UNIVERSIDAD POLITÉCNICA DE CARTAGENA

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES

DEPARTAMENTO DE TECNOLOGIA ELECTRÓNICA

PROYECTO FIN DE CARRERA

INGENIERO TÉCNICO INDUSTRIAL EN ELECTRONICA INDUSTRIAL

"DISEÑO DE UN PROTOTIPO INALÁMBRICO PARA AJUSTAR LA POSICIÓN DE LOS DENDRÓMETROS DE UNA EXPLOTACIÓN AGRONÓMICA"

DIRECTORES: D. Juan Antonio López Riquelme

D. Juan Suardíaz Muro

AUTORA: Rita Patricia Narváez Narváez

CURSO 2012-2013

AGRADECIMIENTOS

- A Juan Antonio por su ayuda y preocupación constante, por estar siempre disponible, aún en periodo de vacaciones y sobre todo por la paciencia.

- A Juan por ayudarme a realizar la documentación de este proyecto.

- A mi madre, por tu apoyo incondicional en todo momento, por creer siempre en mí y por ayudarme en los momentos más difíciles.

- A ti, que siempre estás ahí y que cada día me das tu apoyo, tu compresión y fuerza para seguir adelante.

- A ti Manolo, gracias por la ayuda prestada con el video y las revisiones de la documentación.

- A toda mi familia, que aunque está lejos, siempre han estado pendientes de mis estudios.

ACRÓNIMOS

ACK ACKNOWLEDGEMENT (RECONOCIMIENTO)

AES ADVANCED ENCRYPTION STANDARD (ESTÁNDAR DE CIFRADO AVANZADO)

AGC AUTOMATIC GAIN CONTROL

CCA CLEAR CHANNEL ASSESSMENT

DMA DIRECT MEMORY ACCESS (ACCESO DIRECTO A MEMORIA)

EB EVALUATION BOARD (PLACA DE DESARROLLO)

EM EVALUATION MODULE (MÓDULO DE DESARROLLO)

FCS FRAME CHECK SEQUENCE

FS FREQUENCY SYNTHESIZER

FSM FINITE STATE MACHINE

GPRS GENERAL PACKET RADIO SERVICE (SERVICIO GENERAL DE PAQUETES VÍA RADIO)

ISM INDUSTRIAL, SCIENTIFIC AND MEDICAL

JTAG JOINT TEST ACTION GROUP

LCD LIQUID CRYSTAL DISPLAY

LED LIGHT-EMITTING DIODE

LNA AMPLIFICADOR DE BAJO RUIDO

LPF LOW PASS FILTER (FILTRO PASO BAJO)

LPW LOW POWER WIRELESS

MAC MEDIUM ACCESS CONTROL

MCU MICRO CONTROLLER

MIPS MILLONES DE INSTRUCCIONES POR SEGUNDO

RF RADIO FREQUENCY (RADIO FRECUENCIA)

SMA SubMiniature versión A

SoC System-on-chip

SPI Serial Peripheral Interface

TI Texas Instruments

UART Universal Asynchronous Receive Transmit

UMTS Universal Mobile Telecommunications System (Sistema universal de telecomunicaciones móviles)

USB Universal Serial Bus

ÍNDICE

CAPÍTULO 1

INTRODUCCIÓN

1. Planteamiento inicial del proyecto

Los investigadores del área agronómica de la Universidad Politécnica de Cartagena han realizado diversos estudios de riego deficitario y riego deficitario controlado en las instalaciones de la Estación Experimental Agroalimentaria "Tomás Ferro" durante los últimos años. Para llevar a cabo estos estudios han desplegado redes cableadas de sensores con una arquitectura centralizada, donde el elemento de almacenamiento y adquisición de datos es un *datalogger* del fabricante Campbell Scientific. En relación a los sensores, han utilizado sensores para monitorizar el estado del ambiente, del suelo y de la planta, entre los que caben destacar los siguientes: tensiómetros, sondas FDR, termoradiómetros, y dendrómetros.

Durante el año 2012, se desplegó una red inalámbrica de sensores en paralelo con uno de los sistemas tradicionales de riego deficitario controlado que están instalados en la Estación Experimental Agroalimentaria "Tomás Ferro". Esta red se desplegó con objeto de evaluar las ventajas e inconvenientes de ambas alternativas. Por ello, se utilizaron sensores idénticos a los empleados en la arquitectura cableada.

El dendrómetro es un instrumento de precisión que se utiliza para medir las pequeñas variaciones (del orden de micrómetros) en el diámetro de un tronco, rama o fruto. Vienen siendo utilizados para analizar el estado general de la planta y permiten estimar el estado de estrés del cultivo y el correcto crecimiento de la planta, entre otros.

Debido a que el rango de medida de un dendrómetro es de pocos milímetros, es necesario reajustar la posición de los mismos periódicamente. Este ajuste se realiza

habitualmente de forma aproximada situando la aguja en el centro del cuerpo del sensor de tipo LVDT.

En este proyecto se propone diseñar e implementar un sistema inalámbrico que permita reajustar la posición de los dendrómetros de una forma más precisa y rápida, facilitando así el mantenimiento de los dendrómetros instalados en la parcela que se está monitorizando. Este sistema se realizará sobre la red inalámbrica existente, de forma que el dispositivo a diseñar se pueda comunicar con dicha red, establecer un modo de configuración y recibir las lecturas de los dendrómetros en los dispositivos. Al tratarse de un dispositivo inalámbrico se facilita enormemente las tareas de mantenimiento, ya que no es necesario realizar la conexión con el dendrómetro que se está reajustando.

Para facilitar las tareas de diseño e implementación del sistema inalámbrico se implementará una red inalámbrica de sensores usando placas de desarrollo. Estás placas de desarrollo contendrán los mismos elementos principales (micro-controlador y módulo de radio), que los existentes en los nodos sensores inalámbricos instalados en la parcela de almendros objeto de estudio. De esta forma, se podrá simular el funcionamiento de la red inalámbrica en el laboratorio y facilitar las tareas comentadas anteriormente.

2. Objetivos

El objetivo principal de este proyecto es el diseño de un prototipo inalámbrico que permita cada cierto intervalo de tiempo ajustar los dendrómetros colocados en los árboles de almendro y así conocer su estado hídrico. Para que el objetivo principal se pueda desarrollar correctamente, se debe cumplir con una serie de subobjetivos:

- ✓ Realizar un estudio del estado del arte que permita conocer las aplicaciones de las redes inalámbricas de sensores en la agricultura.
- ✓ Proponer el hardware y el software adecuado que permita resolver la problemática propuesta en el objetivo principal.
- ✓ Implementar el software necesario para conseguir el correcto funcionamiento de la red inalámbrica.
- ✓ Realizar las pruebas necesarias en el laboratorio que permita validar el software implementado.

3. Fases del proyecto

Para el correcto desarrollo del proyecto se han establecido las siguientes fases que se exponen a continuación:

1. Búsqueda y selección de nodo sensores que permita interpretar de forma eficiente los datos obtenidos por los dendrómetros.
2. Estudio del sistema de compilación IAR Embedded Workbench, con el que se programará la red de nodo sensores.
3. Desarrollo de un software dedicado a la monitorización de los datos procedentes de los dendrómetros.
4. Montaje de un prototipo demostrativo.
5. Pruebas y puesta a punto.
6. Redacción del documento final de memoria de proyecto.

4. Desarrollo de la memoria

La memoria de este proyecto se divide en cinco capítulos, en los que se describe todo lo comentado en el apartado anterior. A continuación se muestra un pequeño avance, desarrollado posteriormente en cada uno de ellos.

- Capitulo 1. Introducción.
  En este capítulo se presenta de forma general de que trata este proyecto, indicando brevemente las fases en las que está dividido.

- Capitulo 2. Estado del arte.
  Este capítulo presenta las redes de sensores inalámbricas, indicando sus características, elementos, protocolos que utiliza, sus posibles aplicaciones en distintos ámbitos, etc. Se explicará también el concepto de nodo sensores, su clasificación, sus modos de funcionamiento, componentes de los mismos, etc; detallando a la vez varios nodos comerciales e indicando, entre otros cosas, sus características técnicas y posibles usos.

