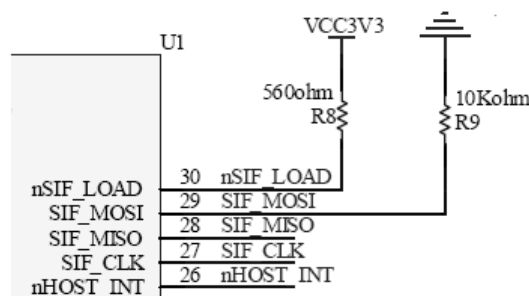


Figura 5.25. Esquema de pull-down y pull-up añadidos a la interfaz SIF

5.4.4 Indicador de actividad en TX/RX y señal TX_ACTIVE

El dispositivo EM260 ofrece la posibilidad de incluir un diodo LED como indicador de actividad de radiofrecuencia, controlado por la señal LINK_ACTIVITY del dispositivo EM260, y que se ilumina cuando se están realizando operaciones de transmisión o recepción de radiofrecuencia.

El circuito indicador de actividad que recomienda el fabricante del EM260 está formado por una rama alimentada con 3.3 V (denotada como VCC3V3), que se encuentra conectada al ánodo del diodo LED, y éste en serie con una resistencia de 510Ω, todo ello conectado a la señal de salida LINK_ACTIVITY (pin 34 del EM260). En la Figura 5.26 se puede observar el esquema del circuito indicador de actividad.

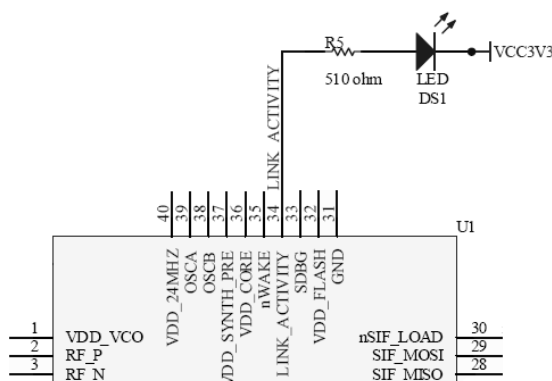


Figura 5.26. Indicador de actividad del circuito de radiofrecuencia

También se debe acondicionar la señal TX_ACTIVE del dispositivo EM260, que tiene como función realizar una conmutación para habilitar el uso de un amplificador de potencia externo en transmisión, y deshabilitarlo en recepción, en el caso de que exista. Como en el presente PFC no se ha empleado, el *datasheet* indica que en la patilla de la señal TX_ACTIVE (pin 10 del EM260) se debe incluir una resistencia de 100KΩ conectada a masa, como se muestra en la Figura 5.27.

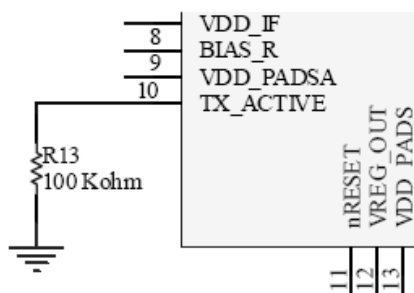


Figura 5.27. Esquema del acondicionamiento de TX_ACTIVE

5.4.5 Conexionado SPI

Como ya se ha comentado, la transmisión y recepción de mensajes entre el dispositivo EM260 y el ATmega168 se realiza mediante el protocolo EZSP, el cual emplea la interfaz SPI para la comunicación. En la Tabla 5.7 se muestran las señales que componen la interfaz SPI y la función que realiza cada una.

Señal	Descripción
MOSI	La señal MOSI (<i>Master Output Slave Input</i>) se encarga de transferir los datos del ATmega168 al EM260.
MISO	La señal MISO (<i>Master Input Slave Output</i>) se encarga de transferir los datos del EM260 al ATmega168.
SCLK	La señal SCLK (<i>Signal Clock</i>) se encarga de transmitir el reloj para la sincronización de la transferencia. Es generada por el ATmega168.
nSSEL	La señal nSSEL (<i>Slave Select</i>) se encarga de seleccionar al EM260 cuando se van a transmitir los datos. Es activa a nivel bajo.

Tabla 5.7. Descripción de señales de la interfaz SPI

El esquema de conexiones de la interfaz SPI entre el dispositivo EM260 y el ATmega168 se muestra en la Figura 5.28.

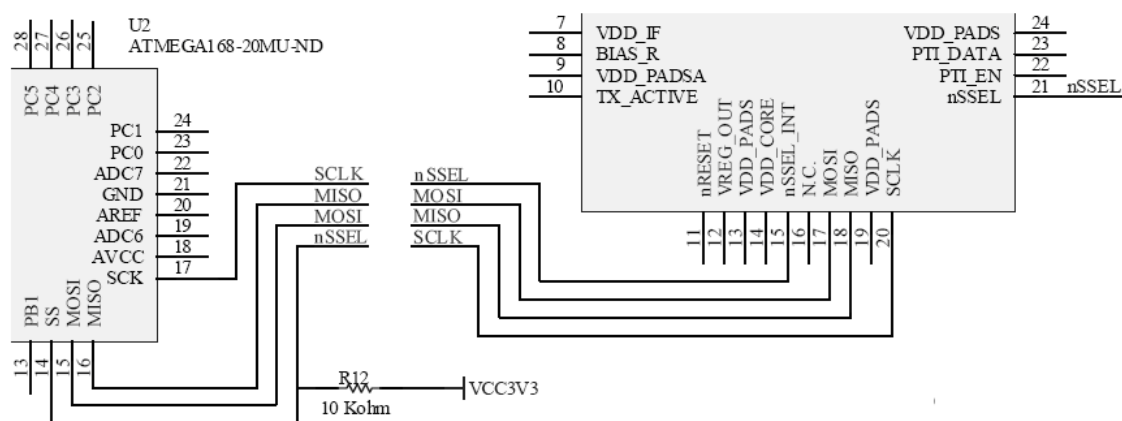


Figura 5.28. Conexión del EM260 y el ATmega168 mediante SPI

Como se puede observar en la Figura 5.28, a la señal nSSEL se le ha añadido una resistencia de *pull-up* que va a permitir tenerla desactivada durante la programación del ATmega168, ya que mientras se está programando, sus puertos se mantienen en alta impedancia. De esta forma, cuando se realiza la programación, se evitará que se puedan malinterpretar comandos por la interfaz SPI del EM260.

5.4.5.1 nHOST_INT y nWAKE

Además de las señales de la interfaz SPI, el protocolo EZSP emplea dos señales adicionales para completar su implementación. Estas señales, denominadas nHOST_INT y nWAKE, se utilizan como mecanismo para facilitar la sincronización entre el ATmega168 y el EM260. El conexionado de estas señales, entre el dispositivo EM260 y el ATmega168, se puede observar en la Figura 5.29.

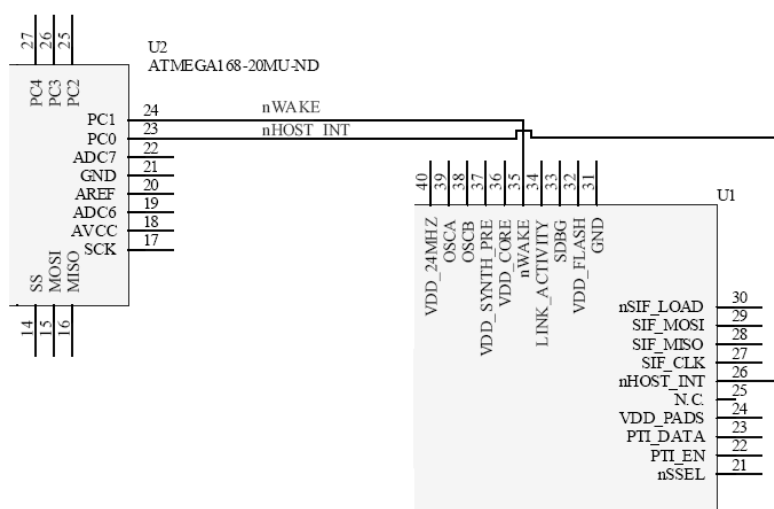


Figura 5.29. Conexión de las señales nWAKE y nHOST_INT

La señal nHOST_INT es generada por el EM260 y se emplea para provocar una interrupción en el dispositivo ATmega168 cuando el EM260 tiene un dato disponible listo ser enviado. Como se puede observar, la señal nHOST_INT del dispositivo EM260 (pin 26) está conectada al puerto de E/S PC0 del ATmega168 (pin 23), que puede ser configurado como interrupción.

La señal nWAKE es generada por el dispositivo ATmega168 y se encarga de despertar al EM260 en caso de que se encuentre en algún modo de bajo consumo. La señal nWAKE del EM260 (pin 35) está conectada con el puerto de E/S PC1 del ATmega168 (pin 24) como se muestra en la Figura 5.29.

5.5 Bloque del dispositivo ATmega168

Como se presento en el apartado correspondiente al bloque del dispositivo EM260, si se desea conseguir que el dispositivo ATmega168 funcione de una manera correcta, es necesario acondicionar ciertas señales, así como disponer de unas señales de reloj estables o un conector para poder realizar la programación del dispositivo. En este apartado se aborda el acondicionamiento de las señales del dispositivo ATmega168.

5.5.1 Reloj del ATmega168

En el caso del dispositivo ATmega168 se decidió emplear un cristal cerámico de montaje superficial de frecuencia 8MHz. La decisión de esta configuración se enmarca en un compromiso entre prestaciones y consumo de potencia, obteniendo buenas prestaciones a una frecuencia de 8MHz con un consumo mínimo. El dispositivo ATmega168 cuenta con un oscilador interno, pero ofrece una frecuencia y prestaciones limitadas, haciendo inviable su uso en aplicaciones que requieran errores de tiempo mínimos, algo no deseado si se intenta lograr que abarque el mayor tipo de aplicaciones posibles. Concretamente, el modelo de cristal seleccionado es el ABMM-8.000MHZ [52] fabricado por Abracon Corporation, el cual posee una tolerancia en frecuencia de ± 20 ppm, a una temperatura ambiente de 25°C. Además del estrecho margen de tolerancia que posee, tiene otras características, como su estabilidad o su tamaño, ideal para dispositivos de tamaño reducido por ser muy plano. En la Figura 5.30 se muestra el diagrama del pinout del cristal ABMM-8.000MHZ.

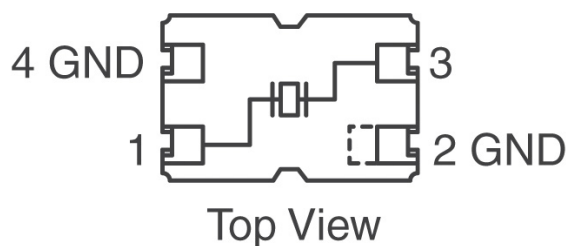


Figura 5.30. Diagrama de pines del ABMM-8.000MHZ

En este diagrama se puede observar el conexionado del cristal se realiza entre los pines 1 y 3 del encapsulado, mientras que los pines 2 y 4 no tienen internamente conexión y deben ser conectados a la masa del circuito. A los pines del cristal, 1 y 3, es necesario conectar un condensador a masa, teniendo en cuenta que los condensadores de cada pin deben ser de igual valor. Con respecto al valor de los condensadores, en el *datasheet* del microcontrolador se proponen unas directrices generales para su elección, pero para conseguir el valor óptimo se recomienda consultar la hoja de características del cristal que se va a emplear, ya que el valor depende de la capacidad de carga del cristal, de las capacidades parásitas, y del ruido electromagnético del medio. En el *datasheet* del cristal ABMM-8.000MHZ se indica que su capacidad de carga es de 18pF. Para realizar una aproximación al valor de los condensadores que se necesitan, llamados C19 y C22 en el esquema de la Figura 5.31, se ha usado la formula recomendada por el fabricante del cristal, que se muestra en la parte superior de la Figura 5.32. Si se simplifica está formula, obviando las capacidades parásitas, y teniendo en cuenta que C19 debe ser igual a C22, queda la formula inferior de la Figura 5.32, que da una valor de 36pF para C19 y C22.

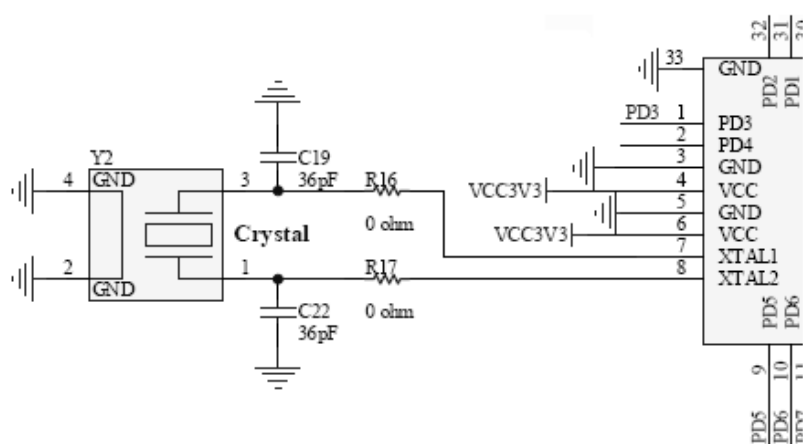


Figura 5.31. Esquema del circuito oscilador del ATmega168

$$C_L = \frac{C_{19} \cdot C_{22}}{C_{19} + C_{22}} + C_{stray} \quad Si \quad \begin{matrix} C_{19} = C_{22} \\ C_{stray} = 0 \end{matrix}$$

$$2C_L = C_{19} = C_{22} = 2 \cdot 18 pF = 36 pF$$

Figura 5.32. Capacidad de carga del circuito oscilador del ATmega168

5.5.2 Conector de programación del ATmega168

El dispositivo ATmega168 se programa a través del conector de programación ISP, el cual permite la programación de la memoria flash del ATmega168 con el *firmware* creado en el presente PFC, sin necesidad de desconectar el ATmega168 de la placa de circuito impreso, permitiendo con ello reprogramar el ATmega168 sin ninguna dificultad y pudiendo programar futuros *firmware* que se desarrollen con nuevas funciones, o simplemente para realizar pruebas en el módulo. Este conector está formado por 6 pines y realiza la programación serie mediante la interfaz SPI. El esquema del conector de programación del dispositivo ATmega168 se puede observar en la Figura 5.33.

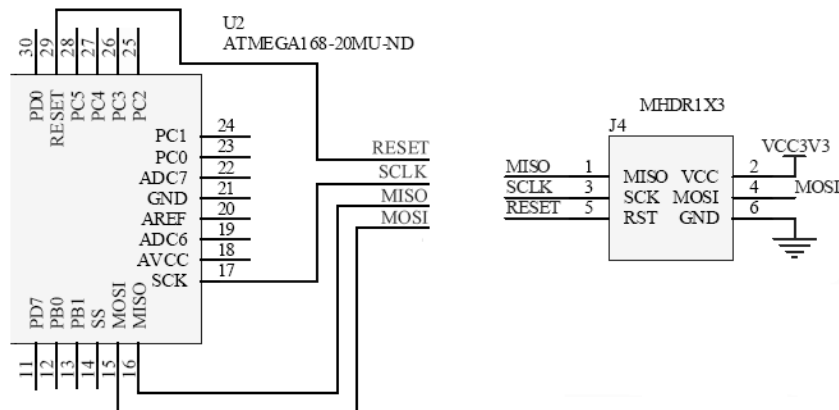


Figura 5.33. Conexión del conector de programación del ATmega168

Para la programación del microcontrolador se hace necesario el uso del programador desarrollado por Atmel, denominado *AVRISP mkII In-System Programmer* [53], que combinado con la plataforma software *AVR Studio*, permite realizar la programación ISP del ATmega168 mediante el puerto USB de un PC, conectado al conector de programación SPI implementado. En la Figura 5.34 se muestra el programador desarrollado por Atmel necesario en la programación del ATmega168, denominado *AVRISP mkII In-System Programmer*.



Figura 5.34. Dispositivo de programación AVRISP mkII In-system Programmer

Las señales implicadas en la programación del dispositivo ATmega168 se indican en la Tabla 5.8.

CONECTOR DE PROGRAMACIÓN DEL ATMEGA168		
Pin	Señal	Descripción
1	MISO	Señal MISO (<i>Master Input Slave Output</i>) de la interfaz SPI.
2	VCC3V3	Tensión de alimentación.
3	SCLK	Señal de reloj de la interfaz SPI.
4	MOSI	Señal MOSI (<i>Master Output Slave Input</i>) de la interfaz SPI.
5	RESET	Señal reset del ATmega168. Se encarga de inicializar el dispositivo.
6	GND	Masa.

Tabla 5.8. Señales del conector de programación del ATmega168

Por otra lado, aprovechando que el conector emplea las señales de la interfaz SPI, las cuales comparte con la comunicación que realiza el ATmega168 con el dispositivo EM260, se decidió ampliar el conector de programación del ATmega168 con la inclusión de 3 señales adicionales, de forma que las señales añadidas, más las señales de la interfaz SPI, forman las señales que emplea el protocolo EZSP. Así, conjuntado los dos conectores se consigue tener uno para el EZSP, lo que permite la posibilidad de realizar otras funciones, como puede ser la monitorización de los comandos y las respuestas del protocolo EZSP mediante un analizador lógico, o la posible depuración mediante el envío de comandos desde un dispositivo externo al EM260, sin necesidad de hacer uso del ATmega168. Las tres señales añadidas que forman el protocolo EZSP son las que se muestran en la Tabla 5.9, mientras que en la Figura 5.35 se puede observar cómo quedaría el esquema completo con los dos conectores.

CONECTOR ADICIONAL PARA EL PROTOCOLO EZSP		
Pin	Señal	Descripción
1	nSSEL	Señal de selección del ATmega168 al EM260.
2	nHOST_INT	Señal de interrupción del EM260 al ATmega168.
3	nWAKE	Señal de interrupción para despertar al EM260.

Tabla 5.9. Señales adicionales para el protocolo EZSP

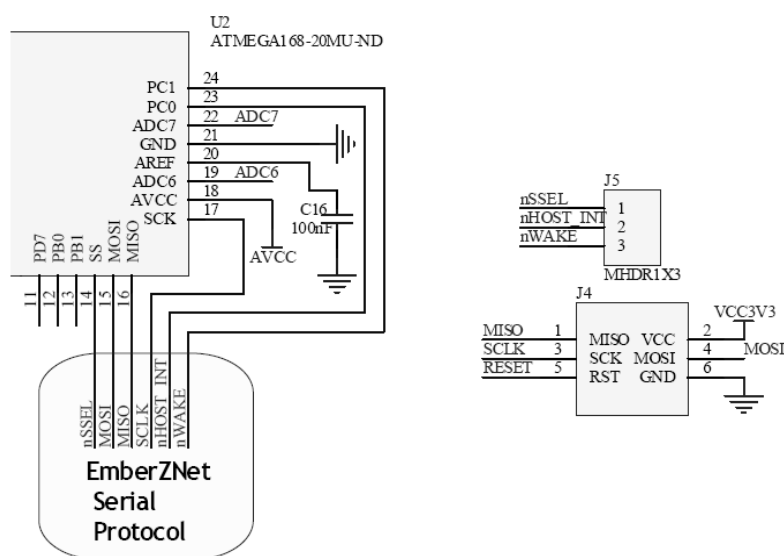


Figura 5.35. Esquema del conector para el protocolo EZSP

5.5.3 Circuito de reset

En todo sistema con microcontrolador se hace necesario disponer de un reset, ya que éste puede entrar en un estado desconocido y operar incorrectamente. La aplicación de un reset a la entrada del microcontrolador lleva a éste a un estado conocido y, por lo tanto, podrá operar según los algoritmos de programación. Para ello, el módulo cuenta con una circuitería adicional, que permite implementar un reset que inicializa externamente el ATmega168. Además, se ha añadido un reset mediante software para el dispositivo EM260. Este reset se realiza a través de un puerto de salida del ATmega168. El diseño del reset está formado por una resistencia y un condensador conectados entre sí. En la Figura 5.36 se muestra el esquema del circuito reset del ATmega168 realizado en este PFC.

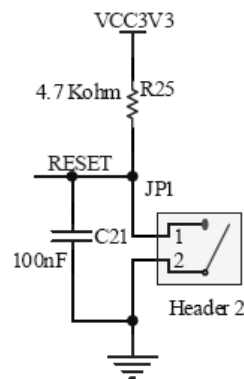


Figura 5.36. Esquema del circuito de reset del ATmega168

Como se puede observar en el esquema de Figura 5.36, el reset se activa a nivel bajo y la etiqueta RESET corresponde al pin 29 del ATmega168. La activación del reset se realiza mediante un *jumper*, que une los dos pin del Header 2.

Por otro lado, como se comentó anteriormente, el reset del dispositivo EM260 se realiza mediante software a través del puerto PC2 del ATmega168, que se encuentra configurado como salida y está conectado al reset del dispositivo EM260, denominado nRESET (pin 11). En la Figura 5.37 se muestra el esquema de conexión del reset del dispositivo EM260 con el ATmega168.

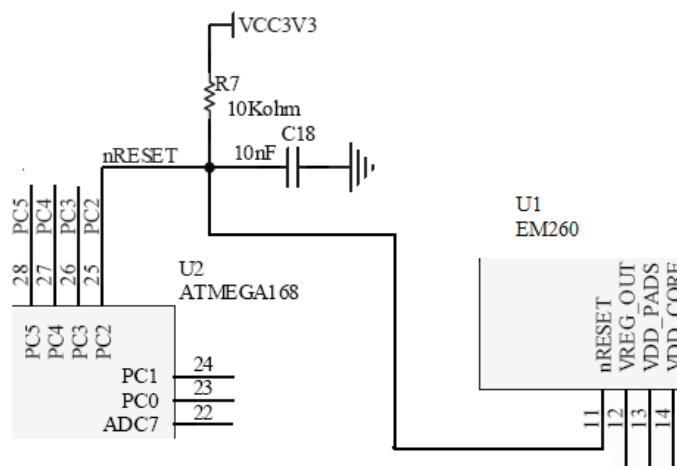


Figura 5.37. Esquema del circuito de reset del EM260

Si se observa el esquema de la Figura 5.37 se puede identificar el condensador de 10nF y la resistencia de 10KΩ que forman el circuito del reset. El condensador actúa como filtro preventivo de *glitches*, para los reset no deseados, y la función de la resistencia es la de *pull-up*, manteniendo el reset desactivado (se activa a nivel bajo) en todo momento hasta que el dispositivo ATmega168 desee resetearlo por software.

5.6 Conector de expansión

Mediante el conector de expansión se amplían las posibilidades de desarrollo de futuras aplicaciones, permitiendo la conexión con otros módulos que completen el diseño, así como llevar a cabo diferentes tareas e incorporar nuevas funciones. En concreto, en el presente PFC provee de las líneas necesarias para la conexión de los sensores empleados. Para ello las líneas empleadas en el conector ofrecen una gran variedad de funciones que facilitan la expansión, así como el empleo de diferentes tipos de sensores. Entre las funciones que pone a disposición el conector se encuentran, convertidores A/D, comparadores, interrupciones, puertos E/S, interfaz serie (tanto síncrona como asíncrona), etc.

El conector de expansión está formado por 16 pines, como se puede observar en el esquema de la Figura 5.38, en el que se indican además las señales seleccionadas que componen el conector.

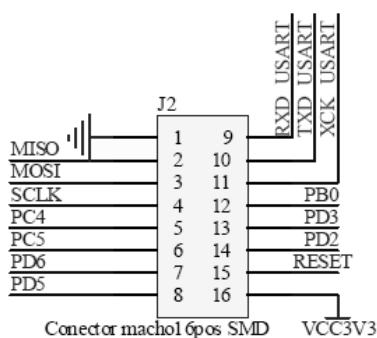


Figura 5.38. Esquema del conector de expansión

En la Tabla 5.10 se indica cada señal que forma el conector de expansión, así como una breve descripción de su funcionalidad.

CONECTOR DE EXPANSIÓN		
Pin	Señal	Descripción
1	GND	Masa.
2	MISO	Señal MISO (<i>Master Input Slave Output</i>) de la interfaz SPI. Puerto de E/S.
3	MOSI	Señal MOSI (<i>Master Output Slave Input</i>) de la interfaz SPI. <i>Output Compare</i> . Puerto de E/S.
4	SCLK	Señal de reloj de la interfaz SPI. Puerto de E/S digital.
5	PC4	Convertidor A/D. Puerto de E/S. Bus I ² C (SDA).
1	PC5	Convertidor A/D. Puerto de E/S. Bus I ² C (SCL).
2	PD6	Comparador analógico, <i>Output Compare</i> . Puerto de E/S.
8	PD5	Timer/contador. <i>Output Compare</i> . Puerto de E/S.
9	RXD_USART	Entrada de datos (RXD) de la USART. Puerto de E/S.
10	TXD_USART	Salida de datos (TXD) de la USART. Puerto de E/S.
11	XCK_USART	Reloj externo de la USART. Timer/contador. Puerto de E/S.
12	PB0	Salida de reloj. <i>Input Compare</i> . Puerto de E/S.
13	PD3	<i>Output Compare</i> . Interrupción externa. Puerto de E/S.
14	PD2	Interrupción externa. Puerto de E/S.
15	RESET	Reset del ATmega168. Puerto de E/S.
16	VCC3V3	Tensión de alimentación.

Tabla 5.10. Señales del conector de expansión

El conector de expansión incluye dos señales antes descritas, PD3 y PD2, a las cuales se han añadido unas resistencias de *pull-up* con el fin de facilitar el empleo como fuente de interrupción externa sin tener que añadir hardware adicional. Además en el caso de usar alguna de estas señales para activar el módulo cuando se encuentre en modo de bajo consumo, se hará activándolo a nivel bajo. En la Figura 5.39 se muestra el conexionado de las resistencias de *pull-up* en la señales de interrupción externa PD2 y PD3.

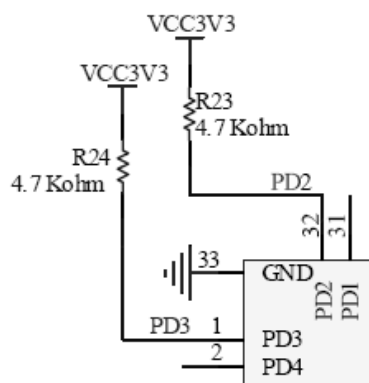


Figura 5.39. Esquema de los pull-up de PD2 y PD3

5.7 Bloque Sensor

En el bloque sensor se encuentran implementados los sensores empleados en el presente PFC. El diseño de este bloque se ha realizado de forma separada del módulo de comunicaciones inalámbricas y la comunicación se realiza mediante el conector de expansión, lo que permite mayor flexibilidad en el diseño del mismo, pudiendo agregar los sensores que se crean convenientes o la posibilidad de conectar otro bloque sensor. Además no sólo se podrá comprobar la funcionalidad del módulo de comunicaciones enviando los datos obtenidos mediante los sensores, sino también verificar las señales del conector de expansión. El bloque está compuesto por tres sensores diferentes; con la incorporación de varios sensores se pretende demostrar la capacidad del módulo para incorporar diferentes tipos de sensores con distintos interfaces sin dificultad.

Los tres sensores implementados en el bloque son: un sensor de temperatura, un sensor de presión y un sensor de movimiento PIR. Además de los sensores, el bloque se completa con un pulsador para resetear el módulo mediante la señal Reset del conector de expansión, dos diodos LEDs como indicadores, y un interruptor. A continuación se describen los diferentes sensores del presente PFC, así como la implantación de cada uno de ellos y el del hardware adicional añadido.

En la Figura 5.40 se muestra el esquemático completo del bloque sensor desarrollado en el presente PFC.

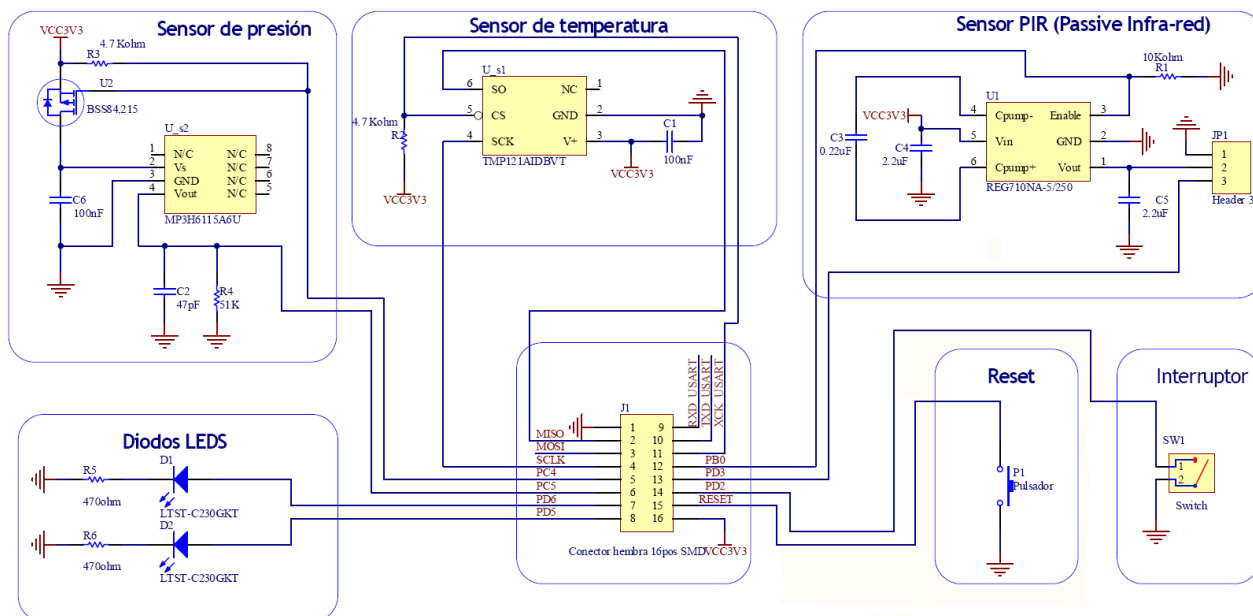


Figura 5.40. Esquema del bloque sensor

5.7.1 Sensor de temperatura

Para el bloque sensor se ha empleado un sensor de temperatura digital cuyo modelo es el TMP121, del fabricante Texas Instruments [54]. El sensor de temperatura TMP121 dispone de una salida digital para realizar la lectura de datos mediante interfaz SPI, ofrece una resolución de 0,0625 °C en los 12 bits, más 1 bit de signo que proporciona el registro de lectura del sensor. Presenta una precisión de ± 1.5 °C en el rango de temperatura de -25°C a $+85^{\circ}\text{C}$. Como se comentó en el bloque de alimentación, requiere una tensión de alimentación comprendido entre 2.7V y 5.5V. Además de estas características, y siguiendo los criterios de diseño comentados al inicio del capítulo, el sensor de temperatura presenta un bajo consumo de energía (50 μA max. en reposo) y un tamaño reducido (encapsulado TINY SOT23-6). Estas características hacen del dispositivo una elección idónea para el presente PFC. En la Tabla 5.11 se exponen las principales características del sensor de temperatura TMP121.

Características principales del TMP121	
- Rango de temperatura de operación	- 55 °C hasta 150 °C
- Precisión (error temperatura)	± 1.5 °C
- Resolución	0,0625 °C
- Rango de tensión de alimentación	2.7 V a 5.5 V
- Consumo activo	35 μA (50 μA max)
- Consumo inactivo	0.1 μA (1 μA max)
- Encapsulado	SOT23-6

Tabla 5.11. Principales características del sensor TMP121

En la Figura 5.41 se muestra el diagrama de pines del sensor de temperatura TMP121 y en la Tabla 5.12 están descritas cada una de las patillas del sensor.

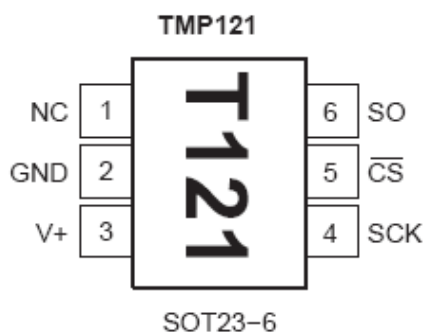


Figura 5.41. Pinout de sensor de temperatura TMP121

Pin	Señal	Descripción
1	NC	Pin sin función, debe ser conectado a masa o dejarse sin conectar.
2	GND	Pin de masa del dispositivo.
3	V+	Alimentación del dispositivo.
4	SCK	La señal SCK (<i>Signal Clock</i>) de la interfaz SPI, se encarga de transmitir el reloj para la sincronización de la transferencia, es generada por el maestro (el ATmega168).
5	CS	La señal <i>Chip Select</i> (CS) se encarga de activar al dispositivo esclavo, en este caso el TMP121. Se activa a nivel bajo.
6	SO	La señal SO funciona como la señal MISO de la interfaz SPI (<i>Master Input Slave Output</i>), se encarga de transferir los datos del TMP121 al ATmega168.

Tabla 5.12. Descripciones de los pines del TMP121

El sensor TMP121 convierte la temperatura en una palabra de 12 bits más el bit de signo con una resolución de 0,0625 °C. La lectura de la temperatura se almacena en un registro de 16 bits, donde los 13 primeros bits se utilizan para indicar la temperatura y los 3 bits restantes, D2, D1 y D0, no tienen ninguna función definida, encontrándose D1 y D0 en alta impedancia y D2=0. En la Tabla 5.13 se puede observar el formato de los 16 bits del registro de temperatura. Como se comentó, el sensor de temperatura TMP121 emplea el protocolo SPI para realizar la lectura de datos, en este caso del registro donde se almacena la temperatura. El dispositivo actúa como *Esclavo*, mientras que el ATmega168 es el *Maestro*, encargándose de generar la señal de reloj para recibir los datos del sensor.

D15	D14	D13	D12	D11	D10	D9	D8
T12	T11	T10	T9	T8	T7	T6	T5

D7	D6	D5	D4	D3	D2	D1	D0
T4	T3	T2	T1	T0	0	Z	Z

Tabla 5.13. Registro de temperatura del TMP121

Para obtener la conversión de la temperatura hay que esperar 0.25s desde que se activa la señal CS. Una vez terminada la conversión, si la señal CS se mantiene activa, entra en estado IDLE durante otros 0.25s, para luego repetir la conversión. El registro de temperatura almacena siempre la conversión más reciente, excepto en el encendido o en el reinicio del dispositivo, en los que el registro almacena 0 °C hasta que se finalice la primera conversión.

En la Figura 5.42 se muestra el esquema del sensor de temperatura TMP121 incorporado en el módulo sensor.

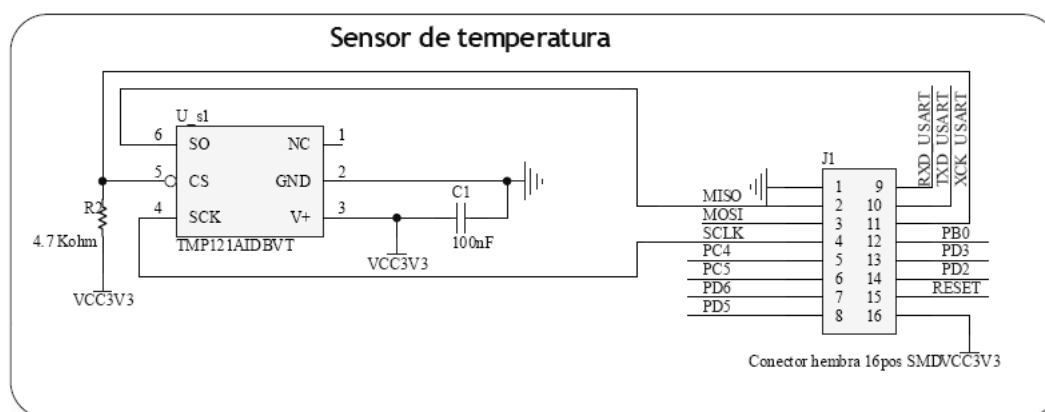


Figura 5.42. Esquema del sensor de temperatura TMP121 en el bloque sensor

Como se observa en la Figura 5.42 las señales SO y SCK del sensor TMP121 se han conectado a los pines 2 y 4 del conector de expansión, que corresponden a la señal MISO y SCLK, respectivamente, de la interfaz SPI del ATmega168. La señal CS del sensor se ha conectado a la señal XCK_USART del ATmega168 (pin 11 del conector de expansión), la cual se ha configurado como puerto de salida y se encarga de activar el sensor de temperatura. Además, a la señal CS se le ha añadido una resistencia de *pull-up* que va a permitir tenerla desactivada durante la programación del ATmega168, ya que mientras se está programando, sus puertos se mantienen en alta impedancia.

Por otro lado, la masa del sensor (GND), que se encuentra en el pin 2, se ha conectado a la masa del conector de expansión. Al pin 3 del sensor, que corresponde con la alimentación, denotada por VCC3V3, se le ha añadido un condensador de 100nF (C1) recomendado por la hoja de características del sensor que actuará como filtro del posible ruido del circuito.

5.7.2 Sensor de presión

El sensor de presión empleado en el presente PFC está basado en un transductor piezoresistivo cuyo modelo es el MP3H6115A6U [55], fabricado por Freescale Semiconductor. El sensor combina técnicas de micromáquinas avanzadas, metalización de películas delgadas y procesamiento bipolar a fin de proporcionar una salida analógica precisa (tensión), proporcional a la presión aplicada.

Entre las características del dispositivo MP3H6115A6U destacan su compensación de temperatura, alcanzando una alta precisión a altas temperaturas, con 1,5% de error máximo en el rango de temperatura comprendido entre 0°C y 85°C. También destaca su encapsulado de reducidas dimensiones y su revestimiento de termoplástico resistente, más concretamente de sulfuro de polifenileno (PPS, *PolyPhenylene sulfide*).

En la Tabla 5.14 se pueden observar las principales características del sensor de presión MP3H6115A6U.

Características principales del MP3H6115A6U	
- Rango de presión	De 15 kPa hasta 115 kPa
- Rango de tensión de alimentación	2.7 V a 3.3 V
- Consumo	4 mA (8 mA max)
- Rango de tensión de salida (V_{OUT})	0.12 V a 2.82 V
- Precisión	$\pm 1.5 \% V_{FSS}^6$
- Sensibilidad	27 mV/kPa
- Warm-Up Time ⁷	20 ms
- Rango de temperatura de operación	- 40 °C hasta 125 °C

Tabla 5.14. Principales características del sensor MP3H6115A6U

⁶ *Voltage Full Scale Output* (V_{FSO}) se define como la tensión máxima de salida o presión nominal total.

⁷ *Warm-up Time* se define como el tiempo requerido por el dispositivo para determinar el voltaje de salida después de que la presión se haya estabilizado.

En la Figura 5.43 se muestra el diagrama de bloques interno del sensor de presión y en la Tabla 5.15 la descripción de cada una de las patillas del sensor de presión MP3H6115A6U.

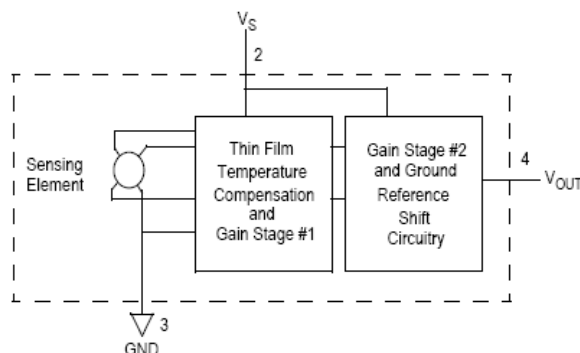


Figura 5.43. Diagrama de bloques del sensor de presión MP3H6115A6U

Pin	Señal	Descripción
1	NC	Pin sin función, debe dejarse sin conectar.
2	V_S	Alimentación del dispositivo.
3	GND	Pin de masa del dispositivo.
4	V_{OUT}	Salida analógica del sensor de presión, proporcional a la presión aplicada al sensor.
5	NC	Pin sin función, debe dejarse sin conectar.
6	NC	Pin sin función, debe dejarse sin conectar.
7	NC	Pin sin función, debe dejarse sin conectar.
8	NC	Pin sin función, debe dejarse sin conectar.

Tabla 5.15. Descripciones de los pines del MP3H6115A6U

La tensión de salida que proporciona el sensor de presión está comprendida entre 0.12 V y 2.82 V, como se indicó en la Tabla 5.14. La salida de tensión es proporcional a la tensión aplicada al dispositivo y está determinada por la función de transferencia de la Figura 5.44, incluida en la hoja de características del sensor de presión MP3H6115A6U.

$$V_{OUT} = V_S \cdot (0.009 \cdot P - 0.095) \pm (\text{Pressure Error} \cdot \text{Temp. Factor} \cdot 0.009 \cdot V_S)$$

$$V_S = 3.0 \pm 0.3V_{DC}$$

Figura 5.44. Función de transferencia del MP3H6115A

El parámetro P corresponde a los kilopascales aplicados al sensor, mientras que *Pressure Error* (error de presión) está determinado por un factor de error de ± 1.5 (KPa) y *Temp. Factor* depende de la temperatura y está caracterizado por la Figura 5.45, que determina su valor dependiendo de la banda de temperatura en la que esté operando el sensor de presión.

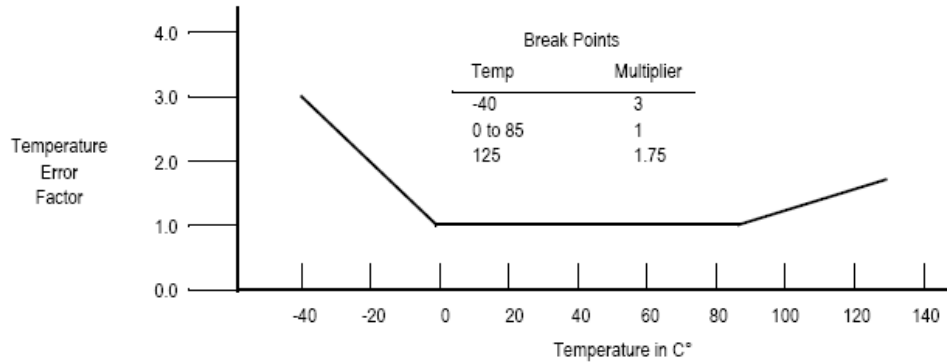


Figura 5.45. Banda de error de temperatura

En la Figura 5.46 se puede observar una gráfica que muestra la salida de tensión respecto a la presión, determinada por la función de transferencia de la Figura 5.44, antes comentada.

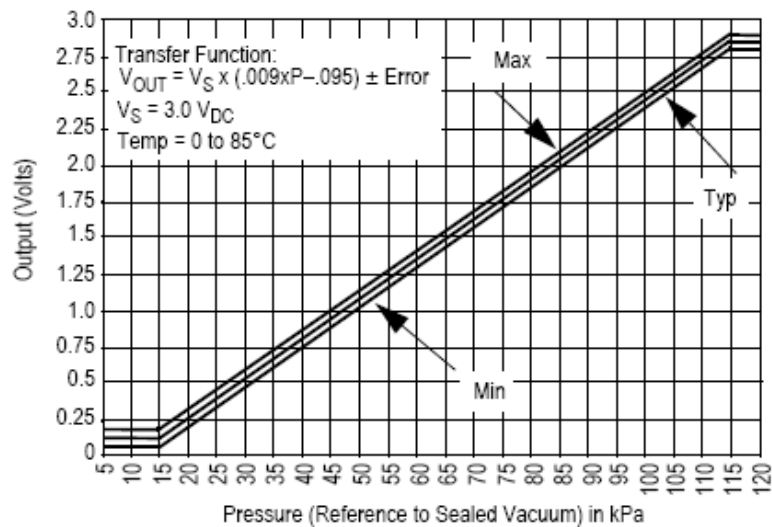


Figura 5.46. Relación entre la tensión de salida y presión en el MP3H6115A

En la Figura 5.47 se muestra el esquema del sensor de presión MP3H6115A6U incorporado y acondicionado en el módulo sensor desarrollado en este PFC.

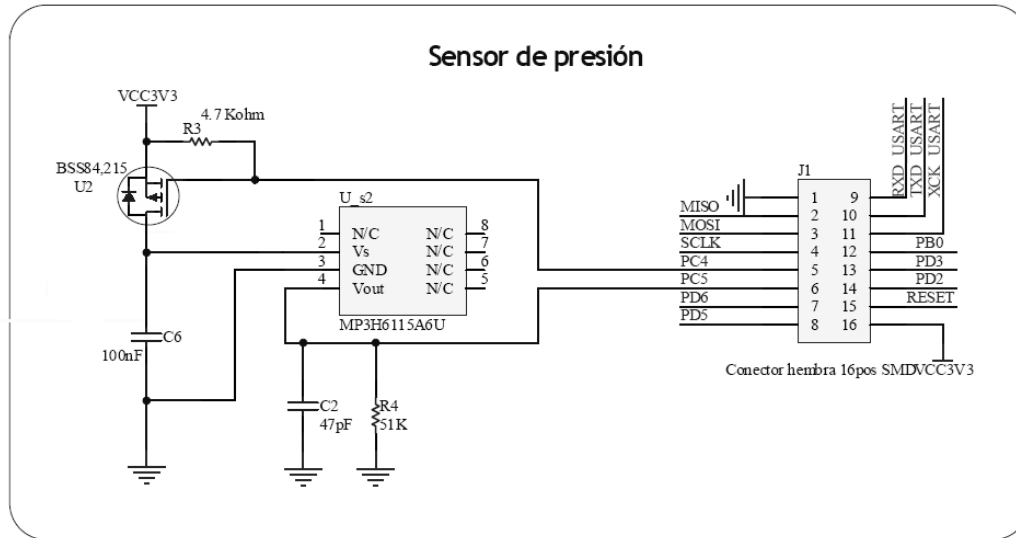


Figura 5.47. Esquema del sensor de presión MP3H6115A en el bloque sensor

Como se puede observar en la Figura 5.47, además del sensor de presión se ha dispuesto de un transistor MOSFET de canal P modelo BSS84.215, fabricado por NXP Semiconductors [56]. La función del transistor MOSFET es la de activar y desactivar el sensor de presión, actuando como un *switch*, evitando así el consumo de corriente innecesario del sensor de presión cuando no se esté usando. El terminal de la fuente del transistor MOSFET se ha conectado a la alimentación de 3.3V (denotada por VCC3V3) que proporciona el pin 16 del conector de expansión, mientras que el terminal de drenador del transistor MOSFET está conectado al pin 2 del sensor de presión, donde obtiene la tensión de alimentación necesaria para su funcionamiento. El terminal de puerta de transistor MOSFET está conectado al pin 5 del conector de expansión, el cual funciona como puerto de E/S del dispositivo ATmega168, y será el encargado de situar al transistor MOSFET en saturación o corte, con el fin de activar o desactivar, respectivamente, el sensor de presión. Además, ha sido necesario añadir una resistencia de *pull-up* en la señal PC4 que está conectada a la puerta del transistor MOSFET; de este modo, en caso de estar programándose el ATmega168 y estar sus puertos en alta impedancia, el transistor MOSFET estará en corte, manteniendo así desactivado el sensor de presión.