- Capitulo 3. Descripción del sistema.
  Como indica el nombre del capítulo en él se describirán todos los elementos que forman este proyecto, explicando todas sus características, sus usos y todas las posibilidades que ofrece. También se el programa de compilación IAR Embedded Workbench para poner en funcionamiento el prototipo y los programas utilizados en los 4 nodos sensores necesarios para el correcto funcionamiento del proyecto.

- Capitulo 4. Conclusiones y trabajos futuros.
  Como su propio nombre indica, este capítulo será un breve resumen de todo lo conseguido y los resultados obtenidos durante la realización del proyecto, concluyendo todos los trabajos realizados y proponiendo posibles proyectos, trabajos o investigaciones relacionados con las redes de sensores inalámbricas en la toma de datos de dendrómetros.

CAPÍTULO 2

ESTADO DEL ARTE

1. Introducción a las Redes Inalámbricos de Sensores (WSN)

Como se ha expuesto en el capitulo anterior, el principal objetivo de este proyecto consiste en el diseño de una red inalámbrica de sensores, la cual nos pueda monitorizar las variaciones que sufre el diámetro del tronco de los árboles. En consecuencia, se va a desarrollar este capítulo donde se hará una introducción a las Redes Inalámbricas de Sensores (*Wireless Sensor Networks* o WSN por sus siglas en ingles); como sus características, aplicaciones, seguridad y las posibilidades que ofrece el mercado.

1.1. Definición

Una red inalámbrica de sensores (en adelante WSN) es una red inalámbrica compuesta por un conjunto de dispositivos de bajo coste y consumo (nodos), capaces de obtener información de su entorno, procesarla localmente, y comunicarla a través de enlaces inalámbricos hasta un nodo central de coordinación, con el fin de supervisar características físicas o condiciones medioambientales en lugares deseados.

La evolución de redes de sensores tiene su origen en iniciativas militares, de ahí que no sea posible hallar mucha información sobre la fuente de la idea. En la actualidad se usan en la mayoría de espacios donde se necesita un control inalámbrico mediante sensores

como por ejemplo en domótica, medicina o en fábricas industriales entre otros. [Huang, 2003].

Cada "*mote* o nodo" (placa de procesador y transmisión/recepción de radio) está compuesto por varios sensores (luminosidad, temperatura, humedad, etc.), un módulo de comunicación inalámbrica, un microcontrolador y una fuente de energía (habitualmente una batería).

El concepto mote viene del proyecto '*Smart Dust'* ("polvo inteligente") propuesto por científicos de la Universidad de California, en el cual, se pretende conseguir una red inalámbrica de sensores donde los nodos sean de un tamaño minúsculo, parecido al de una "mota de polvo", consiguiendo redes inalámbricas muy potentes, ocupando un espacio mínimo.

Es posible usar los sensores inalámbricos pueden para tener varias capacidades de detección si existen diferentes sensores dentro de la misma red. Por lo que es posible disponer de varios tipos de datos al mismo tiempo, que se transmiten hasta la estación base de la red. Además, al tener un microcontrolador interno dentro de estos motes, estos pueden realizar pequeñas modificaciones o tratamientos de los datos, con el objetivo de reducir el tráfico circulante por la red, puesto que enviaremos sólo los datos o las informaciones estrictamente necesarias.

La tendencia actual consiste en la sustitución de sensores complejos por multitud de pequeños sensores simples, siguiendo el principio fundamental de la no utilización de cables, unida a la mejora de las comunicaciones inalámbricas en la sociedad y a la disminución progresiva en tamaño de los componentes electrónicos. [Akyildiz, 2002]

Por tanto, en general, podemos decir que los elementos que forman la WSN son:

- Sensores de todos los tipos, que toman la información del medio y la convierten en señales eléctricas.

- *'Motes'* o nodos, dispositivos capaces de recoger la información recibida por los sensores y transmitirla a otros motes o a la estación base.

- Pasarelas o '*Gateways'*, elementos que conectan la red de sensores a una red de datos.

- Estación base, elemento que recoge todos los datos, que suele tratarse de un ordenador común o un elemento que lo sustituya.

A partir de aquí, se detallarán todas las posibilidades que ofrece el mercado y se tratará todo lo relacionado con la comunicación, descripción y aplicaciones de los *motes*.

1.2. Características

La creación de motes cada vez más pequeños, más potentes y de menor coste, ha dado la posibilidad de crear redes de sensores con una notable mejora de éstas a lo largo del tiempo. La naturaleza de los nodos y la función que éstos realizan puede ser muy

variada y distinta entre los diferentes motes que forman la red, pero todos llevan una serie de características comunes, que se describen a continuación [Akyildiz,2002]:

- Gran escala: Como es posible comprobar, la red inalámbrica de sensores puede crecer a lo largo del tiempo, pudiendo llegar a contener decenas o cientos de nodos, compuestos a su vez por varios sensores, lo que nos puede dejar varios miles de sensores que controlar, siendo este un valor bastante importante, y que demuestra la escala y las dimensiones que puede tener una red de sensores.

- Comunicación: Los nodos sensores usan comunicación por difusión. Además, los niveles de transmisión de potencia se mantienen muy bajos y existen menos problemas de propagación en comunicaciones inalámbricas a largas distancias [Kahn, 1999].

- Funcionamiento autónomo: Prácticamente durante todo el tiempo que está activo el nodo, éste trabaja de forma autónoma según como esté programado, salvo que necesiten cambio de baterías o porque exista algún problema de funcionamiento concreto. Es uno de los factores más sensibles debido a que tienen que conjugar autonomía con capacidad de proceso.

- Topología: Una de las ventajas de la red de sensores es la topología utilizada, la cual es totalmente aleatoria, y no existe una disposición en la forma de los nodos que deba seguirse estrictamente, es decir, es cambiante. Además los nodos desconocen la posición del resto de nodos de la red. Los sensores tienen que adaptarse para poder comunicar nuevos datos adquiridos [Akyildiz, 2002].

- Recursos limitados: Para conseguir un consumo ajustado, se hace indispensable que el hardware sea lo más sencillo posible, esto nos deja una capacidad de proceso limitada, lo cual supone una desventaja.

- Costes de producción: Dada que la naturaleza de una red de sensores tiene que ser, en número, muy elevada para poder obtener datos con fiabilidad, los nodos sensores, una vez definida su aplicación, son económicos de hacer si son fabricados en grandes cantidades [Rabaey, 2000].

1.3. Requisitos para una WSN

Para que una red de sensores funcione correctamente y se cumplan las características del apartado anterior, se deben cumplir los siguientes requisitos:

- Eficiencia energética: Se trata del apartado más importante. Cuanto menor sea el gasto de energía y menos consuma un nodo, mayor será su vida útil y a la vez mayor tiempo de vida tendrá la red de sensores. Esto se consigue disminuyendo los recursos utilizados, desactivando sensores cuando no se necesiten o disminuyendo los datos que enviará el nodo.

- Seguridad: Los datos se transmiten por medios accesibles por cualquier persona ajena, por tanto debemos tener la red protegida contra intrusos y proteger esos datos o enviarlos encriptados de alguna forma que no puedan ser destruidos por cualquiera.

- Organización automática: Los nodos deben tener una organización interna que permita conocer el nodo al que le envían los datos. Por esto, es necesario que los nodos sepan organizarse de manera automática y además sea capaz la red de reestructurarse al ingresar un nuevo nodo.

- Crecimiento sin pérdida de prestaciones: Es evidente que las redes de sensores van creciendo según el paso del tiempo, así que es necesaria una red en la que, conforme se produzca el crecimiento de ésta, no implique un fuerte descenso en las prestaciones de la red.

- Tiempos de envío: Es importante tener en cuenta que la mayoría de los envíos de datos tienen que realizarse conforme a un tiempo establecido y puede suceder que después de ese tiempo el envío no se realice, o lo haga de forma incorrecta, así pues, se deben tomar las medidas necesarias para que esto no suceda.

- Tolerancia a errores: Como hemos comentado anteriormente, se pueden producir momentos en los que el nodo deje de funcionar, bien por el estado de las baterías, por algún problema en la programación o por algún error de comunicación. Por lo que el objetivo es disminuir las consecuencias del error producido en la red de sensores.

Todas estas características y requisitos deben tenerse muy en cuenta a la hora de desarrollar o llevar a cabo una red de sensores, para conseguir un mejor rendimiento y por tanto una red posible.

1.4. Seguridad en una WSN

Este tipo de red presenta multitud de problemas respecto a la seguridad, sobre todo en este caso en el que se trata de un tipo de red muy utilizado para todo tipo de aplicaciones. Cualquier intruso puede acceder a la información procedente de una red de sensores, debido a que los nodos están normalmente distribuidos en un entorno de fácil acceso, y los canales de comunicación inalámbricos son inherentemente inseguros. En consecuencia, cualquier dispositivo puede escuchar o inyectar paquetes en la red de sensores [Mfbarcell].

Es por lo tanto indispensable incluir unas técnicas de seguridad dentro de los nodos para así proporcionar tanto una mínima protección al flujo de información como una base para la creación de protocolos seguros.

Sin embargo no es una tarea sencilla garantizar la seguridad total, cuando estamos hablando de unos nodos limitados en cuanto a características, almacenamiento y gasto de energía.

Por tanto, se pretende evitar todo tipo de intrusos, ataques y problemas relacionados con la seguridad, que generalmente se producen por los siguientes puntos débiles:

- Las restricciones de hardware y software de los nodos, dificultan la incorporación de mecanismos seguros.
- Al tratarse de una comunicación inalámbrica, cualquiera puede acceder a ella utilizando una simple antena.
- La posibilidad de fallo en los nodos que puede provocar una parada en el funcionamiento de la red.

Para garantizar la seguridad en todo momento se emplea las siguientes técnicas: encriptación de los datos por Criptografía de Clave Secreta (*Secret Key Cryptography* ó SKC por sus siglas en ingles), por Criptografía de clave pública (*Public Key Cryptography* ó el PKC por sus siglas en ingles) y Códigos de Autenticación de Mensajes (*Message Authentication Code* ó MAC por sus siglas en ingles). Estas técnicas no garantizan la seguridad total, pero es necesario utilizarlas para tener la máxima protección [Rodríguez, 2011].

1.5. Arquitectura de una WSN

Tomando como elementos de una red los nombrados anteriormente, podemos distinguir dos tipos de arquitecturas:

- Arquitectura Centralizada

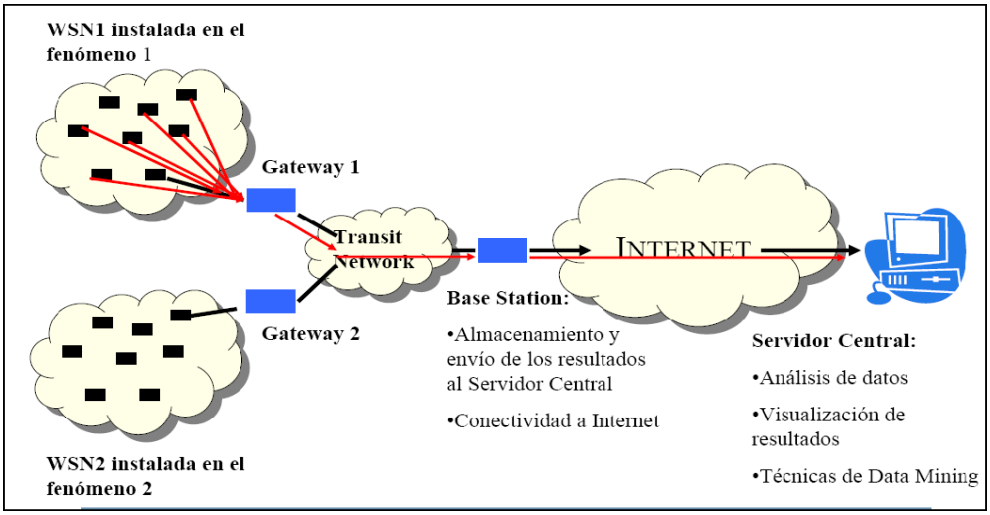


Figura 2.1. Esquema de arquitectura centralizada

En este tipo de red, los nodos encargados de recoger información, enviarán sus datos a la pasarela más cercana que dirigirá el tráfico de la red. Los nodos se comunican únicamente gracias a las pasarelas o '*gateway'*.

Esto produce un problema denominado cuello de botella, que provoca la lentitud de los envíos de los datos y como consecuencia un mayor consumo de energía, disminuyendo el tiempo de vida de los nodos.

- Arquitectura Distribuida

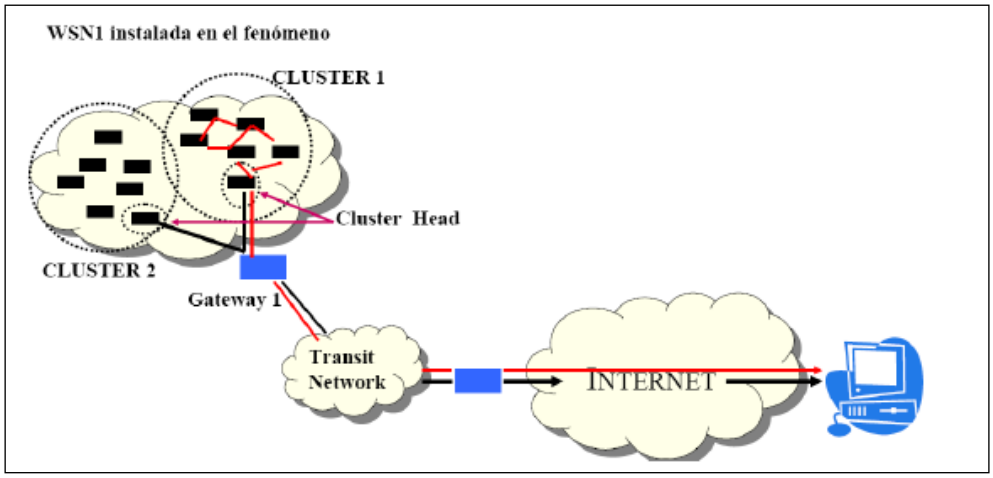


Figura 2.2. Esquema de arquitectura distribuida

Los nodos se comunicarán sólo con otros sensores dentro de su alcance, cooperando y obteniendo una respuesta única que será enviada al '*clúster head*' que se encargará de comunicarse con la estación base. Este tipo de arquitectura evita los problemas de la arquitectura centralizada.

1.6. Aplicaciones de una WSN

Las WSN pueden estar formadas por nodos que tienen conectados diferentes sensores de distintos tipos, como pueden ser sísmicos, magnéticos, térmicos, acústicos, radar, infrarrojos, etc.

Estos tipos de sensores son capaces de monitorizar una gran cantidad de condiciones ambientales y procesos, entre ellos:

- ✓ Temperatura, humedad, presión.
- ✓ Composición del suelo.
- ✓ Condiciones de luz.
- ✓ Movimiento de vehículos.
- ✓ Niveles de ruido.
- ✓ Presencia o ausencia de cierto tipo de objetos…

Además, los nodos sensores pueden adoptar diversas formas de trabajo como actuar en modo continuo, por detección de eventos, por identificación de eventos, toma de datos localizados o como control local de actuadores. Por todo esto y por lo comentado

anteriormente, este tipo de redes tienen gran cantidad de aplicaciones, algunas de ellas son:

- Aplicaciones militares: Entre las más importantes se encuentran la monitorización de fuerzas y equipos enemigos, vigilancia en el campo de batalla, reconocimiento del terreno, detección de ataques biológicos, químicos o nucleares, etc.

- Aplicaciones medioambientales como el seguimiento de animales, monitorización de las condiciones ambientales en cultivos, riego, agricultura de precisión, detección de incendios forestales, detección de inundaciones, etc [Song, 2008].