La señal de salida del sensor de presión V_{OUT} se ha conectado al pin 6 del conector de expansión, que se corresponde con el puerto PC5 del ATmega168, el cual se ha configurado como convertidor A/D, permitiendo obtener la tensión de la salida del sensor para posteriormente calcular la presión. Además, siguiendo las especificaciones de la hoja de características, en la señal de salida del sensor se ha añadido un condensador (C2) de 47pF y una resistencia (R4) de 51KΩ.

Para finalizar con el esquema del sensor de presión, mencionar que el pin 3 del sensor de presión, que corresponde a la masa del dispositivo, se ha conectado a la masa del bloque sensor, y, que en el pin de alimentación del sensor (pin 2) se ha añadido un condensador de 100nF a masa que actuará como filtro del posible ruido del circuito, tal como recomienda el fabricante.

5.7.3 Sensor de movimiento (PIR)

Al bloque sensor también se le ha añadido un sensor de movimiento de infrarrojos pasivo (PIR). El sensor PIR (*Passive Infra-Red*) es un dispositivo piroeléctrico que mide cambios en los niveles de radiación infrarroja emitida por los objetos que se encuentran a su alrededor. En el caso particular de este PFC se ha empleado el sensor de movimiento modelo *PIR Sensor 555-28027*, fabricado por Parallax [57]. Este sensor es capaz de reconocer el movimiento a una distancia máxima de 6 metros. Como respuesta al movimiento, el sensor produce un cambio en el nivel lógico en su salida digital (siendo la polaridad seleccionable mediante un *jumper*), lo que facilita su uso, ya que sólo necesita de una salida para indicar la detección de movimiento. Adicionalmente, es un sensor de bajo coste y tamaño reducido. El resto de características que ofrece el sensor empleado se muestran en la Tabla 5.16.

Características principales del Sensor PIR	
- Rango de medición	Hasta 6 m.
- Rango de tensión de alimentación	3.3 V a 5 V
- Consumo	≈ 3 mA
- Tamaño	32.2 x 24.3 x 25.4 mm.
- Tiempo de calibración	Entre 10 y 60 segundos
- Conexión mediante header SIP de 3 pin	
- Rango de temperatura de operación	0 °C hasta 50 °C

Tabla 5.16. Principales características del sensor de movimiento PIR

En la Figura 5.48 se muestra una representación de la vista inferior del sensor PIR y en la Tabla 5.17 se encuentran descritos cada uno de los pines que posee el conector del sensor.

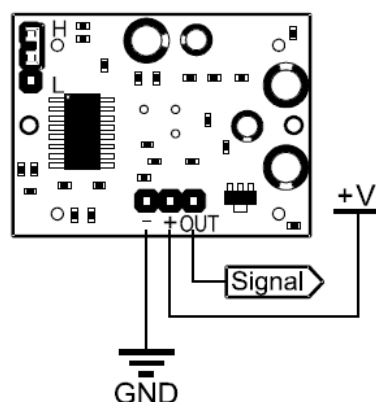


Figura 5.48. Representación del sensor de movimiento PIR (vista inferior)

Pin	Señal	Descripción
1	GND	Pin de masa del dispositivo.
2	+V	Alimentación del dispositivo.
3	OUT	Salida digital del sensor de movimiento PIR. Al reconocer un movimiento, el sensor cambia el nivel lógico de la señal.

Tabla 5.17. Descripciones de los pines del sensor de movimiento PIR

Para lograr un funcionamiento adecuado, el sensor de movimiento debe calibrarse. Para ello el fabricante recomienda que, una vez activada la alimentación del dispositivo, el sensor de movimiento permanezca en la zona donde va a operar manteniendo su rango de medición en ausencia de personas o de objetos en movimiento, entre 10 y 60 segundos. De esta manera, el sensor reconoce el estado de reposo o no movimiento del ambiente, adaptándose así a las condiciones propias de operación del ambiente donde está instalado.

Debido a que el rango de alimentación que requiere el sensor está comprendido entre 3.3V y 5V, como se muestra en la Tabla 5.16. Principales características del sensor de movimiento PIR, y que la tensión de 3V que proporciona la pila no se encuentra dentro de este rango, como se indica en el apartado 5.2.2 del PFC, ha sido necesario emplear un convertidor DC-DC que se encargará de suministrar la tensión adecuada.

El convertidor empleado es un convertidor *Buck/Boost Charge Pump* cuyo modelo es REG710NA-5/250, del fabricante Texas Instruments [58]. El convertidor REG710NA-5/250 proporciona una

salida regulada de 5V ante una entrada no regulada comprendida entre 2.7V y 5.5V. Entre sus características se puede destacar su reducido tamaño (encapsulado SOT23-6), su bajo rizado de salida (entorno a los 35mV_{pp}), el reducido número de componentes externos que requiere, o el modo *Shutdown* que proporciona.

En la Figura 5.49 se representa la distribución del pinout de convertidor REG710NA-5/250 y en la Tabla 5.18 se describe la función de cada pin del convertidor.

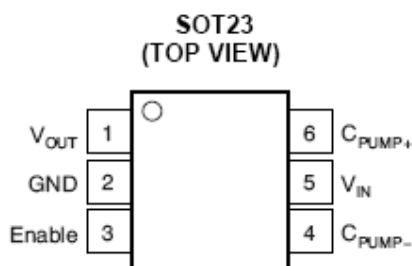


Figura 5.49. Asignación de los pines del convertidor REG710-5/250

Pin	Señal	Descripción
1	V _{OUT}	Tensión de salida regulada a 5V.
2	GND	Pin de masa del dispositivo.
3	Enable	Señal de habilitación (activa a nivel alto). Permite habilitar o deshabilitar el funcionamiento del convertidor. Cuando se deshabilita pasa a <i>modo shutdown</i> , ocasionando un bloqueo en la tensión de salida V _{OUT} , y reduciendo el consumo del dispositivo a 0.01μA.
4	C _{PUMP-}	Pin al cual se conecta el terminal negativo del <i>pump capacitor</i> .
5	V _{IN}	Tensión de entrada del convertidor. Debe estar comprendida entre 2.7V y 5.5V.
6	C _{PUMP+}	Pin al cual se conecta el terminal positivo del <i>pump capacitor</i> .

Tabla 5.18. Descripciones de los pines del convertidor REG710NA-5/250

En la Figura 5.50 se puede observar el esquema del conector (*header* de 3 pines) implementado en el módulo sensor para conectar el sensor de movimiento PIR, además del acondicionamiento de la tensión de alimentación del sensor mediante el convertidor REG710NA-5/250.

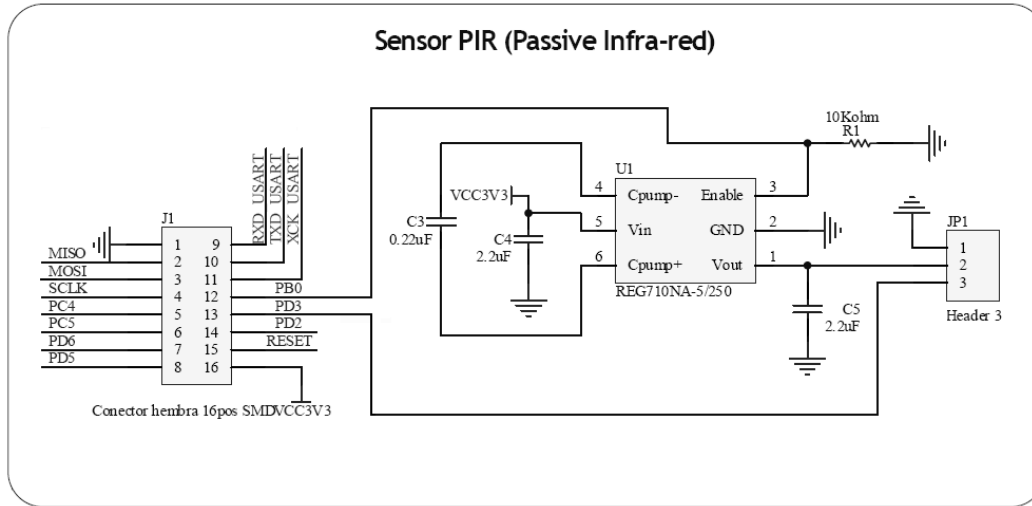


Figura 5.50. Esquema del circuito de acondicionamiento para el sensor de movimiento PIR

Siguiendo las indicaciones de la hoja de características del convertidor REG710NA-5/250, se han añadido tres condensadores; dos condensadores de $2.2\mu\text{F}$ (C4 y C5) conectados a la entrada y salida del convertidor (pin 5 y pin 1, respectivamente) y masa, y un tercer condensador (*pump capacitor*, C3) de $0.22\mu\text{F}$ conectado entre $C_{\text{PUMP-}}$ y $C_{\text{PUMP+}}$ (pin 4 y 6, respectivamente). La hoja de características también indica que los tres condensadores empleados deben ser cerámicos y de montaje superficial con el fin de eliminar respuestas transitorias, así como lograr unas altas corrientes de carga con tensiones de entrada bajas.

Se puede observar en la Figura 5.50 que el *header* de 3 pines, mediante el cual se conecta el sensor de movimiento PIR, presenta las siguientes conexiones. El pin 1 del *header* está conectado a la masa del bloque sensor, el pin 2 del *header* (+V) está conectado a la señal V_{OUT} de convertidor REG710NA-5/250 (pin 1), que se corresponde con la salida regulada de 5V que necesita el sensor para ser alimentado correctamente, y el pin 3 del *header* se ha conectado al puerto PD3 del ATmega168 (pin 13 del conector de expansión), encargado de reconocer el cambio de nivel de la señal de salida del sensor (OUT), cuando se produzca la activación del sensor de movimiento.

Además de lo mencionado, la entrada del convertidor REG710NA-5/250, V_{IN} (pin 5), está conectada al pin 16 del conector de expansión, el cual proporciona la alimentación al módulo sensor, mientras que el pin 2 del convertidor, correspondiente a su masa, va conectada a la masa del módulo sensor (pin 1).

Para finalizar con el acondicionamiento de las señales del sensor de movimiento, indicar que la señal de habilitación del convertidor está conectada con el puerto PB0 del ATmega168 (pin 12 del

conector). Este puerto será el encargado de activar y desactivar el convertidor REG710NA-5/250, y por tanto también el sensor de movimiento PIR.

En la Figura 5.51 se muestra una fotografía del sensor de movimiento PIR empleado en el presente PFC.

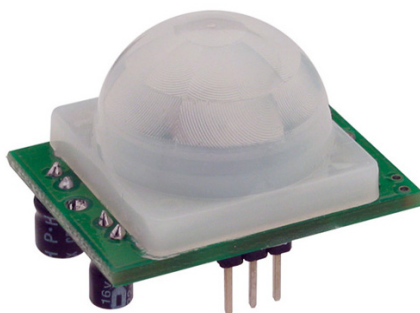


Figura 5.51. Fotografía del sensor de movimiento PIR

5.7.4 Diodos LEDs, interruptor y reset

El bloque sensor se ha completado con la incorporación de dos diodos LEDs, un interruptor y un pulsador de reset. La finalidad de los diodos LEDs es la de indicar estados que se produzcan durante el transcurso de la aplicación empleada en el módulo, o realizar indicaciones en pruebas o test a las que pueda ser sometido el módulo. El interruptor permite al módulo la selección de varias opciones en la aplicación desarrollada, pudiendo seleccionar la configuración más adecuada en cualquier momento. Además, se incorpora un pulsador que se encarga de resetear todo el módulo mediante la activación de la señal reset del ATmega168.

En la Figura 5.52 se muestra el esquema de los dos diodos LEDs incluidos en el modulo sensor, con sus respectivas resistencias de limitación de corriente (R5 y R6). El diodo D1 es de color verde y está conectado al puerto PD6 del conector de expansión, mientras que el diodo D2 es de color rojo y está conectado al puerto PD5 del mismo conector.

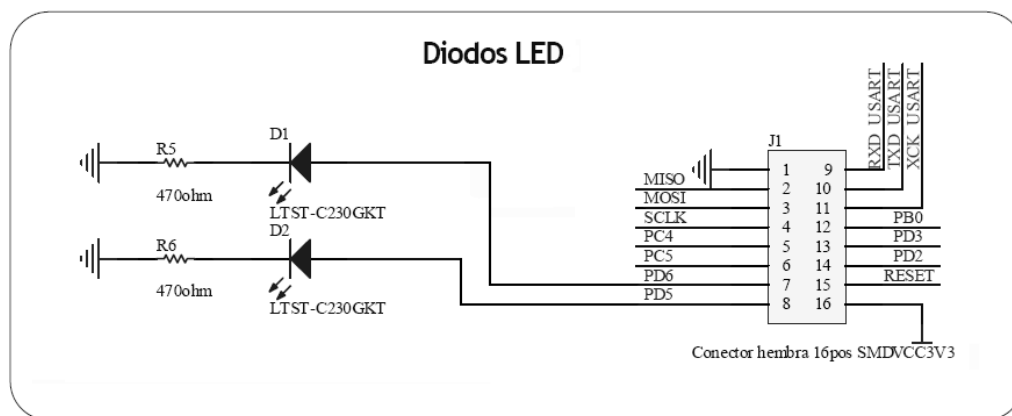


Figura 5.52. Esquema de los diodos LEDs del bloque sensor

En la Figura 5.53 se muestra el esquemático del interruptor del bloque sensor. En este caso se ha implementado mediante un *header* de 2 pines, que podrá activarse o desactivarse mediante un *jumper*. El interruptor está conectado al puerto PD2, el cual incluye una resistencia de *pull-up* en el módulo de de comunicaciones inalámbricas, con la idea de emplear el puerto como fuente interrupción, tal como se describió en el apartado 5.6.

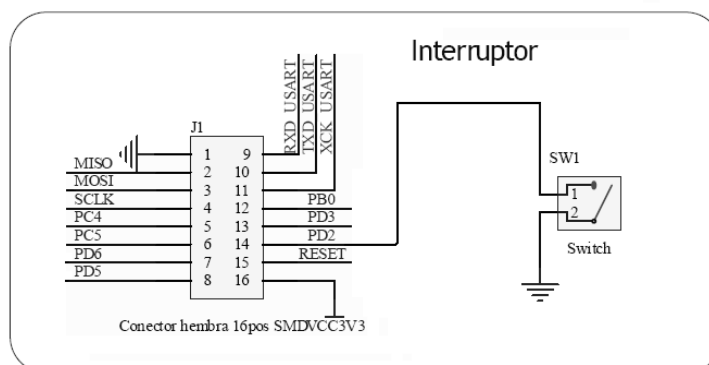


Figura 5.53. Esquema del interruptor del bloque sensor

Además de los diodos LEDs y el interruptor se ha añadido un pulsador mediante el cual se resetea el dispositivo ATmega168. Un terminal del pulsador se ha conectado al pin 15 del conector de expansión, que se corresponde con la señal de reset del ATmega168, y el otro terminal a la masa del módulo sensor, ya que el reset se activa a nivel bajo. En la Figura 5.54 se puede observar el esquema del pulsador implementado en el módulo sensor.

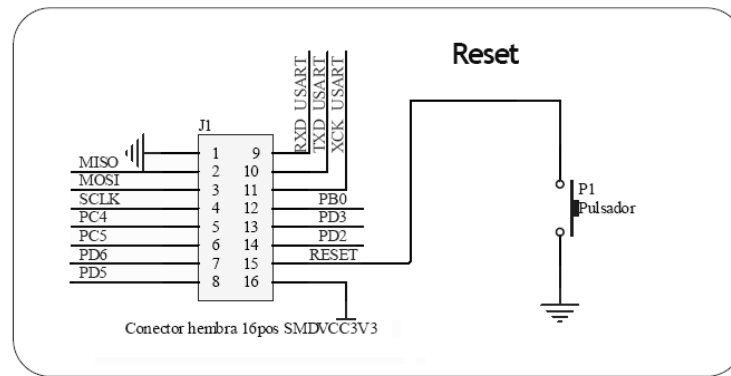


Figura 5.54. Esquema del pulsador de reset del bloque sensor

Capítulo 6

Diseño y montaje de las placas de circuito impreso

Una vez realizado el diseño hardware del módulo de comunicaciones inalámbricas y del módulo sensor, en este capítulo se procederá a detallar los pasos seguidos para la implementación de las placas de circuito impreso diseñadas, obteniendo finalmente los archivos necesarios para su posterior fabricación y montaje, finalizando así la realización física del módulo inalámbrico objetivo del presente PFC.

El diseño de PCB (*Printed Circuit Board*) o placa de circuito impreso, es un punto muy importante. Aunque en algunos casos, por su simplicidad o por el tipo de diseño, no adquiere esa importancia, cada vez son más los casos en los que el diseño de la PCB requiere mayores esfuerzos, debido a factores como las altas frecuencias que hoy día emplean los dispositivos, o al elevado número de interconexiones y componentes que se implementan en una placa de circuito impreso. Estos factores conllevan una serie de problemas en el diseño de PCB, los cuales no se pueden obviar, y deben tenerse en cuenta, ya que un buen diseño de PCB puede determinar que el resto del sistema satisfaga las especificaciones inicialmente establecidas.

6.1 Descripción general

Los diseños de las PCBs realizados en el desarrollo del presente PFC se han confeccionado utilizando el programa software de diseño electrónico DXP 2004 [59] de la compañía Altium. Esta herramienta de diseño cuenta, entre sus diversas funciones, con la de diseñar PCBs, además de permitir crear esquemáticos del diseño hardware realizado, pudiendo elegir los componentes que se van a utilizar en el dispositivo (tamaño, encapsulado, alimentación, tipo de conector, etc.), y muchas funciones adicionales que facilitan la labor de diseño de la PCB.

El primer paso a realizar, una vez ejecutado el software DXP 2004, es crear un nuevo proyecto. Este proyecto contendrá la ruta de todos los archivos necesarios para la creación final de las PCBs, entre los que se encuentran el esquemático, las librerías, tanto de símbolos como de *footprint*, los diseños PCBs, y los ficheros de salida, orientados a la fabricación final de las PCBs.

En la Figura 6.1 se muestra una captura de la ventana del programa DXP 2004, en la que se puede observar la estructura que presenta el proyecto de referencia de este PFC y los ficheros que lo forman, una vez creados.

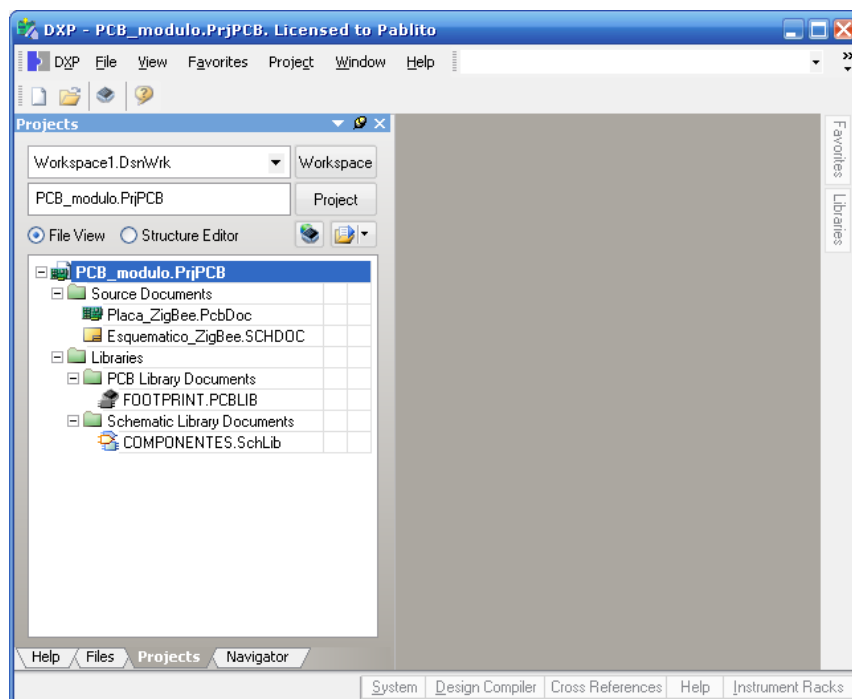


Figura 6.1. Proyecto de diseño de una PCB en el DXP 2004

En el lado izquierdo de la Figura 6.1, justamente debajo de la barra de herramientas del programa DXP, se encuentra la pestaña *Projects*, la cual está encabezada por el nombre del fichero del proyecto (PCB_modulo en este caso), con extensión .PrjPCB. Justo debajo se puede observar la ruta y estructura de los ficheros que componen el proyecto. Existen varias carpetas que ayudan a establecer las relaciones entre los ficheros y su organización. De esta manera se identificarán más fácilmente el tipo de ficheros y su contenido, reconociendo entre ellos los ficheros fuente, librerías de símbolos PCB, librerías de esquemáticos, ficheros de salida, etc.

Para aclarar y especificar mejor el proceso de diseño de la PCB del proyecto, en los siguientes apartados se explicarán de forma detallada, y por separado, cada uno de los ficheros creados.

6.2 Captura del esquemático

Después de crear el proyecto en el software de diseño DXP 2004, y antes de comenzar con la realización del esquemático, se deben crear las librerías necesarias para poder elaborarlo correctamente, las cuales se definen en los siguientes apartados.

6.2.1 Librería de símbolos

La primera librería que se ha creado en el nuevo proyecto de DXP 2004 es la librería de símbolos. Esta librería contendrá todos los símbolos correspondientes a los componentes que se van a emplear en el diseño de la PCB. Estos símbolos se añadirán al esquemático, representando los diferentes componentes que forman el circuito diseñado, como pueden ser resistencias, condensadores, bobinas, diodos, regulador de tensión, conectores, y demás componentes activos y pasivos que componen el módulo. Además, se han incluido en la librería de símbolos el co-procesador ZigBee EM260 y el microcontrolador ATmega168 de Atmel.

La librería debe contener los símbolos de todos los componentes, pero no para todos los componentes hay que crear un símbolo, ya que en muchos casos, éstos ya están creados en las librerías que incorpora por defecto la herramienta de diseño DXP 2004. Es muy común encontrar entre las librerías estándar del software los símbolos más usados, como pueden ser resistencias, condensadores, etc. Esto permite que una vez creada la librería se puedan copiar esos símbolos de las librerías estándar e importarlos a aquella que se creó. De esta manera, se pueden obtener los componentes más típicos de una forma rápida y eficaz, ahorrando mucho tiempo en el diseño de la PCB. Entre las librerías empleadas para obtener símbolos se encuentra la librería *miscellaneous*

devices, en la cual se pueden obtener los símbolos más comunes empleados en el diseño de un circuito impreso.

En la Figura 6.2 y en la Figura 6.3 se pueden observar varios símbolos de los componentes empleados en el diseño de entre los proporcionados en la librería *miscellaneous devices*, correspondientes al símbolo de la resistencia y del condensador, respectivamente.

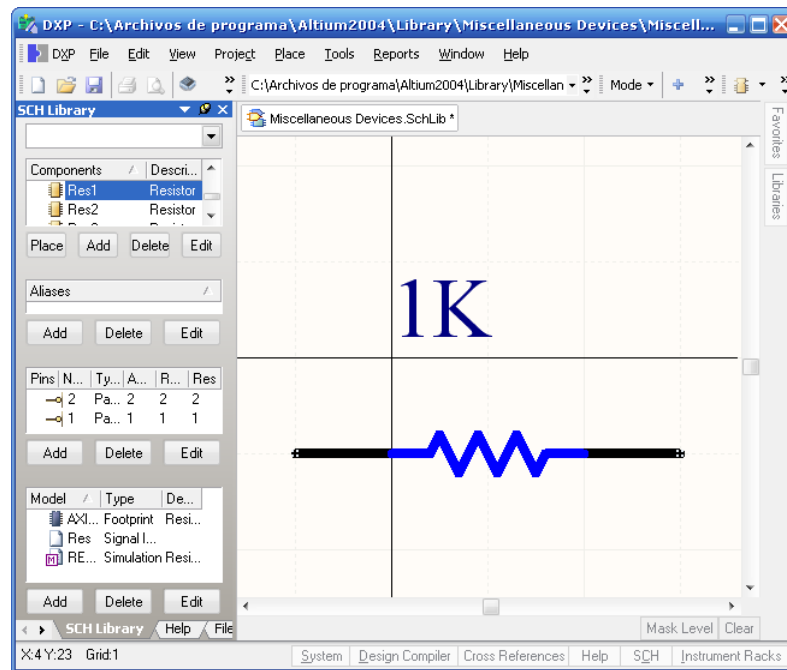


Figura 6.2. Resistencia de la librería estándar de la herramienta DXP 2004

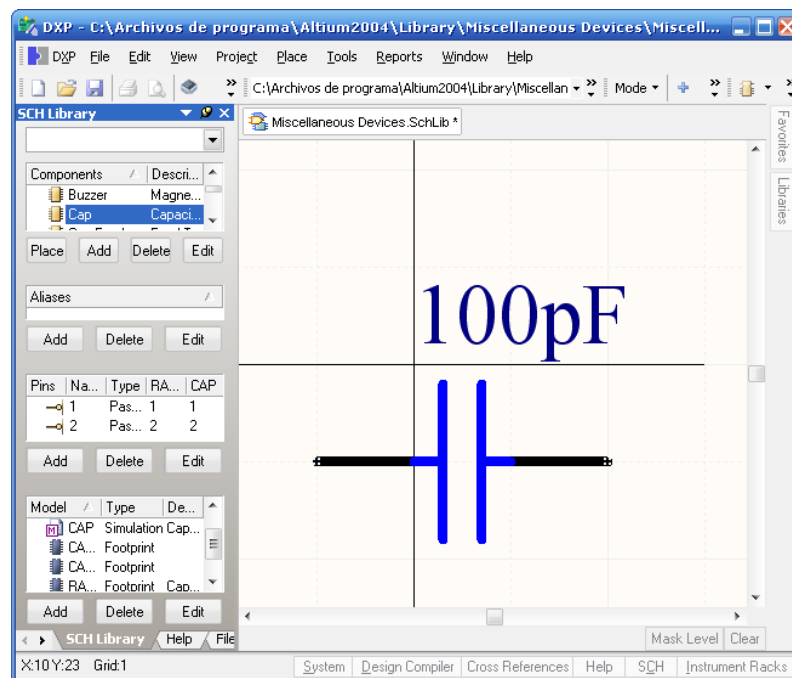


Figura 6.3. Condensador de la librería estándar de la herramienta DXP 2004

En el caso de no encontrarse el símbolo de algún componente empleado en el diseño en las librerías estándar, como puede ser el caso de los dispositivos EM260 o ATmega168, es necesario crearlos para que el diseño pueda completarse correctamente. Para ello, se ha de seleccionar en el menú de la librería de símbolos del proyecto la opción *New Component*. De esta forma se creará un nuevo símbolo, en el cual se deberán definir las líneas que conformarán el símbolo, para posteriormente ubicar cada uno de los *pads* de los que conste el componente correspondiente.

Idealmente, los símbolos de los componentes que se creen deben reproducir fielmente el componente físico que se desee representar, así como sus conexiones. El componente determinará el número de *pads* que se deben añadir al símbolo y su nombre, ya que representará los pines del componente, además de indicar las propiedades de cada uno, identificando si es de entrada, salida, bi-direccional, alimentación, etc.

Como se comentó anteriormente, para el diseño de la PCB se han tenido que crear algunos símbolos que no se encuentran en las librerías estándar proporcionadas por el programa DXP 2004. Entre ellos, el símbolo del dispositivo EM260 de Ember o del microcontrolador ATmega168 de Atmel, que se muestran en la Figura 6.4 y la Figura 6.5, respectivamente.

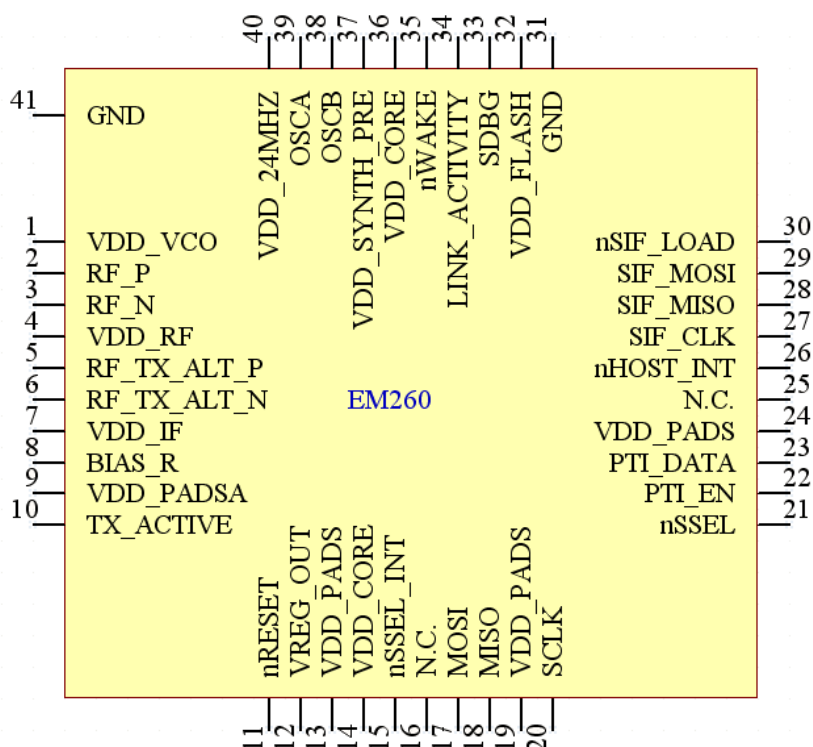


Figura 6.4. Símbolo del dispositivo EM260

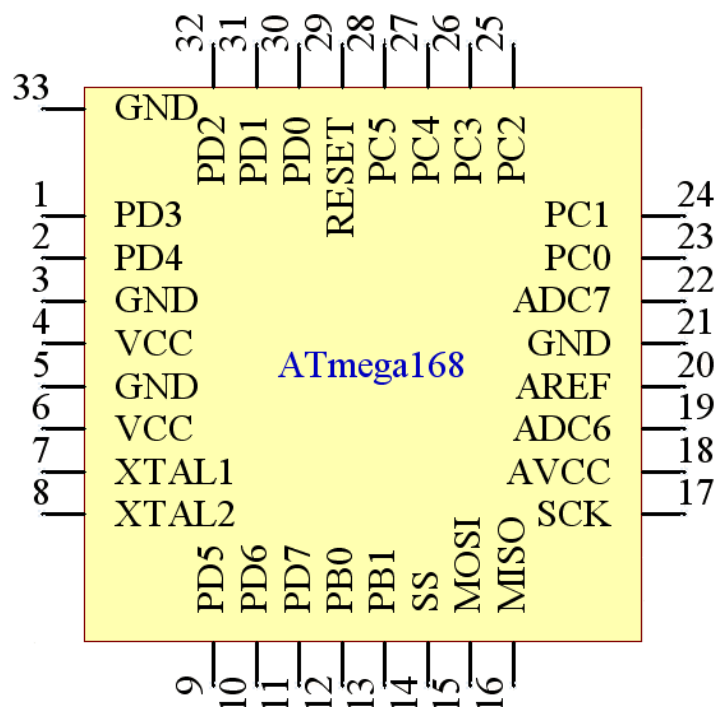


Figura 6.5. Símbolo del microcontrolador ATmega168

6.2.2 Librería de *footprints*

Una vez finalizada la librería de símbolos, incluyendo los símbolos de todos los componentes necesarios, se puede crear la siguiente librería, correspondiente a la librería de *footprints*. Esta librería también se crea dentro del proyecto activo en el software DXP 2004. El *footprint* es la huella o impresión que define el número de pines de un componente, así como su distribución y tamaño (grosor, distancia entre pines, etc.) para que luego se pueda crear la PCB y soldar los componentes correctamente. También se puede definir como eslabón entre el símbolo del componente en el esquemático, y el componente en la PCB. Como norma general, el *footprint* de los componentes viene incluido en su hoja de características, existiendo un apartado dedicado exclusivamente a ello, en el que se especifican las medidas del encapsulado y, en algunos casos, se proporciona el *footprint* recomendado con el fin de facilitar el diseño del mismo. En la Figura 6.6 se puede observar un ejemplo de las especificaciones del encapsulado del dispositivo EM260, cuyo encapsulado es de tipo QFN40 (*Quad Flat No leads* 40 pines), también denominado MLF40 (*MicroLeadFrame* 40 pines).

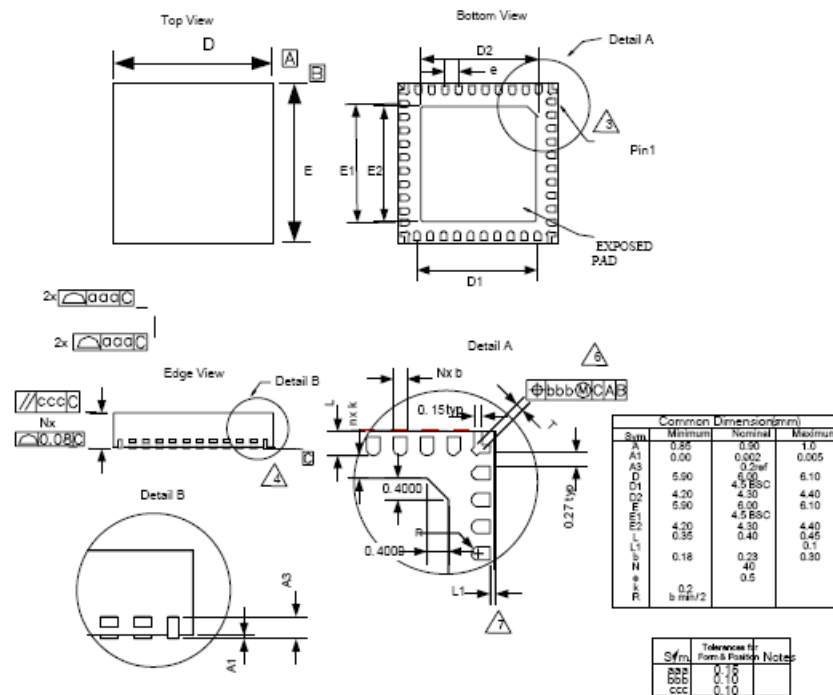


Figura 6.6. Descripción de medidas del encapsulado QFN40 (EM260)

Antes de comenzar con la elaboración de los *footprints* se debe tener en cuenta que muchos componentes pueden ser adquiridos con distintos encapsulados, presentando éstos diferentes tamaños, no compatibles entre ellos. Por esta razón, es muy importante conocer el encapsulado que se ha elegido con el fin de crear el *footprint* idóneo para cada encapsulado.

Con respecto al anterior párrafo y como referencia, en el diseño realizado en este PFC se ha empleado el microcontrolador ATmega168 con encapsulado MLF (*MicroLeadFrame*) de 32 pines, pero el fabricante también comercializa este mismo dispositivo con otros encapsulados. En este caso se puede adquirir además del empleado en este PFC, con encapsulado: TQFP (*Thin Quad Flat Pack*), PDIP (*Plastic dual in-line package*) o MLF pero con 28 pines. Todos difieren entre ellos en forma y tamaño, excepto el MLF de 32 pines y el MLF de 28 pines, que tienen la misma forma pero con diferente tamaño, presentando todos una incompatibilidad manifiesta.

En cuanto a componentes discretos, dado que uno de los criterios empleados en este Proyecto Fin de Carrera es lograr un tamaño reducido del dispositivo final, en el diseño realizado se ha decidido utilizar, siempre que sea posible, componentes de montaje superficial (SMD) 0402 (1mm x 0.5mm). Sin embargo, ha sido necesario incluir componentes de mayor tamaño debido a requerimientos en cuanto a su valor nominal, o de diseño, de manera que se ha hecho uso de

algunos componentes de montaje superficial 0603 (1.6mm x 0.8mm) e, incluso, algunos 0805 (2mm x 1.2mm) y 1206 (3.2mm x 1.6mm), en función del componente requerido en el diseño.

El proceso para la creación de la librería de *footprint* se realiza de la misma forma que en el caso de la librería de símbolos. De esta manera, los *footprints* más usuales se han obtenido de las librerías estándar que incluye la herramienta de diseño DXP 2004. Debido a lo crítico de este proceso, los *footprints* importados de las librerías estándares, como por ejemplo los obtenidos de la librería denominada *miscellaneous devices*, son revisados para comprobar que cuentan con las dimensiones correctas y que la distancia entre pines está en el orden adecuado del componente que va a usar ese *footprint*, ya que un error en éste, podría invalidar el circuito, obligando a repetir el diseño, fabricación y montaje de la placa de circuito impreso.

En la Figura 6.7 se muestra una captura de la pantalla del programa DXP 2004 en la que se puede observar un *footprint* de la librería estándar *miscellaneous devices*, donde se obtienen los *footprints* más comunes. En este caso en particular se muestra el *footprint* C1005-0402.

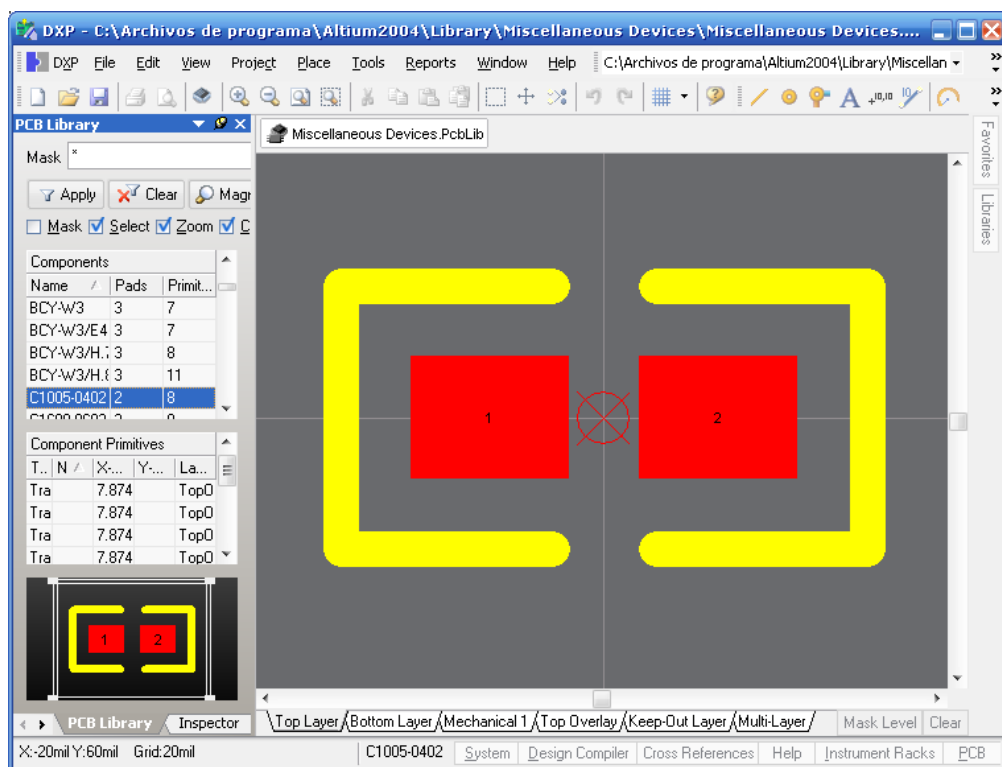


Figura 6.7. Footprint C1005-0402 de la librería *miscellaneous devices*

Sin embargo, como ocurría en las librerías de símbolos, los *footprints* menos usuales o con alguna particularidad se han tenido que elaborar a partir de las dimensiones del encapsulado especificadas en el *datasheet* del componente. Como ejemplo se pueden observar en la Figura 6.8 las

dimensiones del encapsulado que requiere el conector de expansión hembra 54722-0168-C del fabricante Molex, a partir del cual se realiza el *footprint* del componente.

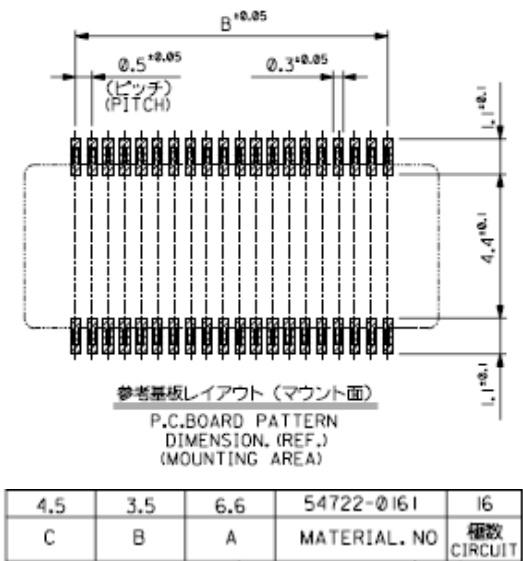


Figura 6.8. Dimensiones del encapsulado del conector Molex 54722-0168-C

Por otro lado, en la Figura 6.9 y la Figura 6.10 se muestran los *footprints* correspondientes a la antena ANT-2.45-CHP de Linx Technologies Inc. y al Micro-conector coaxial MM8430-2600 de Murata Electronics North America, respectivamente. Como otros componentes utilizados en este Proyecto Fin de Carrera, ha sido necesario crearlos a partir de las dimensiones del encapsulado que proporcionan los *datasheets* de los componentes, debido a que en las librerías estándar de la herramienta de diseño no se encontraban editados estos encapsulados.

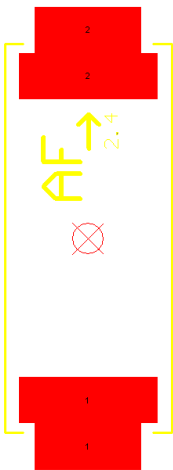


Figura 6.9. *Footprint* de la antena ANT-2.45-CHP de Linx Technologies Inc.

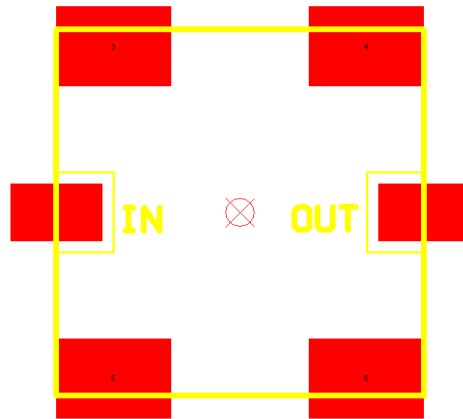
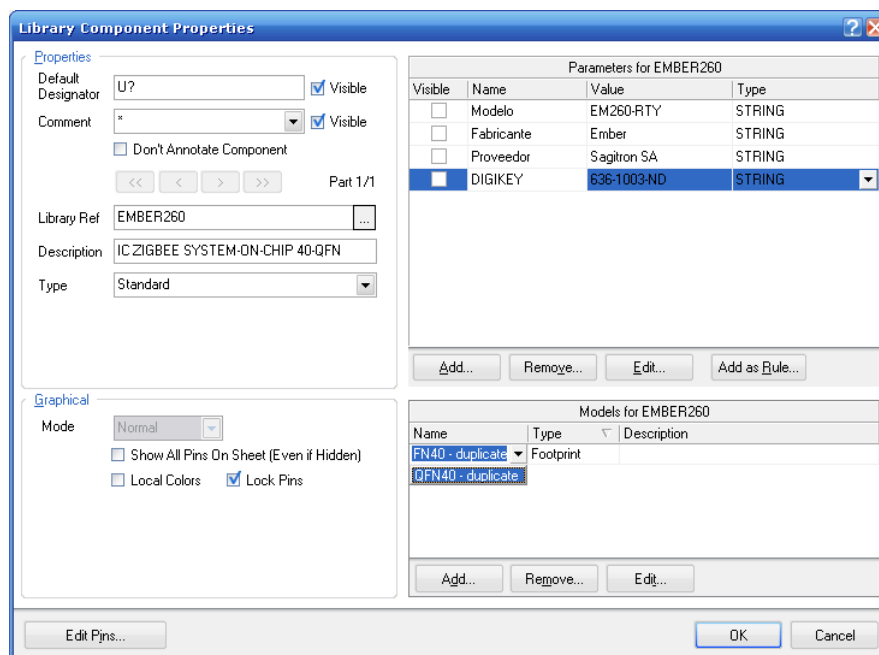


Figura 6.10. *Footprint* de micro-conector coaxial MM8430-2600 de Murata

6.2.3 Enlace entre librerías

Una vez completadas las librerías de símbolos y de *footprint* de todos los componentes necesarios para el diseño de la placa de circuito impreso, se deben enlazar ambas librerías entre sí, de manera que el símbolo de cada componente tenga asignado su correspondiente *footprint*. Para llevar a cabo esta asignación, en la librería de símbolos debe editarse cada símbolo individualmente, con el *footprint* correspondiente. De este modo, cuando se realice el esquemático del diseño y se vayan incluyendo los diferentes símbolos de los componentes, no será necesario especificar el *footprint* del componente desde la librería de símbolos creada. El programa DXP 2004 asignará por defecto el que se haya indicado anteriormente, evitando ir componente a componente y asignar el *footprint* correspondiente.

Para enlazar los símbolos de los componentes con sus correspondientes *footprints*, se ha editado cada uno de los símbolos en la librería de símbolos creada, añadiendo el atributo de *footprint* y seleccionando el encapsulado que le corresponde, ubicados en la librería de *footprint*. En la Figura 6.11 se muestra la ventana de edición de la herramienta DXP 2004 en la que se debe elegir el *footprint* adecuado para el encapsulado de cada símbolo.

Figura 6.11. Enlace entre un *footprint* y un símbolo

Tras la finalización de la creación y enlazado de librerías, se ha elaborado el esquemático completo del módulo ZigBee y del módulo sensor, necesario para realizar el diseño de las PCBs. En la Figura 6. y en la Figura 6.13 se pueden observar los esquemáticos finales del diseño del módulo de comunicaciones inalámbricas y del módulo sensor, respectivamente, donde figuran todos los componentes que integrarán las PCBs, representados por los símbolos de la librería creada, los cuales tienen asignados sus *footprints* correspondientes.

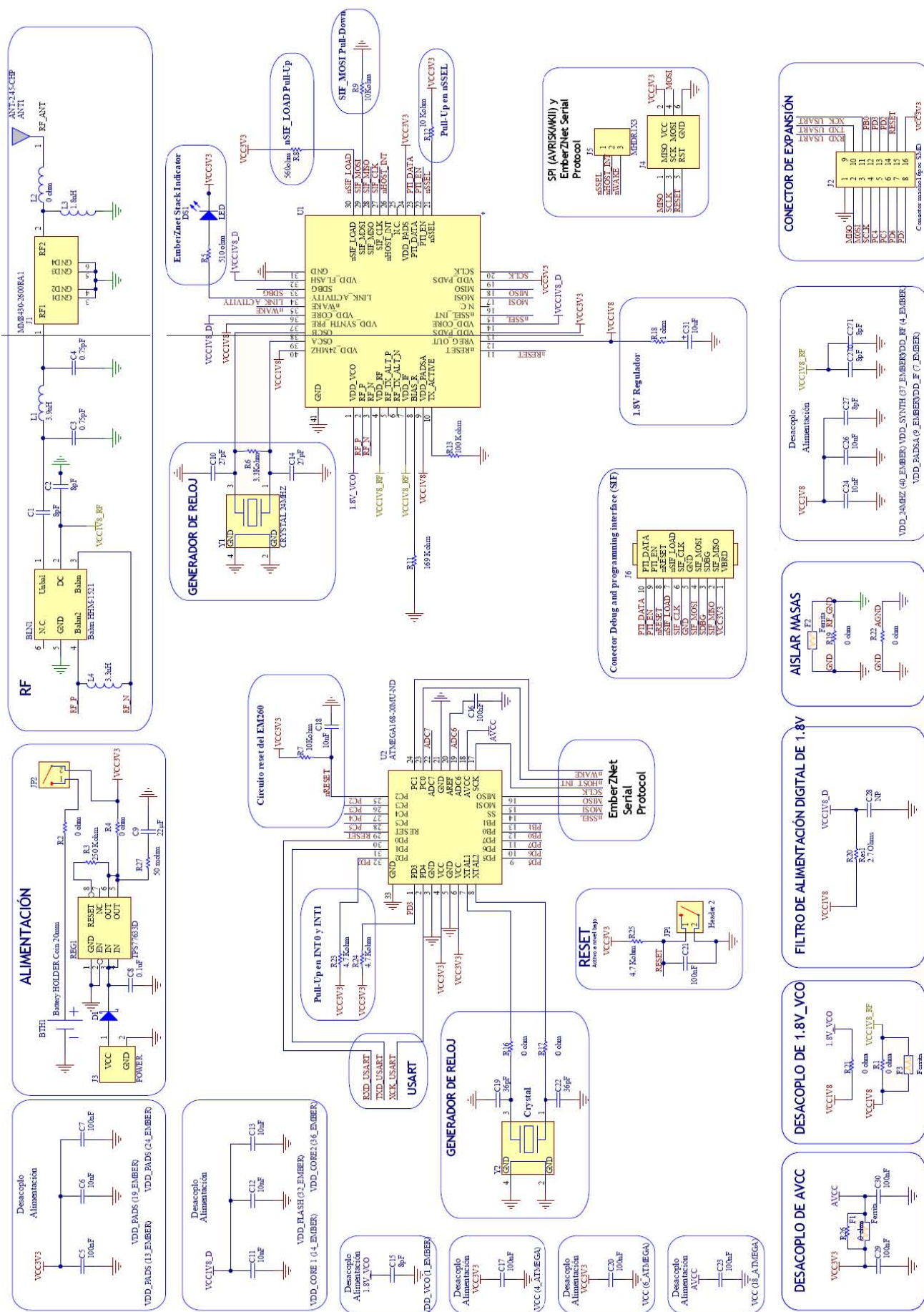


Figura 6.12. Esquemático completo del módulo de comunicaciones inalámbrico

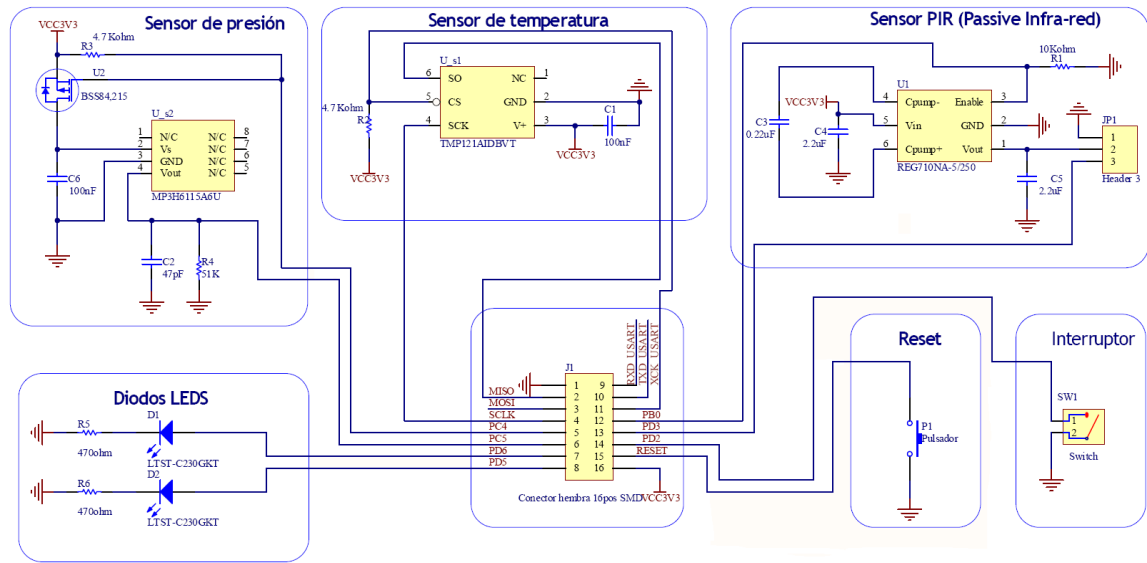


Figura 6.13. Esquemático completo del módulo sensor

Además, a cada componente del esquemático se le han agregado varios parámetros, entre ellos se encuentran los siguientes: modelo, fabricante, proveedor, código DIGIKEY etc. Mediante estos parámetros se intenta ayudar a la localización y consulta de los componentes utilizados en el diseño de la PCB.

En la Tabla 6.1 se muestro una lista parcial de los componentes empleados en el diseño y generada por el programa DXP 2004. En ella se pueden observar los distintos parámetros que se añadieron durante el diseño; de esta manera se facilita la realización de pedidos a los proveedores para adquirir los componentes necesarios, y permite llevar un control más exhaustivo. La lista completa de los componentes necesarios en el diseño final está disponible en el Anexo B de este documento.

Quantity	Designator	Description	Modelo	Fabricante	Footprint	Valor	DIGIKEY
1	ANT1	ANT 2.4GHZ 802.11 BLUETOOTH SMD	ANT-2.45-CHP	Linx Technologies Inc.	ANT-2.45-CHP		ANT-2.45-CHPCT-ND
1	BLN1	XFRMR BALUN RF 2.4GHZ SMD	HHM1521	TDK	BALUN_HHM1521		445-1229-1-ND
1	BTH1	HOLDER COIN CELL 2032 SMD	BK-883-TR	MPD (Memory Protection Devices)	HOLDER Coin 20mm		BK-883-CT-ND
1	C1	CAP CER 8PF 50V C0G 0402	GRM1555C1H8R0DZ01D	Murata Electronics North America	0402	8PF	490-3212-1-ND
1	C2	CAP CER 8PF 50V C0G 0402	GRM1555C1H8R0DZ01D	Murata Electronics North America	0402	8PF	490-3212-1-ND
1	R1	RES ZERO OHM 1/16W 5% 0402 SMD	ERJ-2GE0R00X	Panasonic - ECG	0402	0 OHM	P0.0JCT-ND
1	R3	RES 249K OHM 1/16W 1% 0402 SMD	ERJ-2RFK2493X	Panasonic - ECG	0402	249K OHM	P249KLCT-ND
1	REG1	IC 3.3V 500MA LDO REG 8-SOIC	TPS77633D	Texas Instruments	SOIC8		296-2782-5-ND
1	U1	IC ZIGBEE SYSTEM-ON-CHIP 40-QFN	EM260-RTY	Ember	QFN40 - duplicate		636-1003-ND
1	U2	IC AVR MCU 16K 20MHZ 32MLF	ATMEGA168-20MU-ND	Atmel	MLF32		ATMEGA168-20MU-ND
1	Y1	CRYSTAL 24.000MHZ SMD	ABM3B-24.000MHZ-D-R60-1-W-T	Abracon Corporation	CRYSTAL 24.000MHZ SMD		535-9700-1-ND

Tabla 6.1. Listado parcial de componentes

6.3 Características tecnológicas de las PCBs

Antes de comenzar con el diseño de las PCBs se deben determinar las características físicas que tendrá cada placa de circuito impreso sobre la que se realizarán los circuitos, así como definir los requerimientos tecnológicos impuestos por el fabricante de cada PCB, con el fin de que éste pueda garantizar su fabricación y la calidad de la misma.

6.3.1 Módulo de comunicaciones inalámbricas

Para el diseño del módulo de comunicaciones inalámbricas realizado en este PFC se ha decidido que la placa de circuito impreso conste de 4 capas, las cuales se han organizado de la siguiente forma: 2 capas externas para señales y 2 capas internas para albergar el plano de masa y alimentación. La decisión de que la placa de circuito impreso esté formada por 4 capas con la distribución comentada, se debe principalmente a que la implementación de circuitos de radiofrecuencia requiere el uso de planos de masa bien situados y extendidos en la placa con el objetivo de minimizar tanto el ruido, como la impedancia que pueda existir con la referencia de 0V del circuito. Además, en la parte de radiofrecuencia existirán componentes cuya conexión será realizada a través de líneas *microstrip* de 50Ω de impedancia característica, por lo que se hace totalmente indispensable la existencia de un plano de masa bajo la zona donde se encuentren localizadas las líneas del bloque de radiofrecuencia.

Los planos de masa y alimentación proporcionan otros beneficios que también son importantes, como el de facilitar el ruteado mediante el empleo de vías, de manera que todas las masas y alimentaciones son accesibles directamente con sólo colocar una vía hacia los planos intermedios. El uso de vías logra un ruteado más limpio, además de favorecer uno de los principales criterios de diseño, como es el de un reducido tamaño, ya que el empleo de vías reduce el espacio necesario en la ubicación y conexión de los componentes que contendrá la placa de circuito impreso.

En la Figura 6.14 se muestra un esquema en el que se puede observar la distribución de las capas, así como las principales características físicas de la placa con la que se ha fabricado la PCB del módulo de comunicaciones inalámbricas diseñada en el presente PFC.

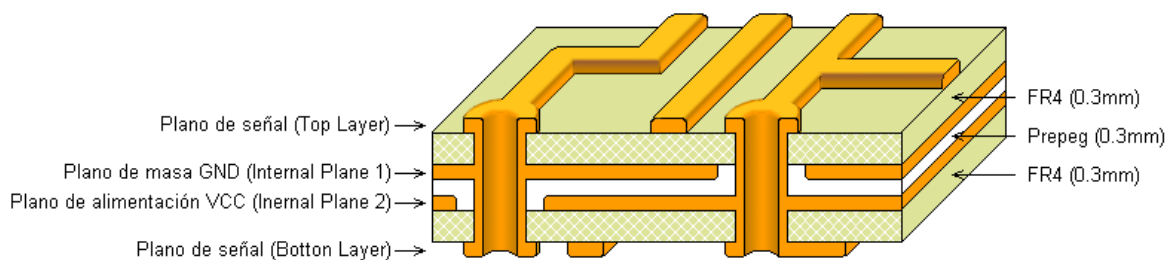


Figura 6.14. Características físicas y distribución de capas de la PCB del módulo de comunicaciones inalámbrico

Antes de comenzar la ubicación y ruteado de los componentes en la PCB, es necesario determinar algunos parámetros, como los anchos de pista mínimos y la separación mínima que debe de existir entre dos pistas diferentes, así como los tamaños mínimos de los taladros de las vías. Todos esos parámetros dependen del proceso de fabricación de la placa de circuito impreso y se denominan genéricamente *especificaciones de fabricación*, las cuales vienen determinadas por el fabricante, que será quien se encargue de producir las PCB diseñadas.

El fabricante elegido para la fabricación de las PCB es PCBCART [60], un fabricante de circuitos impresos de alta calidad emplazado en China.

A continuación se indican las especificaciones de fabricación proporcionadas por este fabricante, las cuales se deberán cumplir con el fin de asegurar que el diseño fabricado posee una buena calidad.

- *Ancho mínimo de pista:* de acuerdo con las especificaciones del fabricante, el ancho de pista que es capaz de fabricar es superior o igual a 8mil (0.2032mm). En consecuencia, se intentará, siempre que sea posible, emplear pistas de mayor anchura, teniendo especial cuidado en las pistas de alimentación o masa, que serán de 20mil (0.508mm) en adelante. En la Figura 6.15 se puede observar de forma gráfica la anchura mínima de pista permitida por el fabricante.

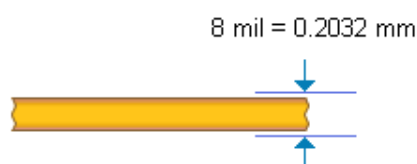


Figura 6.15. Ancho mínimo de pista

- *Separación mínima entre pistas*: ha de ser como mínimo de 8mil (0.2032mm), si bien, al igual que con la anchura de pistas, se intentarán realizar de un ancho mayor, siempre que exista espacio disponible y los requerimientos del diseño lo permitan. En la Figura 6.16 se puede observar de forma gráfica la separación mínima entre pistas permitida por el fabricante.

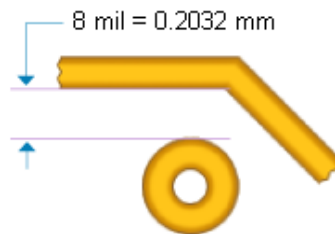


Figura 6.16. Separación mínima entre pistas

- *Diámetro mínimo de las vías*: ha de ser como mínimo de 8mil (0.2032mm). En este caso se ha decidido elegir un tamaño algo mayor para el taladrado de las vías existentes en la placa de circuito impreso; concretamente, se ha tomado como tamaño del taladro de las vías 15.7mil (0.4mm); este tamaño facilita las mediciones para la verificación de la placa PCB. En la Figura 6.17 se puede observar de forma gráfica el diámetro mínimo de las vías realizadas por el fabricante.

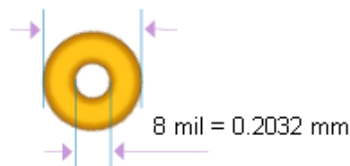
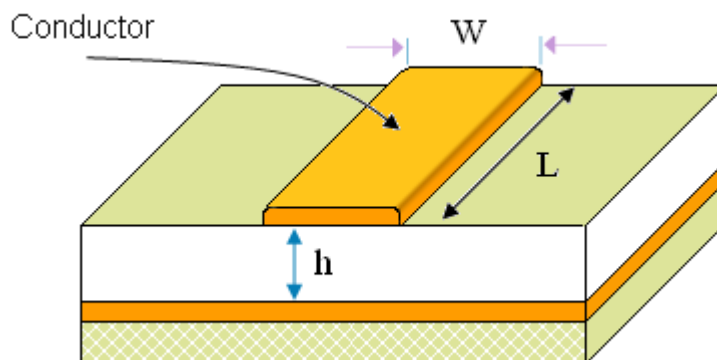


Figura 6.17. Diámetro mínimo de vías

- *Ancho de pistas microstrip para $Z_0=50\Omega$* : de acuerdo con las indicaciones proporcionadas por el fabricante, las pistas adaptadas a 50Ω han de ir en la capa superior, y el plano inmediatamente inferior ha de ser de masa. El ancho de pista ha de ser de 15.8mil (0.40132mm), de manera que $W=15.8\text{mil}$ en la representación de la Figura 6.18.

Figura 6.18. Ancho de pistas *microstrip*

6.3.2 Módulo sensor

En el diseño de la PCB del módulo sensor se ha empleado una placa de circuito impreso de 2 capas, una primera capa se encuentra en la parte superior del circuito impreso, denominada *Top Layer*, y la segunda capa, en la parte inferior del circuito impreso, se denomina *Bottom Layer*. Se ha decidido emplear 2 capas para lograr una reducción de tamaño, ya que ofrece más espacio para ubicar los componentes de montaje superficial, además de lograr un ruteado más limpio, facilitado mediante el empleo de vías.

En la Figura 6.19 se muestra un esquema en el que se puede observar la distribución de las capas, así como las principales características físicas de la placa con la que se ha fabricado la PCB del módulo sensor diseñado en el presente PFC.

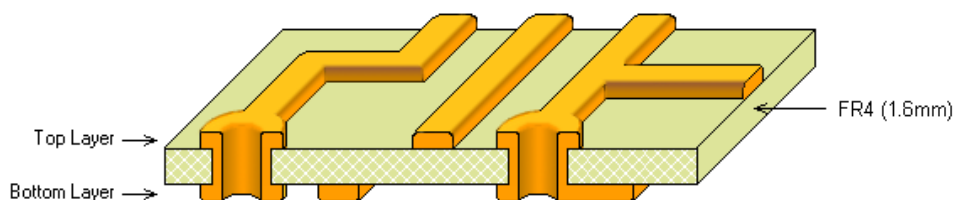


Figura 6.19. Características físicas y distribución de capas de la PCB del módulo sensor

La fabricación de las PCB del módulo sensor la ha realizado el Servicio de Fabricación de Prototipos de Circuitos Impresos y Sistemas Electrónicos del IUMA [69]. En la Tabla 6.2 se

indican las especificaciones de fabricación determinadas por el fabricante de la PCB del módulo sensor.

Especificaciones de fabricación	Valor
Anchura de pista mínimo	10mil (0.254mm)
Separación mínima entre pistas	10mil (0.254mm)
Diámetro mínimo de las vías	15.748mil (0.4mm)

Tabla 6.2. Especificaciones de fabricación de la PCB del módulo sensor

6.4 Diseño de las PCBs

El diseño de la PCB consiste en ubicar los componentes sobre su superficie de forma que faciliten el enrutamiento de todas las conexiones existentes en el diseño coherentemente e intentando minimizar el área total empleada por el circuito impreso. Antes de comenzar a posicionar los componentes y enrutar las conexiones en la placa, es necesario preparar el área de trabajo de la PCB.

En el desarrollo del presente Proyecto Fin de Carrera se han elaborado dos diseño de PCBs, en el primero se implementa el módulo de comunicaciones inalámbricas y en el segundo el módulo sensor, los cuales se conectarán mediante el conector de expansión implementado, obteniendo finalmente el módulo final completamente funcional. En la primera parte de este apartado se describirá el diseño de la PCB del módulo de comunicaciones inalámbricas y se finalizará con el diseño de la PCB del módulo sensor.

6.4.1 Módulo de comunicaciones inalámbricas

En primer lugar, para el diseño de la PCB del módulo de comunicaciones inalámbricas se debe crear un nuevo fichero PCB dentro del proyecto, de manera que pueda transferirse el diseño desde el editor de esquema, al editor de PCB. Dicho fichero será el que albergará toda la información relacionada con la ubicación de los componentes y cómo van conectadas las diferentes señales del diseño realizado.

Para crear un nuevo fichero PCB, el programa DXP 2004 dispone de un asistente de PCB denominado *PCB Board Wizard* que facilita la creación de la superficie de la placa. Durante el

proceso de creación mediante este asistente, es necesario introducir ciertos parámetros relacionados con la placa. El primer parámetro que solicita es la unidad de medida a utilizar, que en este PFC será el milímetro. En la siguiente página del asistente se deben definir las dimensiones de la placa, número de capas, el estilo de vías que se permiten en el diseño, y si se emplean componentes de montaje superficial. Las dimensiones inicialmente establecidas son de 5x5cm, empleando 4 capas (2 de señales y 2 de alimentación) como se comentó al principio de este capítulo, siendo el estilo vía seleccionado el de vías perforantes de lado a lado de la PCB. Además, se activó la casilla correspondiente al empleo de componentes de montaje superficial. La página final del asistente permite incorporar algunas de las reglas que se aplicarán en el diseño, y que se abordarán en el siguiente apartado.

Una vez finalizada la creación del fichero PCB con la superficie de la placa, el diseño del esquemático debe ser transferido al nuevo fichero PCB creado para proceder a posicionar los componentes y enrutar todas las conexiones. Este proceso es el encargado de convertir cada uno de los símbolos del esquemático interconexionados entre sí, en los *footprints* correspondientes, manteniendo el conexionado entre todos los componentes. En el programa DXP 2004 este proceso se realiza seleccionando en la aplicación *Design→Import changes*. Si no se ha producido ningún error, se mostrará el fichero PCB con los componentes listos para ser colocados y comenzar con el ruteado de todas las conexiones.

6.4.1.1 Reglas de diseño

Durante la creación del fichero PCB con el asistente *PCB Board Wizard* del programa DXP 2004, el software solicita la inserción de algunas reglas de diseño básicas que se aplicarán en el diseño de la PCB. Las reglas de diseño básicas, y el valor que se ha asignado en el asistente, son los que se muestran en la Tabla 6.3.

Reglas básica de diseño	Valor
Anchura de pista mínima por defecto	12 mil (0.3048 mm)
Distancia entre pistas diferentes mínima por defecto	12 mil (0.3048 mm)
Diámetro mínimo del taladrado por defecto	15.748 mil (0.4 mm)

Tabla 6.3. Reglas básicas de diseño empleadas en el módulo de comunicaciones inalámbricas

6.4.1.2 Consideraciones especiales sobre la antena

En el diseño del PCB, la ubicación de la antena recibe especial importancia respecto a otros componentes. Esta prioridad se debe a la alta sensibilidad que presenta con respecto al ruido. Así, eligiendo un emplazamiento adecuado se puede minimizar la cantidad de ruido que recibe, mejorando con ello la relación señal/ruido de emisión y recepción.

Una ubicación adecuada se considera aquella que está lo más alejada posible de las fuentes que puedan generar ruido. El ruido puede provenir de diversas fuentes, la mayoría de ellas externas y que no se pueden minimizar, si bien hay otras que se generan en la misma placa de circuito impreso, como señales digitales de los microcontroladores, señales de entrada del convertidor analógico/digital, etc. Estas últimas sí se pueden minimizar con un correcto emplazamiento de la antena, como puede ser uno de los extremos de la placa de circuito impreso, y a ser posible, en una esquina de la misma. El ruido que pueda recibir la antena en una esquina es menor que en cualquier otra ubicación debido a que alrededor se encuentran menos componentes.

Una antena es un dispositivo diseñado con el objetivo de emitir o recibir ondas electromagnéticas hacia o desde el espacio libre. Para ello, la antena se encarga de transformar corrientes eléctricas en ondas electromagnéticas cuando está en transmisión, realizando la función inversa cuando está en recepción.

Como se comentó en el capítulo 5, la antena empleada en el diseño realizado es una antena en chip, indicada para la banda de 2.4GHz, con patrón de radiación omnidireccional, es decir, capaz de radiar con igual intensidad en todas las direcciones. En la Figura 6.20 se puede observar una foto de la antena en chip utilizada.

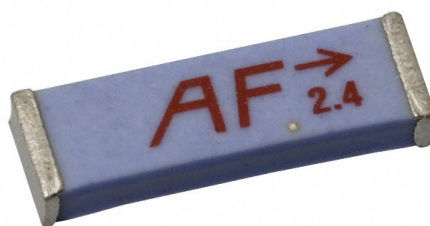


Figura 6.20. Antena empleada en el PFC

Además de ubicar correctamente la antena, también se deben tener en cuenta otras consideraciones importantes en el proceso de diseño de la PCB. Debido a su patrón de radiación omnidireccional, es recomendable que bajo la zona donde se ubique el chip de la antena no exista ningún plano de masa

o alimentación, con el fin de que la antena sea capaz de radiar, en caso de transmisión, o recibir, en caso de recepción, a la máxima potencia posible en todas las direcciones. Otra consideración sobre la antena que fue comentada anteriormente en este capítulo, es el empleo de líneas *microstrip* de 50Ω de impedancia característica en el bloque de radiofrecuencia, lo que implica que la conexión a la antena deba realizarse a través de una línea *microstrip* de 50Ω de impedancia característica. En la Figura 6.21 se puede observar de forma gráfica cómo ha de ir conectada en el diseño del circuito impreso la antena en chip seleccionada, cumpliendo las consideraciones indicadas.

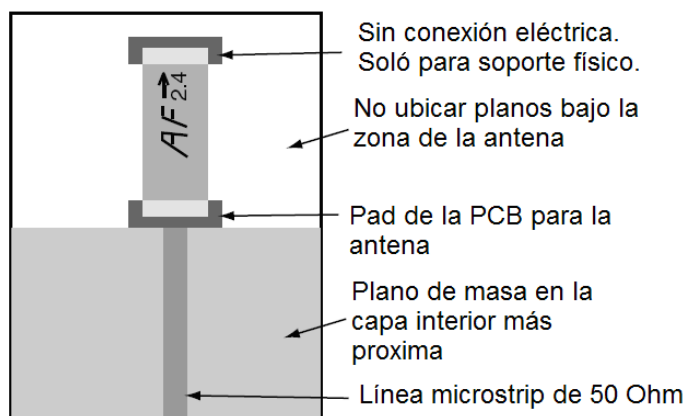


Figura 6.21. Ubicación de la antena y los planos de masa

6.4.1.3 Distribución de componentes por zonas

Es muy importante en el diseño de la PCB, que antes de comenzar a ubicar los diferentes componentes que integran el diseño, se plantee cuál es la distribución que asegura un ruteado limpio y ordenado de las pistas, atendiendo a varios factores, y dándole mayor importancia a determinadas zonas, como aquellas críticas del circuito, las alimentaciones, los distintos tipos de masa, etc. Además de lo comentado, en la distribución de los componentes se ha de intentar que el área sea lo mínima posible, y las pistas de las señales lo más cortas que permita el diseño.

Una vez decidido el emplazamiento más adecuado de los componentes bajo los criterios antes comentados, la placa de circuito impreso presenta una forma rectangular, con varias zonas bien definidas, como la zona de radiofrecuencia en uno de los extremos, y en el otro extremo la zona de alimentación, con un espacio dedicado a la ubicación de la pila de botón.

En la Figura 6.22 se puede observar el aspecto que presenta la distribución seleccionada en el diseño del circuito impreso realizado en el Proyecto Fin de Carrera.

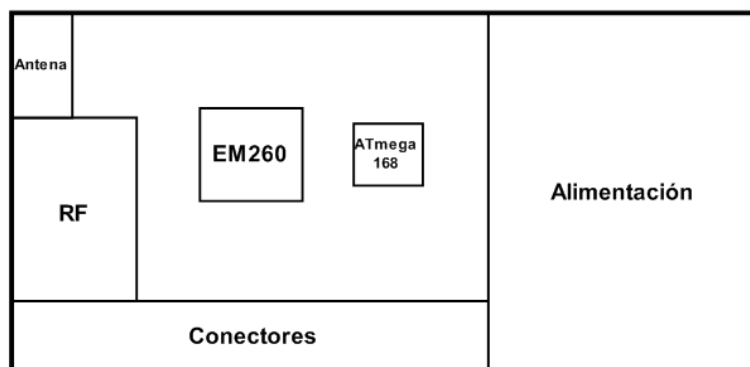


Figura 6.22. Distribución del diseño en la PCB del módulo de comunicaciones inalámbricas

Como se puede observar en la Figura 6.22, en la esquina superior izquierda se ha ubicado la antena, lo más lejos posible de la alimentación y de la lógica digital que presenta el microcontrolador ATmega168; de esta manera se intenta minimizar el ruido que puede afectar a la antena, siguiendo las consideraciones comentadas en el apartado anterior.

Por otro lado, justamente debajo de la antena se encuentra la zona que contiene el hardware de radiofrecuencia, compuesto por el filtro de RF, el conector externo, el balún, y todos los componentes necesarios para el correcto funcionamiento de la circuitería de RF. En la zona RF se han intentado colocar los componentes lo más cerca posible para interconectarlos entre sí con pistas muy cortas, a fin de evitar posibles pérdidas debidas a reflexiones o desadaptaciones en las líneas que interconectan los diferentes componentes.

El microcontrolador ATmega168 y el co-procesador EM260 se encuentran ubicados en el centro de la distribución del circuito impreso, facilitando la conexión SPI entre ellos, así como la conexión con los conectores que se encuentran justo debajo, necesarios para la programación del *firmware*. Además de los conectores de programación, en la zona de conectores se ubica el conector de expansión, mediante el cual se ponen a disposición del circuito sensor las señales del ATmega168.

En el lado derecho del circuito impreso se ha colocado el bloque de alimentación, formado por el regulador de tensión con su circuitería asociada, el diodo Schottky y el contenedor de la pila. Hay que tener especial cuidado en la zona donde va ubicado el contenedor de pila, debido a que la masa de la pila de botón, una vez colocada su parte inferior (masa), hace contacto con la placa de circuito impreso, evitando en ese área colocar vías que no sean masa, ya que se podría producir un cortocircuito por un contacto a masa no esperado en alguna señal.

6.4.1.4 Ruteado de las pistas

Después de analizar las posibilidades de distribución de componentes y haber elegido la más satisfactoria, se procede a ubicar los componentes de las diferentes zonas y a rutearlos entre sí. Se comienza por la zona más crítica del diseño, en este caso la parte superior y central del lado izquierdo, que es donde se ubican la antena y los componentes de radiofrecuencia.

El ruteado de la zona de radiofrecuencia se realiza con especial atención, debido a los problemas que puede causar una mal ruteado, como las pérdidas de en las líneas de transmisiones o las reflexiones. Para evitar estos efectos, los componentes de radiofrecuencia deben estar lo más próximos posibles entre ellos, permitiendo que las conexiones sean lo más cortas y rectas posibles.

Una vez finalizada la zona de radiofrecuencia se procede a rutear el resto del circuito, procurando siempre ocupar el menor área posible. Con el fin de minimizar el área se han empleado ambas caras del circuito impreso para la ubicación de componentes. Además, en muchos casos se han utilizado ambas caras para que las pistas puedan completar las interconexiones entre los componentes. Las interconexiones entre caras se han realizado mediante vías, que son las encargadas de conectar la capa superior e inferior de la PCB, es decir *Top Layer* y *Bottom Layer*, respectivamente. Especial dificultad presentó el ruteado en las proximidades del microcontrolador ATmega168 y el co-procesador EM260, debido a la densidad de pads que se concentran en esta zona, obligando a emplear una gran cantidad de vías para completar la interconexión de los componentes. Las dimensiones de la placa de circuito impreso finalmente se han fijado en 34.5mm x 52.7 mm.

En la Figura 6.23 y en la Figura 6.24 se puede observar el diseño de la cara superior (*Top Layer*) e inferior (*Bottom Layer*), respectivamente, de la PCB diseñada.

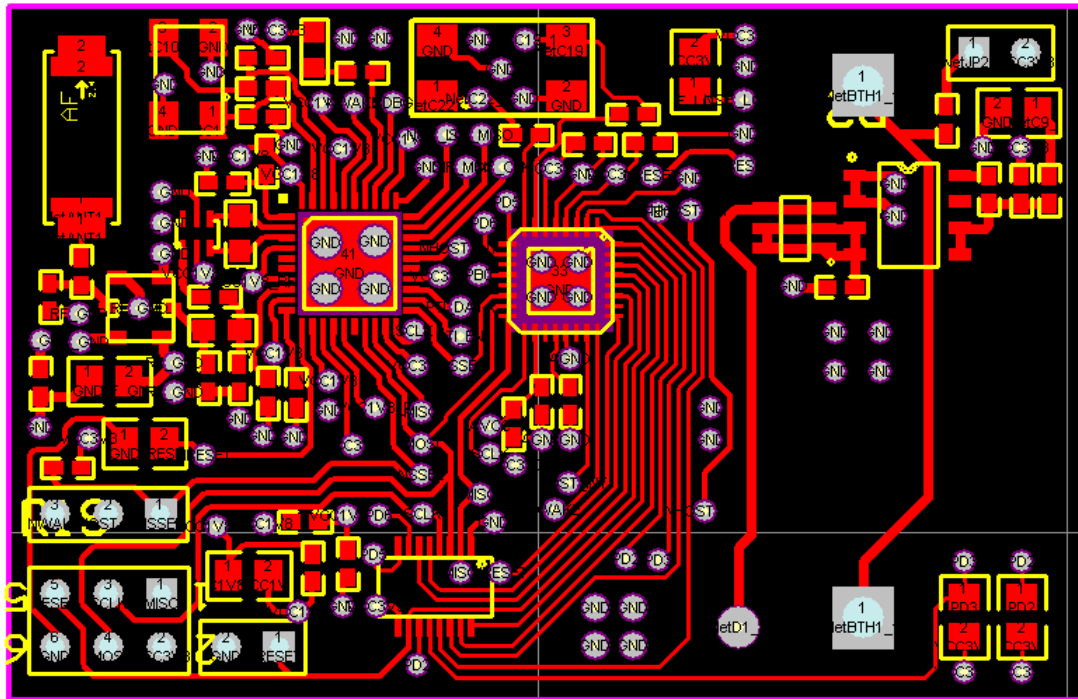


Figura 6.23. Diseño de *Top Layer* del módulo de comunicaciones inalámbricas

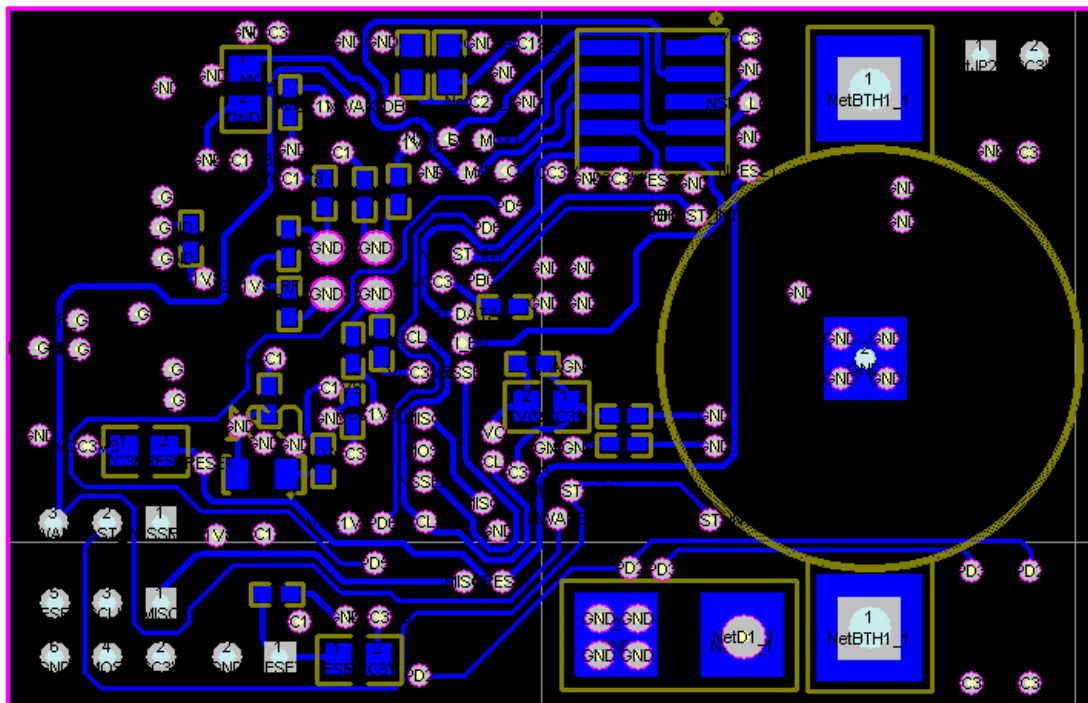


Figura 6.24. Diseño de *Bottom Layer* del módulo de comunicaciones inalámbricas

6.4.1.5 Distribución del plano de masa

El plano de masa ha sido fragmentado en 3 secciones diferentes, y en función de la parte del circuito a la que se encuentren conectados, le corresponde una de las 3 masas existentes en el diseño, denominadas RF_GND, AGND y GND. Cada sección de las masas se distribuye para una serie de conexiones:

- RF_GND: Masa de los componentes que forman el bloque de radiofrecuencia. En esta sección del plano se conectan todas las terminaciones a masa de los componentes y circuitos involucrados en el bloque de radiofrecuencia.
- AGND: Masa analógica, en lo que se conecta toda la circuitería analógica que forma parte del convertidor analógico/digital del microcontrolador ATmega168.
- GND: Masa general, empleada en todas las terminaciones a masa del diseño, excepto las masas antes comentadas, las cuales pertenecen al bloque de radiofrecuencia, y la circuitería del convertidor analógico/digital del microcontrolador ATmega168.

Las 3 secciones que forman el plano de masa están separadas mediante ferrita o, en su defecto, resistencias de 0Ω , con el fin de minimizar el ruido que puedan introducir las masas más ruidosas en otras menos ruidosas, evitando que el ruido proveniente de una parte del circuito se acople en otras partes a través del plano de masa. De este modo, las distintas secciones de masa se encuentran virtualmente aisladas.

El plano de masa está situado en una de las capas interiores, en este caso la interior superior, que se encuentra situada bajo la capa *Top Layer*, y se denomina *Internal Plane 1*. La situación de la capa de masa determina la capa que alberga las líneas de transmisión *microstrip* con impedancia característica de 50Ω , necesarias en el bloque de radiofrecuencia. En este caso, se rutearán en *Top Layer*, ya que estas líneas deben estar situadas a muy poca distancia del plano de masa.

La distribución del plano de masa se ha realizado a medida que se han ido ruteando las diferentes pistas de interconexión que conforman la placa de circuito impreso diseñada. Para realizar la división del plano de masa en las 3 secciones que lo forman, se ha delimitado el área de cada masa en función del número de vías que albergue esa área. Así, en la zona donde se encuentre el mayor número de vías de una masa concreta, se ubicará el plano de esa masa. De esta forma se ha facilitado el ruteado y se ha reducido el tamaño de la placa de circuito impreso diseñada.

En la Figura 6.25, la Figura 6.26 y la Figura 6.27, se puede observar el esquema PCB de la capa *Internal Plane 1*, donde se pueden distinguir de forma resaltada cada una de las 3 masas que componen el plano de masa, correspondientes a RF_GND, AGND y GND, respectivamente.

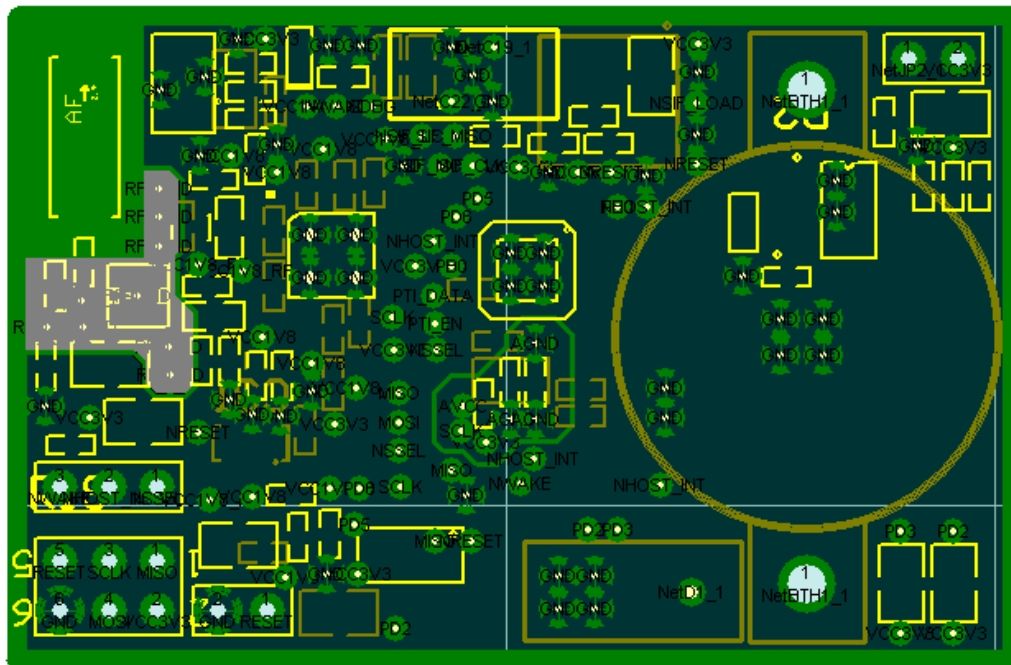


Figura 6.25. Sección RF_GND del plano de masa

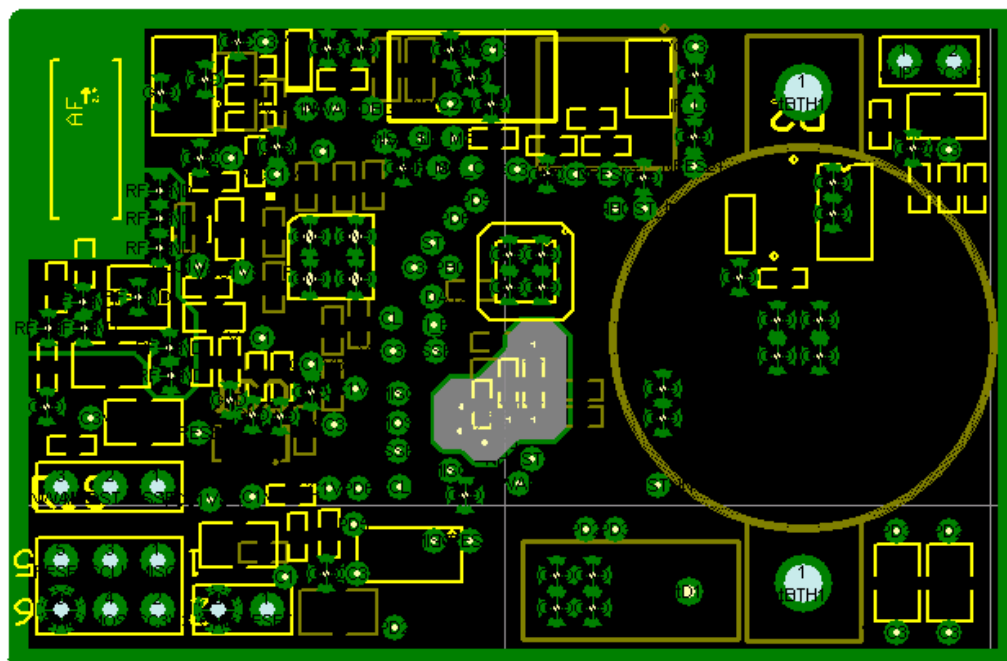


Figura 6.26. Sección AGND del plano de masa

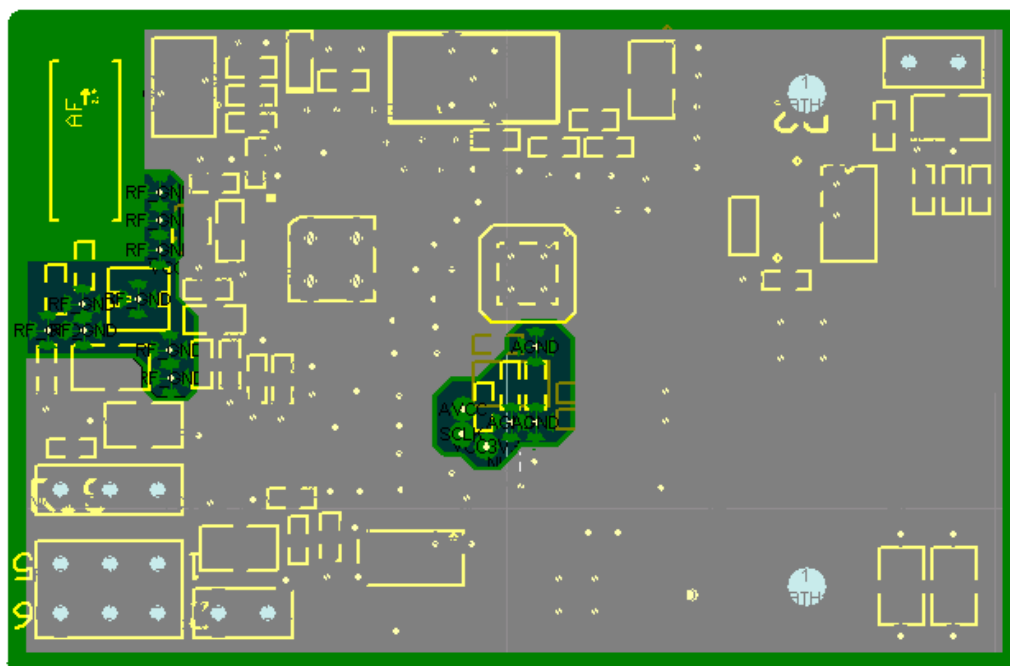


Figura 6.27. Sección GND del plano de masa

Como se puede observar en las figuras anteriores, y siguiendo las recomendaciones del apartado 6.4.1.2, en el área del circuito impreso donde se ubica la antena de radiofrecuencia no se ha situado ningún plano de masa.