- Aplicaciones sanitarias: Para la telemonitorización de datos fisiológicos, diagnostico, administración de medicamentos, seguimiento de médicos y pacientes, etc.[Handle, 2006]

- Aplicaciones en el hogar: En este tipo de aplicaciones encontramos la domótica, el control de electrodomésticos, entornos inteligentes, control ambiental, etc.

- Aplicaciones industriales: Seguimiento de vehículos, control de flota, control de inventarios, etc.

- Aplicaciones turísticas: Interactividad en museos y espacios turísticos, control de acceso, etc.

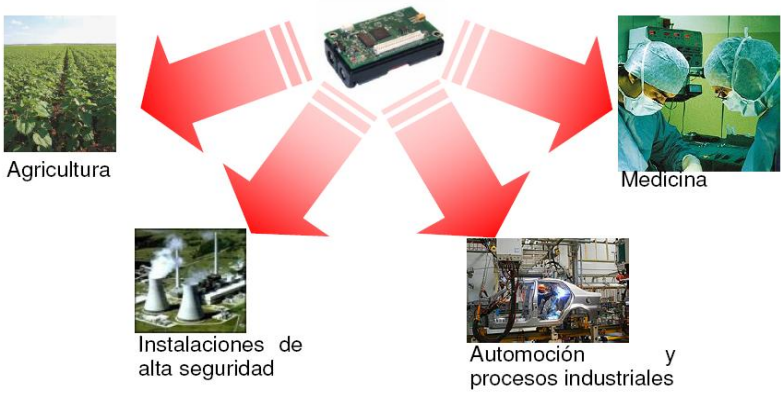


Figura 2.3. Algunas aplicaciones de las WSN

1.7. Nodos Sensores

Las redes de sensores se componen de diversos elementos formando el conjunto total de la red en sí. Estos elementos son las pasarelas, las estaciones base, la tecnología inalámbrica, etc.

Los elementos más importantes de una WSN son los nodos sensores, ya que son la pieza clave que permite interactuar con el entorno, por lo que a continuación se

examinarán sus principales características y los distintos sensores que están apareciendo en el mercado.

Un nodo sensor, también conocido como mote, es un dispositivo capaz de tomar los datos del sensor a través de sus puertas de datos y enviar la información a la estación base. El mote combina las capacidades de recolección, procesado y transmisión de datos en un mismo dispositivo, logrando todo esto con un reducido coste económico, tamaño y consumo de potencia. Otras de sus capacidades, como se verá en secciones posteriores, es adaptar la señal que mide para que pueda interpretarla otro elemento.

Estos nodos llevan incorporados una serie de sensores que son de diferente tipo según el fenómeno o proceso que se desea analizar.

1.7.1. Características generales de los nodos sensores

Con todo lo anterior expuesto, se pueden distinguir las siguientes características:

- Integran sensores de todas clases; es posible medir aceleración, temperatura, movimientos sísmicos, posición global, intensidad de luz, sonido, campos magnéticos, etc.
- Los sensores tienen un coste bajo.
- Normalmente se alimentan de baterías.
- Están limitados en diferentes aspectos.
  - ✓ Disponen unos pocos kilobytes de memoria.
  - ✓ Su procesador opera, como mucho, a unos cuantos Megahercios.
  - ✓ No disponen de gran capacidad de procesamiento.

- Hacen un uso intensivo de la CPU para el procesamiento y de la Radio para enviar y recibir mensajes.
- Alta probabilidad de fallo, debido a las condiciones a las que se exponen y a su bajo coste.
- Son autónomos y se adaptan al entorno.

1.7.2. Componentes básicos de un nodo sensor

- Unidad de Procesado: Se encarga de coordinar al resto de subsistemas, disparando las tareas de recogida de datos y comunicaciones con otros nodos, así como de procesar los datos tomados. Aunque cada vez hay procesadores más pequeños y rápidos, las unidades de procesamiento y almacenamiento en los nodos son recursos escasos.

- Unidad de sensorización: Toma los datos del entorno a través de sus sensores y transforma las medidas analógicas tomadas por el mote, en medidas digitales que el microprocesador puede manejar. Los convertidores A/D adaptan la señal para su posterior utilización, mediante un proceso que suele constar de tres etapas: muestreo, cuantificación y codificación.

- Transceptor: Realiza la tarea de comunicaciones con el resto de nodos**.** Los transceptores pueden ser dispositivos ópticos (activos o pasivos), o dispositivos de radiofrecuencia. En el caso de radiofrecuencia, es necesario una serie de fases: modulación, filtro paso banda, filtrado, demodulación y circuito multiplexor, lo que los hace complejos y caros. Aún así se prefieren transceptores RF en la mayoría de los proyectos porque los paquetes transportados son pequeños y las tasas de transferencia también.

- Unidad de Alimentación: Suministra energía al resto de subsistemas. Normalmente se obtiene de las baterías pero puede estar ayudado de un generador.

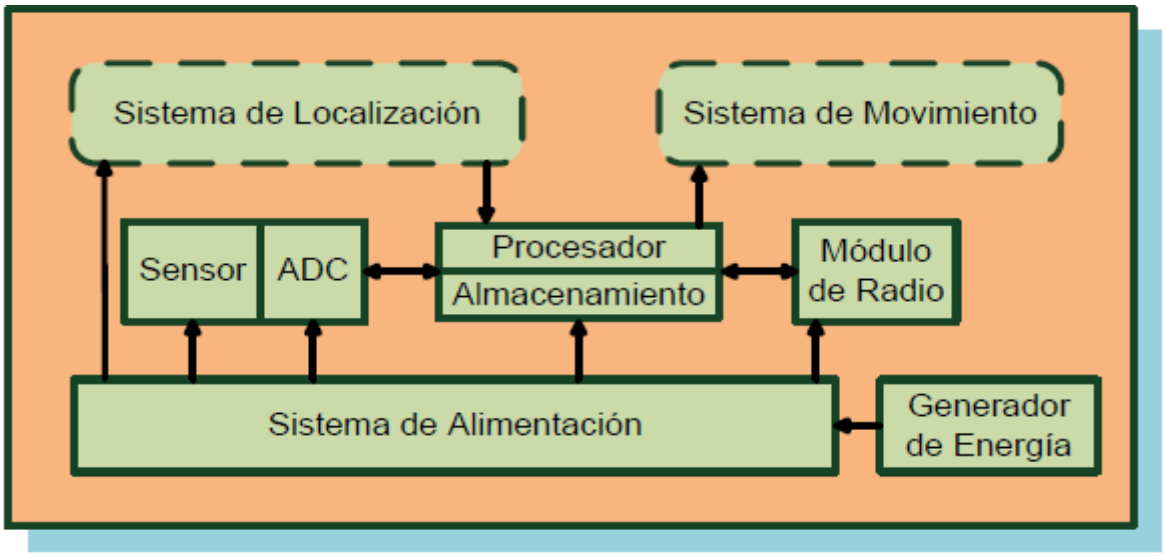


Figura 2.4. Componentes de un nodo sensor o mote

1.7.3. Estados de un nodo sensor

Como se ha explicado con anterioridad, lo motes llevan integrados sensores para medir diferentes aspectos ambientales como la luz, la temperatura o la humedad. También utilizan la radio de forma intensiva para enviar y recibir datos y para procesamiento del microcontrolador.

Pero como también se ha explicado, una de las principales limitaciones es el consumo de energía, por ello se ha desarrollado una estrategia que provoca un ahorro de energía permitiendo una mayor duración de la batería de los motes.

Para ello, un mote tiene tres estados de funcionamiento:

1. *Sleep* (durmiendo): Como su propio nombre indica el mote está en estado dormido. Tenemos que mantener el mote la mayor parte del tiempo en este estado con el fin de que el consumo sea mínimo y permitiendo una mayor duración en la vida de los motes.

2. *Wake-up* (despertando): Es el estado de cambio en el que el nodo se despierta y va a pasar a estado activo. Dicho cambio se produce cuando el sensor recibe algún cambio, estímulo o interrupción programada dentro de sus funciones de

detección y análisis. Uno de los objetivos es minimizar este tiempo para pasar rápidamente al estado de trabajo.