6.4.1.6 Distribución del plano de alimentación

Con el fin de simplificar y facilitar el ruteado en el diseño de la placa de circuito impreso, el plano de alimentación, al igual que el plano de masa, se ha fragmentado en varias secciones, cada una de las cuales alberga una de las alimentaciones que más conexiones presentan en el diseño, de acuerdo a los requerimientos en cuanto a voltaje y estabilidad. El proceso para el diseño del plano de alimentación es idéntico al realizado en el plano de masa, intentado englobar en el área de cada sección de alimentación el mayor número de conexiones correspondientes a cada tipo de alimentación. Entre las 4 alimentaciones que forman el plano de alimentación se encuentran:

- VCC1V8_RF: Alimentación de 1.8V, empleada en los componentes que forman el bloque de radiofrecuencia, además de la parte del dispositivo EM260 que implementa las funciones de radiofrecuencia.

- VCC1V8_D: Alimentación de 1.8V, empleada en los bloques funcionales digitales del dispositivo EM260, entre los que se encuentran la memoria flash integrada y el núcleo del EM260.
- VCC1V8: Alimentación de 1.8V general, destinada a alimentar a todos los bloques funcionales de 1.8V del EM260, exceptuando los comentados anteriormente, además del VCO del dispositivo EM260, que se alimenta con la alimentación denominada 1.8V_VCO.
- VCC3V3: Alimentación de referencia, que proporciona los 3.3V que necesita el EM260, el ATmega168, y el resto de circuitería del diseño, exceptuando la parte del convertidor analógico/digital que se alimenta con la alimentación denominada AVCC.

El plano de alimentación segmentado en las 4 secciones descritas se ha situado en la capa interior inferior, que se encuentra situada sobre la capa *Bottom Layer*, se denomina *Internal Plane 2*.

En la Figura 6.28, la Figura 6.29, la Figura 6.30 y la Figura 6.31, se puede observar el esquema de la PCB de la capa *Internal Plane 2*, mostrándose de forma resaltada cada una de las 4 alimentaciones que forman el plano de alimentación, correspondientes a VCC1V8_RF, VCC1V8_D, VCC1V8 y VCC3V3, respectivamente.

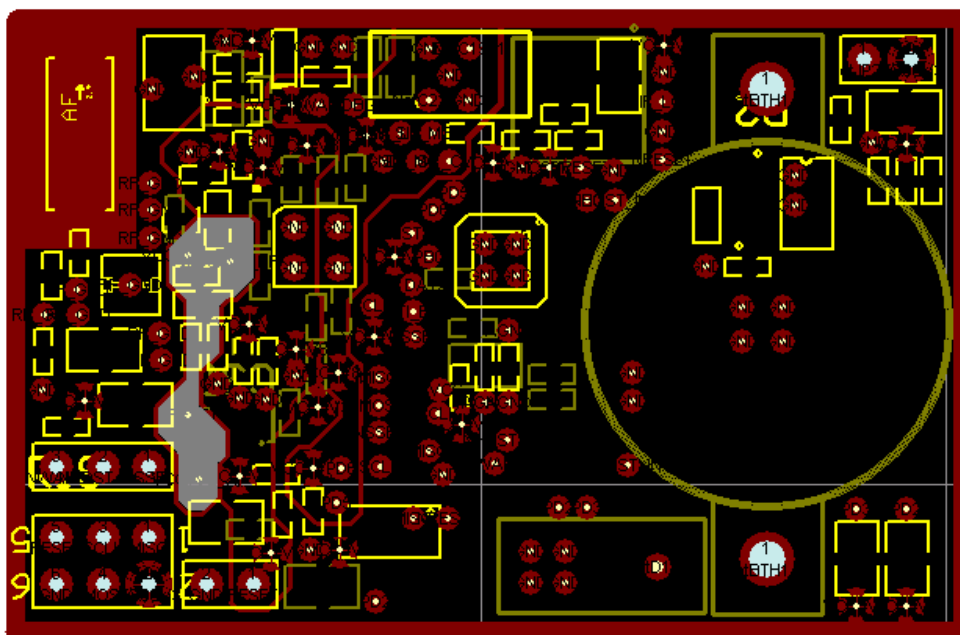


Figura 6.28. Sección VCC1V8_RF del plano de alimentación

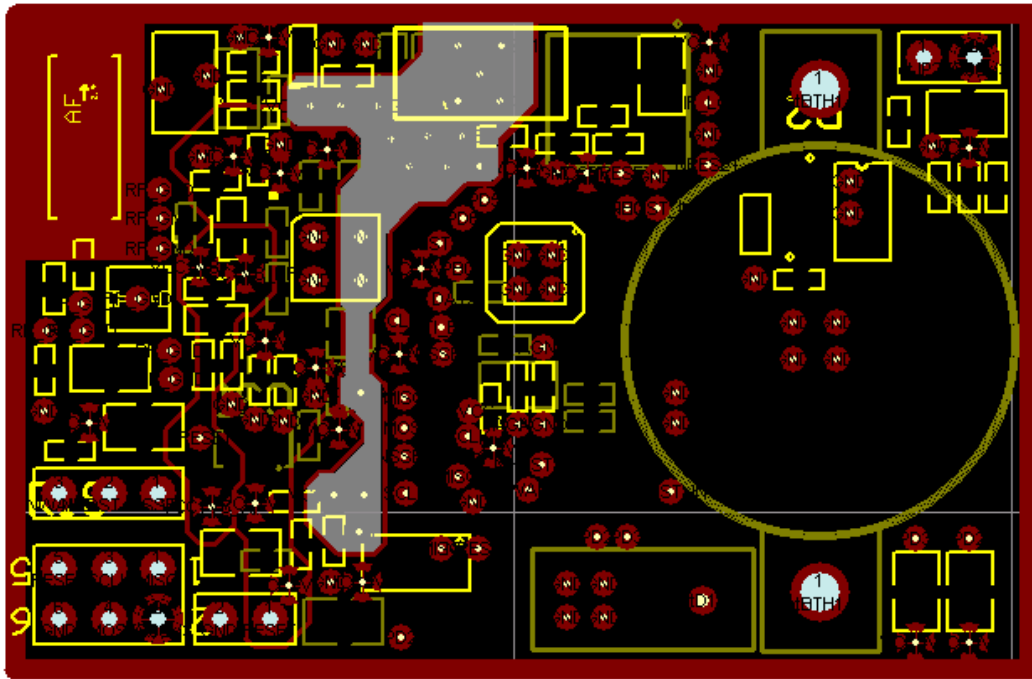


Figura 6.29. Sección VCC1V8_D del plano de alimentación

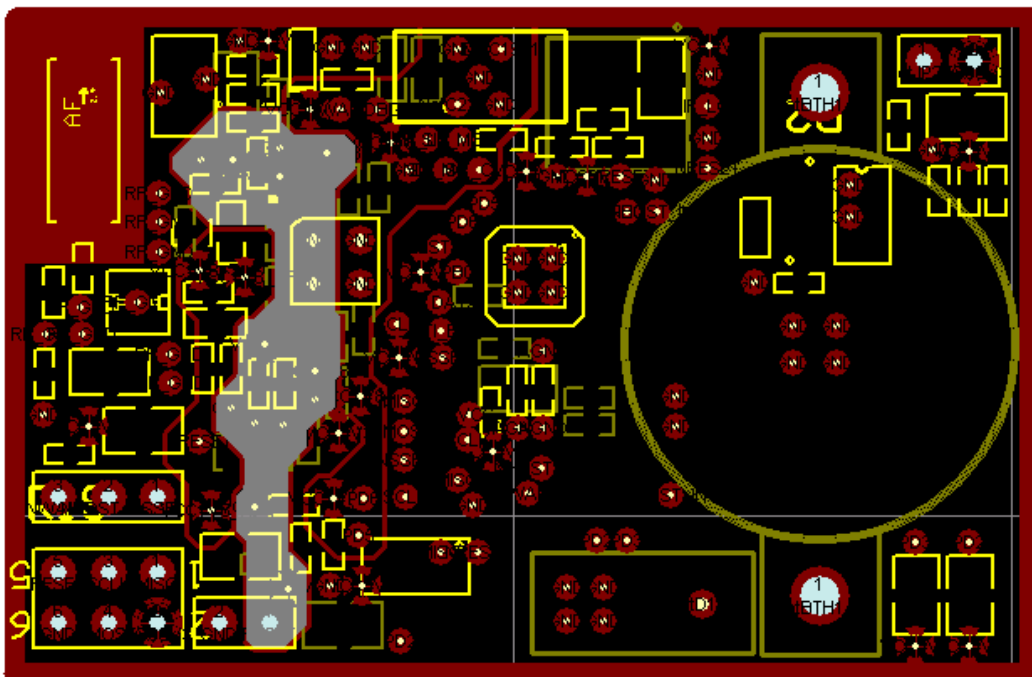


Figura 6.30. Sección VCC1V8 del plano de alimentación

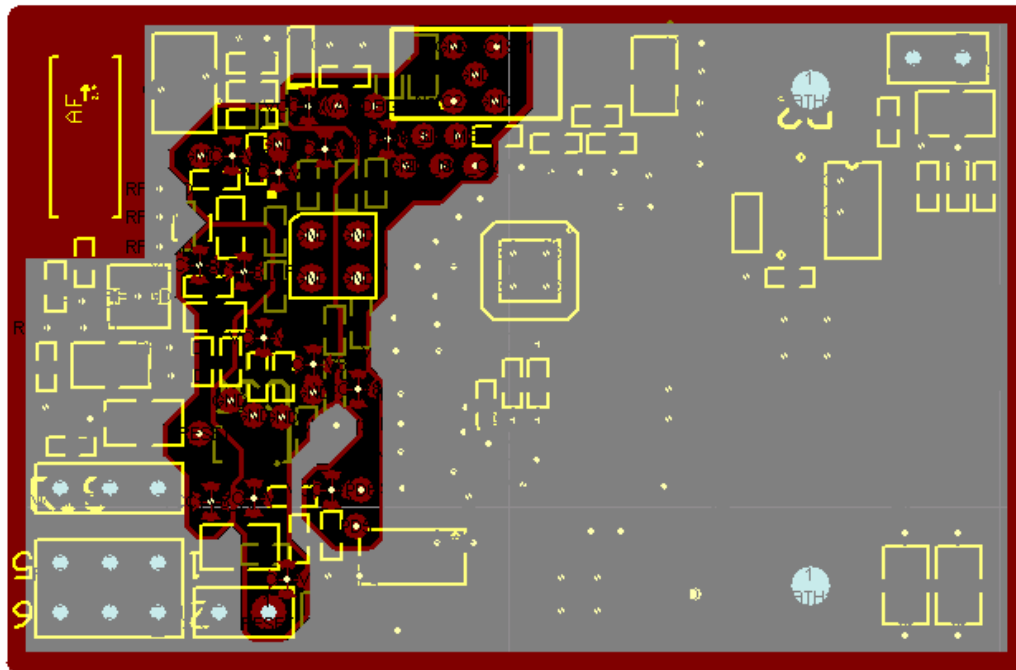


Figura 6.31. Sección VCC3V3 del plano de alimentación

Igual que en el caso del plano de masa, en Figura 6.28, la Figura 6.29, la Figura 6.30 y la Figura 6.31, se puede observar que en el espacio donde se ubica la antena de radiofrecuencia no se ha situado ningún plano de alimentación, siguiendo las recomendaciones del apartado 6.4.1.2.

6.4.1.7 Diseño final

Una vez completada la ubicación de todos los componentes, realizado el ruteado de todas las pistas necesarias en la PCB, y creados adecuadamente los planos de masa y alimentación, se realiza una última revisión de la placa de circuito impreso, incluyendo la verificación de todas las reglas de diseño que se definieron. Si la verificación es satisfactoria, se procede a generar los ficheros de salida del proyecto, conocidos como ficheros *gerber* o de fabricación, que son una serie de ficheros que contienen todas y cada una de las dimensiones de las pistas, así como de los *footprints* separados de cada capa física. Estos ficheros del diseño son los que se entregan al fabricante de la placa de circuito impreso como guía para el proceso de fabricación de la PCB.

Aunque cada capa generará un fichero diferente de salida, los cuales serán entregados al fabricante, en la Figura 6.32 se puede observar cómo queda la salida del fichero *gerber* con todas las capas visibles y diferenciadas mediante colores.

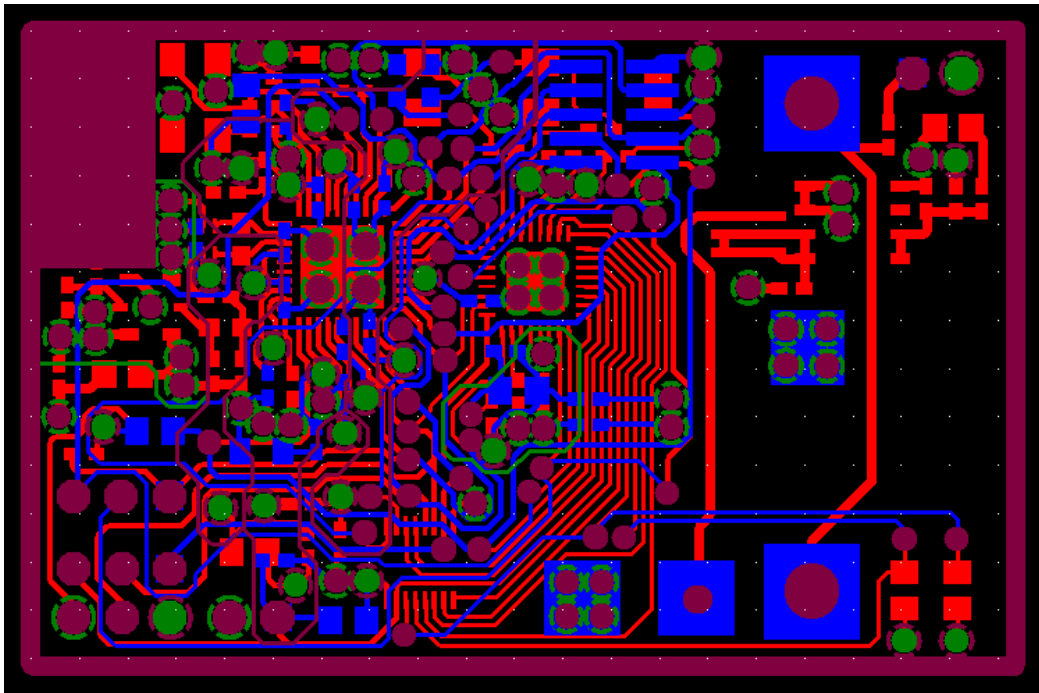


Figura 6.32. Fichero de salida *gerber* del módulo de comunicaciones inalámbricas diseñado

En la Figura 6.33 y la Figura 6.34 se muestra el aspecto que presentan las capas *Top Layer* y *Bottom Layer*, respectivamente, de la placa de circuito impreso, una vez fabricada.

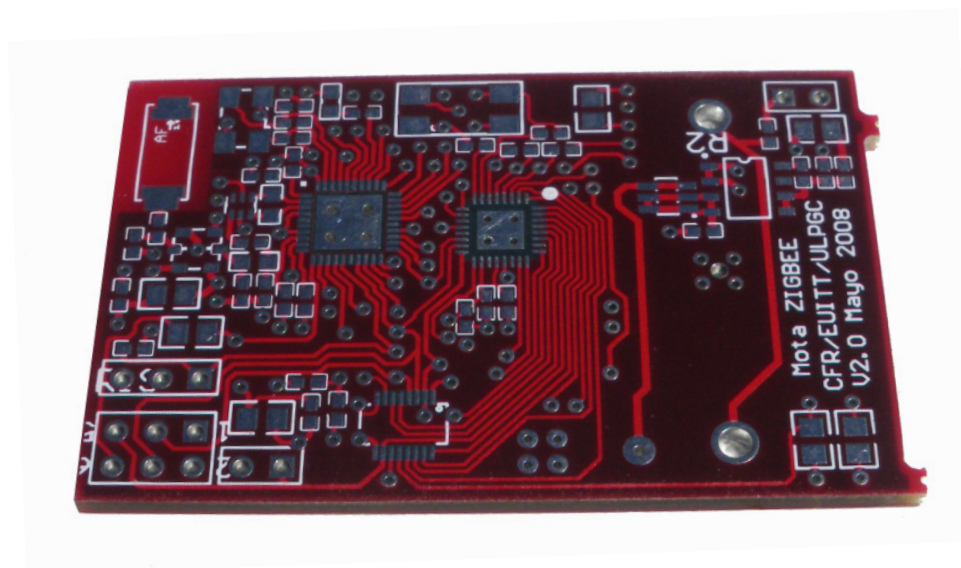


Figura 6.33. Capa *Top Layer* de la PCB fabricada

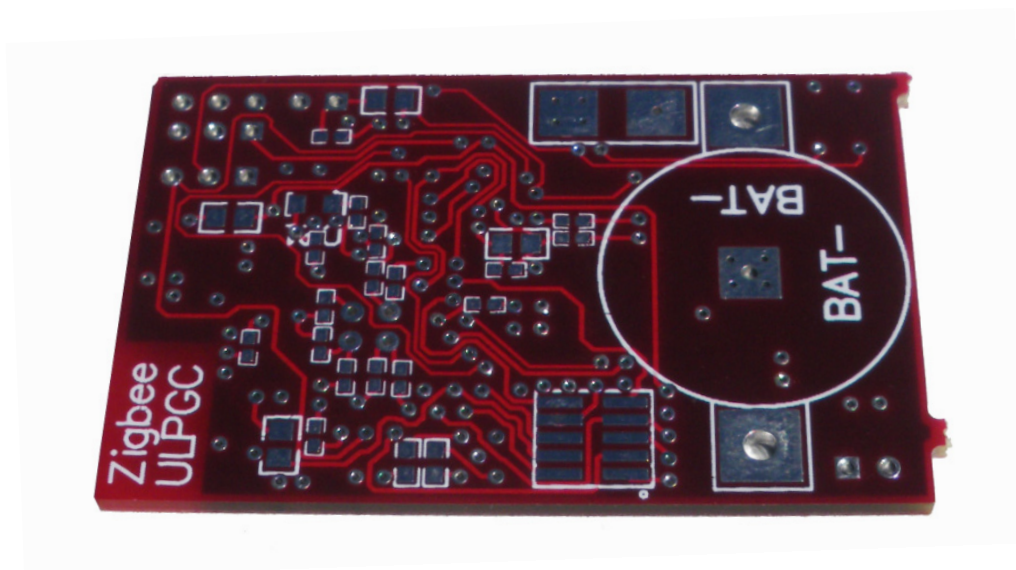


Figura 6.34. Capa *Bottom Layer* de la PCB fabricada

6.4.2 Módulo sensor

Para crear el circuito impreso del bloque sensor, repitiendo los pasos seguidos en el diseño de la PCB del módulo de comunicaciones inalámbricas, en primero lugar y mediante el asistente de PCB, el *PCB Board Wizard*, se crea el nuevo fichero PCB dentro del proyecto, de manera que se pueda transferir el esquemático del bloque sensor al editor de PCB.

En el asistente PCB del programa DXP 2004 se introducen los parámetros necesarios, como son unidad de medida, dimensiones de la placa (inicialmente establecidas), número de capas, estilo de las vías, tipos de componentes y reglas de diseño requeridas en la PCB. En la Tabla 6.4 se pueden observar los parámetros establecidos en el asistente *PCB Board Wizard*. Las reglas de diseño se especificaran en el siguiente apartado.

Parámetros introducidos en el asistente de PCB	Valor
Unidad de medida	Métrico
Dimensiones (inicialmente establecidas)	15mm x 40 mm
Número de capas	2 (Señales)
Estilo de las vías	Perforantes de lado a lado
Tipos de componentes	Montaje superficial (SMD)

Tabla 6.4. Parámetros establecidos en el asistente de PCB para el módulo sensor

Una vez finalizado la creación del fichero PCB, sólo queda transferir el esquemático creado al editor de PCB para comenzar a ubicar los componentes y rutear las pistas. De la misma manera que se procedió para el módulo de comunicaciones inalámbricas, se utiliza el comando *Design→Import changes*, dentro del editor de esquema del programa DXP 2004. Finalizado este paso con éxito, ya se encuentra en disposición de comenzar el colocado y ruteado de las pistas de la PCB.

6.4.2.1 Reglas de diseño

En la Tabla 6.5 se pueden observar las reglas de diseño básicas que se han introducido durante la creación del fichero PCB en el asistente *PCB Board Wizard* del programa DXP 2004.

Reglas básica de diseño	Valor
Anchura de pista mínima por defecto	0.3048 mm (12mil)
Distancia entre pistas diferentes mínima por defecto	0.3048 mm (12mil)
Diámetro mínimo del taladrado por defecto	0.4 mm (15.748 mil)

Tabla 6.5. Reglas básicas de diseño empleadas en el módulo sensor

6.4.2.2 Distribución de componentes

La distribución de los componentes se ha realizado atendiendo a varios factores, el más importante y que no se puede obviar, es la zona crítica que rodea el conector de expansión en la cara inferior de la PCB (*Bottom Layer*), debido a que la conexión entre los módulos, mediante el conector de expansión, provoca una superposición entre las placas de circuito impreso de los módulos. Así, se debe evitar ubicar componentes en la zona adyacente al conector que imposibiliten u obstaculicen una conexión óptima entre el módulo de comunicaciones inalámbricas y el módulo sensor. En la Figura 6.35 se representa gráficamente la zona crítica donde se superpone el módulo sensor sobre el módulo de comunicaciones inalámbrico.

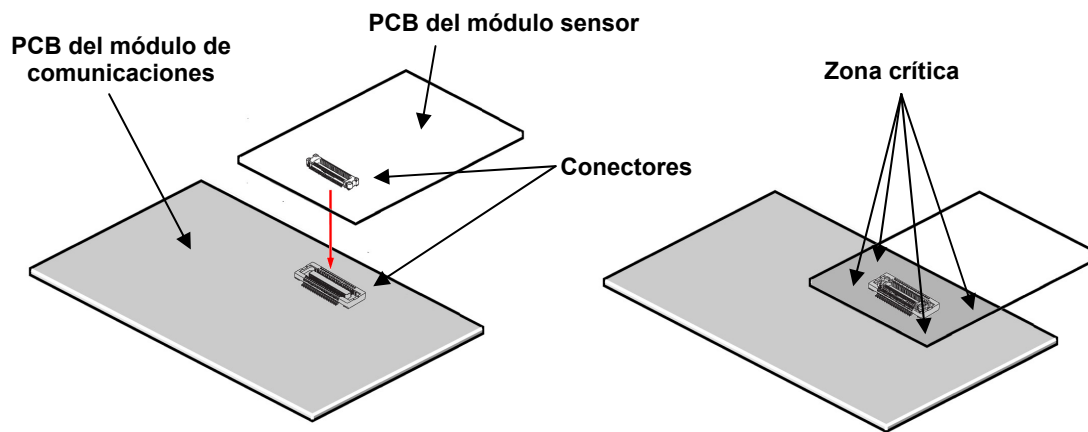


Figura 6.35. Conexión PCB del módulo de comunicaciones inalámbricas y el módulo sensor

Además de la zona crítica mencionada, también influyen en la distribución aspectos específicos de algunos componentes, como la necesidad de permitir una buena visibilidad a los diodos LEDs o la ubicación del interruptor y del pulsador en zonas de fácil acceso. Igualmente importante es lograr un ruteado limpio con las pistas ordenadas, sin olvidar criterios del diseño como su tamaño reducido y pistas lo más cortas posibles. En algunos casos se tiene que llegar a un compromiso de diseño entre diferentes factores, ya que lograr ruteados limpios con las pistas ordenadas y tamaño reducido puede conducir a una contradicción.

Finalmente, con la ubicación de los componentes decididos, la placa de circuito impreso presentará una forma rectangular con una dimensión final de 35.5mm x 13.4mm, logrando que el área de superposición del módulo de comunicaciones inalámbricas y el módulo sensor sea mínimo. Además como se comentó en el párrafo anterior, tanto los diodos LEDs como el pulsador y el interruptor se ubicaran en los extremos de la placa y en la cara superior, logrando buena visibilidad y facilitando el acceso a su manipulación, respectivamente. En la Figura 6.36 y la Figura 6.37 se puede observar la distribución de los componentes más relevantes del módulo sensor en la cara superior (*Top Layer*) e inferior (*Bottom Layer*), respectivamente.

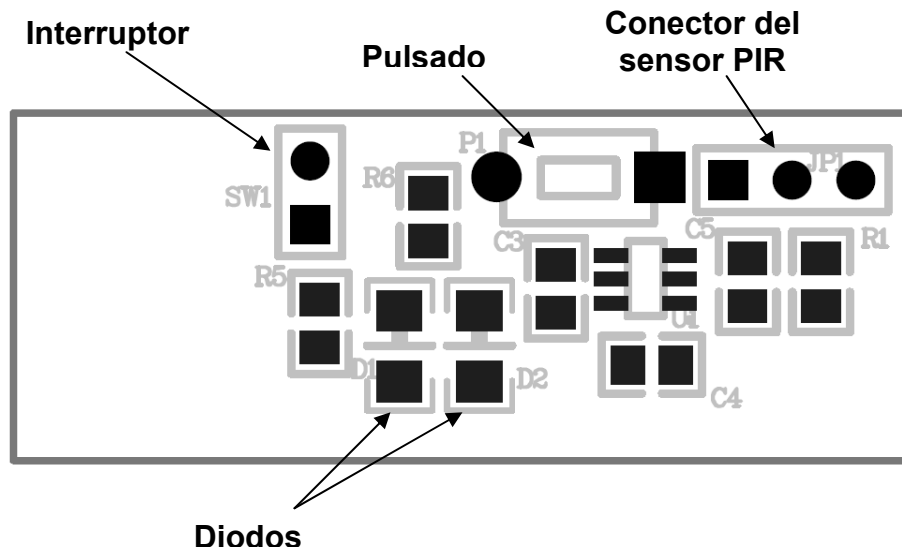


Figura 6.36. Distribución de componentes de la PCB del módulos sensor (*Top Layer*)

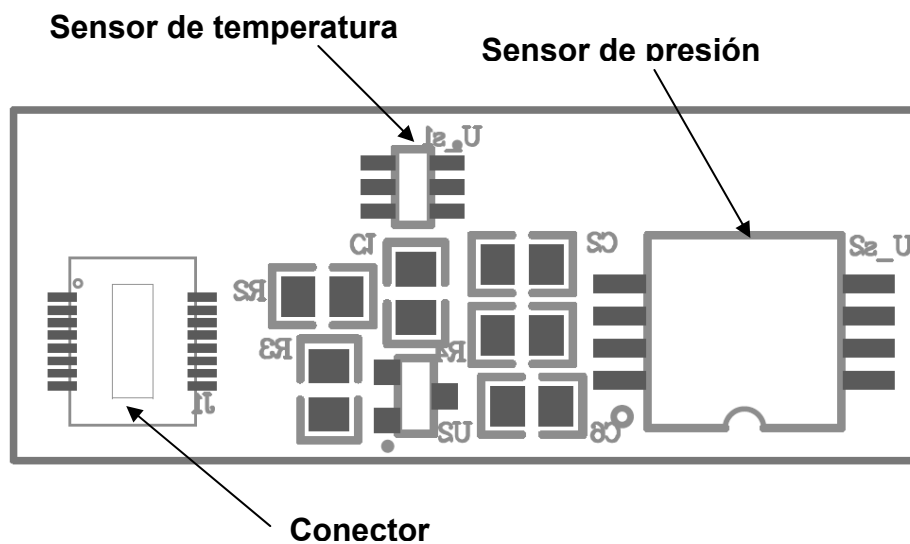


Figura 6.37. Distribución de componentes de la PCB del módulos sensor (*Bottom Layer*)

6.4.2.3 Ruteado de las pistas

Una vez realizada la distribución de los componentes, se puede comenzar a rutear las pistas. Primero se rutearán las zonas donde el ruteado presenta una mayor dificultad, como en las proximidades del conector de expansión, donde las pistas se encuentran muy próximas entre sí, y el espacio para distribuirlas al resto de los componentes es ajustado.

Posteriormente se ruteará el resto del circuito intentando ocupar el mínima área posible empleando para ello ambas caras en la ubicación de los componentes y el ruteado de las pistas. La capa superior e inferior (*Top Layer* y *Bottom Layer*) se unen mediante vías donde ha sido necesario.

En la Figura 6.38 y la Figura 6.39 se puede observar el diseño de la cara superior (*Top Layer*) e inferior (*Bottom Layer*), respectivamente, de la PCB diseñada.

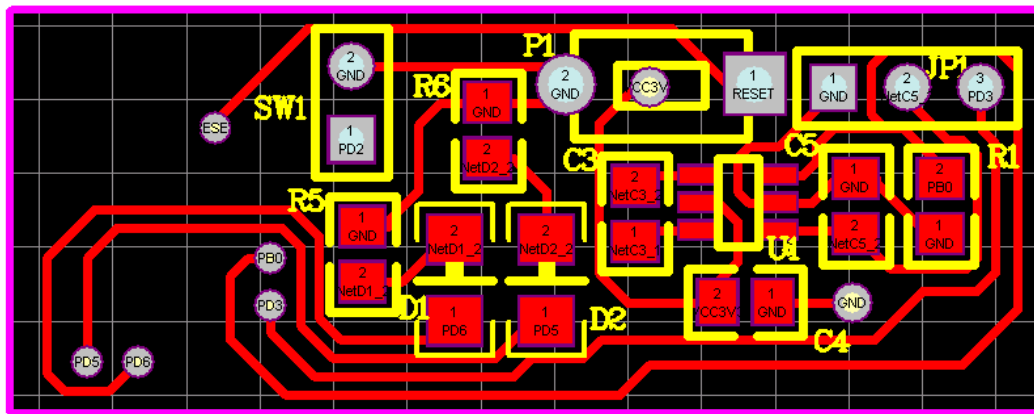


Figura 6.38. Diseño de la *Top Layer* del módulo sensor

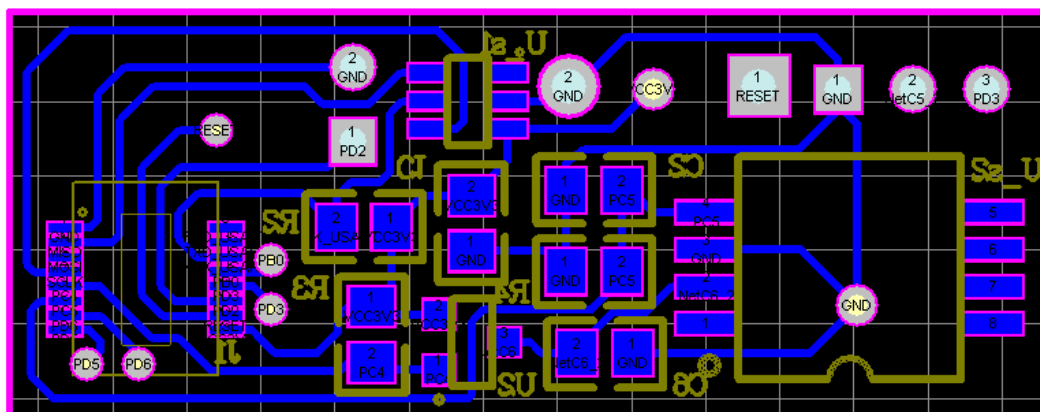


Figura 6.39. Diseño de la *Bottom Layer* del módulo sensor

6.4.2.4 Diseño final

Una vez completada la ubicación de todos los componentes, realizado el ruteado de todas las pistas necesarias en la PCB, y verificadas todas las reglas de diseño anteriormente definidas, si la verificación es satisfactoria, se procede a generar los ficheros de salida del proyecto, conocidos como ficheros *gerber* o de fabricación, explicados anteriormente en el apartado 6.4.1.7.

En la Figura 6.40 se puede observar cómo queda la salida del fichero *gerber* de la PCB diseñada con todas las capas visibles y diferenciadas mediante colores, cara superior (*Top Layer*) e inferior (*Bottom Layer*), respectivamente.

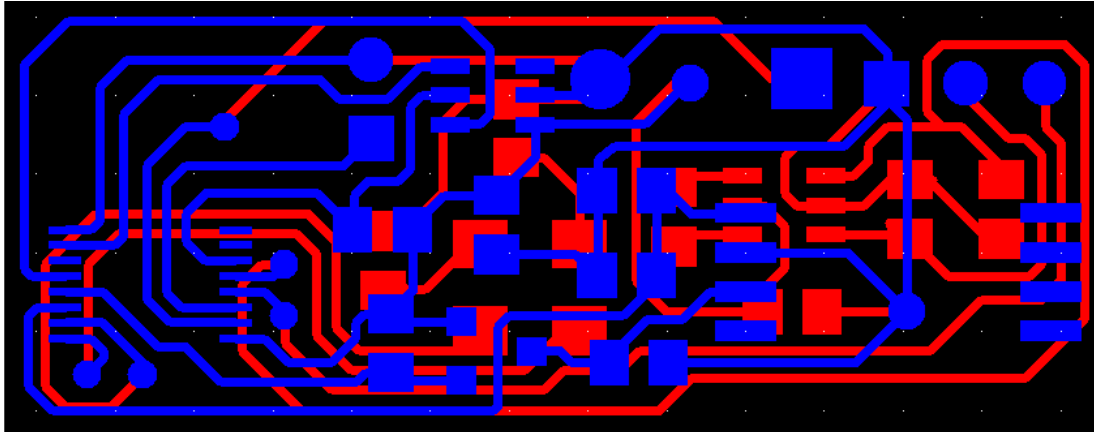


Figura 6.40. Fichero de salida gerber del módulo sensor diseñado

En la Figura 6.41 y la Figura 6.42 se muestra el aspecto que presentan las capas *Top Layer* y *Bottom Layer*, respectivamente, de la placa de circuito impreso del módulo sensor, una vez fabricada.

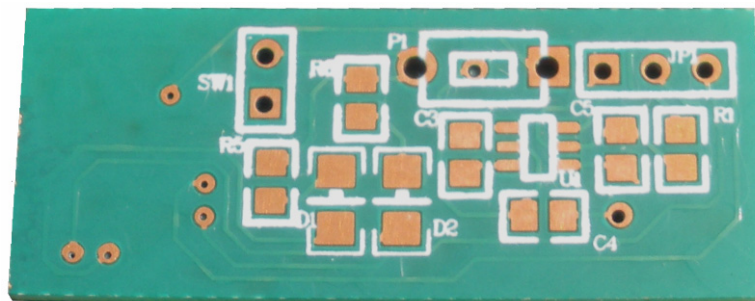


Figura 6.41. Capa *Top Layer* de la PCB fabricada del módulo sensor

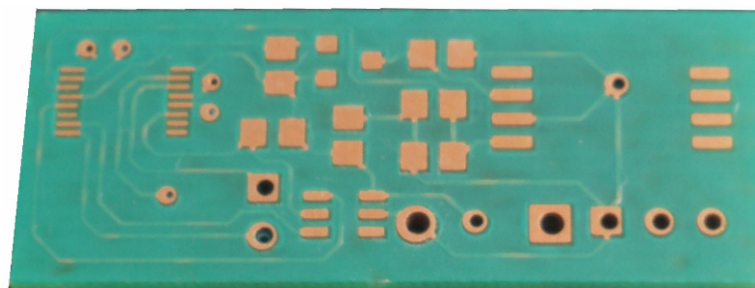


Figura 6.42. Capa *Bottom Layer* de la PCB fabricada del módulo sensor

6.5 Montaje de las PCBs

Con los circuitos impresos ya fabricados, y antes de comenzar a soldar los componentes que componen los módulos, se realizan una serie de verificaciones con el fin de asegurar que las placas de circuito impreso no presentan ningún defecto que impida su correcto funcionamiento. Estas verificaciones se centran en comprobar que no existan cortocircuitos entre secciones de los planos internos de masa y alimentación del módulo de comunicaciones inalámbricas, además de constatar que no hay cortocircuitos en las vías de las placas de los circuitos impresos fabricados.

Después de comprobar las verificaciones comentadas, se elabora una guía de fabricación, incluida en el Anexo C del presente documento, la cual estructura el orden en el que se deben soldar los componentes en la placa de circuito durante el proceso de soldadura, facilitando así el montaje de la PCB. Además, provee la orientación correcta en la que deben soldarse los componentes, evitando así errores en su funcionamiento una vez montados en la placa de circuito impreso.

La soldadura de los componentes se ha realizado con una estación soldadora con módulo de aire caliente. Debido al reducido tamaño de la mayoría de los componentes, las soldaduras se han efectuado situando el componente mediante unas pinzas y aplicando calor con la estación soldadora y una punta de soldador muy fina, habiendo aplicado *flux* con antelación, para facilitar la soldadura. Una vez finalizadas las soldaduras más pequeñas, se han revisado a través de una lupa electrónica para verificar mediante inspección visual que éstas se han realizado de forma eficiente.

En la Figura 6.43 y la Figura 6.44 se puede observar el aspecto que presenta la capa superior e inferior, respectivamente, de la placa de circuito impreso del módulo de comunicaciones inalámbrica, una vez finalizada.

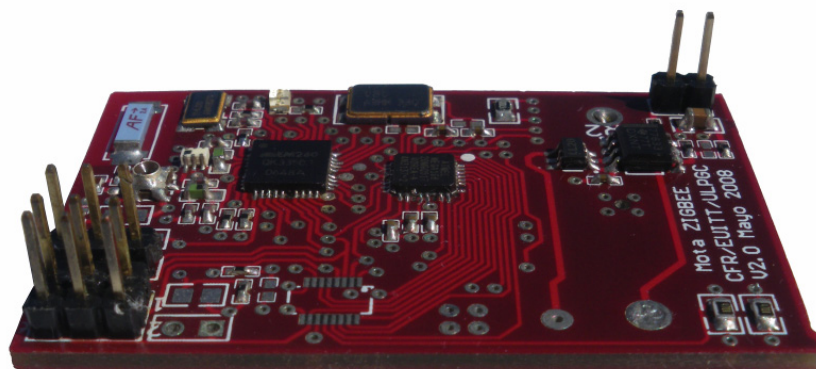


Figura 6.43. Aspecto de la cara superior de la PCB finalizada

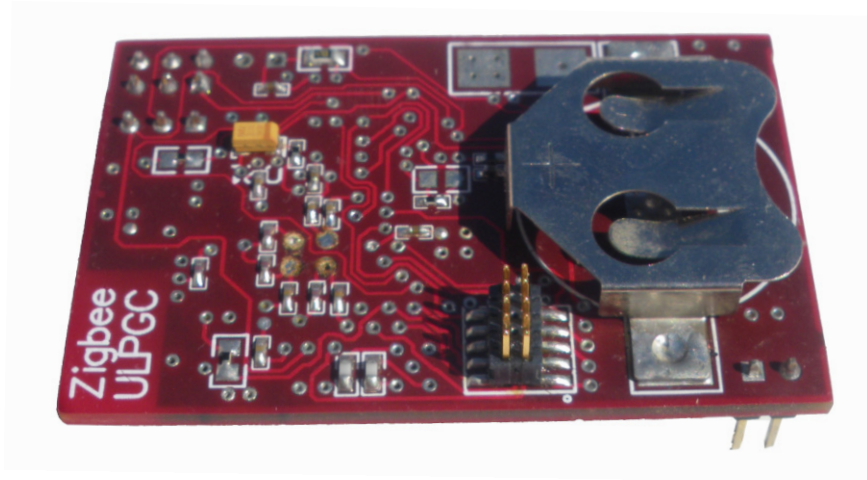


Figura 6.44. Aspecto de la cara inferior de la PCB finalizada

También se puede observar en la Figura 6.45 y la Figura 6.46 el aspecto que presenta la capa superior e inferior, respectivamente, de la placa de circuito impreso del módulo de comunicaciones sensor, finalizada.

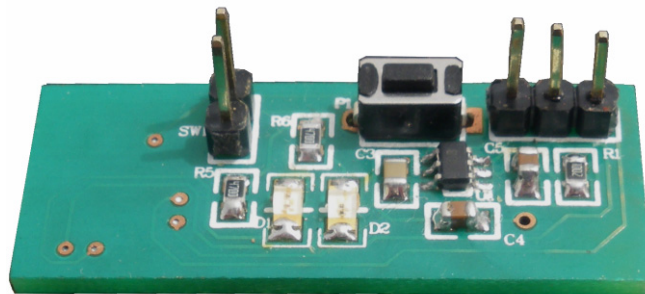


Figura 6.45. Aspecto de la cara superior de la PCB del módulo sensor finalizado

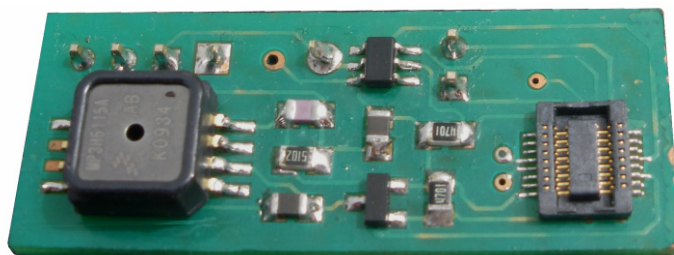


Figura 6.46. Aspecto de la cara inferior de la PCB del módulo sensor finalizado

Finalmente en la Figura 6.47 se muestran los circuitos impresos del módulo de comunicaciones inalámbricas y del módulo sensor totalmente terminados y conectados entre sí, dando lugar al nodo sensor inalámbrico desarrollado en el presente Proyecto Fin de Carrera.

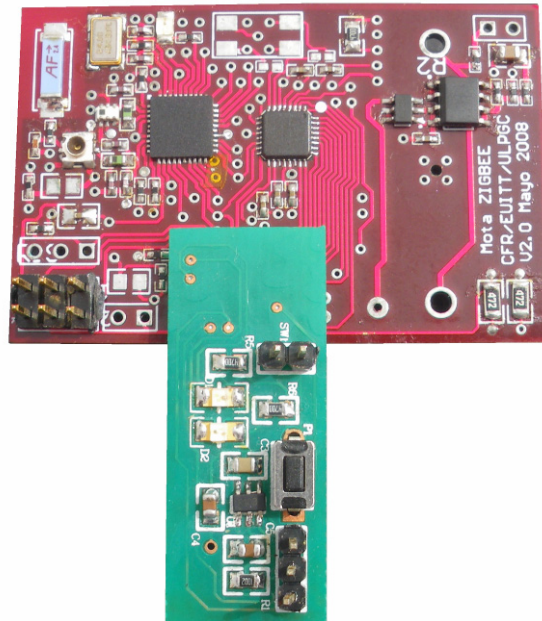


Figura 6.47. Nodo sensor inalámbrico terminado

Capítulo 7

Desarrollo del firmware

En este capítulo se realiza la descripción a nivel funcional del *firmware* desarrollado con el objetivo de verificar la funcionalidad del módulo sensor de comunicaciones inalámbrico implementado en el presente Proyecto Fin de Carrera.

7.1 Descripción general

Para llevar a cabo las pruebas funcionales del módulo sensor de comunicaciones inalámbricas, en este PFC se ha implementado un *firmware* que obtendrá los datos de los sensores incluidos en el módulo y realizará las funciones necesarias para agregarse a una red ZigBee que haya sido creada, enviando a continuación la información obtenida por los sensores al coordinador de la red ZigBee.

Como ya se ha comentado en los capítulos anteriores, el módulo diseñado dispone de un sensor de temperatura [54] y un sensor de presión [55], que conectados al módulo de comunicación inalámbrica es capaz de enviar los datos obtenidos por éstos. Para la verificación del funcionamiento del módulo implementado se ha empleado un módulo de comunicaciones ya existente, cuya funcionalidad ha sido comprobada con antelación [61]. Este módulo desempeñará la función de coordinador, encargándose de crear la red ZigBee y actuar como interfaz entre el PC y la pila ZigBee, permitiendo verificar los datos obtenidos y realizar la gestiones de la red ZigBee

con mayor facilidad. El módulo diseñado en el presente PFC actuará como nodo en la red ZigBee, en este caso concreto como dispositivo final (*End Device*), encargándose de enviar paquetes con información de forma periódica al coordinador de la red ZigBee. Para demostrar la funcionalidad del dispositivo diseñado actuando como *End Device*, se ha creado una aplicación que cada 4 segundos envía datos de temperatura y presión de la ubicación donde se encuentren los nodos, hacia el coordinador. Para lograr este objetivo, el nodo debe estar conectado a la red ZigBee creada por el dispositivo coordinador. La aplicación no sólo se encarga de transmitir los datos de los sensores, sino que debe gestionar la unión a la red ZigBee y mantenerlo conectado ante cualquier pérdida de la conexión, además de poder resolver e interpretar los posibles errores o estados que puedan producirse en el dispositivo EM260 y que necesiten respuesta para que continúe proporcionando un funcionamiento óptimo. En la aplicación se incluyen funciones adicionales, como la recepción e indicación de mensajes enviados por el coordinador, o la señalización de posibles errores en el módulo. En la Figura 7.1 se expone de forma gráfica la red ZigBee que la aplicación ha dispuesto para verificar la funcionalidad del dispositivo.

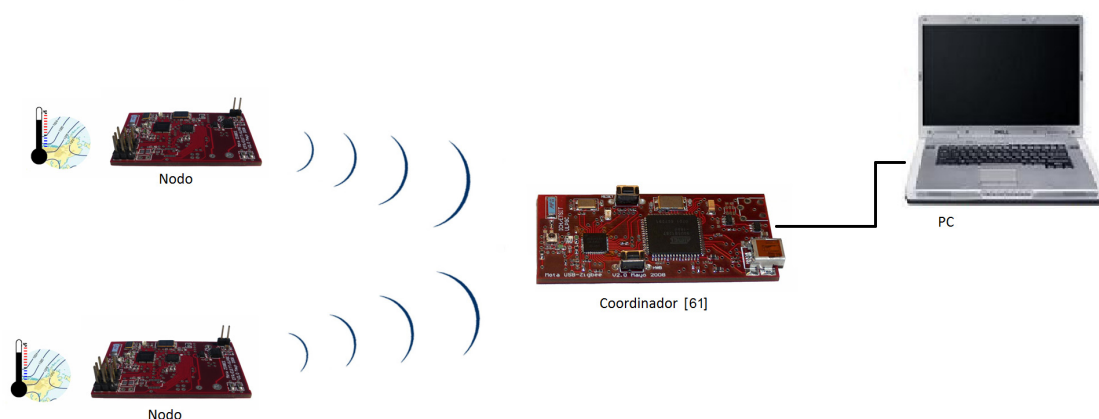


Figura 7.1. Red de nodos implementada en la aplicación

7.2 Programación de los dispositivos

En la Figura 7.5 se puede observar el diagrama de flujo del *firmware* implementado en el dispositivo ATmega168 del módulo de comunicaciones desarrollado en el presente PFC. El código del *firmware* se ha realizado en el lenguaje de programación C, siendo el entorno de programación utilizado *IAR Embedded Workbench Integrated Development Environment* (IDE) [62]. Este entorno de programación incluye un compilador y un depurador, así como un completo

administrador de proyectos. El compilador integrado destaca por una generación de código eficiente y optimizado para los dispositivos que soporta, como es el caso del ATmega168.