3. ***Active*** (activo): Es el estado activo del mote, donde realiza todo el proceso de adquisición y transmisión de datos. Por supuesto, este tiempo debe ser mínimo para volver cuanto antes al estado '*sleep'*, ya que el consumo será el mayor de los tres que se dan en cada fase.

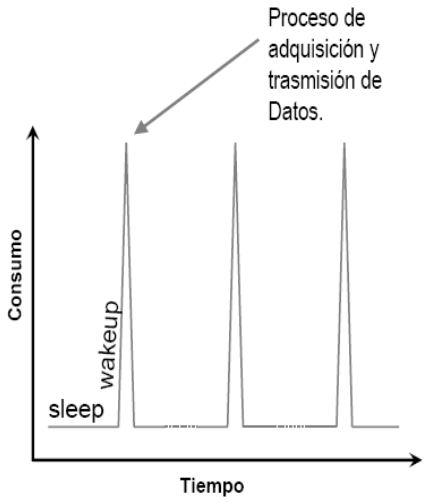


Figura 2.5. Estados de un nodo sensor

1.7.4. Clasificación de los sensores

Se pueden clasificar los sensores de diferentes formas: [Pérez, 2008]

- Atendiendo al fundamento físico: Es decir, según la propiedad física cuya variación produce excitación.
    - Resistivos: Miden variaciones de la resistencia.
    - Capacitivos: Determinan variaciones de la capacidad.
    - Inductivos: Obtienen variaciones de la inductancia electromagnética o magnitud de flujo magnético.
    - Generadores de tensión o intensidad: La magnitud física provoca la generación de tensión o intensidad en el dispositivo, sin necesidad de alimentación externa.

- Atendiendo a la alimentación
    - Activos: Ellos mismos generan una tensión o corriente, no requiriendo por tanto una alimentación externa. Se basan en diferentes efectos: termoeléctrico, piezoeléctrico, etc.
    - Pasivos: Requieren de una alimentación o excitación externa para generar una señal.

- Atendiendo a la salida.

    - Analógicos: La salida del transductor es un nivel de tensión o intensidad que varía de forma continua con la variable a medir dentro del rango de medida del transductor. Suelen emplearse valores normalizados a 0-10V y 4-20 mA.
    - Digitales: La salida está codificada mediante un código binario o en forma de pulsos. Son codificaciones habituales la binaria, Gray, el BCD, etc
    - Todo-Nada: Se consideran un caso particular de los digitales. La salida sólo presenta dos estados: activa o no activa.
      Detectores de presencia…

- Atendiendo a la magnitud a medir: posición, velocidad, aceleración, temperatura, fuerza, nivel, presión, etc.

Por otro lado, podemos establecer una serie de factores que evalúan y catalogan también los tipos de nodos sensores. Estos factores serían: energía, flexibilidad, robustez, seguridad, comunicación, computación, sincronización, tamaño y coste.

2. Protocolos de una WSN

2.1 Wi-Fi

La tecnología Wi-Fi es similar a la red Ethernet tradicional y por tanto se necesita una configuración previa para establecer la comunicación. Muchas veces, se le denomina Wi-Fi al "Ethernet sin cables", para dar una gran idea de las ventajas e inconvenientes que tiene respecto a otras alternativas. Además, Wi-Fi permite conexiones mucho más rápidas y rangos de distancias mayores y sobre todo nos brinda mejores mecanismos de seguridad.

Los dispositivos Wi-Fi permiten una gran movilidad que es uno de las ventajas de esta tecnología, por el contrario tenemos que es de muy fácil acceso para personas ajenas si la red no está bien configurada o bien protegida, siendo necesaria la seguridad para evitar este último problema.

Utiliza el mismo espectro de frecuencia que Bluetooth con una potencia de salida mayor que lleva a conexiones más sólidas. Cabe aclarar que esta tecnología no es compatible con otros tipos de conexiones sin cables como Bluetooth, GPRS, UMTS, etc.

Existen varios tipos de dispositivos que se pueden clasificar en dos grupos:

- Dispositivos de distribución o red, entre los que cabe destacar los routers, repetidores o puntos de acceso.
- Dispositivos terminales, que suelen ser las tarjetas receptoras empleadas en la comunicación con el ordenador (tarjetas PCI), tarjetas PCMCIA que fueron utilizadas en los primeros portátiles y ahora se encuentran en desuso, y tarjetas USB que son más comunes en la actualidad.

2.2. Bluetooth

El protocolo Bluetooth fue diseñado con el objetivo de reemplazar la tecnología con cables usando una conexión de radio muy seguro y de corto alcance. Nos permite la conexión de dispositivos a un bajo costo y a pequeñas distancias. El alcance es generalmente de 10 metros, puesto que se hace para establecer una conexión utilizando el mínimo consumo de energía de las baterías. A pesar de esto, podemos conseguir un alcance mucho mayor, similar al Wi-Fi, con el inconveniente del aumento de consumo. También es recomendable que no exista nada físico entre los elementos que se conectan mediante este protocolo para una mejor comunicación. Este sistema de comunicación está basado en el estándar IEEE 802.15.1 y puede trabajar a una velocidad de transmisión de datos de 1 Mbps [Rodríguez, 2011].

Podemos diferenciar varios tipos de clases de dispositivos:

- **Clase 1** para una potencia máxima de 100 mW o 20 dBm y cuyo rango sería de aproximadamente unos 100 metros.
- **Clase 2** serían aquellos dispositivos de 2,5 mW o 4 dBm de potencia y con un rango de unos 20 m.
- Finalmente la **Clase 3** son los de mínimo consumo (1mW) y mínimo alcance (1m aproximadamente).

También podemos diferenciar los dispositivos por su ancho de banda, donde tendremos la Versión 1.2 de 1Mbits/s, Versión 2.0+EDR de 3Mbits/s y la versión 3.0+HS de 24 Mbps.

2.3. IEEE 802.15.4

IEEE 802.15.4 es un estándar que define el nivel físico y el control de acceso al medio de redes inalámbricas de área personal con tasas bajas de transmisión de datos (*Low-Rate Wireless Personal Area Network* o LR-WPAN). También es la base sobre la que se define la especificación de ZigBee, cuyo propósito es ofrecer una solución completa para este tipo de redes construyendo los niveles superiores de la pila de protocolos que el estándar no cubre.

El propósito del estándar es definir los niveles de red básicos para dar servicio a un tipo específico de red inalámbrica de área personal (*Wireless Personal Area Network* o WPAN) centrada en la habilitación de comunicación entre dispositivos ubicuos con bajo coste y velocidad (en contraste con esfuerzos más orientados directamente a los usuarios medios, como *Wireless Fidelity* ó WiFi). Se enfatiza el bajo coste de comunicación con nodos cercanos y sin infraestructura o con muy poca, para favorecer aún más el bajo consumo.

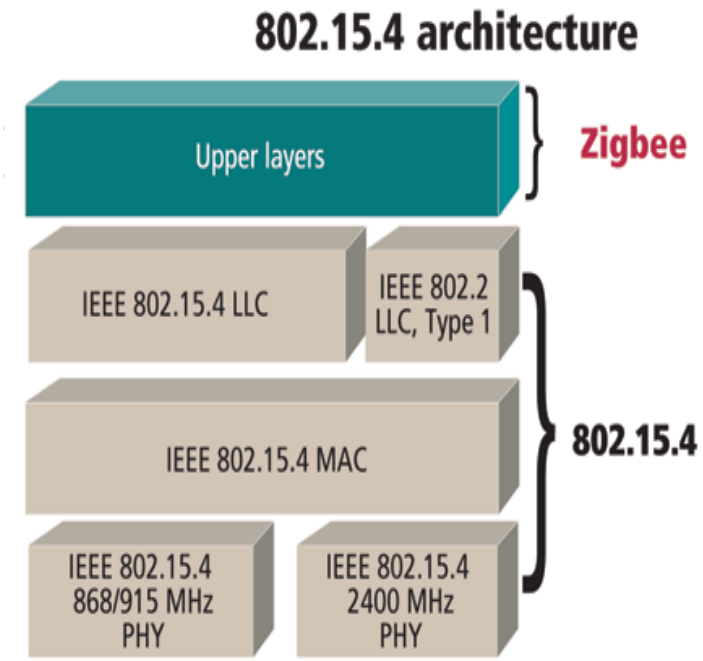


Figura 2.6. Pila del protocolo IEEE.802.15.4

En su forma básica se concibe un área de comunicación de 10 metros con una tasa de transferencia de 250 Kbps. Se pueden lograr tasas aún menores con la consiguiente reducción de consumo de energía. Como se ha indicado, la característica fundamental de 802.15.4 entre las WPAN's es la obtención de costes de fabricación excepcionalmente bajos por medio de la sencillez tecnológica, sin perjuicio de la generalidad o la adaptabilidad.