A continuación se detallan los estados que componen el flujograma de la Figura 7.5 describiendo así su funcionamiento:

- En el estado “Inicialización y Configuración” se lleva a cabo la inicialización del dispositivo ATmega168, estableciendo el valor inicial de las variables de entorno, las variables de índices, señales indicativas de estado, contadores de tiempo, y señales de reset. Además, se lleva a cabo la configuración de los registros del puerto SPI, los registros del convertidor analógico/digital, y las señales de todos los puertos de E/S. En la Figura 7.2, la Figura 7.3 y la Figura 7.4 se pueden observar las señales asignadas a cada puerto del dispositivo ATmega168. Este bloque se ejecuta una sola vez, al encender o inicializar el dispositivo ATmega168.

Bit	7	6	5	4	3	2	1	0	
	NULL	NULL	SCLK(EM260)/ SCK(TMP121)	MISO(EM260)/ SO(TMP121A)	MOSI (EM260)	nSSEL (EM260)	NULL	Enable (PIR)	PBn
In/Out			O	I	O	O		O	

Figura 7.2. Asignación de señales al puerto B del dispositivo ATmega168

Bit	7	6	5	4	3	2	1	0	
	-	RESET	Vout (MP3H6)	PMOS (neg.) (MP3H6)	NULL	nReset (EM260)	nWAKE (EM260)	nHOST_INT (EM260)	PCn
In/Out		I	I	O		O	O	I	

Figura 7.3. Asignación de señales al puerto C del dispositivo ATmega168

Bit	7	6	5	4	3	2	1	0	
	NULL	LED1	LED2	CS (neg.) (TMP121)	Pir (PIR)	SW1	NULL	NULL	PDn
In/Out		O	O	O	I	I			

Figura 7.4. Asignación de señales al puerto D del dispositivo ATmega168

- En el estado “Conectar a la Red ZigBee” se realiza la unión a la red ZigBee creada por el coordinador, mientras que en el estado “¿Conectado a la Red?” se verifica que el módulo

se encuentra conectado a la red. En el caso de no estar conectado a la red, vuelve a intentar unirse, mientras que en caso contrario, continúa con el siguiente estado.

- En el estado “Activar contador (4 seg.)” se pone en funcionamiento el *Timer1*, el cual controla los 4 segundos que debe mantenerse haciendo *polling* en el bucle que le precede. En el estado “Capturar el estado del PIR” se comprueba que la señal de salida del PIR, asociado al pin PD3, esté o no, activada. En caso de que se active, quedará almacenado para, posteriormente, en el estado “¿El contador ha llegado a los 4 seg. o el PIR está activado?” sea evaluado y salga del bucle. En el estado “Gestionar el estado del EM260” se comprueba el pin PC0, asociada a la señal nHOST_INT del dispositivo EM260. En el caso de que se produzca un cambio en la señal nHOST_INT, el dispositivo ATmega168 interpretará el comando EZSP que envíe el dispositivo EM260 y realizará las funciones oportunas.
- En el estado “¿El contador ha llegado a los 4 seg. o el PIR está activado?” se comprueba si el contador ha alcanzado los 4 segundos para salir del *polling* y comunicar al coordinador los datos de temperatura y presión. En el caso de que la señal de salida del PIR se active, o se produzca una desconexión con la red ZigBee, también se saldrá del bucle aunque no se hayan alcanzado los 4 segundos.
- En los estados “Leer Temperatura” y “Leer Presión” se obtiene la información de los sensores de temperatura y de presión, mediante el puerto SPI y el puerto del convertidor A/D, respectivamente.
- En el estado “Transmitir Temperatura, Presión y estado del PIR al coordinador” se realiza la transmisión de los datos de las medidas obtenidas por los sensores al coordinador de la red. Una vez transmitidos, se vuelve al bucle del *polling*, ejecutando nuevamente todo el flujo, pero comprobando con antelación si el módulo sigue conectado a la red ZigBee.

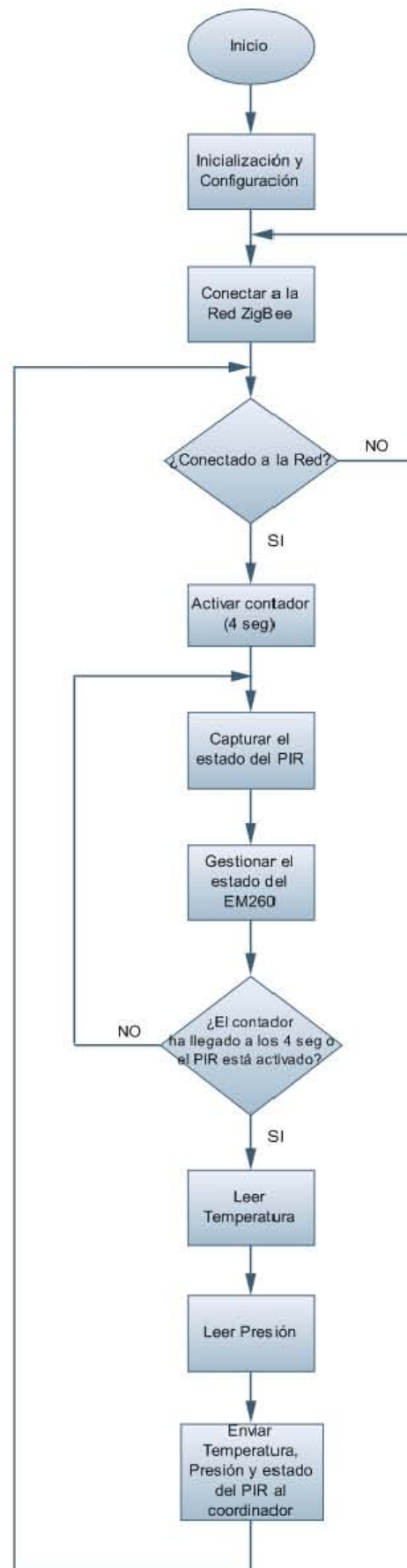


Figura 7.5. Flujograma de la aplicación desarrollada

Para finalizar, con el fin de conocer el estado del dispositivo en cada momento, se han implementado una serie de funciones para la señalización mediante el uso de diodos LEDs, que indicarán en qué estado se encuentra el dispositivo, avisando si se ha producido algún error o si ha realizado alguna de las funciones más importantes del *firmware*. En la Tabla 7.1. Señalización de los diodos LEDs del *firmware* desarrollado

Señalización	Causa
LED verde encendido.	El módulo está conectado a la red ZigBee
LED verde y rojo encendidos.	El módulo está obteniendo las medidas de temperatura y presión de los sensores
LED verde parpadea 2 veces.	La transmisión de los datos obtenidos por los sensores ha llegado satisfactoriamente al coordinador.
LED rojo y verde parpadean 3 veces (sincronizados).	El módulo ha recibido un mensaje y se dispone a mostrarlo mediante los LEDs.
LED rojo parpadea 1 vez.	Se ha producido un error. No se encontró terminador de la trama recibida, o trama recibida muy larga.
LED rojo y verde parpadean 1 vez (sincronizados).	Se ha producido un error durante el reset del EM260.
LED rojo parpadea 2 veces.	Se ha producido un error de <i>oversized</i> (trama muy larga) en el envío de la trama al EM260.
LED rojo parpadea 3 veces.	Se ha producido un error de transacción abortada por el EM260.
LED rojo parpadea 4 veces.	Se ha producido un error en el envío de la trama al EM260. No se encontró terminador (0xA7).
LED rojo parpadea 5 veces.	Se ha producido un error en el envío de la trama al EM260. El comando SPI no es soportado, la trama contiene un byte SPI no soportado.
LED rojo y verde parpadean 1 vez (no sincronizados).	Se ha producido un error desconocido.
LED rojo y verde parpadean 2 veces (no sincronizados).	Se ha producido un error en el envío de la trama al EM260. El comando EZSP enviado no es válido.
LED rojo parpadea 6 veces.	Se ha producido un error. La versión del EZSP empleada no es soportada por el EM260.
LED rojo encendido y LED verde parpadea 1 vez.	Se ha producido un error al intentar ejecutar el comando EZSP enviado.
LED rojo encendido y LED verde parpadea 2 veces.	Se ha producido un error debido a que no existe red ZigBee.
LED rojo y verde parpadean 3 veces (no sincronizados)	Se ha producido un error en la trama recibida del EM260. El comando EZSP recibido no es conocido.
LED rojo encendido y LED verde parpadea 3 veces.	Se ha producido un error en la transmisión de los datos obtenidos por los sensores al coordinador. No han llegado correctamente.

Tabla 7.1. Señalización de los diodos LEDs del firmware desarrollado

7.3 Medida del consumo de corriente

Uno de los parámetros más relevantes cuando se especifican las características de cualquier dispositivo electrónico es su consumo de potencia, y más en el caso de dispositivos portátiles donde la energía proporcionada por las baterías o pilas es limitada, y la vida útil de éstas depende directamente de su consumo.

Para determinar los requerimientos en cuanto a consumo del nodo sensor inalámbrico implementado en el presente Proyecto Fin de Carrera, éste se ha alimentado mediante una fuente de alimentación, que además de permitir ajustar la alimentación necesaria, indica la corriente que proporciona al dispositivo conectado. De esta manera es posible medir de manera simple la corriente total consumida por el nodo sensor inalámbrico bajo diferentes modos de operación, como son el modo de reposo, el modo de transmisión y el modo lectura de sensores.

La medida se ha realizado empleando un ordenador portátil, una fuente de alimentación modelo HM8142 fabricada por Hameg, un modulo de comunicaciones inalámbrico ya implementado con conexión USB [61], que actúa como coordinador y está conectado al ordenador portátil mediante puerto USB, y el nodo sensor inalámbrico desarrollado en el presente Proyecto Fin de Carrera, conectado a la fuente de alimentación antes indicada. De esta forma será posible ejecutar la aplicación desarrollada y determinar la corriente consumida en cada uno de los modos de operación. En la Figura 7.6 se pueden observar las condiciones en las que se ha realizado la medida del consumo para el dispositivo implementado.

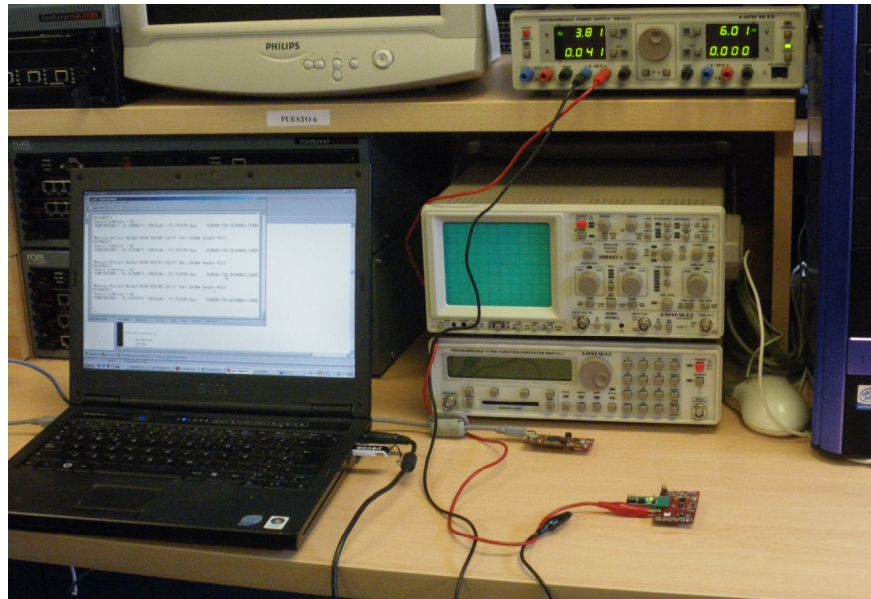


Figura 7.6. Medida de consumo del nodo sensor inalámbrico

En la Tabla 7.2 se indican los consumos de corriente obtenidos en los distintos modos de operación para el circuito implementado en el presente Proyecto Fin de Carrera, proporcionados por la fuente de alimentación, como se puede observar en la Figura 7.7, la Figura 7.8, y la Figura 7.9, para cada caso.

Modo de Operación	Consumo
Reposo	38 mA.
Transmisión	41 mA.
Lectura sensores	45 mA.

Tabla 7.2. Consumo del nodo sensor inalámbrico



Figura 7.7. Consumo del nodo sensor inalámbrico en modo reposo

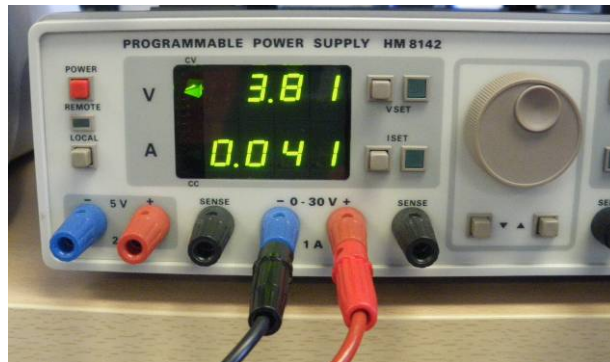


Figura 7.8. Consumo del nodo sensor inalámbrico en modo transmisión

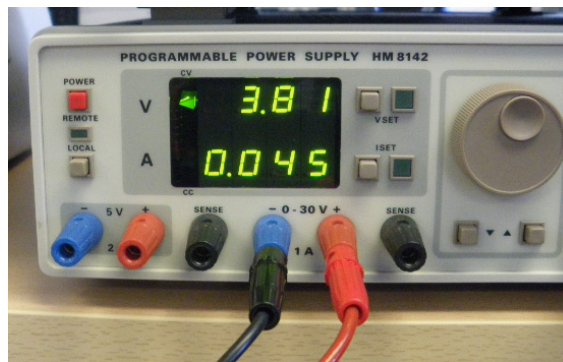


Figura 7.9. Consumo del nodo sensor inalámbrico en modo lectura sensores

Los resultados presentados en la Tabla 7.3 se han obtenido sin optimizar ni reducir las prestaciones de los componentes del nodo sensor inalámbrico. De esta manera, no se ha empleado ninguno de los modos de bajo consumo disponibles en el dispositivo EM260 o en el ATmega168, además, en el microcontrolador ATmega168 se ha usado una frecuencia de trabajo de 8Mhz, muy superior a la necesaria en una aplicación como la desarrollada en el presente Proyecto Fin de Carrera.

Con el fin de reducir el consumo de corriente, especialmente en modo reposo, donde los resultados obtenidos no resultan óptimos para un nodo sensor inalámbrico de bajo consumo, se han realizado varias modificaciones en el firmware del dispositivo. De esta manera los aspectos modificados en el presente PFC con el fin de lograr una disminución en el consumo de corriente, son los siguientes:

- Reducir la frecuencia de trabajo del microcontrolador ATmega168.
- Emplear el modo de bajo consumo IDLE en el microcontrolador ATmega168, durante el modo de operación Reposo.

- Emplear el modo de bajo consumo *Power down* en el co-procesador EM260 durante el modo de operación Reposo.
- Desactivar las indicaciones de los diodos LEDs.
- Desactivar el modo boost de transmisión y recepción del dispositivo EM260.

Finalmente, utilizando una frecuencia de trabajo de 128KHz, y empleando los modos de bajo consumo comentados además de desactivar el modo boost y los diodos LEDs, se han obtenido los resultados que se pueden observar en la Tabla 7.3, mucho más acordes con lo que se espera de un nodo sensor inalámbrico de bajo consumo.

Modo de Operación	Consumo
Reposo	430 μ A.
Transmisión	32 mA.
Lectura sensores	8 mA.

Tabla 7.3. Consumo reducido del nodo sensor inalámbrico

En la Tabla 7.3 se puede observar que los consumos de corriente han disminuido en todos los modos de operación, aunque en el modo de operación Transmisión la disminución no es significativa, en los modos de operación Reposo y Lectura de sensores, si es apreciable, sobre todo en el modo de operación Reposo, donde se ha pasado de un consumo de 38mA a 430 μ A, lo que representa un consumo de corriente casi 100 veces menor.

Aunque existen más soluciones que se pueden adoptar para continuar reduciendo el consumo de corriente del nodo sensor inalámbrico, éstas necesitan un estudio más exhaustivo de los componentes y del diseño hardware, que no se ha abordado en el presente PFC.

7.4 Test de alcance

Otro aspecto importante en cualquier dispositivo inalámbrico es la máxima distancia a la que éste es capaz, tanto de transmitir como de recibir datos sin perder la conexión. Por ello se han realizado una serie de pruebas de alcance bajo determinadas condiciones conocidas, permitiendo delimitar la distancia aproximada que es capaz de cubrir el nodo implementado en el presente PFC. El test de alcance de la aplicación desarrollada se ha realizado empleando un ordenador portátil, el módulo de comunicaciones inalámbrico ya implementado con conexión USB [61] y conectado al

ordenador portátil, que actúa como coordinador, además del nodo sensor inalámbrico implementado en el presente Proyecto Fin de Carrera. A continuación se detalla la configuración con la que se ha realizado el test de alcance del dispositivo coordinador y el nodo sensor inalámbrico:

- *Coordinador*: Configurado como coordinador de la red con una potencia de salida fijada a +3dBm, correspondiente con el modo *Boost* de funcionamiento. El dispositivo emplea la antena integrada en el mismo.
- *Nodo*: Configurado como dispositivo final de la red con una potencia de salida fijada a +3dBm, correspondiente con el modo *Boost* de funcionamiento. Se han empleado dos configuraciones diferentes, una primera configuración empleando la antena integrada que posee, y una segunda configuración haciendo uso de una antena externa del fabricante Pulse con una ganancia de 5dBi y patrón de radiación omnidireccional [70], conectado al nodo sensor inalámbrico mediante el conector coaxial tipo SWD que incluye. En la Figura 7.10 se muestra una fotografía de la antena externa empleada en el test de alcance realizado.



Figura 7.10. Antena omnidireccional externa auxiliar

Debido a que el alcance de cualquier dispositivo inalámbrico depende de multitud de factores (situaciones atmosféricas, obstáculos, etc.), se han realizado tests en diferentes escenarios con el fin de determinar un alcance aproximado del nodo desarrollado en el presente PFC. A continuación se realiza una descripción de los escenarios en los que se han realizado los tests empleando, o no, la antena externa.

7.4.1 Escenario A

En este caso, se ha realizado la prueba de alcance en condiciones de espacio abierto y visión directa. En la Figura 7.11 se puede observar una imagen que muestra el escenario y la distancia alcanzada entre el nodo sensor inalámbrico y el módulo coordinador sin emplear antena externa. En estas condiciones se ha obtenido un alcance de 21 metros y la potencia recibida por el módulo coordinador ha sido de -94dBm.

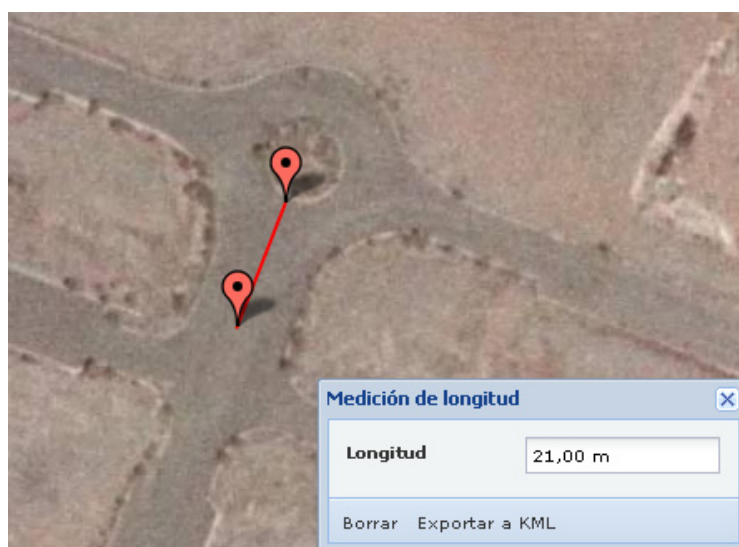


Figura 7.11. Alcance máximo en espacio abierto sin emplear antena externa

En las mismas condiciones, pero empleándose una antena externa en el nodo sensor inalámbrico, se logra aumentar el alcance, obteniéndose un valor de 43 metros. La potencia recibida por el receptor en estas condiciones empleando la antena externa ha sido de -93dBm. En la Figura 7.12 se puede observar una imagen que muestra el alcance máximo obtenido en espacio abierto con visión directa empleando la antena externa.



Figura 7.12. Alcance máximo en espacio abierto empleando antena externa

7.4.2 Escenario B

Con el fin de determinar el rendimiento del nodo sensor inalámbrico en entornos cerrados, se ha seleccionado como escenario los pasillos del Pabellón A del Edificio de Electrónica y Telecomunicación de la ULPGC. La prueba correspondiente se ha realizado en condiciones de visión directa y sin antena externa, obteniendo un alcance de 34 metros, como se puede observar en la Figura 7.13. La potencia recibida por el módulo coordinador en las condiciones descritas ha sido de -89dBm.



Figura 7.13. Alcance obtenido en el interior del Pabellón A sin antena externa

Por otro lado, en una segunda prueba en el mismo escenario (escenario B), pero empleando antena externa y sin visión directa se ha obtenido un alcance de 74 metros, como se puede observar en la Figura 7.14. La potencia recibida por el módulo coordinador en estas condiciones ha sido de -93dBm.

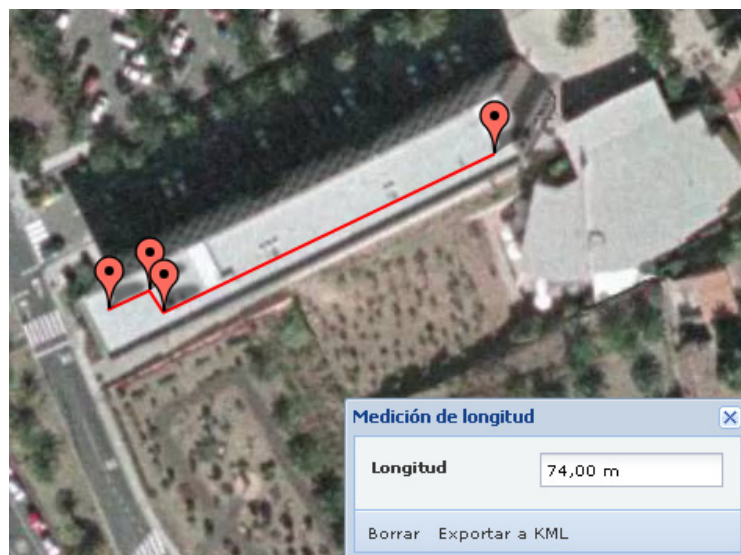


Figura 7.14. Alcance obtenido en el interior del Pabellón A con antena externa

7.4.3 Escenario C

Para el tercer escenario se ha elegido el pasillo del Pabellón C del Edificio de Electrónica y Telecomunicación de la ULPGC, tal como ocurrió con el escenario B, también se va a determinar el rendimiento del nodo sensor inalámbrico en entornos cerrados. En la Figura 7.15 se puede observar una imagen que muestra el alcance máximo obtenido en el interior del Pabellón C, sin emplear antena externa en el nodo sensor inalámbrico, obteniendo un alcance de 27 metros. La potencia recibida por el módulo coordinador en estas condiciones ha sido de -92dBm.



Figura 7.15. Alcance obtenido en el interior del Pabellón C sin antena externa

En este escenario, pero empleando una antena externa, se ha obtenido un alcance de 53 metros, como se puede observar en la imagen de la Figura 7.16. La potencia recibida por el módulo coordinador en estas condiciones ha sido de -95dBm.

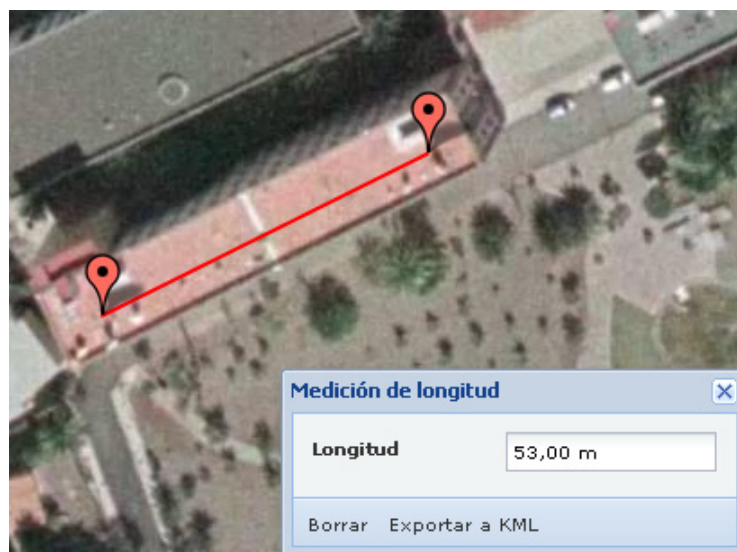


Figura 7.16. Alcance obtenido en el interior del Pabellón C con antena externa

En la Tabla 7.4 se muestra un resumen del alcance máximo desde el nodo sensor inalámbrico al módulo coordinador en los tres escenarios descritos (A, B y C), además de la potencia recibida por el coordinador, todo ello bajo las condiciones que se especifican en la tabla.

Condiciones	Antena externa	Potencia recibida	Distancia máxima
Espacio abierto y visión directa.	No	-94dBm	21 m.
Espacio abierto y visión directa.	Sí	-93dBm	43 m.
Pasillos del Pabellón A y visión directa.	No	-89dBm	34 m.
Pasillos del Pabellón A sin visión directa.	Sí	-93dBm	74 m.
Pasillos del Pabellón C y visión directa.	No	-92dBm	27 m.
Pasillos del Pabellón C y visión directa.	Sí	-94dBm	53 m.

Tabla 7.4. Alcance máximo del nodo sensor inalámbrico

Capítulo 8

Conclusiones y líneas futuras

En el presente capítulo se expondrá de forma breve las conclusiones a las que se ha llegado tras la realización de este Proyecto Fin de Carrera. Además, se planteará una serie de posibilidades en cuanto a la ampliación del mismo, que permita continuar su desarrollo.

8.1 Conclusiones

Una vez completada la implementación del nodo sensor inalámbrico y verificado éste, es capaz de establecer una correcta comunicación entre dispositivos a través del *firmware* desarrollado, pudiendo enviar y recibir datos con un bajo consumo de energía y un coste poco significativo, se puede afirmar que se han satisfecho los objetivos impuestos al comienzo del presente Proyecto Fin de Carrera. Además de su bajo coste y su bajo consumo de energía, el dispositivo implementado posee un reducido tamaño (34.5mm x 52.7mm), haciendo posible su empleo en un amplio espectro de aplicaciones, lo cual lo convierte en una solución idónea en aplicaciones sin gran complejidad, donde su sencillez y su facilidad en la gestión de redes evita la necesidad de desplegar redes de gran capacidad para aplicaciones que no lo requieran.

Además de lo comentado, el Proyecto Fin de Carrera desarrollado ha permitido realizar una evaluación general de las posibilidades del estándar ZigBee mediante el empleo del dispositivo EM260. A partir del estudio realizado sobre el estándar de comunicaciones ZigBee, se deduce un estándar ideal para las redes que necesiten una baja transferencia de datos y unos consumos muy reducidos, permitiendo diferentes topologías de redes y tolerando un elevado número de nodos. Desde el punto de vista de las aplicaciones, estas características lo convierten en un estándar muy flexible donde la adaptación a cualquier tipo de aplicación que no requiera una gran tasa de transferencia indica la excelente integración que puede lograr en cualquier entorno. En concreto, en el dispositivo empleado para implementar el estándar ZigBee en el nodo sensor inalámbrico desarrollado, el co-procesador EM260, se puede destacar que incluye una pila de comandos muy completa, proporcionando funciones muy interesantes para el envío de mensajes, la reducción de consumo o la gestión de redes. También es destacable su robustez y fiabilidad, ya que no ha ocasionado ningún problema de mención durante las comprobaciones y la aplicación desarrollada. No obstante, aunque en líneas generales el funcionamiento del dispositivo EM260 ha sido correcto, hay algunos aspectos que no resultan tan positivos, como la lentitud en el manejo de los procesos de conexión y en el envío de los mensajes, o el moderado consumo que presenta a pleno rendimiento, así como la necesidad de incluir en el diseño un hardware específico de programación, restando potencialidad y sencillez a las excelentes prestaciones de las que dispone el dispositivo, lo que complica su uso para aplicaciones simples.

En lo referente al diseño del nodo sensor inalámbrico, algunas regiones han precisado de mayor consideración, como la zona de radiofrecuencia, donde se requiere una ubicación de componentes muy próxima, dándole prioridad respecto a otras zonas. También es relevante señalar la eliminación de los planos de masa y alimentación en la regiones donde se encuentra ubicada la antena, además de situarla en un extremo de la placa de circuito impreso, con el fin de maximizar sus prestaciones. Además de estas consideraciones, durante la implementación del nodo sensor inalámbrico, las tareas de mayor dificultad han sido las relacionadas con la ubicación de los componentes y el ruteado de las pistas que se encuentran próximas a los dispositivos ATmega168 y EM260, debido al reducido área que se ha tratado de emplear y a la densidad de pads adyacentes a éstos. Igualmente complicada y laboriosa ha sido la etapa de soldar los componentes, ya que el reducido tamaño de algunos elementos de montaje superficial, como los de encapsulado 0402 (1mm x 0.5mm) o los dispositivos EM260 o ATmega168, la han dificultado sensiblemente.

Por otro lado, algunos resultados obtenidos por el nodo sensor inalámbrico implementado no han cumplido exactamente con las expectativas previstas inicialmente. De este modo, se preveía un consumo algo menor, ya que el parámetro obtenido en la aplicación del *firmware* desarrollada se

puede considerar como moderado, teniendo en cuenta que si el nodo se alimenta con una batería, es necesario un consumo de corriente máximo del orden de 15-20mA, para prolongar su vida útil. No obstante, el consumo de corriente medido en el nodo sensor inalámbrico se ha logrado reducir considerablemente empleando los modos de bajo consumo, aunque la reducción puede aumentarse si se realiza una optimización adecuada del software, ya sea empleando otros modos de bajo consumo soportados, tanto por el dispositivo ATmega168 como por el EM260, disminuyendo la frecuencia de trabajo del microcontrolador ATmega168 al mínimo necesario, manteniendo inactivo el transceptor cuando no se envíen mensajes, restringiendo el uso de los diodos LEDs para indicar situaciones críticas, etc. Respecto al alcance, el resultado obtenido tampoco ha sido el deseado; así, la cobertura obtenida en espacio abierto y con visión directa ha sido de unos 20 metros aproximadamente, lo que no representa un valor especialmente satisfactorio con lo que se estimaba al inicio del PFC, aunque con el uso de una antena externa se mejora considerablemente.

Se puede concluir indicando que el nodo sensor inalámbrico implementado en el presente Proyecto Fin Carrera ha demostrado poseer una funcionalidad completa, siendo verificado a través de un *firmware* desarrollado para una aplicación específica. El nodo desarrollado facilita su despliegue en diversas estructuras de red, debido a su flexibilidad, ofreciendo una elevada versatilidad gracias a la posibilidad de incorporar una gran variedad de sensores mediante el conector de expansión, convirtiéndose así en una solución práctica para múltiples aplicaciones.

8.2 Líneas Futuras

A medida que se ha ido desarrollando el presente Proyecto Fin de Carrera han ido surgiendo diferentes posibilidades de mejora que no han podido ser contempladas debido a que los esfuerzos requeridos para su consecución sobrepasan los estimados para la conclusión de este trabajo, proponiéndolos como líneas futuras. A continuación se expondrán las que se consideran más relevantes.

El *firmware* desarrollado en el presente Proyecto Fin de Carrera no contempla el uso de los diferentes niveles de seguridad y encriptación disponibles en el protocolo ZigBee, aspecto muy importante en este protocolo. De este modo, en un futuro podrían implantarse en los nodos sensores inalámbricos los sistemas de seguridad y encriptación que soporte el protocolo ZigBee, para posteriormente comprobar el nivel de seguridad que es posible alcanzar, así como la vulnerabilidad de ésta.

Por otro lado, como se comentó en el apartado anterior, es posible obtener una reducción mayor en el consumo de energía, para lo cual sería necesario un estudio más en profundidad de los modos de bajo consumo soportados por el microcontrolador ATmega168 y el co-procesador EM260 con el fin de emplearlos adecuadamente, y optimizar así el firmware desarrollado. Además de emplear los modos de bajo consumo, existen otros métodos de lograr la optimización del software, como puede ser reducir los datos a enviar (mediante codificación), ya que cuanto más reducido sea el volumen de datos enviados, menor es el tiempo necesario para su transmisión, además de ampliar los intervalos de envío de información, etc. También se debe analizar la mínima frecuencia de trabajo necesaria en el dispositivo ATmega168 para que la aplicación desarrollada funcione correctamente, evitando que éste consuma más energía mediante la generación de una frecuencia mayor que la estrictamente necesaria. Todas estas medidas y algunas más de hardware, como el estudio de los valores óptimos de las resistencias de pull-up para minimizar su consumo, pueden representar otra línea futura de trabajo interesante.

Finalmente, podría mejorarse el alcance del nodo sensor inalámbrico mediante la inclusión en el diseño de un amplificador de potencia. El dispositivo EM260 permite emplear un amplificador que proporcione un aumento de la ganancia del dispositivo. De esta manera se puede incrementar la cobertura varios cientos de metros, aumentando así las posibles aplicaciones del nodo, si bien se debe considerar el compromiso entre consumo y ganancia, que dependerá de la aplicación final del nodo implementado.

Bibliografía

- [1] Baker, W J. History of the Marconi Company 1894-1965. (1970, 1996).
- [2] “Wireless Communications”, Goldsmith, Andrea. History of Wireless Communications, pp. 1-4. 2005.
- [3] The Wi-Fi Alliance, <http://www.wi-fi.org/index.php>.
- [4] IEEE 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - IEEE 802.11 working group, <http://www.ieee802.org/11/>.
- [5] “IEEE 802.16 Task Group d”, <http://www.ieee802.org/16/tgd/>.
- [6] GSM Association (GSMA), <http://www.gsm.org/>.
- [7] IEEE Standard 521-2002: Standard Letter Designations for Radar-Frequency Bands
- [8] ISO - International Organization for Standardization, www.iso.org.
- [9] “Bluetooth Specification”, <http://www.bluetooth.com/Bluetooth/Default.htm>.
- [10] “ZigBee Specification”, The Zigbee Alliance, Diciembre de 2007. Documento 053474r17ZB. Disponible en www.ZigBee.org.
- [11] “The RF in RFID : passive UHF RFID in practice”, Daniel M. Dobkin. History and

- Practice of RFID, pp. 7-49. 2007.
- [12] IEEE 802.3 Standard, <http://www.ieee802.org/3/>.
- [13] “A Survey on Sensor Networks”, Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. IEEE Communications Magazine, Vol. 40, N°. 8, August 2002.
- [14] “Redes de sensores inalámbricos”, Francisco Ramos Pascual. Disponible en <http://www.radioptica.com/Radio/wsn.asp>.
- [15] “Instrumenting the World with Wireless Sensor Networks”, D. Estrin, L. Girod, G. Pottie, and M. Srivastava, Proc. Int’l Conf. Acoustics, Speech, and Signal Processing (ICASSP 2001), May 2001.
- [16] “Wireless Integrated Network Sensors”, G.J. Pottie and W.J. Kaiser, Communications of the ACM, vol. 43, no. 5, pp. 51-58, May 2000.J.
- [17] Sound Surveillance System (SOSUS). Disponible en <http://www.globalsecurity.org/intell/systems/sosus.htm>.
- [18] “Monitoring the global ocean through underwater acoustics”. Disponible en <http://www.pmel.noaa.gov/vents/acoustics/sosus.html>.
- [19] “Information processing and routing in wireless sensor networks”, Yang Yu, Viktor K. Prasanna, Bhaskar Krishnamachari, V. K. Prasanna Kumar. pp. 5-10.
- [20] North American Aerospace Defense Command (NORAD). Disponible en <http://www.norad.mil/>.
- [21] Defense Advanced Research Projects Agency. Disponible en <http://www.darpa.mil/>.
- [22] A History of the ARPANET: The First Decade (Bolt, Beranek and Newman, 1981)
- [23] Carnegie Mellon University. Disponible en <http://www.cmu.edu/>.
- [24] Massachusetts Institute of Technology. Disponible en <http://web.mit.edu/>.

- [25] IEEE 802.15 WPAN™ Task Group 4 (TG4), <http://ieee802.org/15/pub/TG4.html>.
- [26] “HART Specification”, HART Communication Foundation. Disponible en <http://www.hartcomm2.org/index.html>.
- [27] ISA100, Wireless Systems for Automation . Disponible en www.isa.org/isa100/.
- [28] 6LOWPAN (IPv6 Over Low power WPAN). Disponible en <http://www.6lowpan.org/1.html>.
- [29] “Co-existence of Zigbee and WLAN - a performance study”, Shuaib, K. Boulmalf, M. Sallabi, F. Lakas, A. , Proceedings of the WOCN, Abril del 2006.
- [30] “Cluster Label-based ZigBee Routing Protocol with High Scalability”, Lee, Kwang Koog; Kim, Seong Hoon; Park, Hong Seong; Second International Conference on Systems and Networks Communications, 2007. ICSNC 2007. 25-31 Aug. 2007.
- [31] “A Mesh Routing Protocol using Cluster Label in the ZigBee Network”, Kwang Koog Lee, Seong Hoon Kim, Yong Soon Choi and Hong Seong Park. 2006 IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS). pp 801-806. Oct. 2006.
- [32] “Ad hoc On-Demand Distance Vector (AODV) Routing”, Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir Das. IETF RFC 3561. Disponible en <http://tools.ietf.org/html/rfc3561>.
- [33] “Data communications, computer networks and open systems”, Fred Halsall, 1995.
- [34] “Spread Spectrum Systems”, Robert C. Dixon. John Wiley & Sons Inc; 2nd edition (August 8, 1984).
- [35] GPS - Global Positioning System <http://www.gps.gov/>.
- [36] “LMDS: Local Mutipoint Distribution Service”, Clint Smith, August 11, 2000.
- [37] “Wireless Sensor Networks: Technology, Protocol, and Applications”, Kazem Sohraby, Daniel Minoli, Taieb Znati. pp 26-28. 2007.
- [38] Documentación de diseño del procesador de red EM260. Disponible en la web

- http://www.ember.com/products_zigbee_chips_e260.html.
- [39] XAP2 ASIC Processor. Disponible en <http://www.cambridgeconsultants.com/downloads/asic/ASICs-SB-006.pdf>.
- [40] EmberZNet API Reference. Disponible en www.ember.com, documento 120-3019-000 rev D.
- [41] EmberZNet Serial Protocol Reference Guide. Disponible en www.ember.com, documento 120-3009-000 rev D.
- [42] Documentación de diseño del microcontrolador Atmel AT90USB1287. Disponible en la web www.atmel.com.
- [43] Hoja de características del dispositivo EM260 de Ember. Disponible en www.ember.com, documento EM260 ZigBee/802.15.4 Network Processor – 120-1003-000M.
- [44] Hoja de características del dispositivo ATmega168 de Atmel. Disponible en www.atmel.com, documento ATmega48/88/168.
- [45] Hoja de características del ZLLS2000TA. Disponible en <http://www.zetex.com/3.0/pdf/ZLLS2000.pdf>.
- [46] Hoja de características del TPS77633D. Disponible en <http://focus.ti.com/docs/prod/folders/print/tps77633.html>.
- [47] Hoja de características del HHM-1521. Disponible en http://www.tdk.co.jp/tefe02/e8balun_hhm.pdf.
- [48] Hoja de características del MM8430-2600RA1. Disponible en <http://www.murata.com/catalog/o30e8.pdf>.
- [49] Hoja de características de la antena en chip ANT-2.45-CHP-T. Disponible en http://www.antennafactor.com/documents/ANT-xxx-CHP-x_Data_Guide.pdf.
- [50] Hoja de características del ABM3B-24.000MHZ. Disponible en <http://www.abracon.com/Resonators/abm3b.pdf>.

- [51] Documentación del dispositivo de programación InSight USB Link. Disponible en la web http://www.ember.com/products_zigbee_development_tools_usb.html.
- [52] Hoja de características del ABMM-8.000MHZ. Disponible en <http://www.abracon.com/Resonators/ABMM.pdf>.
- [53] Documentación del dispositivo de programación AVRISP mkII In-System Programmer. Disponible en la web http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3808.
- [54] Hoja de características del TMP121. Disponible en <http://focus.ti.com/lit/ds/symlink/tmp121.pdf>.
- [55] Hoja de características del MP3H6115A6U. Disponible en http://www.freescale.com/files/sensors/doc/data_sheet/MP3H6115A.pdf.
- [56] Hoja de características del BSS84,215. Disponible en http://www.nxp.com/documents/data_sheet/BSS84.pdf.
- [57] Hoja de características del sensor PIR 555-28027. Disponible en <http://www.parallax.com/Portals/0/Downloads/docs/prod/audiovis/PIRSensor-V1.2.pdf>.
- [58] Hoja de características del convertidor REG710NA-5/250. Disponible en <http://focus.ti.com/lit/ds/symlink/reg71050.pdf>.
- [59] Software de diseño Altium Design eXPlorer. <http://www.altium.com>.
- [60] Fabricante PCB PCBCART. <http://www.pcbcart.com>.
- [61] “Desarrollo e implementación hardware de un módulo de comunicaciones inalámbrico de propósito general y bajo consumo”, Ignacio Del Castillo Hernández. Proyecto Fin de Carrera realizado para la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universidad de Las Palmas de Gran Canaria. 2009.
- [62] Software para el desarrollo y depuración de aplicaciones en dispositivos integrados IAR Embedded Workbench Integrated Development Environment. <http://www.iar.com>.
- [63] “Wireless Networks in industrial environments: State of the art and Issues”, Xavier Carcelle, Tuan Dang, Catherine Devic. International Federation for Information

- Processing (IFIP) Volume 212, Ad-Hoc Networking, ed. Al Agha, K., (Boston: Springer), pp. 141-156. 2006.
- [64] “Emerging standards for wireless mesh technology”, Lee, M.J.; Jianliang Zheng; Young-Bae Ko; Shrestha, D.M.; IEEE Wireless Communications Volume 13, No. 2, Abril 2006 pp:56 – 63.
- [65] “Will IEEE 802.15.4 make ubiquitous networking a reality?: a discussion on a potential low power, low bit rate standard” Jianliang Zheng; Lee, M.J.; IEEE Communication Magazine. Vol. 42, No. 6, 2004. pp 140-146.
- [66] “Commercial Applications of Wireless Sensor Networks Using ZigBee”, Wheeler, A.; IEEE Communications Magazine. Vol 45, No. 4. Abr. 2007. pp 70-77.
- [67] “Propuesta de baremos orientativos para el cálculo de honorarios” establecida por el Colegio Oficial de Ingenieros Técnicos de Telecomunicación a partir de 1-01-2006. Disponible en <http://www.coitt.es>.
- [68] Documentación de la alianza ZigBee, *ZigBee Alliance*. Disponible en la web www.zigbee.org.
- [69] Servicio de Fabricación de Prototipos de Circuitos Impresos y Sistemas Electrónicos del IUMA. <http://www.iuma.ulpgc.es/services/sfp/>.
- [70] Hoja de características de la antena S151AH-2450S. Disponible en <http://media.digikey.com/pdf/Data%20Sheets/Nearson%20PDFs/S151xx-2450S.pdf>

Presupuesto

Una vez completada la implementación del nodo sensor inalámbrico y comprobado su correcto funcionamiento, en los siguientes apartados se realizará un estudio económico con los costes, tanto parciales como totales, para concluir con el Proyecto Fin de Carrera.

Baremos utilizados

El cálculo del presupuesto de este proyecto se ha realizado según la “Propuesta de baremos orientativos para el cálculo de honorarios” establecida por el Colegio Oficial de Ingenieros Técnicos de Telecomunicación (COITT) a partir de 1-01-2006 [67]. Esta propuesta establece que para “Trabajos tarifados por tiempo empleado” se aplique:

$$H = H_n \cdot 65 + H_e \cdot 78 \quad ,$$

donde:

- H son los honorarios a percibir.
- H_n son las horas contabilizadas en jornada normal.
- H_e son las horas contabilizadas fuera de la jornada normal de trabajo.

No se han aplicado los baremos correspondientes al año 2010 pues los costes estimados de trabajos profesionales ya no son publicados por el COITT por indicación de las autoridades de la competencia del Ministerio de Economía y Hacienda.

Los honorarios que se obtengan por la aplicación de la clave “H” se reducirán a medida que aumente el número de horas, a cuyo efecto serán multiplicados por los coeficientes reductores, con arreglo a lo detallado en la Tabla P.1.

HORAS			COEFICIENTE
Hasta	36 horas		C = 1
Exceso de	36 horas	Hasta 72 horas	C = 0,9
Exceso de	72 horas	Hasta 108 horas	C = 0,8
Exceso de	108 horas	Hasta 144 horas	C = 0,7
Exceso de	144 horas	Hasta 180 horas	C = 0,65
Exceso de	180 horas	Hasta 360 horas	C = 0,60
Exceso de	360 horas	Hasta 510 horas	C = 0,55
Exceso de	510 horas	Hasta 720 horas	C = 0,50
Exceso de	720 horas	Hasta 1080 horas	C = 0,45
Exceso de	1080 horas		C = 0,40

Tabla P.1. Coeficientes reductores

Cálculo del presupuesto

P.1. Coste de debidos a los recursos humanos

En este apartado se incluyen los honorarios a percibir por el ingeniero técnico en el desarrollo del proyecto, en función de las horas de trabajo que ha empleado en la realización del mismo.

Para la realización del presente Proyecto Fin de Carrera se estima que se ha trabajado durante un año a tiempo completo, es decir, 52 semanas hábiles, trabajando 5 días semanales y 8 horas diarias, por lo que el número de horas totales trabajadas resulta:

$$H_n = 8 \text{ horas / día} \cdot 5 \text{ días / semana} \cdot 52 \text{ semanas} = 2.080 \text{ horas}$$

En definitiva, se necesitaron un total de 2.080 horas para la realización de este proyecto, consideradas en su totalidad del tipo de jornada normal, con lo que el cálculo “H” resulta:

$$H = 2.080 \cdot 65 = 135.200 \text{ €}$$

Aplicando los coeficientes correctivos, proporcionados por el COITT, a los tramos correspondientes, resultan unos honorarios de:

$$H = 135.200 \cdot 0,40 = 54.080 \text{ €}$$

Por tanto, el trabajo tarifado por tiempo empleado asciende a la cantidad de **cincuenta y cuatro mil ochenta euros (54.080,00 €)**.

P.2. Coste de amortización de los recursos hardware

A continuación se detallan, en la Tabla P.2, los costes relacionados con la utilización de equipos informáticos y equipamiento diverso de laboratorio, como una estación de soldadura con aire caliente, una lupa de gran aumento y un polímetro, empleados durante la elaboración de este proyecto.

Los costes están divididos entre el número de usuarios que accedan a éstos, puesto que la mayoría de estos recursos son compartidos entre varios usuarios. El cálculo del coste asociado estará en función del número de personas estimadas que lo utilicen y del período de amortización asumido para cada uno de los recursos.

Los periodos de amortización y el número de usuarios van a depender del recurso utilizado. Así, para el ordenador portátil se ha estimado un periodo de amortización de 18 meses, suponiéndose que será un solo usuario el que haga uso del recurso. Igualmente, la impresora se ha estimado con un periodo de amortización de 18 meses, si bien el número de usuarios estimado es de 30. En cuanto al material de laboratorio, se asume que tanto el polímetro como la estación de soldadura tienen un periodo de amortización de 24 meses, y que hacen uso de ellos 20 personas.

Concepto	Coste unitario	Amortización	Coste Mensual/Usuario	Tiempo	Coste
Ordenador	1.500,00 €	18 meses	83,34 €	12 meses	1.000,08 €
Impresora	300,00 €	18 meses	0,56 €	12 meses	6,72 €
Estación de soldadura	800,00 €	24 meses	1,67 €	12 meses	20,04 €
Lupa	100,00 €	24 meses	0,2 €	12 meses	2,40 €
TOTAL					1.029,24 €

Tabla P.2. Costes asociados a los recursos hardware

El coste total de amortización de los recursos hardware asciende a la cantidad de **mil veintinueve euros con veinticuatro céntimos (1.029,24 €)**.

P.3. Coste de amortización de los recursos software

El coste de los recursos software se obtiene a partir del valor de las licencias y el mantenimiento de cada uno de los programas utilizados. En la Tabla P.3 se resumen estos costes.

Concepto	Tiempo Empleado	Coste
Entorno de diseño Altium DXP 2004 con Service Pack 3 (Coste único)	7 meses	1.000,00 €
Entorno de diseño IAR AVR Embedded Workbench (Coste único)	3 meses	350,00 €
Microsoft Office XP (Coste único)	2 meses	653,00 €
TOTAL		2.003,00 €

Tabla P.3. Costes asociados a los recursos software

El coste total de amortización de los recursos software asciende a la cantidad de **dos mil tres euros (2.003,00 €)**.

P.4. Coste de fabricación de prototipos

El coste de fabricación de prototipos ha de ser contemplado en el presupuesto final, ya que éste forma parte del desarrollo del proyecto. El coste de fabricación contempla los diferentes procesos necesarios para obtener unos prototipos totalmente funcionales. En la Tabla P.4 se resumen estos costes para la fabricación correspondiente a 10 prototipos con el fabricante PCBCART [60] y el fabricante Servicio de Fabricación de Prototipos de Circuitos Impresos y Sistemas Electrónicos del IUMA [69].

Concepto	Coste Unitario	Coste
Fabricación PCB del módulo inalámbrico	10,00 €	100,00 €
Fabricación PCB del módulo sensor	3,00 €	30,00 €
Componentes para la fabricación	13,00 €	130,00 €
Montaje Prototipos	78,00 €	780,00 €
TOTAL		1.040,00 €

Tabla P.4. Costes asociados a la fabricación de prototipos

El coste total de la fabricación de 10 prototipos del nodo sensor inalámbrico asciende a la cantidad de **mil cuarenta euros (1.040,00 €)**.

P.5. Coste de la redacción del proyecto

Según los criterios establecidos por el COIT, los gastos generados para la redacción del Proyecto Fin de Carrera son calculados de acuerdo a la siguiente fórmula:

$$H = 0,07 \cdot P_T \cdot C + 0,03 \cdot P_C \cdot C \quad ,$$

donde:

- H son los honorarios a percibir por la redacción del proyecto.
- C es el coeficiente por tramos para el presupuesto.
- P_T es el presupuesto de ejecución del proyecto.
- P_C es el presupuesto de obra civil.

Con los costes obtenidos en las partidas anteriores se calcula el presupuesto de ejecución del proyecto. Por tanto el valor de P_T es:

$$P_T = 54.080,00 + 1.029,24 + 2.003,00 + 1.040,00 = 58.152,24 \text{ €}$$

El presente PFC no tiene asociada ninguna obra civil, por lo que el presupuesto de ésta es nulo.

El COITT establece un coeficiente reductor de valor $C = 0,9$ para el tramo entre 30.050 € y 60.101 €. Por tanto, el coste de redacción calculado es:

$$H = 0,07 \cdot 58.152,24 \cdot 0,9 = 3.663,59 \text{ €}$$

El coste total de la redacción del proyecto asciende a la cantidad de **tres mil seiscientos sesenta y tres euros con cincuenta y nueve céntimos (3.663,59 €)**.

P.6. Coste por derechos de visado

El coste de visar un proyecto en el COIT durante el año 2010 viene establecido por la siguiente fórmula:

$$V = 0,006 \cdot P_1 \cdot C_1 + 0,003 \cdot P_2 \cdot C_2 \quad ,$$

donde:

- V es el coste de visado del proyecto.
- P_1 es el presupuesto de ejecución material de sistemas de telecomunicación.

- P_2 es el presupuesto de ejecución material correspondiente a obra civil.
- C_1 es el coeficiente reductor por tramos para el presupuesto P1.
- C_2 es el coeficiente reductor por tramos para el presupuesto P2.

El presupuesto de ejecución material de sistemas de telecomunicación asciende a la cantidad acumulada en las partidas de costes anteriores. Por tanto el valor de P_1 es:

$$P_1 = 54.080,00 + 1.029,24 + 2.003,00 + 1.040,00 + 3.663,59 = 61.815,83 \text{ €}$$

No hay asignado presupuesto de ejecución material correspondiente a obra civil, por lo que $P_2 = 00,00 \text{ €}$. Por otro lado, el coeficiente reductor por tramos establecido por el COIT en el año 2010 para el tramo entre 60.101 € y 90.152 €, es $C_1 = 0,8$. Por tanto, el coste para visar el proyecto es:

$$V = 0,006 \cdot 61.815,83 \cdot 0,8 = 296,71 \text{ €}$$

Los costes de visado del presente PFC ascienden a la cantidad de **doscientos noventa y seis euros con setenta y un céntimos (296,71 €)**.

P.7. Coste de tramitación y envío

Son debidos a los costes generados por enviar los tomos a la sede del COIT para que sean visados. Para esta partida se aplica una tarifa fija de **nueve euros (9,00 €)**.

P.8. Gastos generales

En este apartado se incluyen los costes derivados del uso de material necesario para la elaboración del proyecto y del mantenimiento de los laboratorios, ascendiendo aproximadamente a un 5% sobre el total de los gastos anteriores.

P.9. Presupuesto total del proyecto

Para finalizar en la Tabla P.5 se recoge el coste total del presente PFC en función de los costes parciales comentados en las secciones anteriores.

Concepto		Importe
P.1.	Coste de Recursos Humanos	54.080,00 €
P.2.	Costes de Recursos Hardware	1.029,24 €
P.3.	Costes de Recursos Software	2.003,00 €
P.4.	Fabricación Prototipos	1.040,00 €
P.5.	Coste de la redacción del proyecto	3.663,59 €
P.6.	Costes por derechos de visado	296,71 €
P.7.	Coste de tramitación y envío	9,00 €
Subtotal		62.121,54 €
P.8.	Gastos Generales (5%)	3.106,07 €
Subtotal		65.227,61 €
IGIC (5%)		3.261,38 €
TOTAL		68.488,99 €

Tabla P.5. Presupuesto final del Proyecto Fin de Carrera

El presupuesto total del Proyecto Fin de Carrera asciende a un total de sesenta y ocho mil cuatrocientos ochenta y ocho euros con noventa y nueve céntimos (**68.488,99 €**).

D. Carlos M. Fernández Reyes declara que el proyecto “Implementación de un sensor inalámbrico de propósito general con bajo consumo” asciende a un total de sesenta y ocho mil cuatrocientos ochenta y ocho euros con noventa y nueve céntimos.

Fdo. Carlos M. Fernández Reyes

DNI: 42201374.

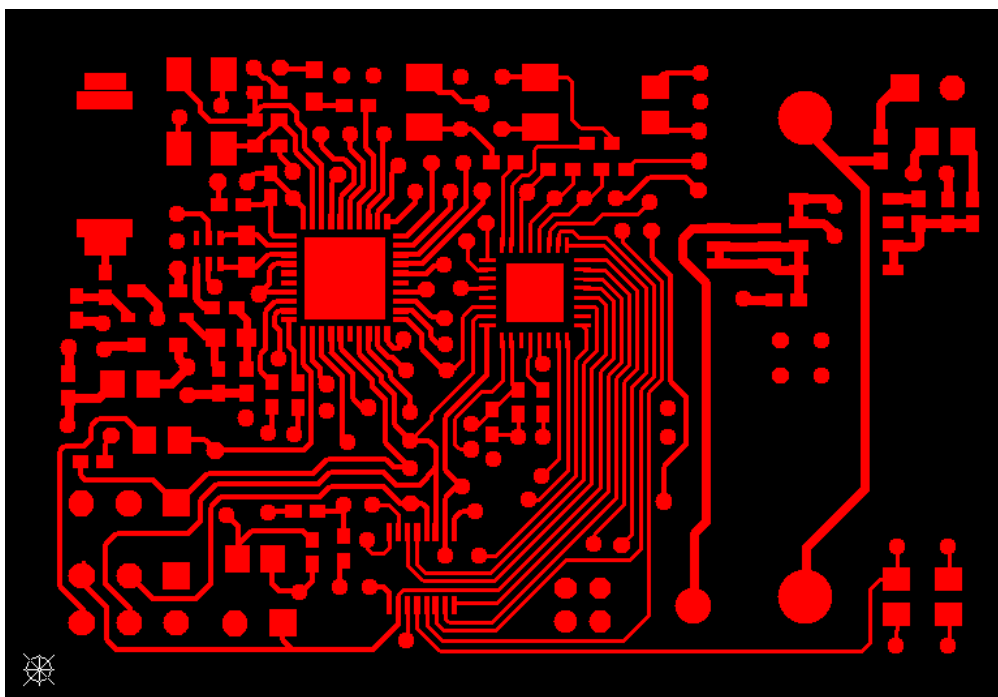
Las Palmas Mayo de 2010

Anexo A

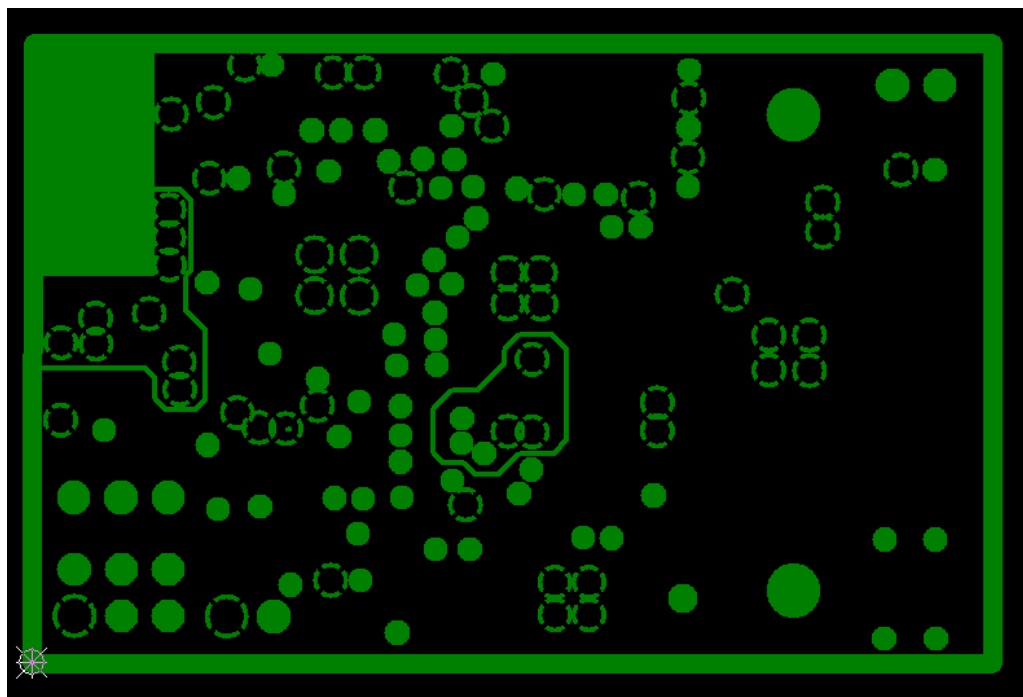
Ficheros Gerber

Módulo de comunicaciones inalámbricas

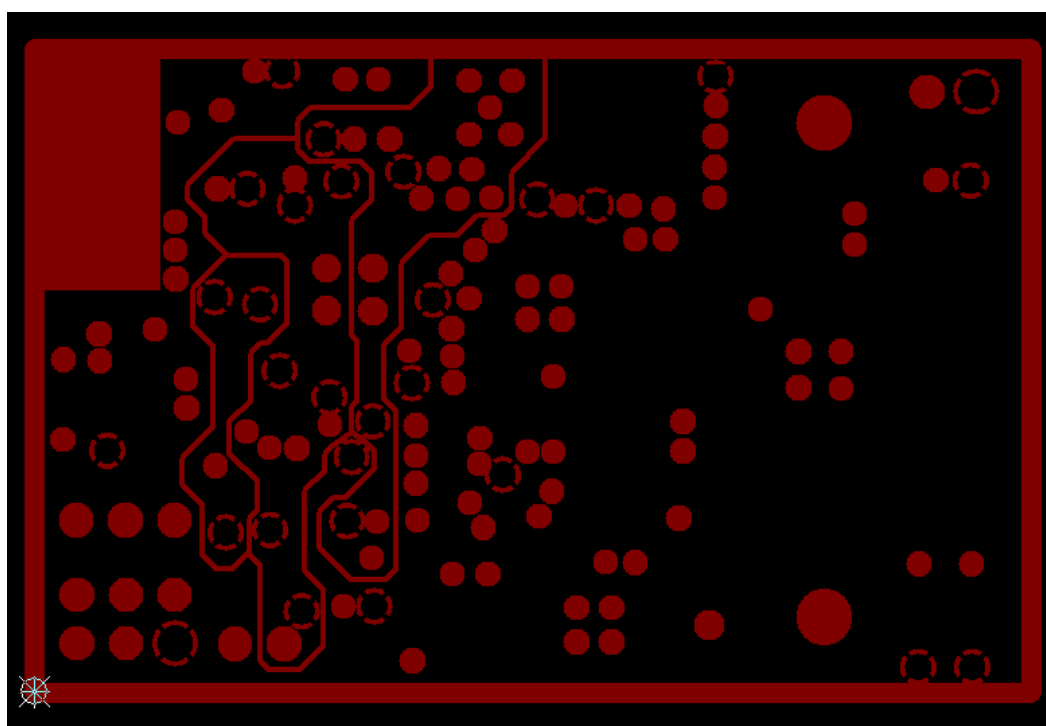
Top Layer



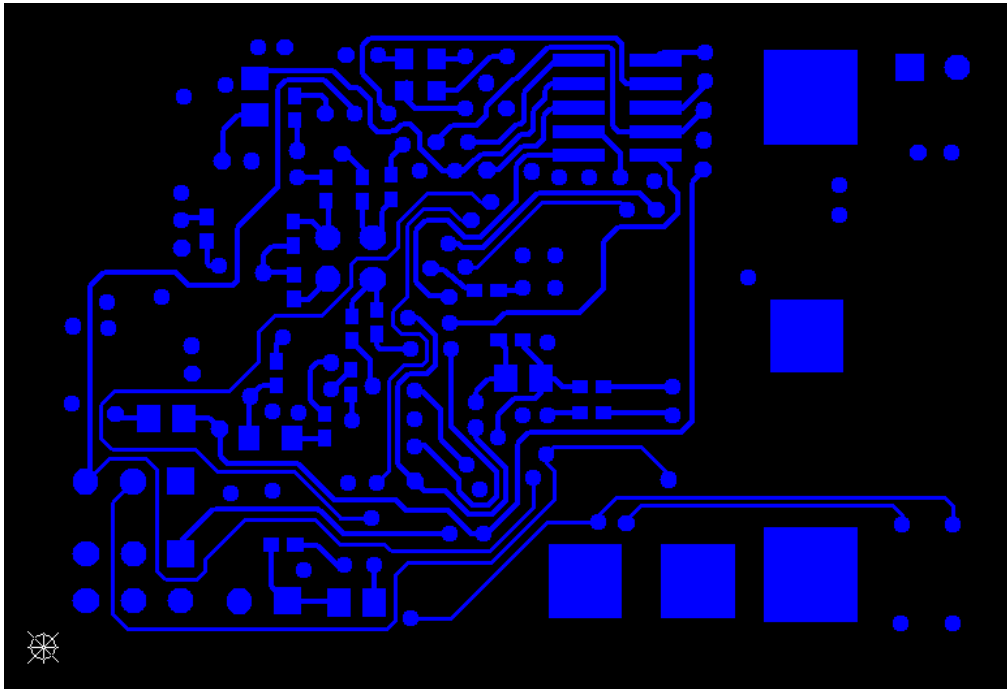
Interplane 1



Interplane 2

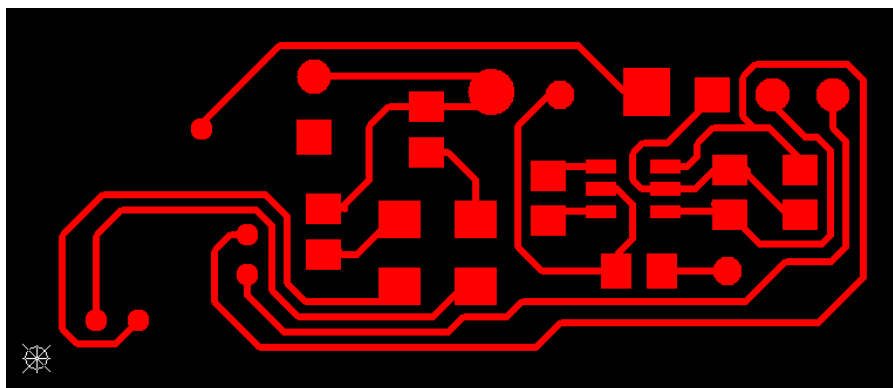


Bottom Layer

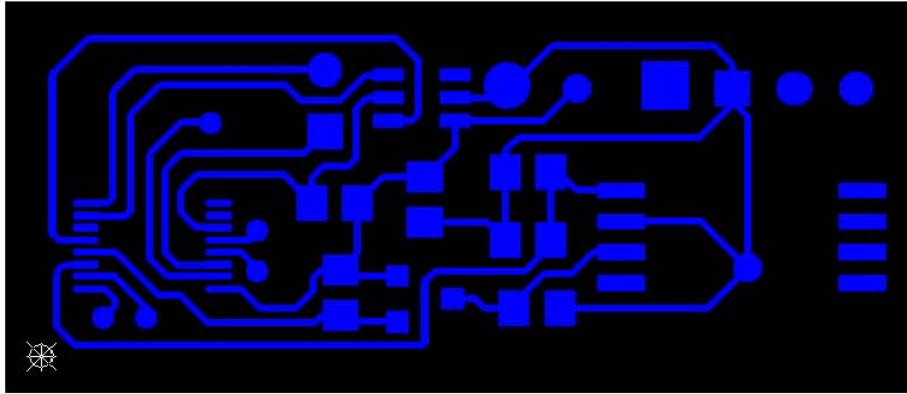


Módulo sensor

Top Layer



Bottom Layer



Anexo B

Listado de Componente

Módulo de comunicaciones inalámbricas

Cantidad	Nombre	Descripción del componente	Referencia del fabricante
1	ANT1	ANT 2.4GHZ 802.11 BLUETOOTH SMD	ANT-2.45-CHP
1	BLN1	XFRMR BALUN RF 2.4GHZ SMD	HHM1521
1	BTH1	HOLDER COIN CELL 2032 SMD	BK-883-TR
6	C1 C2 C15 C27 C270 271	CAP CER 8PF 50V C0G 0402	GRM1555C1H8R0DZ01D
2	C10 C14	CAP CER 27PF 50V 5% C0G 0402	GRM1555C1H270JZ01D
2	C19 C22	CAP CER 36PF 50V 5% C0G 0603	GRM1885C1H360JA01D
1	C3 C4	CAP CER .75PF 50V C0G 0402	GRM1555C1HR75CZ01D
1	C31	CAPACITOR TANT 10UF 10V 10% SMD	T491A106K010AT
10	C5 C7 C8 C16 C17 C20 C21 C23 C29 C30	CAP CER .1UF 16V X7R 0402	GRM155R71C104KA88D
7	C6 C11 C12 C13 C18 C24 C26	CAP CER 10000PF 25V 10% X7R 0402	GRM155R71E103KA01D
1	C9	CAP CERAMIC 22UF 6.3V X5R 0805	ECJ-2FB0J226M
1	D1	DIODE SCHOTTKY 40V 2.2A SOT23-6	ZLLS2000TA

Cantidad	Nombre	Descripción del componente	Referencia del fabricante
1	DS1	LED ALINGAP RED WHT DIFF 0603SMD	SML-LX0603SIW-TR
3	F1 F2 F3	SMD EMI Filters/Chip Ferrite Beads	BLM21AH102SN1D
1	J1	CONN MW COAXIAL WITH SWITCH	MM8430-2600RA1
1	J2	CONN PLUG 16POS 1.5MM SMD .5MM	55560-0168-C
1	J4	CONECTOR DEL AVRISP mkII In-System Programmer Header, 6-pin	
1	J5	Header, 3-Pin	
1	J6	HEADER, SHROUDED, 10 PIN, DUAL ROW, VERTICAL, 0.050"	FTSH-105-01-F-DV-K
2	JP1 JP2	Header, 2-Pin	
1	L1	INDUCTOR 3.9NH 450MA 0603	LQG18HN3N9S00D
1	L3	INDUCTOR 1.8NH 280MA 0402	LQP15MN1N8B02D
1	L4	INDUCTOR 3.3NH 500MA 0603	LQG18HN3N3S00D
10	R1 R2 R4 R16 R17 R19 R21 R22 R26 L2	RES ZERO OHM 1/16W 5% 0402 SMD	ERJ-2GE0R00X
1	R11	RES 169K OHM 1/16W 1% 0402 SMD	ERJ-2RK1693X
1	R13	RES 100K OHM 1/16W 5% 0402 SMD	ERJ-2GEJ104X
1	R18	RESISTOR 1.0 OHM 1/16W 5% 0402	ERJ-2GEJ1R0X
1	R20	RESISTOR 2.7 OHM 1/10W 5% 0402	ERJ-2GEJ2R7X
1	R27	RESISTOR .051 OHM 1/8W 5% 0402	ERJ-2BWJR051X
1	R3	RES 249K OHM 1/16W 1% 0402 SMD	ERJ-2RK2493X
1	R5	RES 510 OHM 1/16W 5% 0402 SMD	ERJ-2GEJ511X
1	R6	RES 3.3K OHM 1/16W 5% 0402 SMD	ERJ-2GEJ332X
4	R7 R8 R9 R12	RES 10K OHM 1/16W 5% 0402 SMD	ERJ-2GEJ103X
1	REG1	IC 3.3V 500MA LDO REG 8-SOIC	TPS77633D
1	U1	IC ZIGBEE SYSTEM-ON-CHIP 40-QFN	EM260-RTY
1	U2	IC AVR MCU 16K 20MHZ 32MLF	ATMEGA168-20MU-ND
1	Y1	CRYSTAL 24.000MHZ SMD	ABM3B-24.000MHZ-D-R60-1-W-T
1	Y2	CRYSTAL 8.000MHZ 18PF FUND SMD	ABMM-8.000MHZ-B2-T

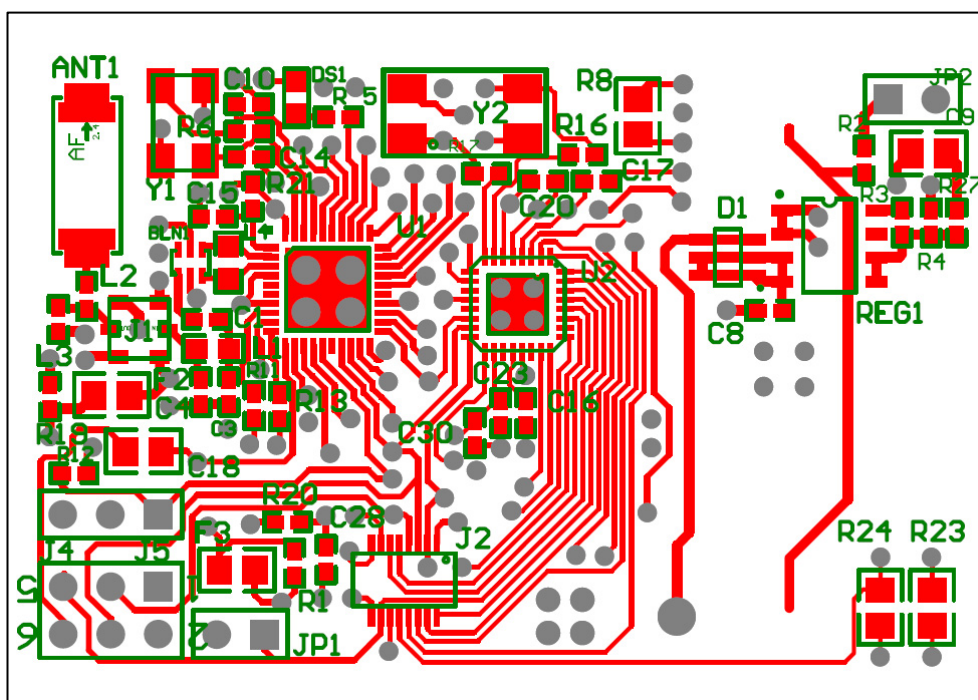
Módulo sensor

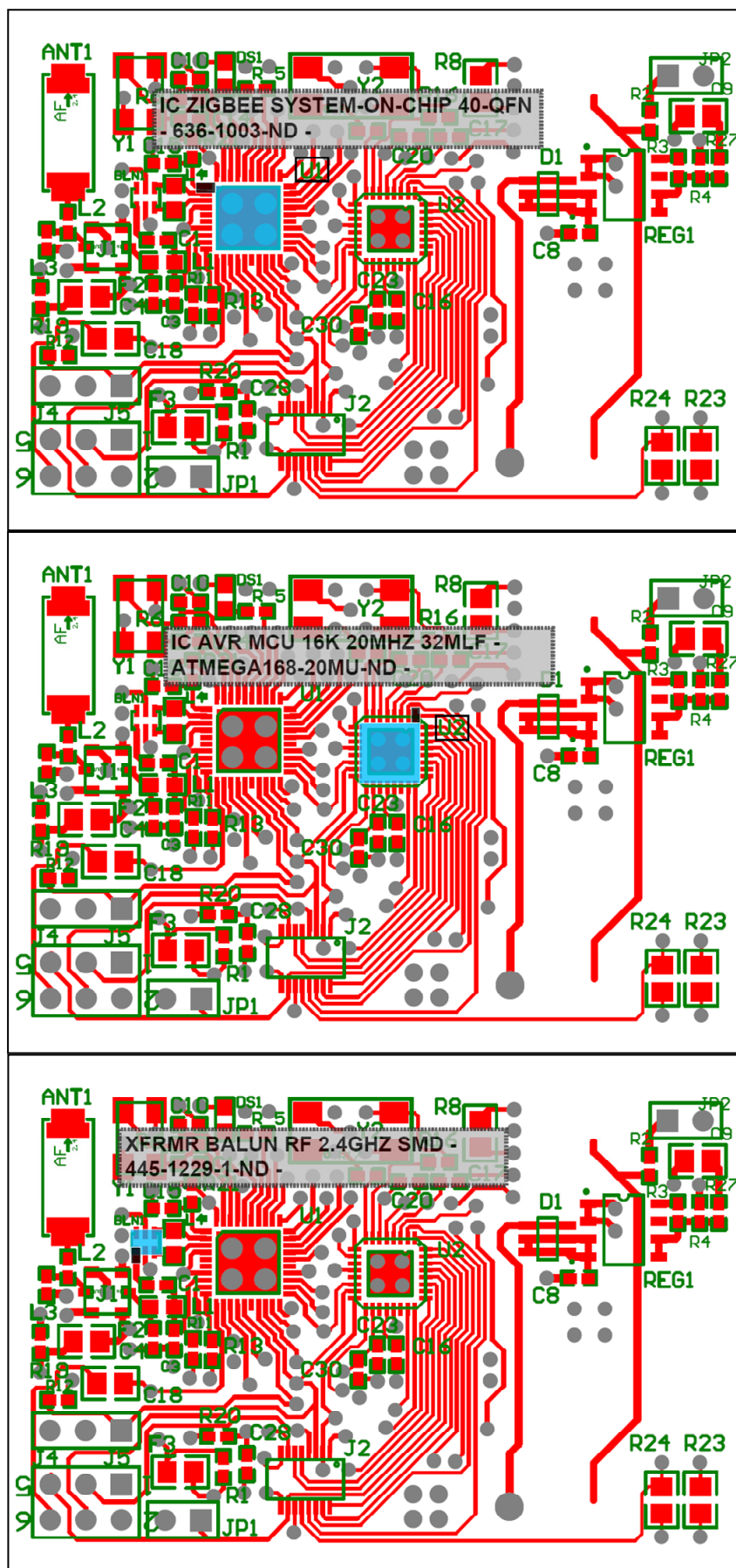
Cantidad	Nombre	Descripción del componente	Referencia del fabricante
3	C1,C2,C6	CAP CER .1UF 16V X7R 0805	ECJ-2VB1C104K
1	C3	CAP .22UF 16V CERAMIC X7R 0805	ECJ-2VB1C224K
2	C4,C5	CAP 2.2UF 6.3V CERAMIC X5R 0805	ECJ-2YB0J225K
1	D1	SMT LED Red, Clear 638nm	LTST-C230EKT
1	D2	SMT LED Green, Clear 569nm	LTST-C230GKT
1	J1	CONN RECEPT 16POS 1.5MM SMD .5MM	54722-0168-C
1	JP1	Header, 3-Pin	
1	P1	SWITCH TACT 6X3.5MM H=4.3MM 180G	TL1107AF180WQ
1	PIR	SENSOR MOTION PASSIVE INFRARED	555-28027
1	R1	RES 10K OHM 1/8W 5% 0805 SMD	MCR10EZPJ103
2	R2,R3	RES 4.7K OHM 1/8W 5% 0805 SMD	MCR10EZPJ472
1	R4	RES 51K OHM 1/8W 5% 0805 SMD	MCR10EZHJ513
1	R5,R6	RES 470 OHM 1/8W 5% 0805 SMD	MCR10EZPJ471
1	SW1	Header, 2-Pin	
1	U_s1	IC DIGITAL TEMP SENSOR SOT23-6	TMP121AIDBVT
1	U_s2	IC PRESSURE SENSOR 16.7PSI 8SSOP	MP3H6115A6U
1	U1	IC CONV BUCK/BOOST 60MA SOT23-6	REG710NA-5/250
1	U2	MOSFET P-CH 50V 130MA SOT-23	BSS84,215