Entre los aspectos más importantes se encuentra la adecuación de su uso para tiempo real por medio de slots de tiempo garantizados, prevención de colisiones por CSMA/CA (Operador de multiple sentido de acceso con prevención de colisiones ó *Carrier Sense Multiple Access with Collision Avoidance* por sus siglas en ingles) y soporte integrado a las comunicaciones seguras. También se incluyen funciones de control del consumo de energía como calidad del enlace y detección de energía.

Un dispositivo que utilice el 802.15.4 puede transmitir en las siguientes bandas de frecuencia:

- En Europa entre 868 y 868.8 MHz, permitiendo hasta tres canales.
- En América del Norte entre 902 y 928 MHz, extendido a treinta canales.
- 2400-2483.5 MHz de uso para todo el mundo, con la posibilidad de hasta dieciséis canales.

En cuanto a la arquitectura de IEEE 802.15.4, la definición de los niveles se basa en el modelo de interconexión de sistemas abiertos (*Open System Interconection* u *OSI* por sus siglas en ingles). Aunque los niveles inferiores se definen en el estándar, se prevé la interacción con el resto de niveles, posiblemente por medio de un subnivel de control de enlace lógico basado en IEEE 802.2, que acceda a MAC (*Media Access Control*) a través de un subnivel de convergencia. La implementación puede basarse en dispositivos externos o integrarlo todo en dispositivos autónomos.

El control de acceso al medio (MAC) gestiona tramas MAC usando para ello el canal físico. Además del servicio de datos, ofrece una interfaz de control y regula el acceso al

canal físico y al balizado de la red. También controla la validación de las tramas y las asociaciones entre nodos, y garantiza slots de tiempo, además ofrece puntos de enganche para servicios seguros.

El nivel físico (PHY) provee el servicio de transmisión de datos sobre el medio físico propiamente dicho, así como la interfaz con la entidad de gestión del nivel físico, por medio de la cual se puede acceder a todos los servicios de gestión del nivel y que mantiene una base de datos con información de redes de área personal relacionadas. De esta forma, PHY controla el transceptor de radiofrecuencia y realiza la selección de canales junto con el control de consumo y de la señal. Opera en una de tres posibles bandas de frecuencia de uso no regulado.

El estándar no define niveles superiores ni subcapas de interoperabilidad. Existen extensiones, como la especificación de ZigBee, que complementan al estándar en la propuesta de soluciones completas.

2.4. ZIGBEE

2.4.1. Introducción

ZigBee es el nombre de la especificación de un conjunto de protocolos de comunicación de alto nivel para su utilización con radios digitales de bajo consumo, basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (*Wireless Personal Area Network*, WPAN). Conlleva el objetivo de realizar las aplicaciones en las que necesitamos una comunicación segura con una baja tasa de envío de los datos y la información y a la vez, que la vida útil de las baterías sea lo más larga posible.

Es muy utilizado en las aplicaciones de domótica por tres características:

- Bajo consumo
- Topología en red de malla
- Fácil integración, lo que permite crear nodos con muy poca electrónica.

ZigBee utiliza la banda ISM en concreto, 868MHz para Europa y 915MHz para Estados Unidos, además de 2,4GHz para todo el mundo, para usos industriales o científicos, aunque generalmente se usará sobre todo la última de todas.

Los niveles Físico/ Red /Aplicación del protocolo Zigbee son las representadas en la figura 2.7:

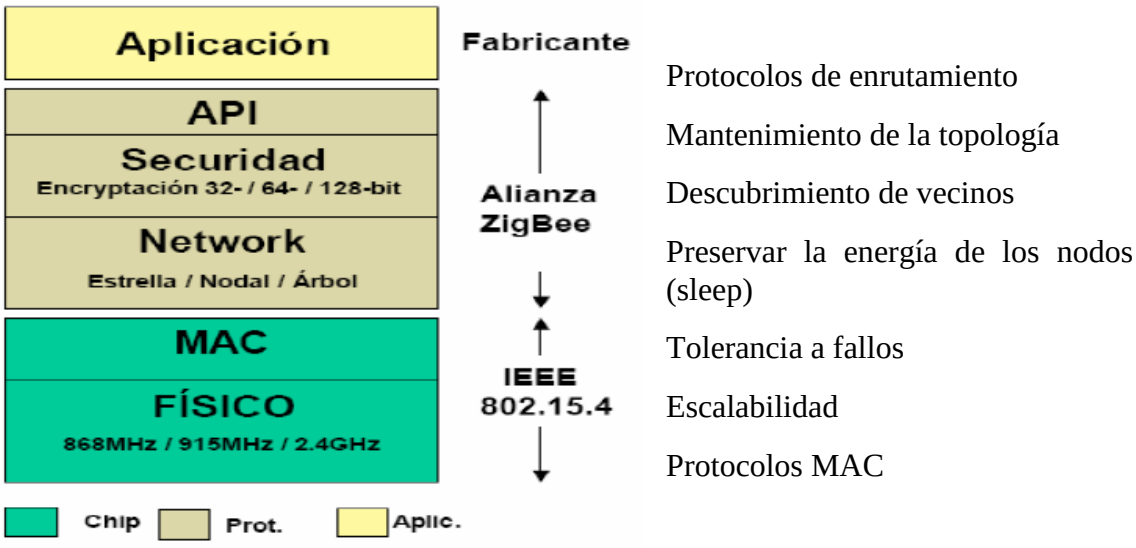


Figura 2.7. Capas del protocolo Zigbee

Los protocolos ZigBee están definidos para su uso en aplicaciones que exigen requerimientos muy bajos de transmisión de datos y consumo energético. Se pretende su uso en aplicaciones de propósito general con características auto organizativas y bajo costo (redes en malla, en concreto). Puede utilizarse para realizar control industrial, albergar sensores empotrados, recolectar datos médicos, ejercer labores de detección de humo, intrusos o domótica. La red en su conjunto utilizará una cantidad muy pequeña de energía de forma que cada dispositivo individual pueda tener una autonomía de hasta 5 años antes de necesitar un recambio en su sistema de alimentación.

2.4.2. Tipos de dispositivos ZIGBEE

Se definen tres tipos distintos de dispositivo ZigBee según su papel en la red:

- **Coordinador ZigBee** (*ZigBee Coordinator*, ZC). El tipo de dispositivo más completo. Debe existir uno por red. Sus funciones son las de encargarse de controlar la red y los caminos que deben seguir los dispositivos para conectarse entre ellos.
- **Router ZigBee** (*ZigBee Router*, ZR). Interconecta dispositivos separados en la topología de la red, además de ofrecer un nivel de aplicación para la ejecución de código de usuario.
- **Dispositivo final** (*ZigBee End Device*, ZED). Posee la funcionalidad necesaria para comunicarse con su nodo padre (el coordinador o un router), pero no puede transmitir información destinada a otros dispositivos. De esta forma, este tipo de nodo puede estar dormido la mayor parte del tiempo, aumentando la vida media de sus baterías. Un ZED tiene requerimientos mínimos de memoria y es por tanto significativamente más barato.

2.4.3. ZIGBEE vs Bluetooth

ZigBee es muy similar a Bluetooth pero con algunas diferencias:

- Una red ZigBee puede constar de un máximo de 64000 nodos, frente a los 8 máximos de una red Bluetooth.
- Menor consumo eléctrico que el ya de por sí bajo de Bluetooth. ZigBee tiene un consumo de 30 mA transmitiendo y de 3 mA en reposo, frente a los 40mA transmitiendo y 0.2 mA en reposo que tiene el Bluetooth.
- Tiene una velocidad de hasta 250 Kbps, mientras que en Bluetooth es de hasta 3Mbps.
- Debido a las velocidades de cada uno, tenemos que Bluetooth se usa para aplicaciones como teléfonos móviles o informática, mientras que el ZigBee se suele utilizar en la domótica.

3. Módulos de Texas Instruments

3.1. CC2530

El CC2530, comercializado por Texas Instruments, es un '*system-on-chip'*(SoC), solución para los protocolos IEEE 802.15.4, Zigbee y aplicaciones RF4CE. Este, además permite construir red de nodos que pueden ser construidas a bajo coste.

Este módulo combina el excelente funcionamiento de un transceptor RF con un estándar industrial mejorado MCU 8051, un sistema de memoria flash programable, 8 KB de RAM, y muchas otras características potentes. El CC2530 se encuentra en 4 diferentes versiones flash: CC22530F32/64/128/256, con memoria flash de 32/64/128/256KB respectivamente.

Incluye varios modos de operación, haciéndolo altamente adecuado para sistemas donde es requerido consumos ultra bajos de energía, además cortas transiciones de tiempo entre modos de operación asegurando bajos consumos de energía.

Presenta las siguientes características:

- Tiene una gran memoria flash de hasta 256 KB.
- Excelente conexión (102 dBm).
- Cuatro modos de alimentación flexibles para el menor consumo de energía.
- Cinco canales potentes DMA.

Figura 2.8. CC2530

Para la programación de estos motes se emplea la Placa de Evaluación *SmartRF05EB*. Esta placa actúa como interfaz entre el PC y los motes a través del puerto serie, permitiendo la programación de los dispositivos acoplados.

La placa SmartRF05EB tiene un controlador USB por el cual se comunica con el PC. El controlador USB se comunica con el módulo de evaluación usando SPI, UART y / o la
Debug Interface (system-on-chip solamente) [TI].

Figura 2.9. Placa de evaluación *SmartRF05EB*

3.1.1. Protocolos

Uno de los principales desafíos de diseño en WSN es la comunicación de datos intentando al mismo tiempo prolongar la vida útil de la red, empleando técnicas agresivas de administración de energía. Se entiende por protocolo al conjunto de estándares que controlan la secuencia de mensajes que ocurren durante una comunicación entre entidades que forman una red.
En cuanto a los protocolos que soporta el CC2530 cabe destacar los siguientes: SimpliciTI, Z-Stack y RemoTI.

3.1.1.1. SimpliciTI

SimpliciTI ™ es un protocolo de radio frecuencia de baja potencia dirigido a las redes pequeñas y simples. Este software de código abierto es un excelente punto de partida para la construcción de una red de dispositivos inalámbricos usando un sistema integrado RF de baja potencia de Texas Instruments (SoC) o de ultra baja potencia MSP430, MCU y un transceptor de RF de Texas Instruments. SimpliciTI fue diseñado para una fácil implementación y despliegue externo en múltiples plataformas RF de Texas Instruments tales como: las MCU MSP430, los transceptores CC1XXX/CC25XX y SoCs. La última versión de SimpliciTI soporta ahora el nuevo hardware CC2530 y CC430 de Texas Instruments.

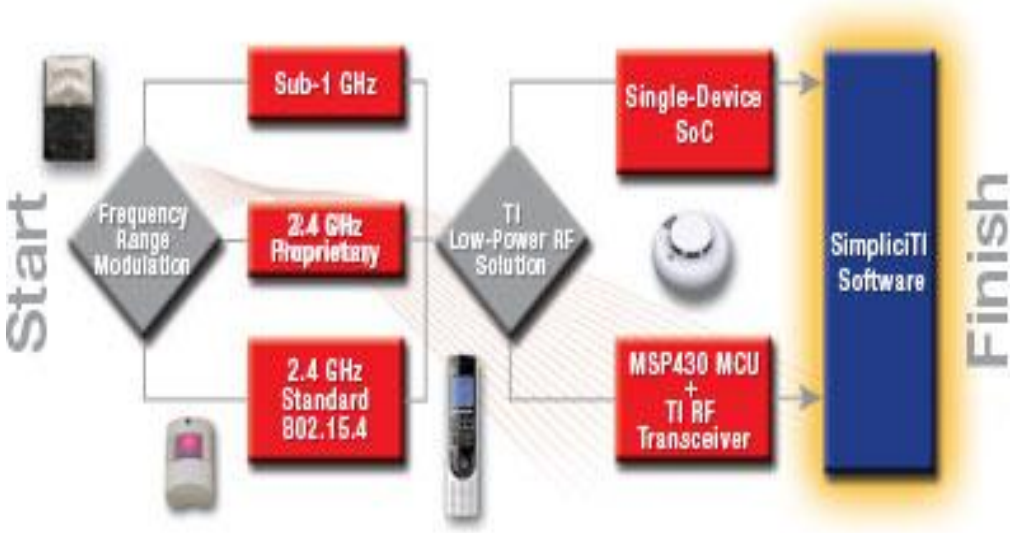


Figura 2.10. Esquema protocolo SimpliciTI

En cuanto a las características principales resaltan:

- Bajo consumo.
- Flexible:
  - ✓ Comunicación directa de los dispositivos.
  - ✓ Configuración en estrella con punto de acceso para almacenar y enviar al '*end device'*(dispositivo final).
- Simple: Utiliza una API de cinco comandos.
- Baja tasa de datos y ciclo de trabajo bajo.
- Facilidad de uso.

Por otro lado, las principales aplicaciones en las cuales se suele utilizar este protocolo son las siguientes:

- Alarma y seguridad: sensores de presencia, sensores de luz, sensores de monóxido de carbono, detectores de rotura de cristal.
- Los detectores de humo.
- Lectura automática de contadores: medidores de gas, medidores de agua, etc.

SimpliciTI trabaja con los dispositivos de baja potencia de RF y herramientas mencionados en la figura 2.11:

Dispositivo *	Descripción
CC2530	Segunda generación System-on-Chip solución para 2,4 GHz IEEE ® 802.15.4/RF4CE/ZigBee
CC2510DK-MINI	El CC2510 Mini Kit de Desarrollo es una plataforma de desarrollo económico y flexible para CC2510Fx TI RF System-on-Chip soluciones y es todo lo que necesita para construir un prototipo de aplicación del sensor nodo inalámbrico! (El kit de desarrollo CC2510 Mini no es una parte del instalador, sin embargo, puede ser manual integrado en Simpliciti. Por favor, vea la nota de diseño DN118)
CC1110DK-MINI	El kit CC1110 Mini Desarrollo es una plataforma de desarrollo económico y flexible para CC1110Fx TI RF System-on-Chip solución y es todo lo que necesita para construir un prototipo de aplicación del sensor nodo inalámbrico! (El kit de desarrollo CC1110 Mini no es una parte del instalador, sin embargo, puede ser manual integrado en Simpliciti. Por favor, vea la nota de diseño DN118)
CC430	La industria de energía más bajo, de un solo chip de radiofrecuencia solución para las aplicaciones basadas en MCU
CC1101	Sub-1 GHz, altamente integrado, multi-canal de RF del transceptor diseñado para aplicaciones inalámbricas de bajo consumo
CC111x	Sub-1 GHz SoC con una mejora de MCU 8051, hasta 32 kB de memoria flash programable y USB opcional
CC2520	La segunda generación de 2,4 GHz RF transceptor para propietarios y ZigBee ® / IEEE 802.15.4 circuitos integrados de RF con selectividad estado-of-the-art / co-existencia, presupuesto del enlace excelente, rango extendido de temperatura (-40 ° C a +125 ° C) y el funcionamiento de bajo voltaje
CC2430	2,4 GHz SoC diseñado específicamente para aplicaciones IEEE 802.15.4 y ZigBee ®; CC2430 está disponible en tres versiones de flash diferentes: 32/64/128 kB de memoria flash
CC2500	2,4 GHz transceptor RF diseñado para aplicaciones de baja potencia inalámbricos en la banda de 2,4 GHz ISM
CC251x	2,4 GHz de bajo consumo SoC integrado con MCU, hasta 32 kB de memoria Flash y opcional USB de alta velocidad integrado
eZ430-RF2500	Desarrollo memoria USB herramienta basada en la combinación de la capacidad de la potencia ultra-baja MSP430 con bajo consumo de energía transceptor RF en un solo dispositivo

Figura 2.11: Listado de dispositivos

3.1.1.2. Z-Stack

Z-Stack ™ es una implementación de la pila Zigbee de comunicación inalámbrica. Z-Stack es compatible con la especificación ZigBee 2007, soportando características tanto Zigbee como Zigbee Pro establecidas en el chip integrado CC2530, MSP430-CC2520 y Stellaris LM3S9B96 +. Z-Stack ™ soporta perfiles de energía inteligente y automatización del hogar.