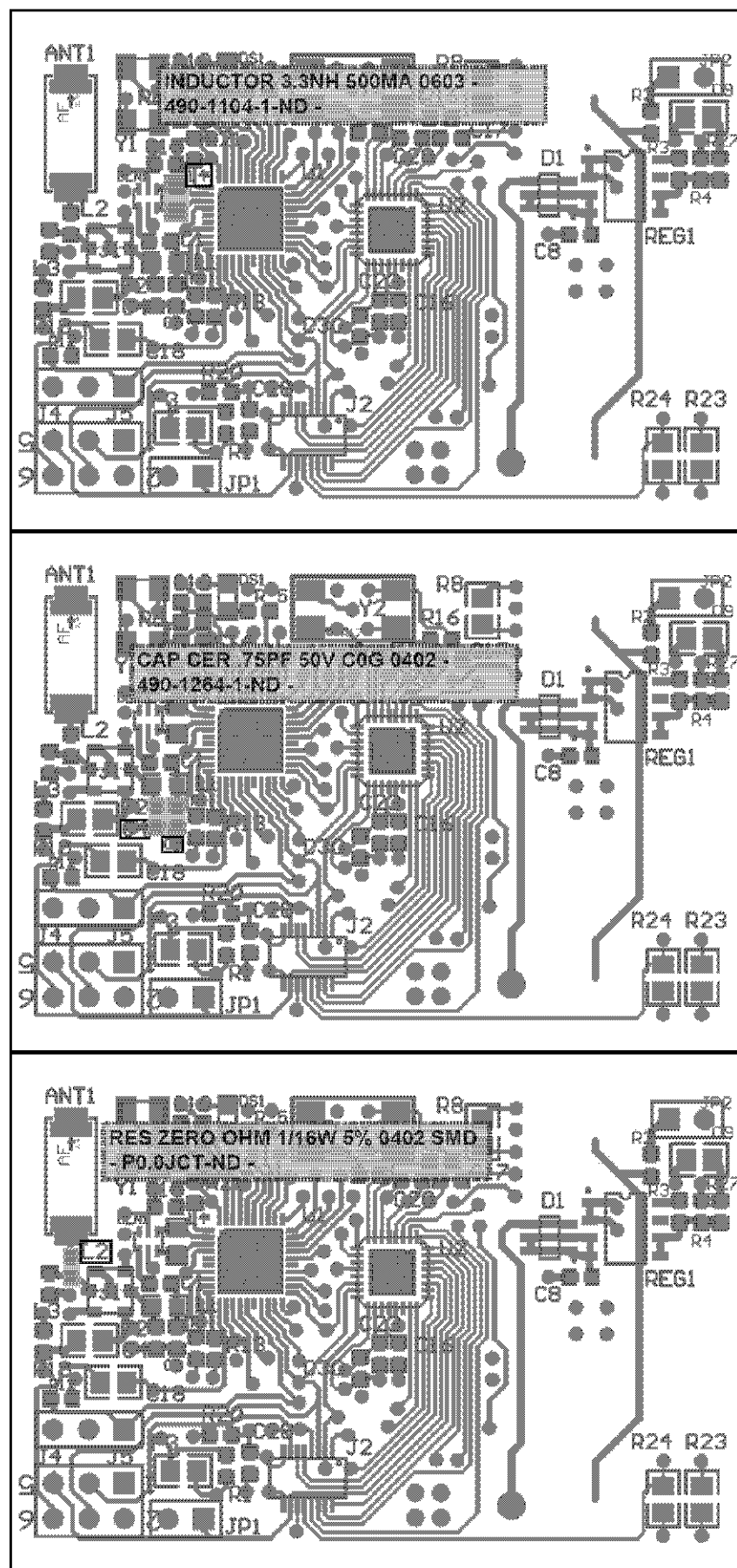
Anexo C

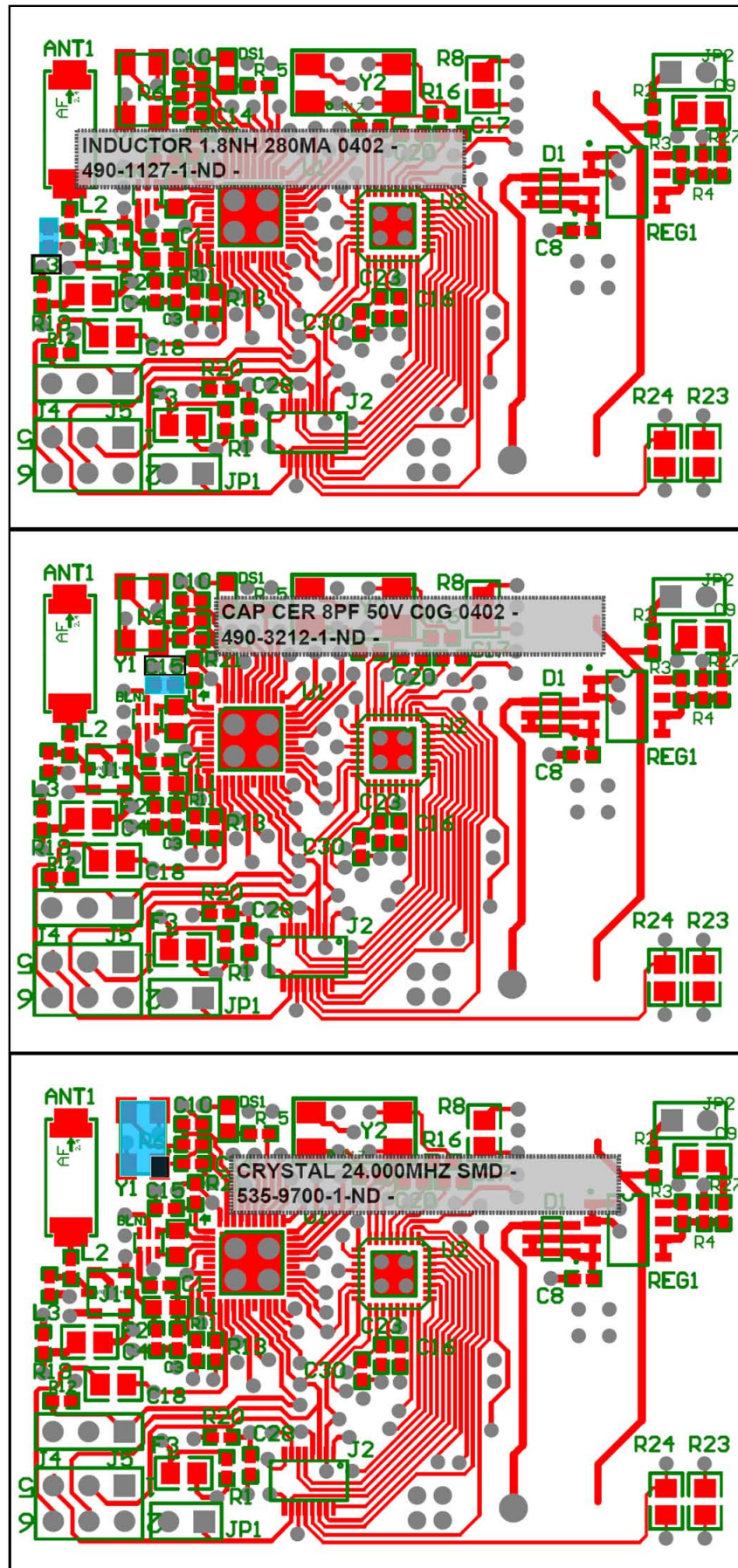
Guía de fabricación

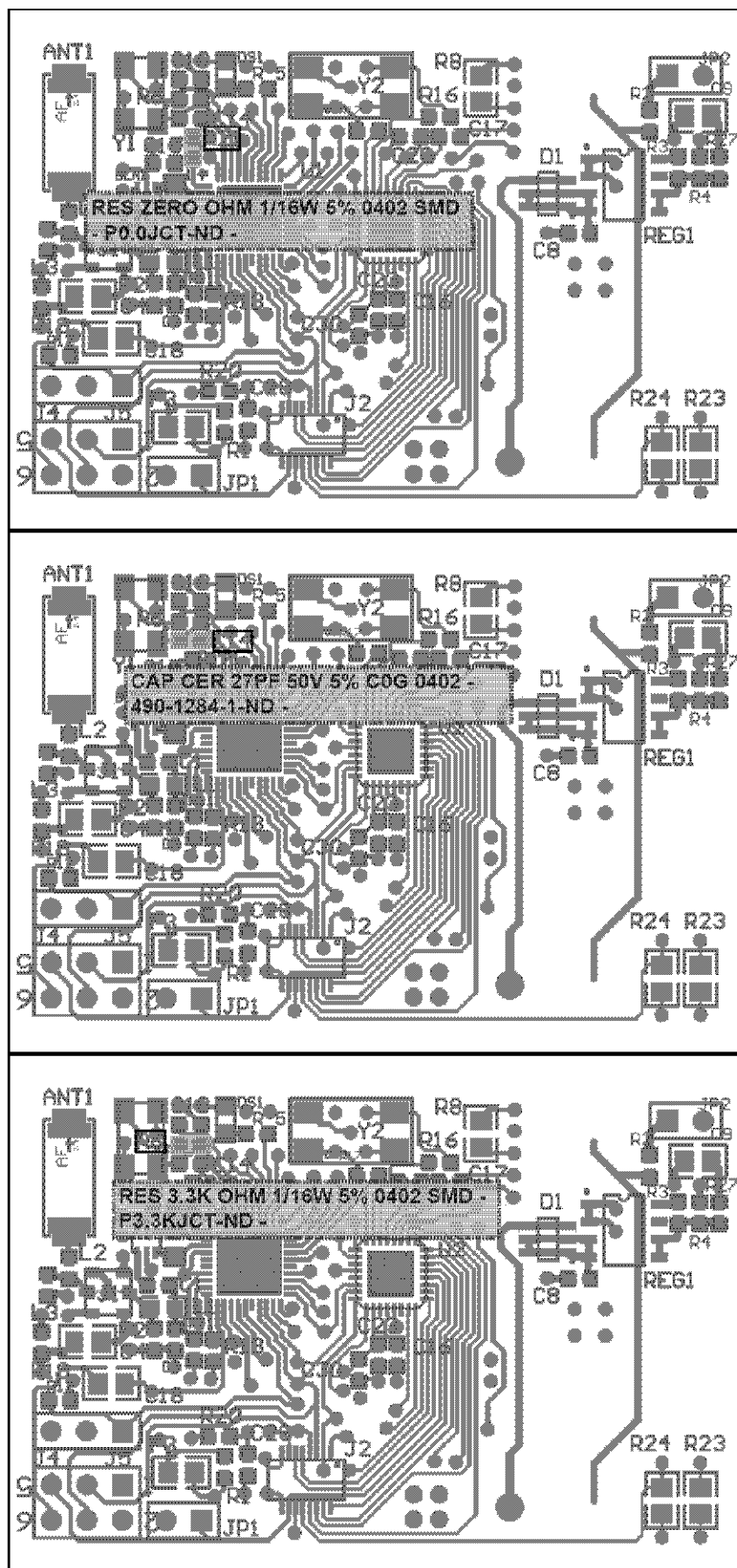
Top Layer

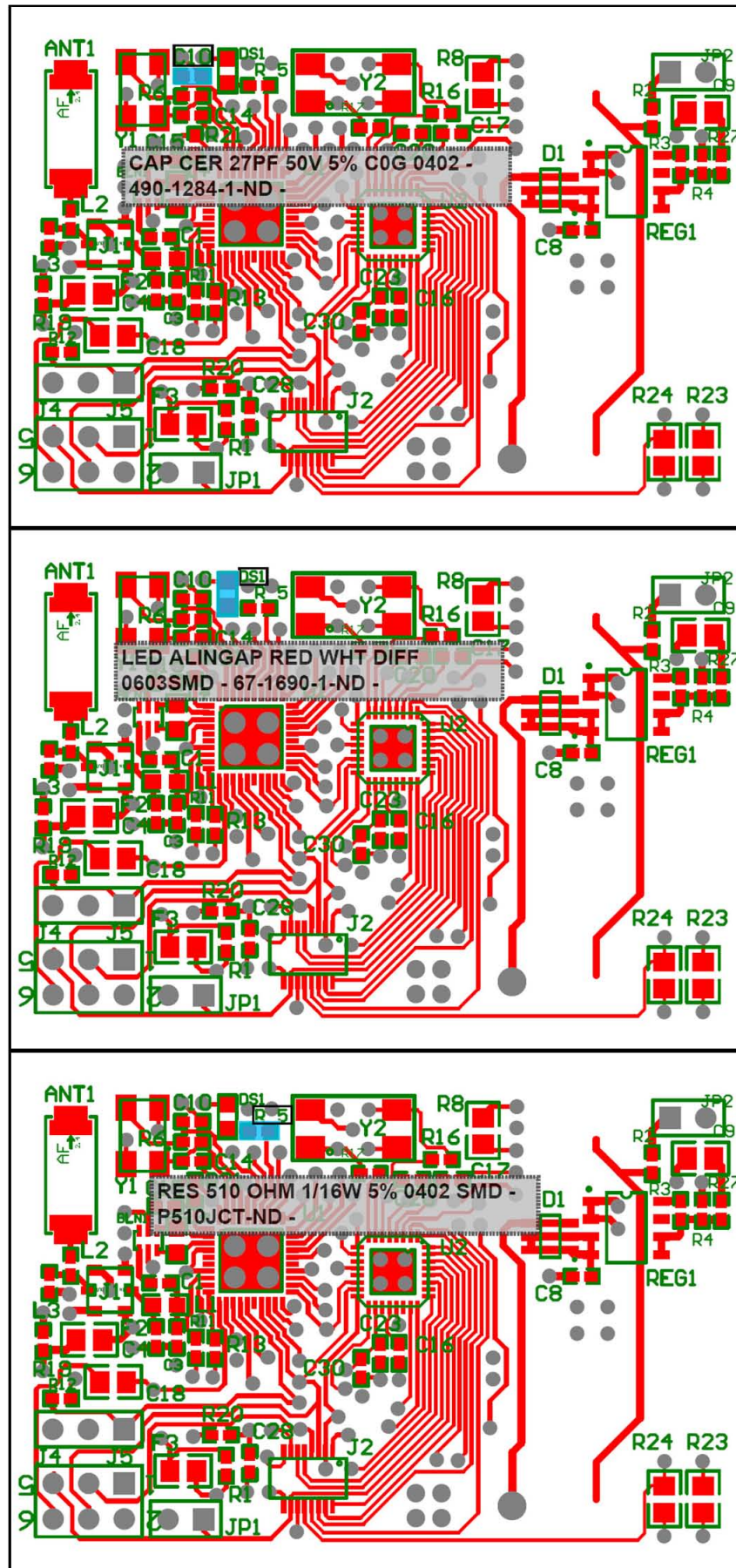


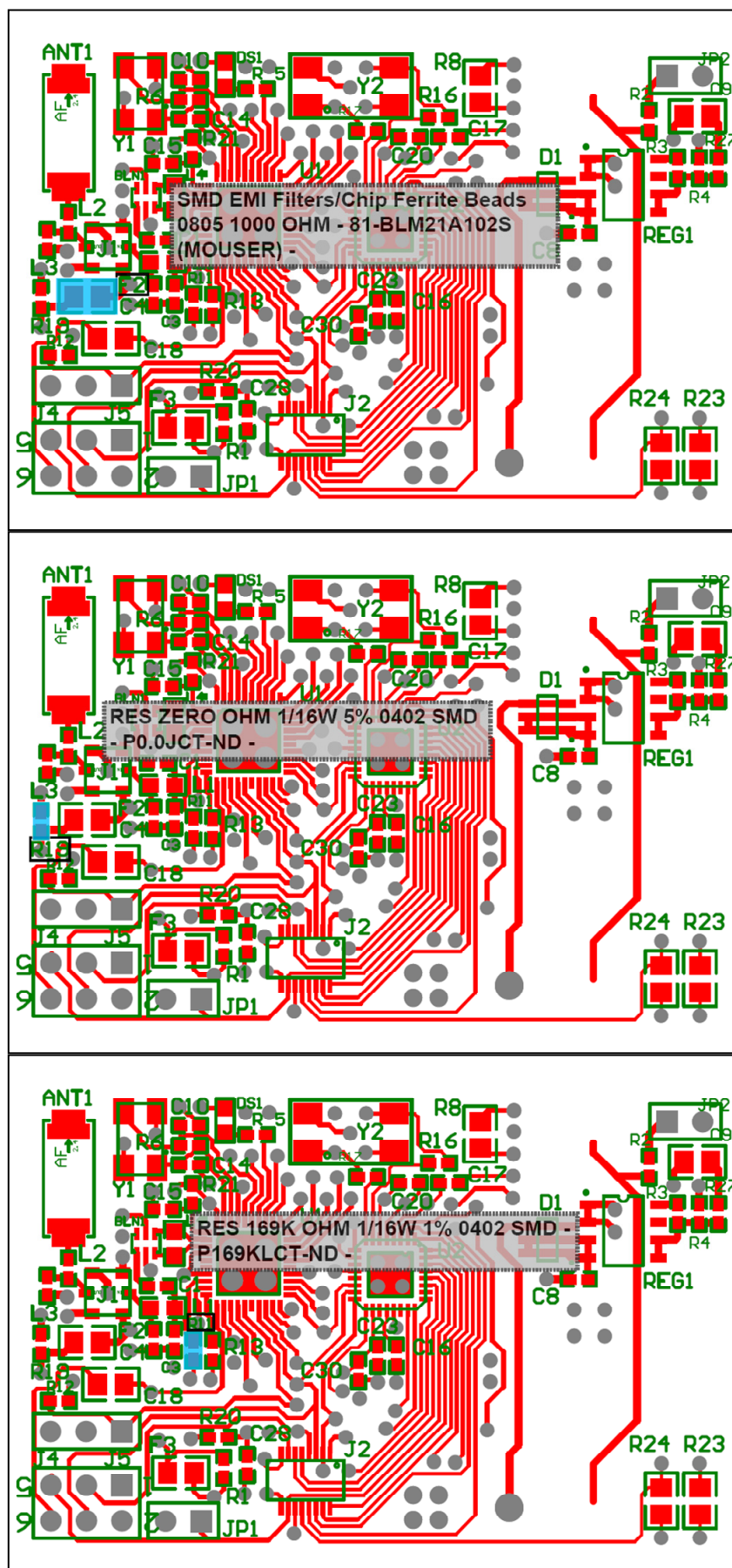


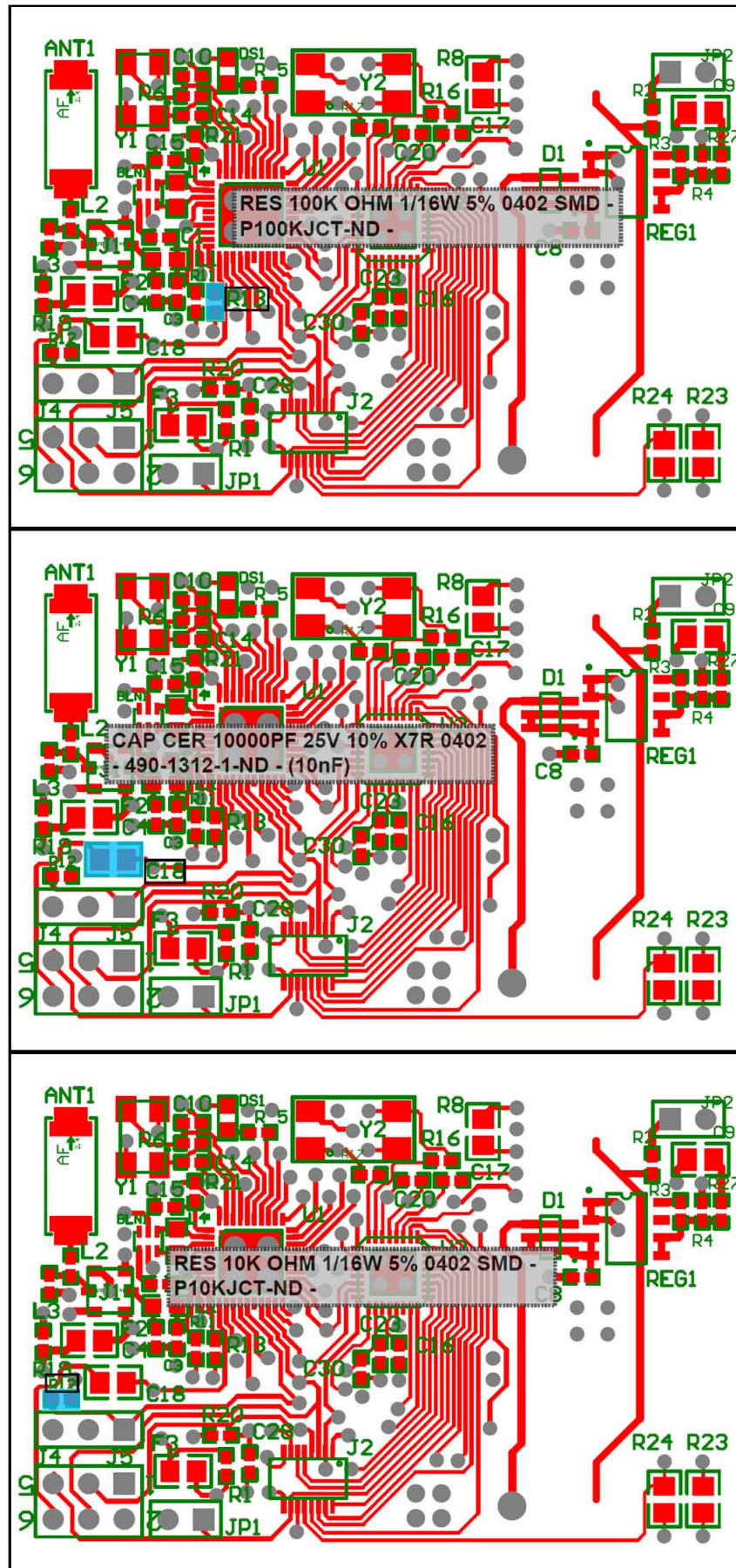


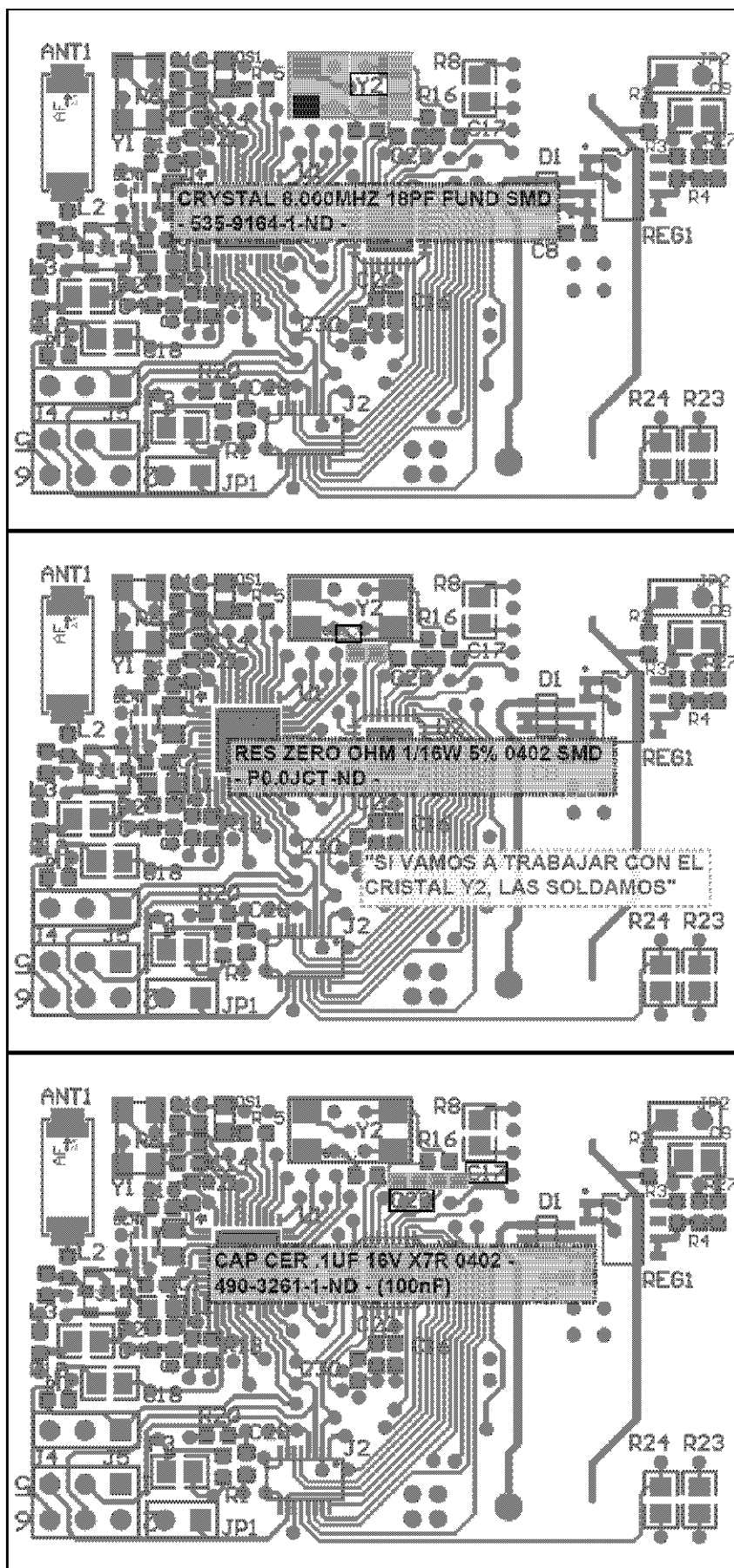


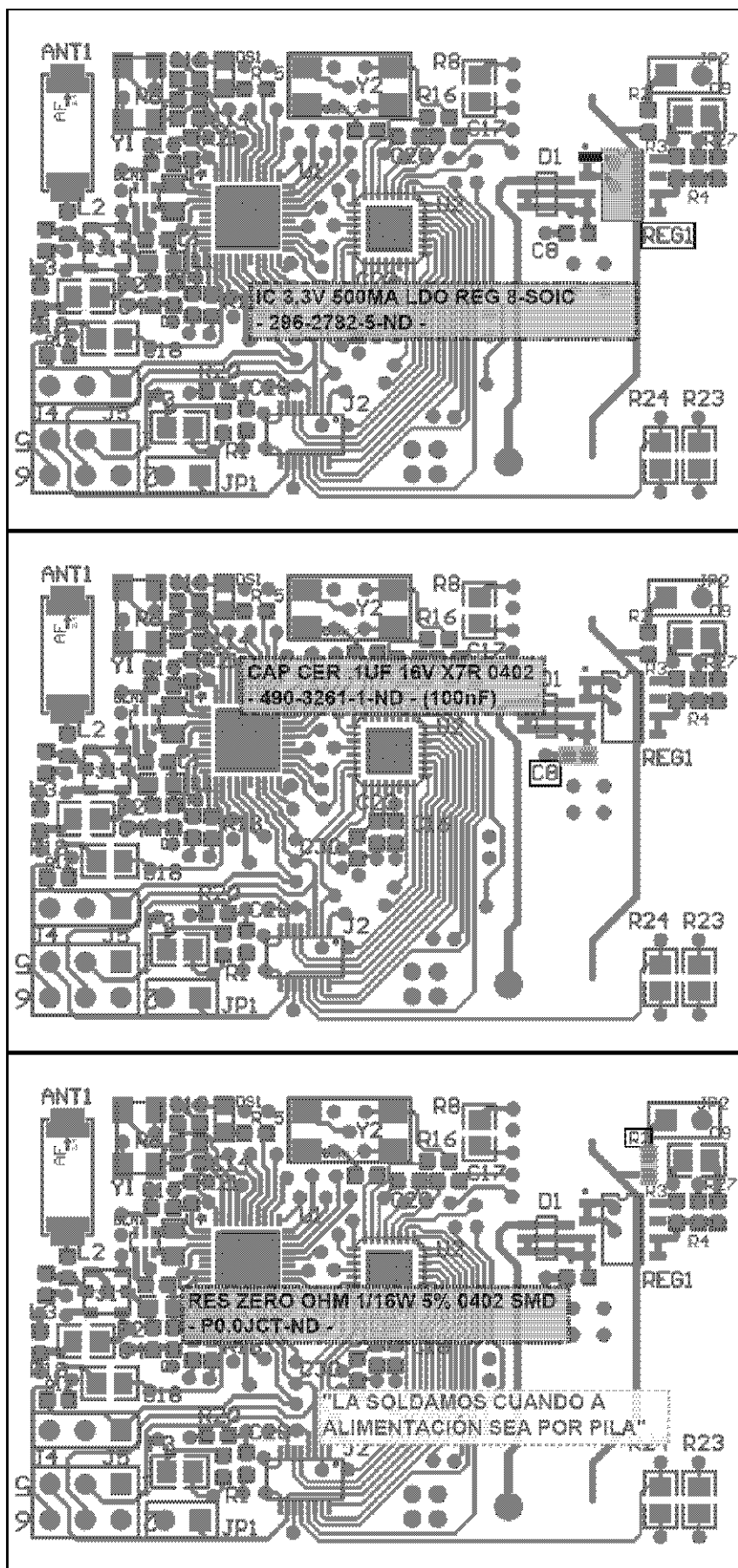


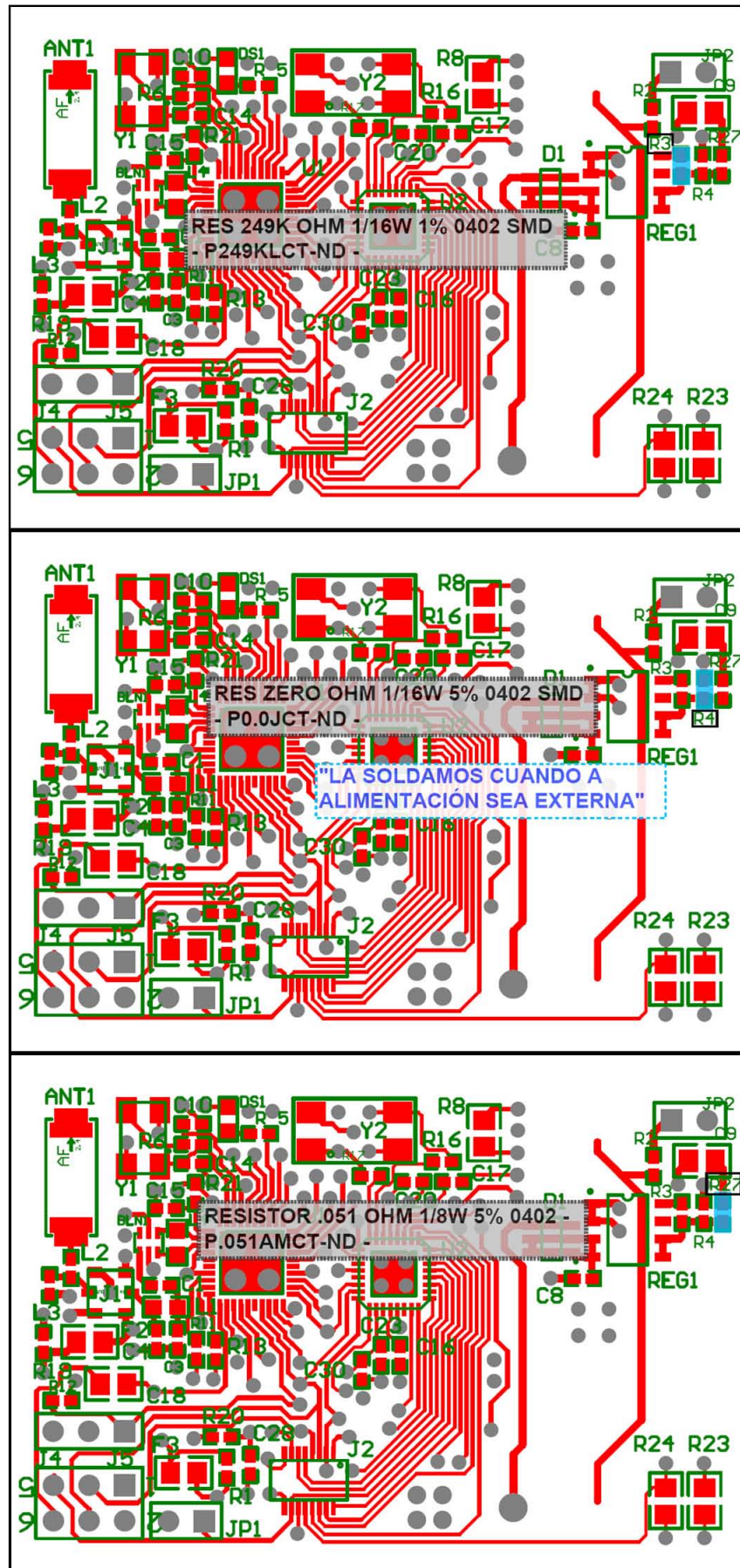


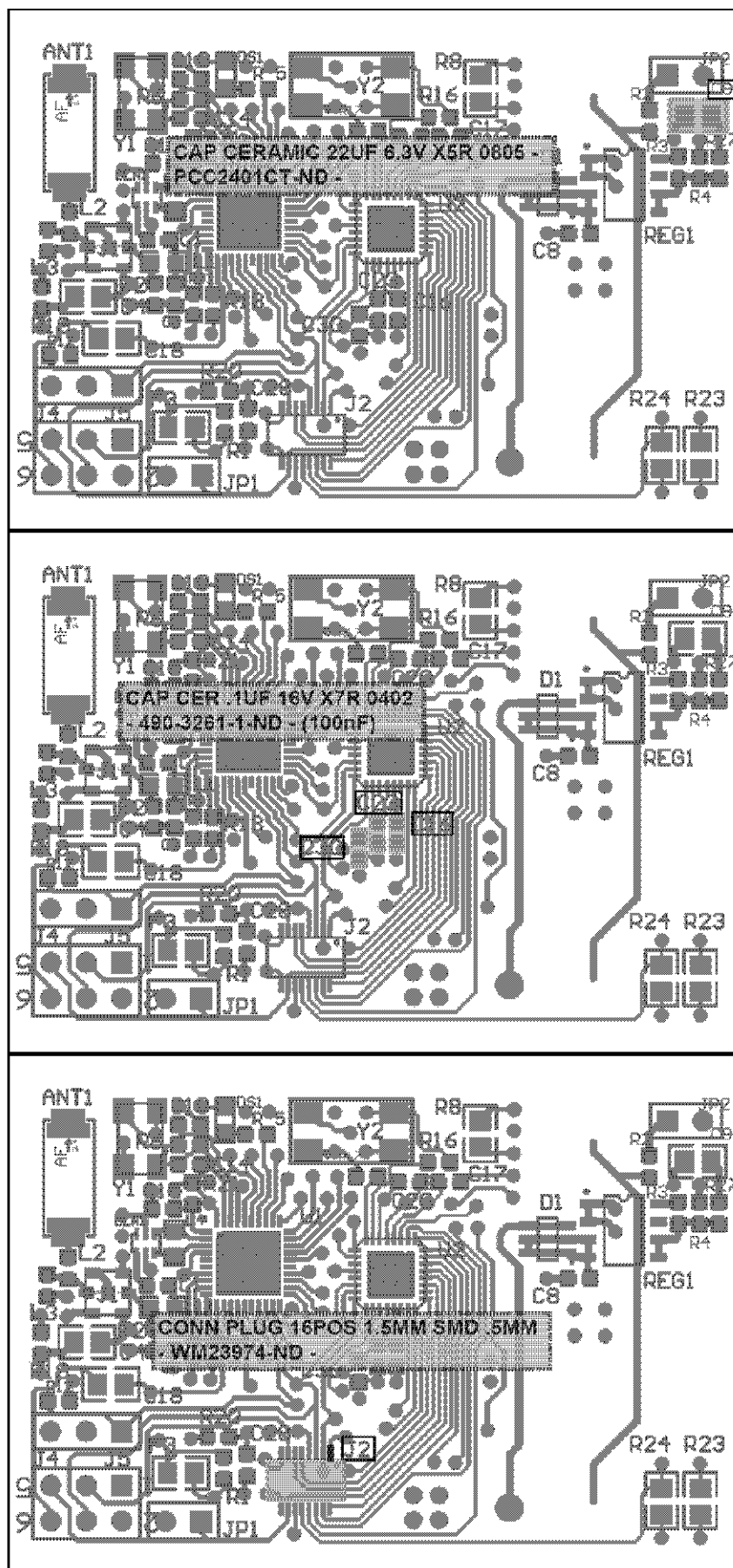


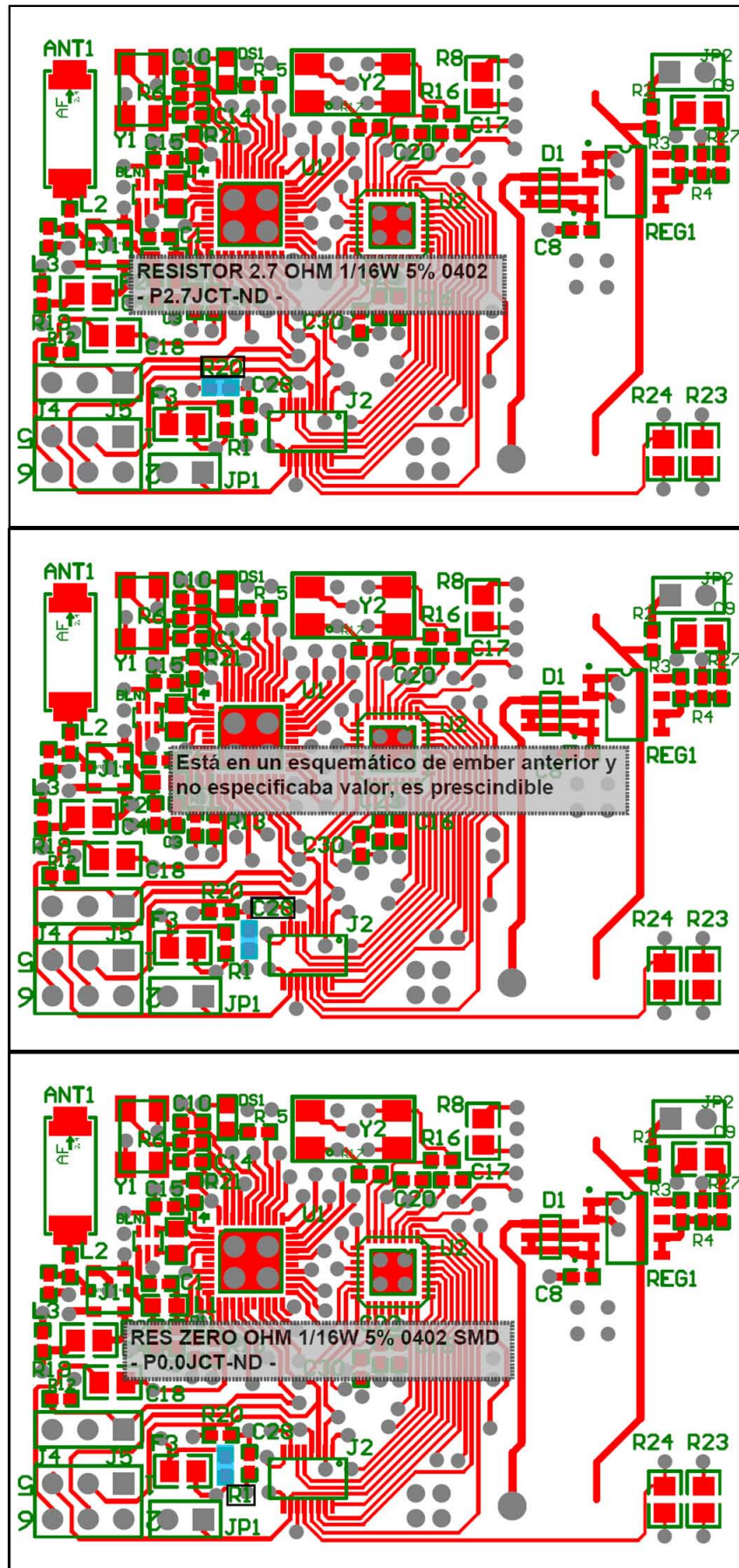


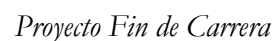


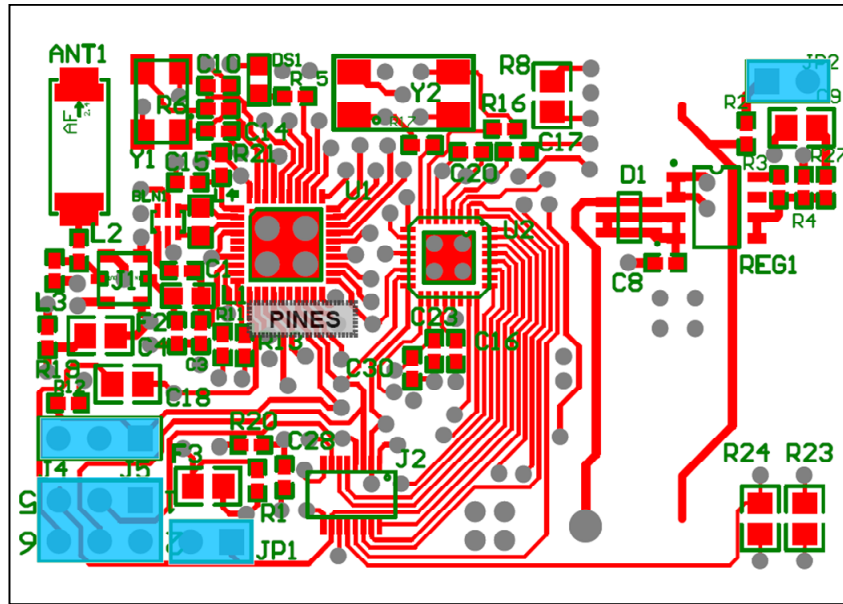




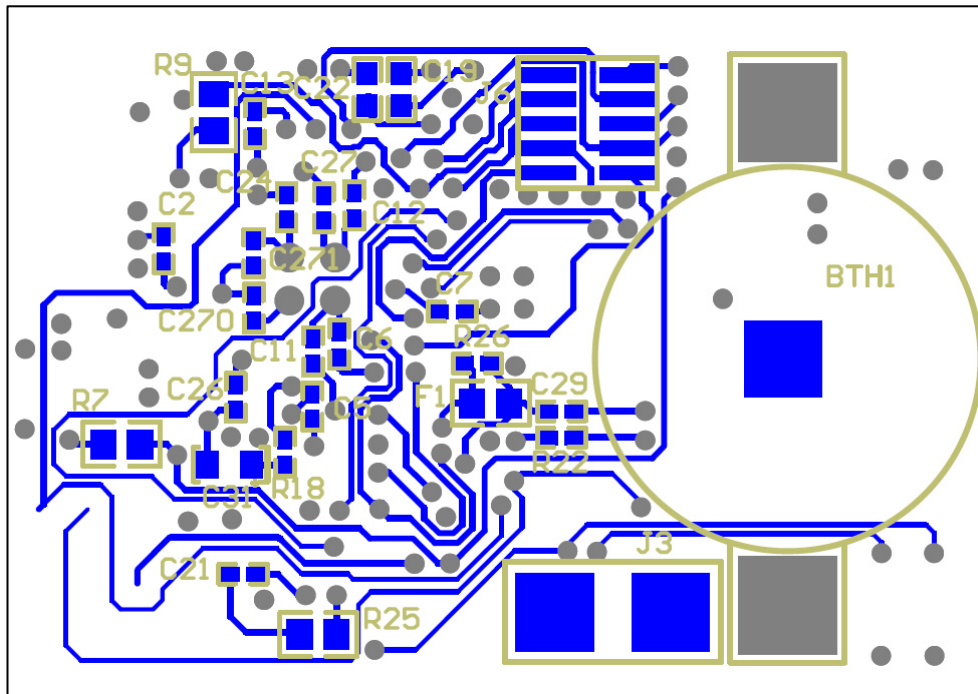


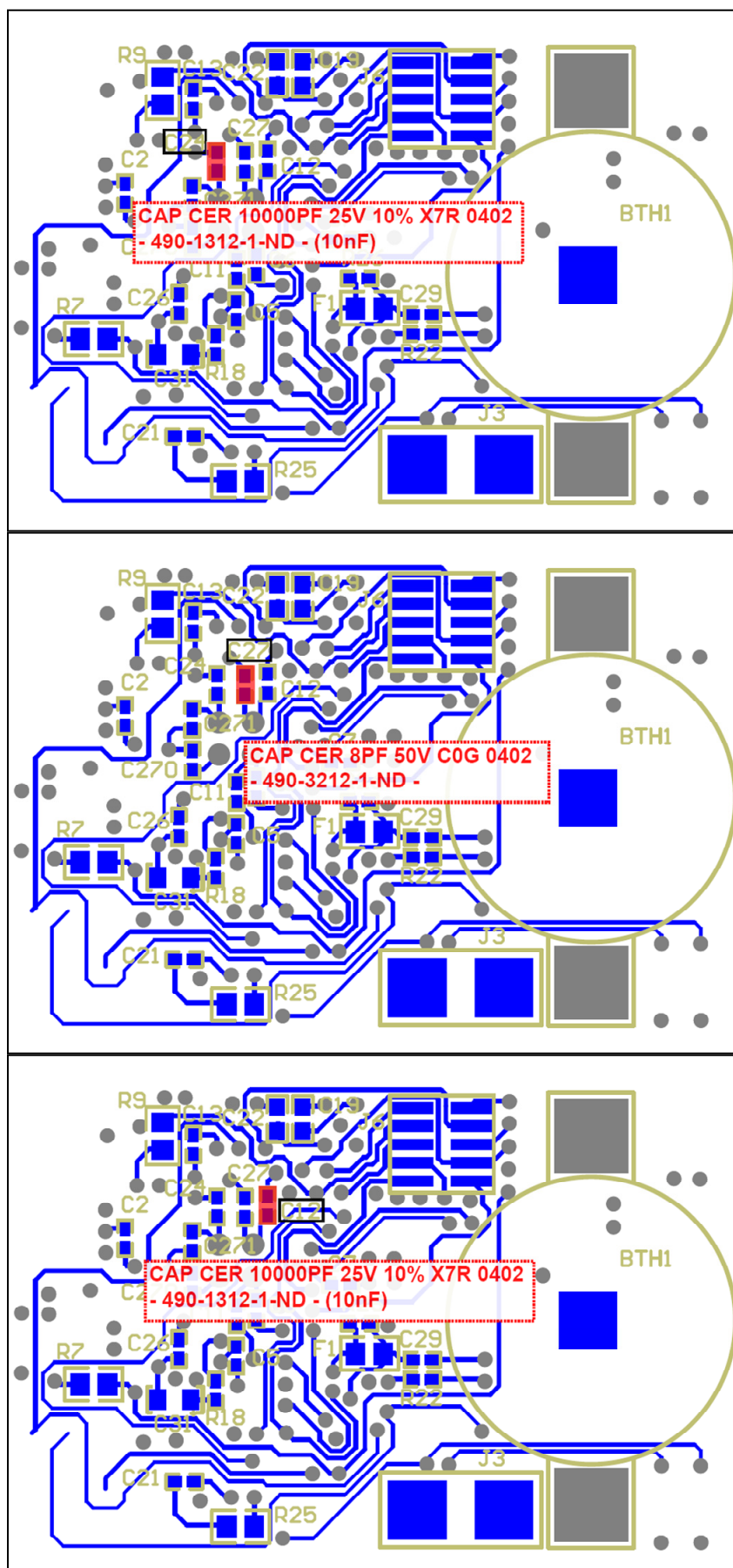


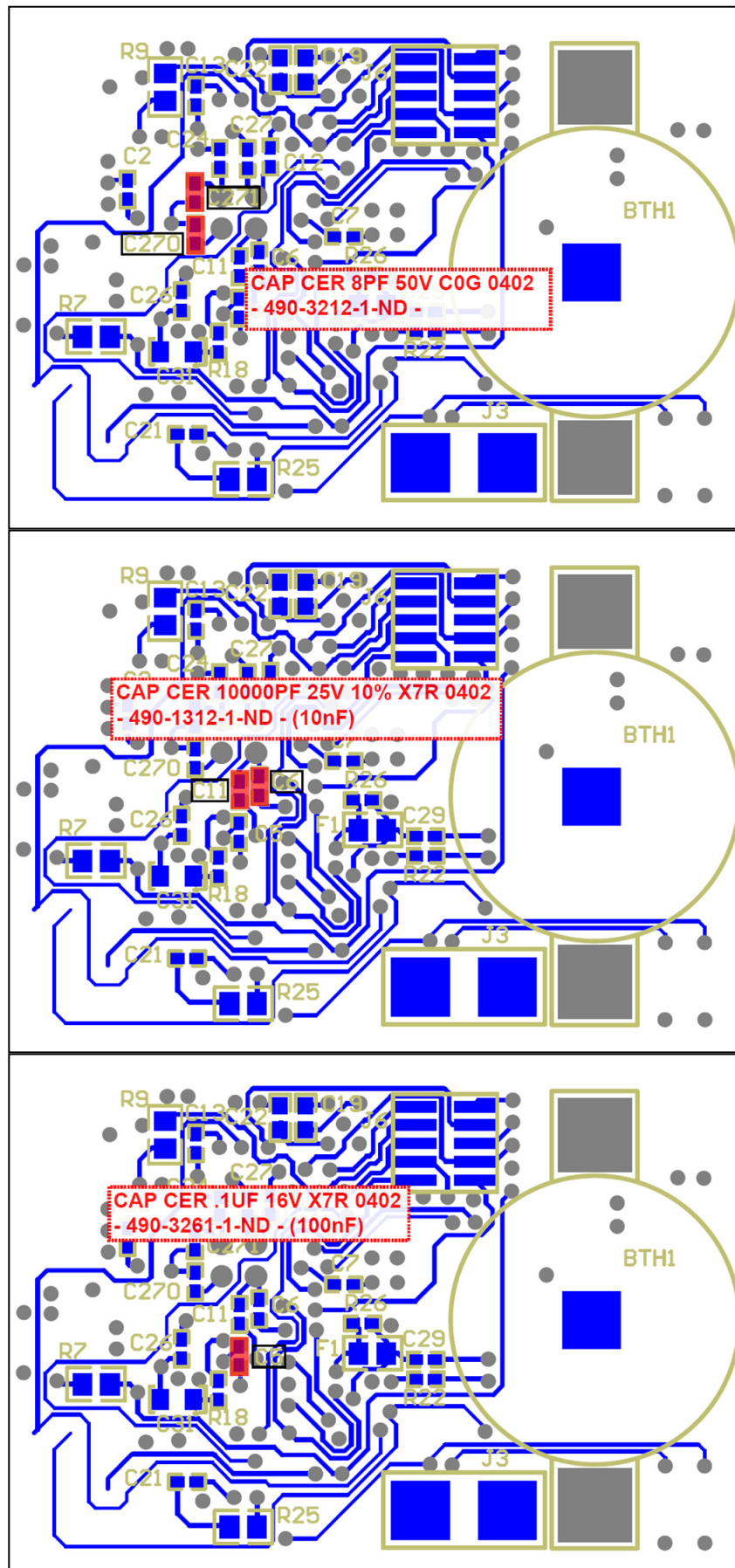


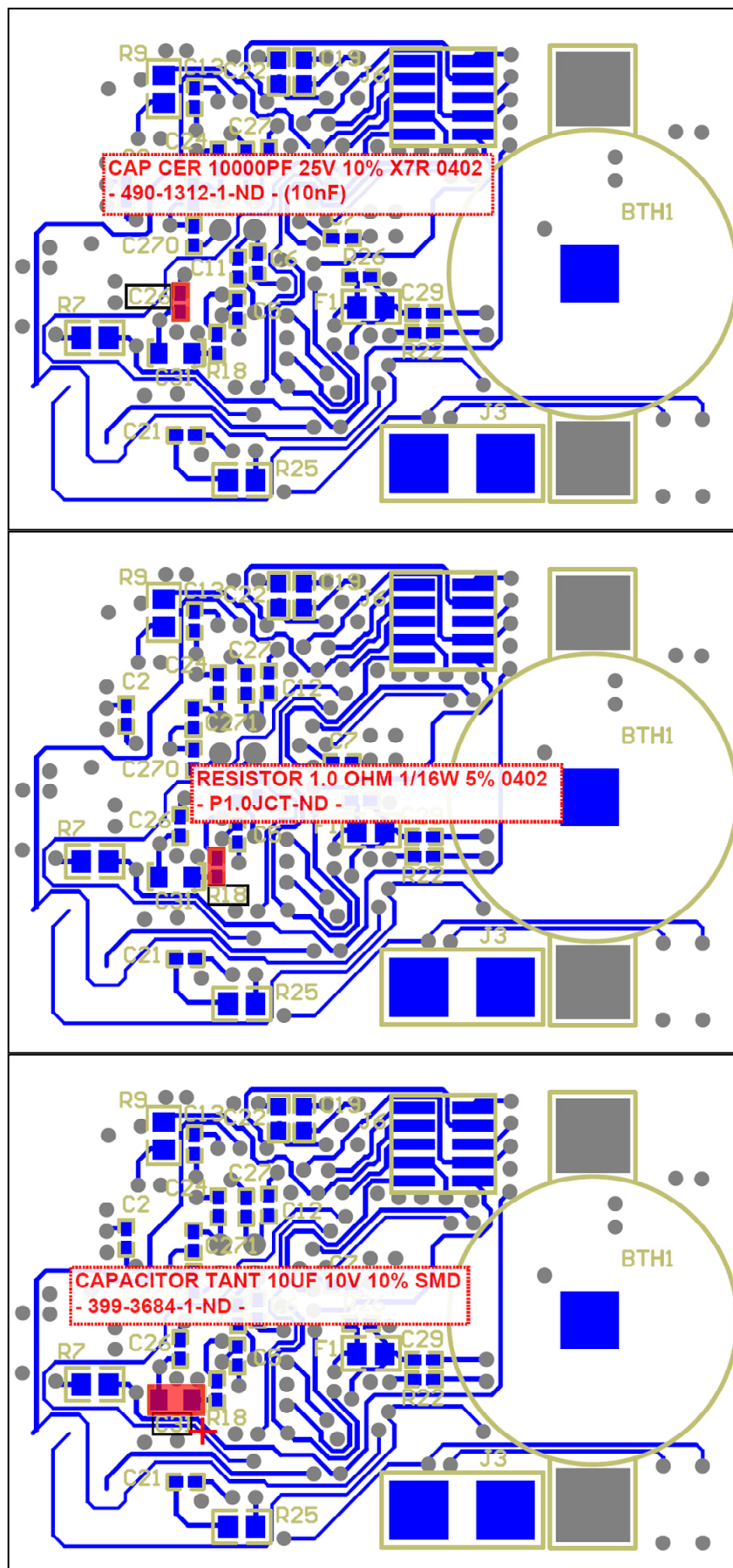


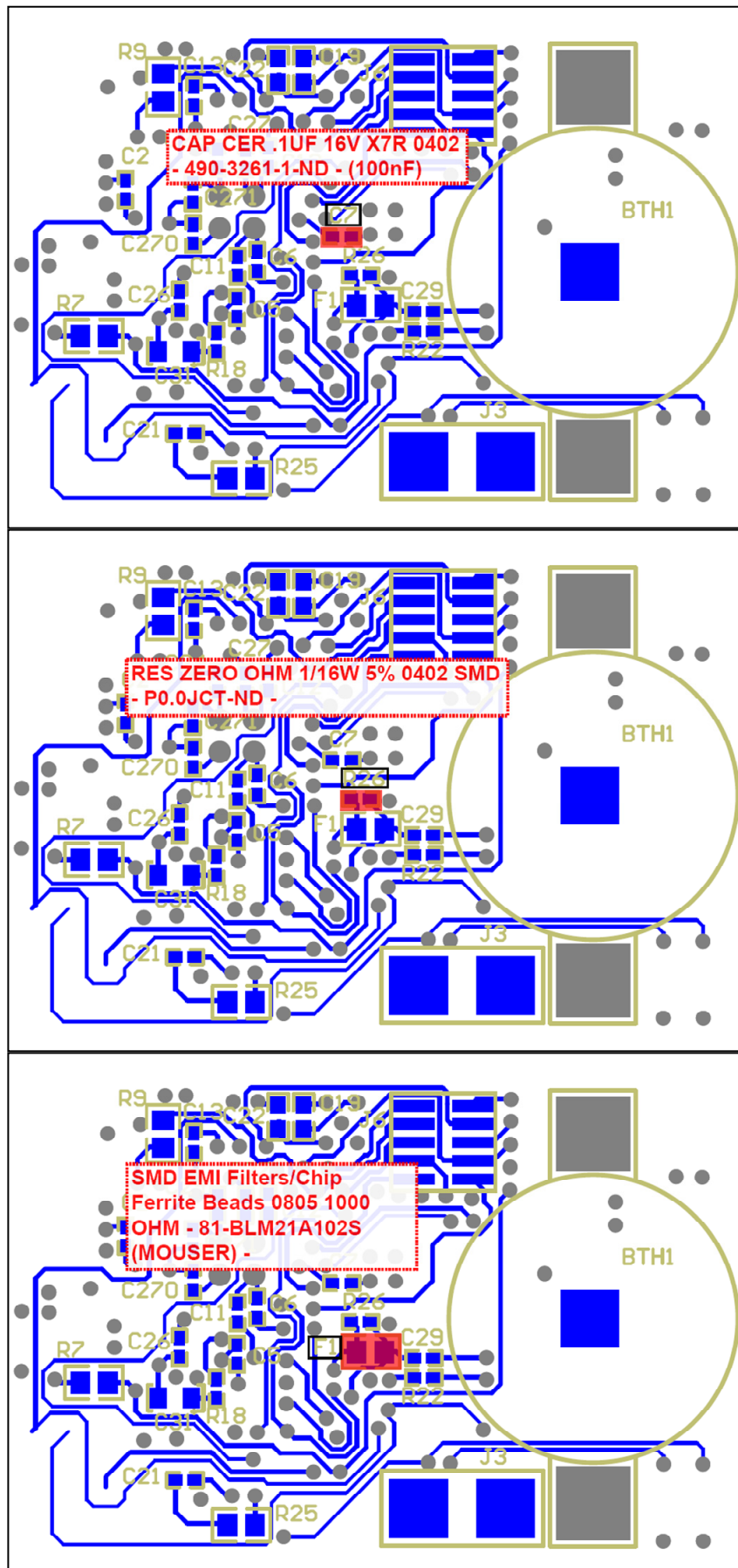
Bottom Layer

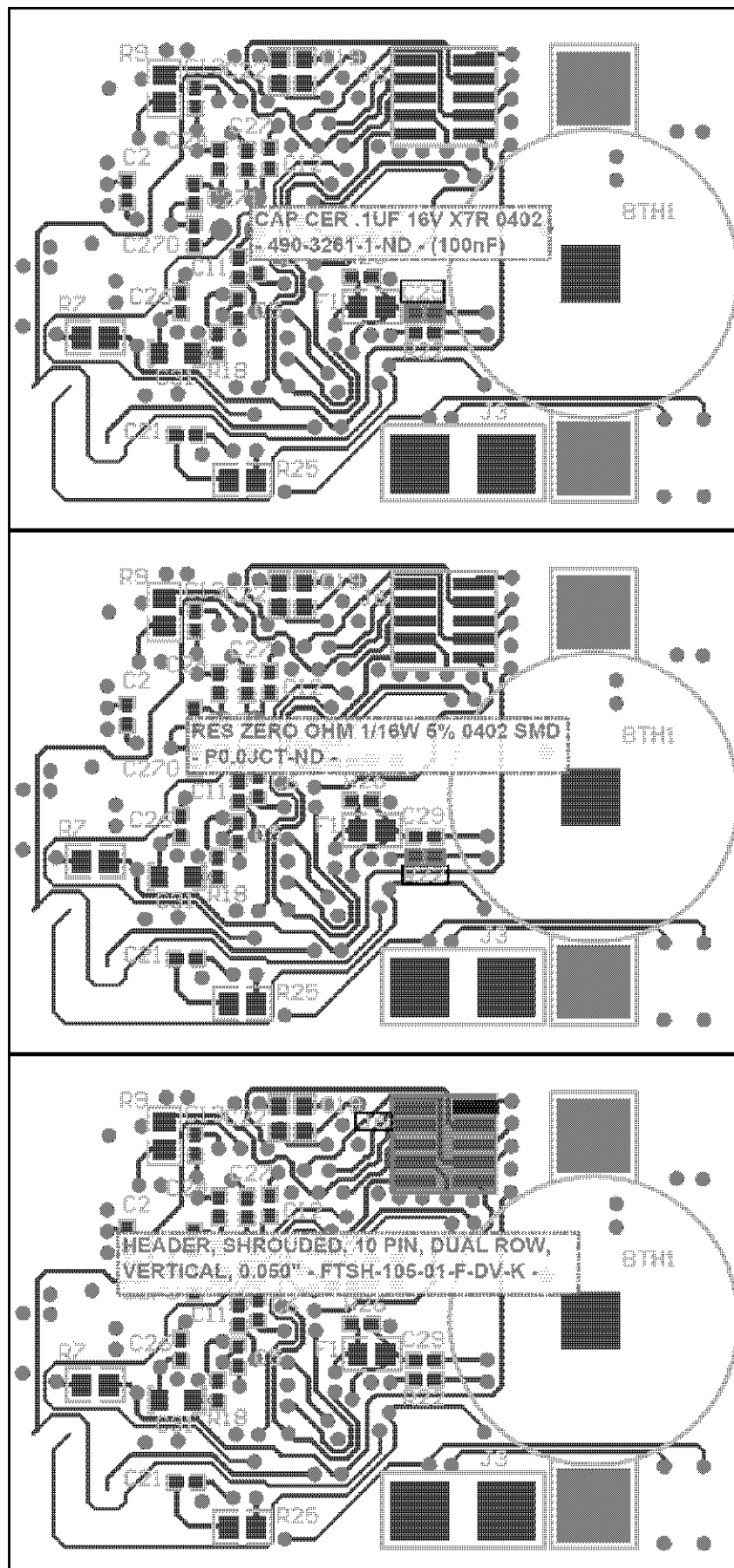


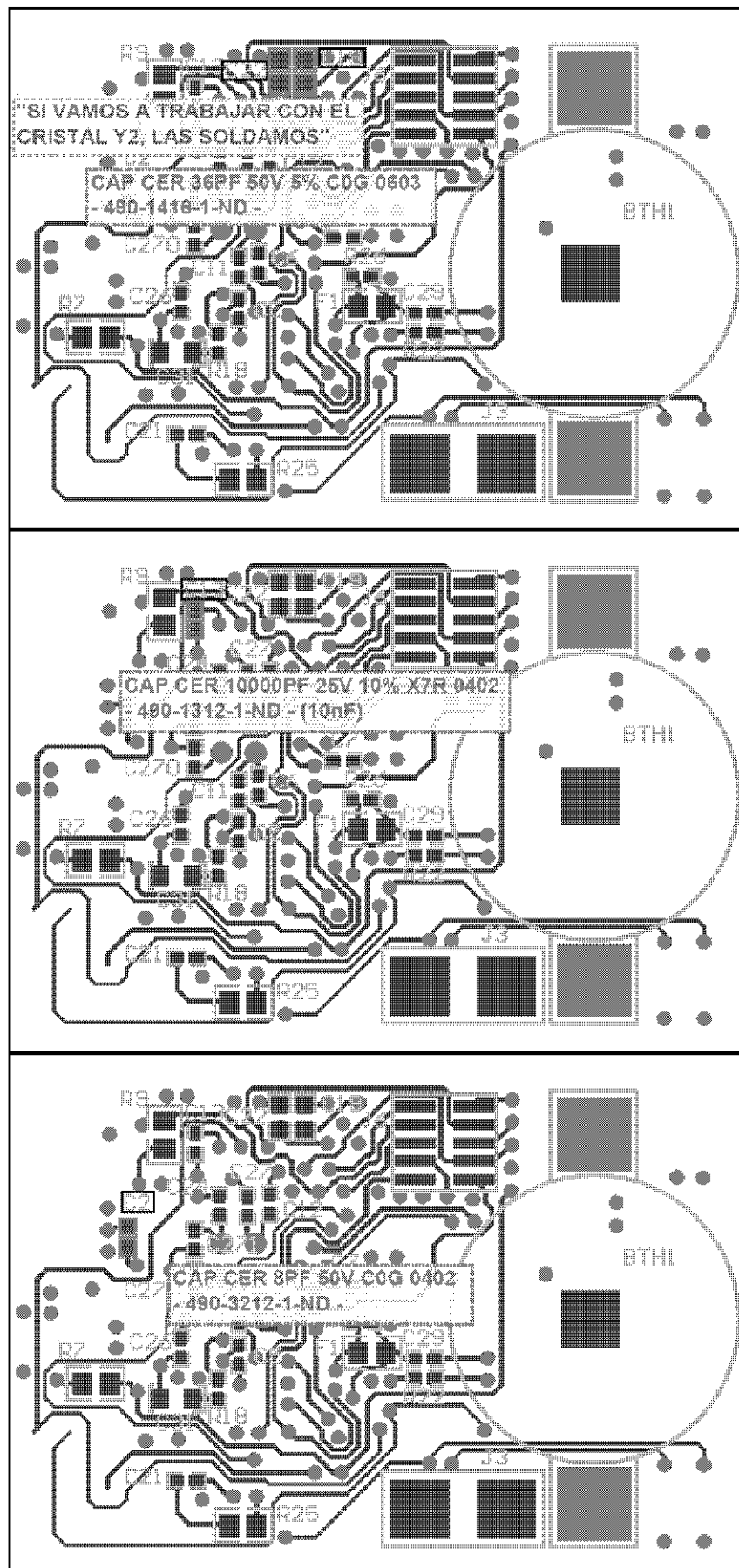


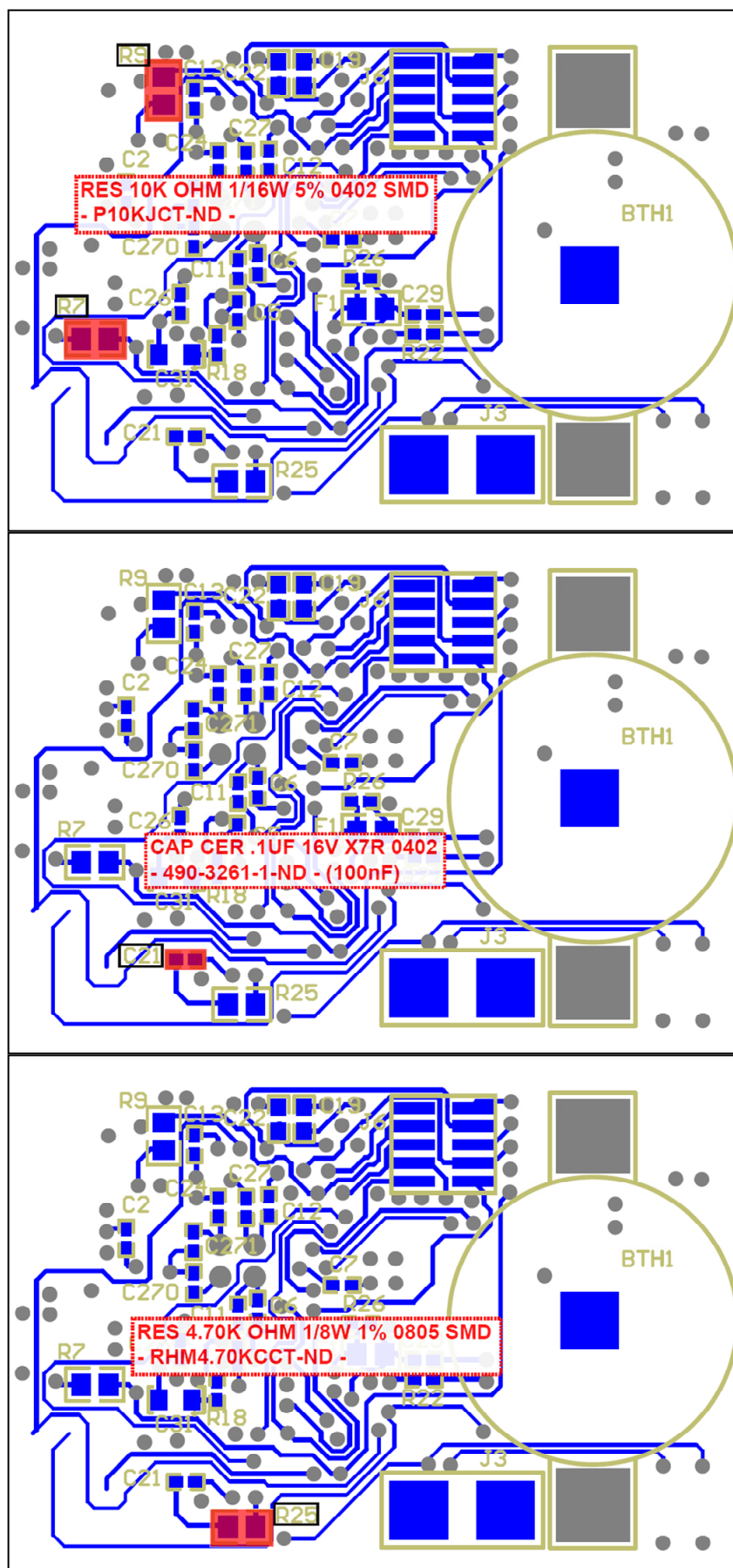


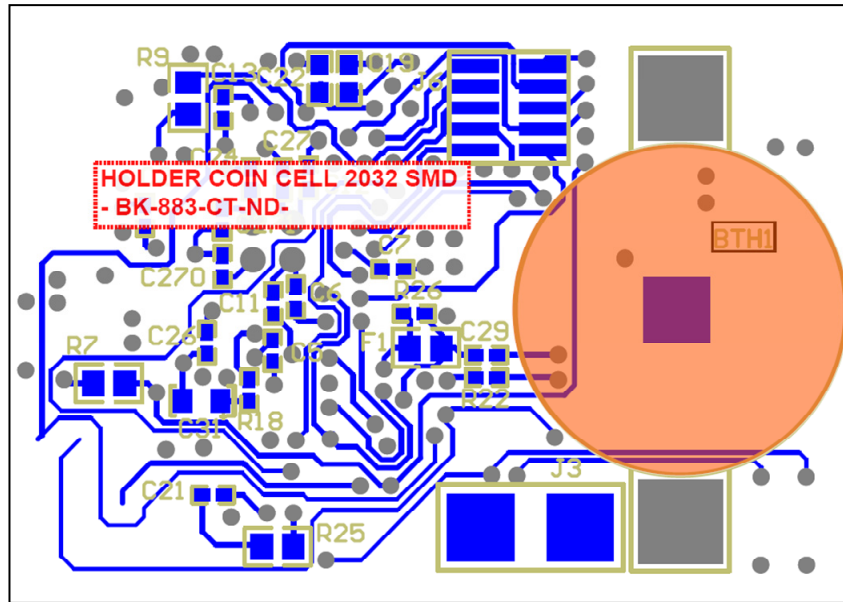












Anexo D

Código fuente

Fichero main.c

```
////////////////////////////////////  
////////////////////////////////////  
////                               main.c                               ////  
////                               ////  
////     Este es fichero contiene el código del firmware     ////  
////     desarrollado para el nodo sensor inalámbrico,       ////  
////     implementado en el PFC titulado "IMPLEMENTACIÓN DE   ////  
////     UN SENSOR INALÁMBRICO CON BAJO CONSUMO DE PROPOSITO  ////  
////                               GENERAL".                     ////  
////////////////////////////////////  
////////////////////////////////////
```

```
#include <stdio.h>  
#include <ctype.h>  
#include <stdlib.h>  
#include <iom168.h>  
#include <intrinsics.h>
```

```

typedef unsigned char  u8;
typedef unsigned short u16;
typedef unsigned long  u32;
typedef unsigned long long u64;
typedef char  s8;
typedef short s16;
typedef int  s32;
typedef long long s64;

void modulo_init          ( void );
void SPI_Transmit         ( u8 );
u8 SPI_Receive            ( void );
void reset_ember          ( void );
u8 protocol_version       ( u8 );
u8 leer_tramaEM260        ( void );
void identificar_trama     ( u8 );
void EmberIncomingMessage ( void );
void checkVersionEZSP     ( void );
void estado_nodo           ( void );
void estado_red            ( void );
void EZSP_version          ( void );
void EZSPStatus            ( void );
void EZSPStatusMSG        ( void );
void EzsetConfigurationValue (u8,u16);
void EZcallback            ( void );
void EZnetworkState        ( void );
void EzjoinNetwork        ( u8 nodeType,
                           u8 *ExPanId ,
                           u16 PanId ,
                           s8 radioTxPower,
                           u8 radioChannel );
void EZsendUnicast         ( u8 MsgType,
                           u16 DestAdr,
                           u16 ProfileId,
                           u16 ClusterId,

```

```
        u8 SrcEP,
        u8 DstEP,
        u16 ApsOptions,
        u16 GroupId,
        u8 MsgSeq,
        u8 MsgTag,
        u8 MsgLen,
        u8 *TxBuf);

void intermitente_leds      ( u8 tipo, u8 repeticiones );
void errores                ( u8 idError );
void leer_temp              ( void );
void leer_presion           ( void );
u8  comprobar_PIR          ( void );
void ordenar_datos         ( void );
void config_EM260           ( void );
void desconexion_red        ( void );
void retardo_ms             ( u16 );
void retardo_us             ( char );
void limpiar_trama         ( void );
void EZleaveNetwork         ( void );
void Bajo_consumo           ( void );
void despertar_EM260        ( void );
void start_timer            ( void );
void stop_timer             ( void );
```

```
//////////////////////////////////////      _____DEFINICIONES_____      ////////////////////////////////////////
```

```
#define true 1
#define false 0
```

```
#define CHK_nHOST  ( PINC & 0x01 )
#define SET_nHOST  ( PORTC |= 0x01 )
```

```
#define CHK_nSSEL  ( PINB & 0x04 )
#define SET_nSSEL  ( PORTB |= 0x04 )
#define CLR_nSSEL  ( PORTB &= 0xFB )
```

```
#define CHK_nRESET_EM    ( PINC & 0x04 )
#define SET_nRESET_EM    ( PORTC |= 0x04 )
#define CLR_nRESET_EM    ( PORTC &= 0xFB )

#define CHK_nWAKE ( PINC & 0x02 )
#define SET_nWAKE ( PORTC |= 0x02 )
#define CLR_nWAKE ( PORTC &= 0xFD )

#define CHK_SCLK    ( PINB & 0x20 )
#define SET_SCLK    ( PORTB |= 0x20 )
#define CLR_SCLK    ( PORTB &= 0xDF )

#define CHK_MOSI    ( PINB & 0x08 )
#define SET_MOSI    ( PORTB |= 0x08 )
#define CLR_MOSI    ( PORTB &= 0xF7 )

#define CHK_MISO    ( PINB & 0x10 )

#define LED2ON      ( PORTD |= 0x40 ) // Led verde
#define LED2OFF     ( PORTD &= 0xBF )
#define LED1ON      ( PORTD |= 0x20 ) // Led rojo
#define LED1OFF     ( PORTD &= 0xDF )

#define ON_PIR      ( PORTB |= 0x01 )
#define OFF_PIR     ( PORTB &= 0xFE )

#define CHK_PIR     ( PIND & 0x08 )

#define ON_PRESSION ( PORTC &= 0xEF )
#define OFF_PRESSION( PORTC |= 0x10 )

#define ON_TEMPERA( PORTD &= 0xEF )
#define OFF_TEMPERA ( PORTD |= 0x10 )

#define CHK_SW1     ( PIND & 0x04 )
```



```
#define SPI_RDY      (SPSR & 0x80)
#define SPI_BUSY     (! SPI_RDY )

#define EMBER_NETWORK_UP           0x90
#define EMBER_NETWORK_DOWN        0x91
#define EMBER_JOIN_FAILED          0x94
#define EMBER_MOVE_FAILED          0x96

#define EMBER_NO_NETWORK           0x00
#define EMBER_JOINING_NETWORK      0x01
#define EMBER_JOINED_NETWORK       0x02
#define EMBER_JOINED_NETWORK_NO_PARENT 0x03
#define EMBER_LEAVING_NETWORK      0x04

#define EZSP_SUCCESS               0x00

#define EMBER_UNKNOWN_DEVICE       0x00
#define EMBER_COORDINATOR          0x01
#define EMBER_ROUTER               0x02
#define EMBER_END_DEVICE           0x03
#define EMBER_SLEEPY_END_DEVICE    0x04
#define EMBER_MOBILE_END_DEVICE    0x05

#define idNEIGHBOUR                0xFFFF
#define idCOORDINADOR              0x0000

#define EMBER_OUTGOING_DIRECT      0x00

#define ONLY_RED                   0x00
#define ONLY_GREEN                 0x01
#define RED_GREEN_SIC              0x02
#define RED_GREEN_ASIC             0x03
```

////////////////////////////////////// _____ DECLARACIÓN DE VARIABLES _____ //

```
u32 contador;                // Contador de los 4 segundos
```

```

u8 SEQ; // Secuencia de comandos del EM260
u8 SEQmsg; // Secuencia de mensajes al EM260
u8 conectado; // Estado de la conexión
u8 SPI_trama[160]; // Trama SPI
u8 verEZSP; // Versión del EZSP que se va a enviar
u8 act_PIR;
u8 ExPanId[] = {0xFE,0xED,0xC0,0xDE,0xDE,0xC0,0xED,0xFE};
u8 radioChannel;
s8 radioTxPower;
u16 PanId;
u8 EndPointID;
u8 TxData[6] = {0x00,0x00,0x00,0x00,0x00,0x00};
u8 TempData[2]= {0x00,0x00};
u8 PresionData[2]= {0x00,0x00};

```

```

int main( void )
{
    modulo_init();
    reset_ember ();
    config_EM260 ();
    //ON_PIR;
    while(1)
    {
        do
        {
            if (!CHK_nHOST)
            {
                EZcallback();
                identificar_trama(leer_tramaEM260());
            }
            switch (conectado)
            {
                case true:
                    LED2ON;
                    break;
                case false:

```

```
        EzjoinNetwork ( EMBER_END_DEVICE,
                        ExPanId,
                        PanId,
                        radioTxPower,
                        radioChannel );

        identificar_trama (leer_tramaEM260());
        break;
    case 0x02:
        retardo_ms(500);
        break;
    }
} while ( conectado!=1);
contador=0x000000;
while ( ( contador!=0x058C74 ) & ( conectado==true ) )
{
    //act_PIR=comprobar_PIR ();
    if (!CHK_nHOST)
    {
        EZcallback();
        identificar_trama(leer_tramaEM260());
    }
    contador++;
}
if ( conectado==true )
{
    leer_temp    ();
    leer_presion ();
    ordenar_datos ();
    EZsendUnicast( EMBER_OUTGOING_DIRECT,
                  idCOORDINADOR,
                  0x0000,
                  0x0000,
                  0x01,
                  EndPointID,
                  0x0000,
                  0x0000,
                  SEQmsg++,
```

```

                                0x64,
                                0x06,
                                TxData);
    identificar_trama(leer_tramaEM260());
    TempData[0]= 0x00;
    TempData[1]= 0x00;
    PresionData[0]= 0x00;
    PresionData[1]= 0x00;
}
//act_PIR=false;
}
}

void modulo_init()
{
    DDRB = 0x2D;      // SCK, MOSI, nSSEL y Enable(PIR) salida, MISO entrada
    DDRC = 0x16;      // PMOS(MP3H6), nReset, nWAKE (EM260) salida, RESET,
                    // Vout (MP3H6) y nHOST_INT entrada
    DDRD = 0x70;      // Led1, Led2, CS(TMP121A) salida, Pir y SW1 entrada

    SPCR = 0x51;      // Se configura el puerto SPI, Enable SPI, Master, clock rate
                    // (f/4),CPOL=0, CPHA=0 (Mode 0)

    ADCSRA = 0x83;    // Se configura conversor A/D, Enable A/D, prescaler 8
    ADMUX = 0x45;      // MUX ADC5(PC5), ADLAR=1, resultado almacenado a la
                    // izq. con el bit más signifivo en ADCH

    OFF_PIR;
    OFF_PRESSION;
    OFF_TEMPERA;

    CLR_nRESET_EM;    // El EM260 inicialmente reseteado.
    SET_nHOST;
    SET_nSSEL;
    SET_nWAKE;
    SET_SCLK;
    SET_MOSI;

```

```
    LED1OFF;
    LED2OFF;

    SEQ = 0x01;
    SEQmsg = 0x01;
    contador = 0x000000;
    //act_PIR = false;
    act_PIR = 0x02; //Deshabilitado
    conectado = false;
    verEZSP=0x02;
    radioChannel = 0x10;
    radioTxPower = 0x03;
    PanId = 0xABCD;
    EndPointID = 0x01; //Identificador del nodo
}

void reset_ember()
{
    CLR_nRESET_EM;           // Se activa nRESET (poniendolo a 0)
    retardo_us(0x10);
    SET_nRESET_EM;           // Se desactiva nRESET (poniendolo a 1) nRESET=PC2
    retardo_ms(0x0064);
    while(CHK_nHOST)          // Se espera por la interrupción nhost_int
    ;
    retardo_us(0x10);
    EZcallback();             // Se envía un callback para finalizar el reset y comprobar
    identificar_trama(leer_tramaEM260()); // que se ha producido adecuadamente
}

void config_EM260 ()          //Configuraciones iniciales del EM260
{
    EZSP_version();
    identificar_trama(leer_tramaEM260()); // Comprobamos la versión del EZSP
    EZsetConfigurationValue(0x17,0x0001); // Se activa el Boost mode
    identificar_trama(leer_tramaEM260());
    EZsetConfigurationValue(0x0D,0x0000); // Se deshabilita la seguridad
}
```

```

    identificar_trama(leer_tramaEM260());
    EZsetConfigurationValue(0x18,0x0000);    // Permite al nodo retransmitir mensajes
    identificar_trama(leer_tramaEM260());
    EZsetConfigurationValue(0x13,0x00FF);    // Se habilita un tiempo si poll de unos
    identificar_trama(leer_tramaEM260());    //256 minutos aproximadamente.
}

```

void ordenar_datos()

```

{
    TxData[0]=0xAA;
    TxData[1]=TempData[0];
    TxData[2]=TempData[1];
    TxData[3]=PresionData[0];
    TxData[4]=PresionData[1];
    TxData[5]=act_PIR;
}

```

void leer_temp ()

```

{
    ON_TEMPERA;
    LED1ON;
    LED2ON;
    retardo_ms(1000);
    TempData[0]=SPI_Receive();
    TempData[1]=SPI_Receive()>>3;
    OFF_TEMPERA;
    TempData[1]=TempData[1]+(TempData[0]<<5);
    TempData[0]=TempData[0]>>3;
    LED1OFF;
    LED2OFF;
}

```

void leer_presion ()

```

{
    ON_PRESION;
    LED1ON;
}

```

```
    LED2ON;
    retardo_ms(500);
    ADCSRA = 0xC0;           // Se habilita el conversor Analógico-Digital (110)
    while (ADCSRA&0x40)      // Se espera a que la conversión este completada
    ;                          // se descarta la primera conversión
    ADCSRA = 0xC0;           // Se vuelve a habiliar la el comienzo de la conversión
    while (ADCSRA&0x40)      // Se espera a que la conversión esté completada
    ;
    ADCSRA = 0x00;           // Se detiene el conversor (000)
    PresionData[0]=ADCH;
    PresionData[1]=ADCL;
    OFF_PRESION;
    LED1OFF;
    LED2OFF;
}

u8 comprobar_PIR ()
{
    if (CHK_PIR!=0x00)
    {
        return true;
    }
    else
    {
        return false;
    }
}

void errores (u8 idError)
{
    LED1OFF;
    LED2OFF;
    switch (idError)
    {
        case 0x01:
            intermitente_leds(ONLY_RED,0x01);
            break;
    }
}
```

```

case 0x02:
    intermitente_leds(RED_GREEN_SIC,0x01);
    reset_ember ();
    config_EM260 ();
    break;
case 0x03:
    intermitente_leds(ONLY_RED,0x02);
    break;
case 0x04:
    intermitente_leds(ONLY_RED,0x03);
    desconexion_red ();
    break;
case 0x05:
    intermitente_leds(ONLY_RED,0x04);
    break;
case 0x06:
    intermitente_leds(ONLY_RED,0x05);
    desconexion_red ();
    break;
case 0x07:
    intermitente_leds(RED_GREEN_ASIC,0x01);
    desconexion_red ();
    break;
case 0x08:
    intermitente_leds(RED_GREEN_ASIC,0x02);
    break;
case 0x09:
    intermitente_leds(ONLY_RED,0x06);
    break;
case 0x0A:
    LED1ON;
    intermitente_leds(ONLY_GREEN,0x01);
    LED1OFF;
    desconexion_red ();
    break;
case 0x0B:

```



```
        LED1ON;
        intermitente_leds(ONLY_GREEN,0x02);
        LED1OFF;
        break;
    case 0x0C:
        intermitente_leds(RED_GREEN_ASIC,0x03);
        break;
    case 0x0D:
        LED1ON;
        intermitente_leds(ONLY_GREEN,0x03);
        LED1OFF;
        desconexion_red ();
        break;
    case 0x0E:
        intermitente_leds(RED_GREEN_SIC,0x02);
        desconexion_red ();
        break;
    }
}

void desconexion_red ()
{
    if (conectado==true)
    {
        EZleaveNetwork();
        identificar_trama(leer_tramaEM260());
    }
}

void intermitente_leds (u8 tipo, u8 repeticiones)
{
    u8 i;
    LED1OFF;
    LED2OFF;
    for (i=0;i<repeticiones;i++)
    {
        switch (tipo)
```

```

    {
        case ONLY_RED:
            LED1ON;
            retardo_ms(1000);
            LED1OFF;
            retardo_ms(1000);
            break;
        case ONLY_GREEN:
            LED2ON;
            retardo_ms(1000);
            LED2OFF;
            retardo_ms(1000);
            break;
        case RED_GREEN_SIC:
            LED1ON;
            LED2ON;
            retardo_ms(1000);
            LED1OFF;
            LED2OFF;
            retardo_ms(1000);
            break;
        case RED_GREEN_ASIC:
            LED1ON;
            LED2OFF;
            retardo_ms(1000);
            LED1OFF;
            LED2ON;
            retardo_ms(1000);
            LED2OFF;
            break;
    }
}
retardo_ms(2000);
}

```

u8 leer_tramaEM260()

```
{
u8 dato_temp;
    u8 i;
    retardo_us(0x20);
    do
    {
        dato_temp=SPI_Receive();
    }while(dato_temp==0xFF);
    for (i=0;(i<155); i++)
    {
        SPI_trama[i]=dato_temp;
        if (dato_temp==0xA7)
        {
            SET_nSSEL;           // Desactivamos nSSEL (poniendolo a 1).
            retardo_ms(0x04);
            return false;
        }
        else
        {
            dato_temp=SPI_Receive();
        }
    }
    SET_nSSEL;           // Se desactiva nSSEL (poniendolo a 1)
    retardo_ms(0x04);    //mínimo 1 ms, tiempo necesario para que el EM260
    return true;         // este listo para procesar otro comando.
}

void SPI_Transmit(u8 cData)    // Función que envía un byte mediante el puerto SPI
{
    SPDR = cData;             // Comienza la transmisión
    while(SPI_BUSY);          // Espera a que la transmita (SPIF se pone a 1 cuando se
}                               // completa).

u8 SPI_Receive()
{
    SPDR = 0xFF;              // Se genera los pulsos de reloj del SPI.
```

```

while(!SPI_RDY);          // Se espera a que la transmisión se complete
return SPDR;              // Se devuelve el dato leído
}

void identificar_trama(u8 error_lec)
{
    if ( error_lec == false )
    {
        if ( SPI_trama[0] != 0xFE )
        {
            switch(SPI_trama[0])
            {
                case 0x00:
                    errores(0x02);
                    break;          // Error de reset (Error 2)
                case 0x01:
                    errores(0x03);
                    break;          // Error de oversized (Error 3)
                case 0x02:
                    errores(0x04);
                    break;          // Error de transacción abortada(Error4)
                case 0x03:
                    errores(0x05);  // Error perdido terminador 0xA7
                    break;          // (Error 5)
                case 0x04:
                    errores(0x06);  // Error de comando SPI no soportado
                    break;          // (Error 6)
                default:
                    errores(0x07);
                    break;          //Error desconocido (Error 7)
            }
        }
    }
    else
    {
        switch(SPI_trama[4])
        {

```

```
case 0x00:           // Respuesta de la versión del EZSP
    checkVersionEZSP();
    break;
case 0x05:           // Respuesta del comando Nop
    break;
case 0x07:           // Indica que no hay callback pendientes
    break;
case 0x18:           // Comando que recibe el estado del
    estado_nodo();    // nodo en la red
    break;
case 0x19:           // Callback que indica el estado de la
    estado_red();     // red.
    break;
case 0x20:           // Comprueba si la red está caída.
    EZSPStatus ();
    if (SPI_trama[5] == EZSP_SUCCESS)
    {
        while(CHK_nHOST);
        EZcallback();
        identificar_trama(leer_tramaEM260());
    }
    break;
case 0x1F:           // Respuesta al comando JoinNetwork
    EZSPStatus ();
    if (SPI_trama[5] == EZSP_SUCCESS)
    {
        while(CHK_nHOST);
        EZcallback();
        identificar_trama(leer_tramaEM260());
    }
    break;
case 0x34:           // Respuesta al comando sendUnicast
    EZSPStatus ();
    if (SPI_trama[5] == EZSP_SUCCESS)
    {
        while(CHK_nHOST);
        EZcallback();
```

```
        identificar_trama(leer_tramaEM260()));
    }
    break;
case 0x3F:          // Respuesta messageSentHandler
    EZSPStatusMSG (); // (si ha llegado correctamente el msg)
    break;
case 0x45:          // Se ha recibido un mensaje
    EmberIncomingMessage(); // Mostramos el mensaje recibido
    break;          // mediante los diodos leds
case 0x53:          // Respuesta al comando
    EZSPStatus ();   // setConfigurationValue
    break;
case 0x58:          // Comando EZSP invalido
    errores(0x08);    //(Error 8)
    break;
default:
    errores(0x0C);    // Error de trama desconocida
    break;           // ( Error 12)
    }
    }
}
else
{
    errores(0x01);    // No se encontro terminador de trama.
}
// o trama muy larga ( Error 1)
limpiar_trama();
}

void limpiar_trama()
{
    u8 i;
    for (i=0; i<160; i++)
    {
        SPI_trama[i]=0x00;
    }
}
```

```
void estado_nodo()
{
    switch ( SPI_trama[5] )
    {
        case EMBER_NO_NETWORK:
            conectado=0;
            LED2OFF;                // Desactivamos led verde
            errores(0x0B);           // No hay red (Error 11)
            retardo_ms(1000);
            break;
        case EMBER_JOINING_NETWORK:
            conectado=2;
            break;
        case EMBER_JOINED_NETWORK:
            conectado=1;
            LED2ON;                  //Activamos led verde
            break;
        case EMBER_JOINED_NETWORK_NO_PARENT:
            conectado=1;
            LED2ON;                  //Activamos led verde
            break;
        case EMBER_LEAVING_NETWORK:
            conectado=0;
            LED2OFF;                //Desactivamos led verde
            retardo_ms(2000);
            break;
        default:
            conectado=0;
            LED2OFF;                //Desactivamos led verde
            errores(0x0E);           //Error al intentar la conexión (Error14)
            retardo_ms(1000);
            break;
    }
}
```

```
void estado_red ()
```

```

{
    switch ( SPI_trama[5] )
    {
        case EMBER_NETWORK_UP:
            conectado=1;
            LED2ON;           //Activamos led verde
            break;
        case EMBER_NETWORK_DOWN:
            conectado=0;
            LED2OFF;           // Desactivamos led verde
            errores(0x0B);      // No hay red creada (Error 11)
            break;
        case EMBER_JOIN_FAILED:
            conectado=0;
            LED2OFF;           // Desactivamos led verde
            break;
        case EMBER_JOINED_NETWORK_NO_PARENT:
            conectado=1;
            LED2ON;           //Activamos led verde
            break;
        case EMBER_MOVE_FAILED:
            conectado=0;
            LED2OFF;           // Desactivamos led verde
            break;
        default:
            conectado=0;
            LED2OFF;           // Desactivamos led verde
            break;
    }
}

```

// Procesamos el mensaje entrante y se muestra mediante los diodos LEDs

void EmberIncomingMessage()

```

{
    u8 i;
    u8 j;

```



```
    u8 cuenta;
    LED1OFF;
    LED2OFF;
    intermitente_leds(RED_GREEN_SIC,0x03);
    for (i=0; i<SPI_trama[23]; i++)
    {
        cuenta=SPI_trama[24+i]>>4;
        for(j=0; j<cuenta; j++)
        {
            intermitente_leds(ONLY_GREEN,0x01);
        }
        intermitente_leds(ONLY_RED,0x01);
        cuenta=SPI_trama[24+i]&0x0F;
        for(j=0; j<cuenta; j++)
        {
            intermitente_leds(ONLY_GREEN,0x01);
        }
    }
    LED2ON;
}

void checkVersionEZSP ()
{
    if( SPI_trama[5] != verEZSP )
    {
        errores(0x09);          // La versión del EZSP no es soportada por el EM260
    }                           // (Error 9)
    else
    {
        intermitente_leds(ONLY_GREEN,0x01);
    }
}

// Comprobamos si se ha llegado el comando correctamente
void EZSPStatus ()
{
```

```
if( SPI_trama[5] == EZSP_SUCCESS )
{
    intermitente_leds(ONLY_GREEN,0x01); //Comando recibido por el
}                                     // EM260 correctamente
else
{
    errores(0x0A);      //Error al intentar ejecutar el comando enviado (Error 10)
}
}
```

// Comprobamos si se ha llegado el MSG correctamente

void EZSPStatusMSG ()

```
{
    if ( SPI_trama[20] == EZSP_SUCCESS )
    {
        intermitente_leds(ONLY_GREEN,0x02);          //Comando recibido por
    }                                                  // el EM260 correctamente
    else
    {
        errores(0x0D);      //Error en la llegada del msg (Error 13)
    }
}
```

void retardo_ms(u16 miliseg){

```
    while(miliseg!=0x0000)
    {
        miliseg=miliseg-1;
        __delay_cycles(0x1F34);
    }
}
```

void retardo_us(char useg)

```
{
    while(useg!=0x00)
    {
        useg=useg-1;
    }
}
```

```
        __delay_cycles(0x06);
    }
}

Bajo_consumo()
{
    EZnop();                // Comando nop para pasar al modo
    identificar_trama(Leer_tramaEM260()); // bajo consumo en el EM260
    start_timer ();          // Activamos timer con el tiempo en espera
    PCMSK1 = 0x02;           // Mascara PCINT9, señal nhost_int y
    PCICR = 0x02;            // se habilita la interrupción de nhost_int
    __enable_interrupt();     // Se habilitan las interrupciones del ATmega168
    SMCR= 0x01;              // Se selecciona y habilita el modo Idle
    __sleep();               // Activamos el modo de bajo consumo Idle
}

void despertar_EM260 ()
{
    CLR_nWAKE;
    retardo_us(0x40);
    while(CHK_nHOST)
    ;
    SET_nWAKE;
}

void start_timer ()
{
    TCNT1= 0xA000;           // Contador a A000.
    TIMSK1 = 0x01;           // Se Habilita interrupción por desbordamiento.
    TCCR1B = 0x03;           // Prescaler a 64 y activación de timer.
}

void stop_timer ()
{
    TCCR1B = 0x00;           // Stop del timer.
    TIMSK1 = 0x00;           // Se deshabilita la interrupción por desbordamiento
}
```

```

/////////////////////////////////////////////////////////////////
//////// Interrupciones empleadas en las modificaciones realizadas en el firmware del //////////
//////// presente PFC para reducir el consumo de corriente ///////////////////////////////////
/////////////////////////////////////////////////////////////////

```

// Interrupción por desbordamiento del Timer

#pragma vector = TIMER1_OVF_vect

__interrupt void interrupcion_timer(void)

```

{
    __disable_interrupt();    //Se deshabilitan las interrupciones del ATmega168
    stop_timer();             // Se para el timer.
    despertar_EM260 ();        // Se saca al EM260 del modo de bajo consumo.
}

```

// Interrupción de la señal nHost_int (PCINT9)

#pragma vector = PCINT1_vect

__interrupt void interrupcion_nhostint(void)

```

{
    __disable_interrupt();    //Se deshabilitan las interrupciones del ATmega168
    stop_timer();             // Se para el timer.
    PCICR = 0x00;             // Se deshabilita la interrupción PCINT9.
    if (!CHK_nHOST)           // Se comprueba si el EM260 tiene alguna trama.
    {
        EZcallback();
        identificar_trama(leer_tramaEM260());
    }
}

```

```

/////////////////////////////////////////////////////////////////
//// Funciones que envían las tramas de los comandos empleados en el presente PFC al EM260 ////
/////////////////////////////////////////////////////////////////

```

// Comando para especificar de la versión del EZSP a utilizar (0x02).

void EZSP_version()

```
{
    CLR_nSSEL;
    retardo_us(0x10);
    SPI_Transmit( 0xFE );
    SPI_Transmit( 0x04 );
    SPI_Transmit( SEQ++);
    SPI_Transmit( 0x00 );
    SPI_Transmit( 0x00 );
    SPI_Transmit( verEZSP );
    SPI_Transmit( 0xA7 );
}

// Comando que permite introducir valores de configuración en el EM260
void EZsetConfigurationValue(u8 Id,u16 value)
{
    CLR_nSSEL;
    retardo_us(0x10);
    SPI_Transmit( 0xFE );
    SPI_Transmit( 0x06 );
    SPI_Transmit( SEQ++);
    SPI_Transmit( 0x00 );
    SPI_Transmit( 0x53 );
    SPI_Transmit( Id );
    SPI_Transmit( value & 0xFF );
    SPI_Transmit( (value >> 8) & 0xFF );
    SPI_Transmit( 0xA7 );
}

// Comando que realiza un callback al EM260.
void EZcallback()
{
    CLR_nSSEL;
    retardo_us(0x10);
    SPI_Transmit( 0xFE );
    SPI_Transmit( 0x03 );
    SPI_Transmit( SEQ++);
    SPI_Transmit( 0x00 );
```

```

        SPI_Transmit( 0x06 );
        SPI_Transmit( 0xA7 );
    }

// Comando que permite obtener el estado de la red,
void EZnetworkState()
{
    CLR_nSSEL;
    retardo_us(0x10);
    SPI_Transmit( 0xFE );
    SPI_Transmit( 0x03 );
    SPI_Transmit( SEQ++ );
    SPI_Transmit( 0x00 );
    SPI_Transmit( 0x18 );
    SPI_Transmit( 0xA7 );
}

// Comando para unirse a una red ya creada.
void EZjoinNetwork ( u8 nodeType,
                    u8 *ExPanId,
                    u16 PanId,
                    s8 radioTxPower,
                    u8 radioChannel )
{
    CLR_nSSEL;
    retardo_us(0x10);
    SPI_Transmit ( 0xFE );
    SPI_Transmit ( 0x10 );
    SPI_Transmit ( SEQ++ );
    SPI_Transmit ( 0x00 );
    SPI_Transmit ( 0x1F );
    SPI_Transmit ( nodeType );
    SPI_Transmit ( ExPanId[0] );
    SPI_Transmit ( ExPanId[1] );
    SPI_Transmit ( ExPanId[2] );
    SPI_Transmit ( ExPanId[3] );
}

```

```
SPI_Transmit ( ExPanId[4] );
SPI_Transmit ( ExPanId[5] );
SPI_Transmit ( ExPanId[6] );
SPI_Transmit ( ExPanId[7] );
SPI_Transmit ( PanId & 0xFF );
SPI_Transmit ( (PanId >> 8) & 0xFF );
SPI_Transmit ( radioTxPower );
SPI_Transmit ( radioChannel );
SPI_Transmit ( 0xA7 );
}

// Comprobación de la versión del protocolo SPI empleado.
u8 protocol_version(u8 version)
{
    u8 error;
    CLR_nSSEL;          /* Activamos nSSEL (poniendolo a 0) */
    SPI_Transmit(0x0A);  /* Se transmite el comando SPI Protocol Version Request */
    SPI_Transmit(0xA7);  /* Transmitimos el frame terminator A7 */
    error=leer_tramaEM260();
    if (error==true || (SPI_trama[0]!=version))
    {
        errores (0x01);    //No se encontro terminador de trama o
        return false;      // trama muy larga. (Error 1)
    }
    else
    {
        return true;
    }
}

// Comando para realizar el envios de mensajes.
void EZsendUnicast ( u8 MsgType,
                    u16 DestAdr,
                    u16 ProfileId,
                    u16 ClusterId,
                    u8 SrcEP,
                    u8 DstEP,
```

```
        u16 ApsOptions,
        u16 GroupId,
        u8 MsgSeq,
        u8 MsgTag,
        u8 MsgLen,
        u8 *TxBuf)
{
    u8 i;
    CLR_nSSEL;          /* Activamos nSSEL (poniendolo a 0) */
    retardo_us(0x10);
    SPI_Transmit ( 0xFE );
    SPI_Transmit ( 19+MsgLen );
    SPI_Transmit ( SEQ++ );
    SPI_Transmit ( 0x00 );
    SPI_Transmit ( 0x34 );
    SPI_Transmit ( MsgType );
    SPI_Transmit ( DestAdr & 0xFF );
    SPI_Transmit ( (DestAdr >> 8) & 0xFF );
    SPI_Transmit ( ProfileId & 0xFF );
    SPI_Transmit ( (ProfileId >> 8) & 0xFF );
    SPI_Transmit ( ClusterId & 0xFF );
    SPI_Transmit ( (ClusterId >> 8) & 0xFF );
    SPI_Transmit ( SrcEP );
    SPI_Transmit ( DstEP );
    SPI_Transmit ( ApsOptions & 0xFF );
    SPI_Transmit ( (ApsOptions >> 8) & 0xFF );
    SPI_Transmit ( GroupId & 0xFF );
    SPI_Transmit ( (GroupId >> 8) & 0xFF );
    SPI_Transmit ( MsgSeq );
    SPI_Transmit ( MsgTag );
    SPI_Transmit ( MsgLen );
    for (i=0; i<MsgLen; i++)
    {
        SPI_Transmit( TxBuf[i]);
    }
    SPI_Transmit( 0xA7 );
}
```



```
}

// Comando para abandonar la red que está conectado.
void EZleaveNetwork()
{
    CLR_nSSEL;
    retardo_us(0x10);
    SPI_Transmit( 0xFE );
    SPI_Transmit( 0x03 );
    SPI_Transmit( SEQ++ );
    SPI_Transmit( 0x00 );
    SPI_Transmit( 0x20 );
    SPI_Transmit( 0xA7 );
}

// Mediante el comando nop pasamos al modo de bajo consumo Power Down en el EM260.
void EZnop()
{
    CLR_nSSEL;
    retardo_us(0x10);
    SPI_Transmit( 0xFE );
    SPI_Transmit( 0x03 );
    SPI_Transmit( SEQ++ );
    SPI_Transmit( 0x02 );           // Power Down
    SPI_Transmit( 0x05 );
    SPI_Transmit( 0xA7 );
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//////// Función main empleada en las modificaciones realizadas en el firmware del //////////
//////// presente PFC para reducir el consumo de corriente //////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

int main ( void )
{
    modulo_init();
    reset_ember ();
}
```

```
config_EM260 ();
//ON_PIR;
while(1)
{
    do
    {
        if (!CHK_nHOST)
        {
            EZcallback();
            identificar_trama(leer_tramaEM260());
        }
        switch (conectado)
        {
            case true:
                LED2ON;
                break;
            case false:
                EZjoinNetwork(EMBER_END_DEVICE,
                               ExPanId,
                               PanId ,
                               radioTxPower,
                               radioChannel);
                identificar_trama(leer_tramaEM260());
                break;
            case 0x02:
                retardo_ms(500);
                break;
        }
    }
    Bajo_consumo();
    if ( conectado==true )
    {
        leer_temp  ();
        leer_presion ();
        ordenar_datos ();
        EZsendUnicast(EMBER_OUTGOING_DIRECT,
```

```
        idCOORDINADOR,
        0x0000,
        0x0000,
        0x01,
        EndPointID,
        0x0000,
        0x0000,
        SEQmsg++,
        0x64,
        0x06,
        TxData);

    identificar_trama(leer_tramaEM260());
    TempData[0]= 0x00;
    TempData[1]= 0x00;
    PresionData[0]= 0x00;
    PresionData[1]= 0x00;
}
//act_PIR=false;
}
}
```


Anexo E

Contenido del CD del PFC

A continuación se especifica el contenido de cada uno de los directorios existentes en el CD de contenidos del presente Proyecto Fin de Carrera.

\Memoria

El este directorio se encuentra localizada la memoria completa del Proyecto Fin de Carrera.

\Datasheets

En este directorio se incluyen las hojas de características de los componentes que ha sido necesario consultar durante el desarrollo de este Proyecto Fin de Carrera.

\Fab

En este directorio se incluye los esquemáticos completos, los ficheros gerber y una lista de componentes empleados en el desarrollo del presente Proyecto Fin de Carrera.