Z-Stack incluye:

- Una característica totalmente compatible con ZigBee PRO situado en las plataformas CC2530 y CC2520 + MSP.
- Es totalmente compatible con ZigBee incluidas en la familia CC2530 SoC y una amplia familia de microcontroladores MSP430 junto con la última y mejor en clase de transceptor CC2520.
- Las aplicaciones simples incluyen el soporte para los perfiles ZigBee *Smart Energy* 1.1 y ZigBee *Home Automation***.**
- Soporta el perfil '*over the air download*'.

En cuanto a las características tenemos:

- Durante el '*over the air download'* permite futuras actualizaciones para el hardware desplegado.
- Soporte para los perfiles *ZigBee Smart Energy* 1.1 (ZSE), domótica *ZigBee* (ZHA) y puesta en marcha de *ZigBee Cluster Application*.
- API simple para reducir el tiempo de desarrollo.

3.1.1.3 RemoTI

RemoTI ™ es el líder industrial de la arquitectura de software compatible de RF4CE. Esta construido en el conocido IEEE 802.15.4 TIMAC de Texas Instruments. RemoTI ™ ofrece un marco de software intuitivo, simple, fácil de usar y todas las herramientas, documentación y soporte necesario para construir un producto RF4CT. RemoTI ™ permite reducir el tiempo y coste de desarrollo.

RemoTI ™ es un kit completo de desarrollo de software y hardware de Texas Instruments que permite realizar aplicaciones compatibles con RF4CT. Se basa en el CC2530, una verdadera solución SoC (System-on-Chip) de 2,4-GHz. El CC2530 combina un transceptor RF de 802.15.4, un microcontrolador, una memoria flash programable de hasta 256 KB, 8 KB de memoria RAM y una completa gama de periféricos.

El paquete de software RemoTI incluye una pila de protocolo compatible RF4CE, ejemplos del código de aplicación y herramientas de PC.

En cuanto a la arquitectura se compone de los siguientes componentes:

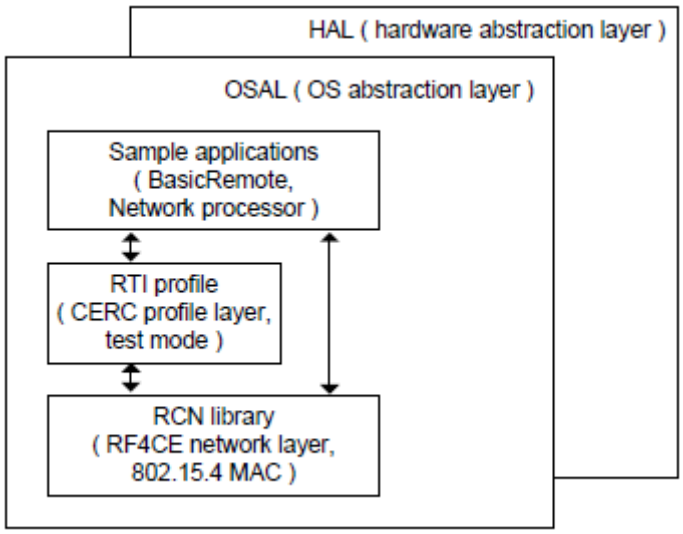


Figura 2.12: Arquitectura del software RemoTi

- Sistema Operativo de la capa de abstracción(*Operating System Abstraction Layer* u OSAL por sus siglas en ingles)

Este es un simple entorno de funcionamiento del sistema para el SoC. Incluye funciones para tareas de gestión, paso de mensajes, colas, gestión de memoria, temporizadores, etc. Este componente se incluye como código fuente.

- Capa de abstracción de hardware (*Hardware Abstraction Layer* o HAL)

Este componente proporciona una interfaz abstracta para el hardware disponible en el chip y en la placa. Incluye firmware para el emisor/receptor asíncrono universal (*Universal Asynchronous Receiver/Transmitter* ó UART) y para la interfaz de comunicación SPI (*Serial Peripheral Interfac*), teclado en el mando a distancia, LEDs y la generación de IR. Este código está incluido en fuente para permitir al usuario realizar modificaciones para adaptarse al hardware disponible.

- Librería RCN (RCN *library*)

Esto es el núcleo de la pila RF4CE e incluye la capa de red RF4CE, la capa IEEE 802.15.4 MAC y el firmware de radio. Este componente está incluido como un objeto en el código de la librería.

- Perfil RTI (RTI *profile*)

Esta es una implementación del perfil de aplicación del Consumidor Electrónico para Control Remoto (*Consumer Electronics for Remote Controls* ó CERC). Se incluye el mecanismo de emparejamiento definido en el perfil de CERC. También hay otras características del modo de prueba disponibles. Este componente está incluido en el código fuente.

- Aplicaciones de ejemplo

Esto es código de ejemplo que muestra cómo construir fácilmente aplicaciones diferentes con la pila RemoTI. La aplicación '*BasicRemote'* implementa las características básicas de un mando a distancia. La aplicación '*NetworkProcessor'* implementa un nodo de destino que se comunica con un procesador anfitrión través de la interfaz UART o SPI.

Ampliando las características, incluye la característica '*over air download'* y la característica de gestor de arranque serie. La característica '*over air download'* está disponible con la aplicación '*BasicRemote'* y puede ser usada para actualizar el firmware en el control remoto del nodo objetivo. La característica '*serial bootloader'* está disponible con la aplicación '*NetworkProcessor'* y puede ser usado para actualizar el firmware en nodo objetivo del procesador anfitrión.

En cuanto a características destacan:

- Su API es simple e intuitivo.
- Contiene UART, SPI, I2C, USB, teclado, soporte para controlador LED y otros.

- Soporte de perfiles ZRC (*ZigBee Remote Controller*) y ZID (*ZigBee Input Device*)
- Muestra el código completo de una aplicación de prueba.
- Cuenta con herramientas de soporte para facilitar el desarrollo y los tests.

3.2. CC2520

El CC2520 es la segunda generación del transceptor RF de Texas Instruments ZigBee ® / IEEE 802.15.4 para la banda de 2,4 GHz sin licencia ISM. Este chip ofrece a las aplicaciones industriales selectividad en el estado del arte, especificación de enlace excelente, operaciones de hasta 125 ° C y de bajo voltaje.

Además, el CC2520 proporciona soporte de ampliación hardware para gestión de servicio, el búfer de datos, transmisiones de ráfaga, el cifrado de datos, canal de valoración limpio u enlace de indicación de calidad. Estas características reducen la carga en el controlador principal.

En un sistema típico, el CC2520 se emplea junto con un microcontrolador y unos pocos componentes pasivos adicionales.

Presenta las siguientes características:

- Rango de temperatura extendida (-40 a +125 ° C).
- Especificación de enlace excelente (103dB).
- Amplia banda de alimentación: 1,8 V - 3,8 V.
- Excelente enlace (103dB)
- Módulo de seguridad AES-128.

Figura 2.13: CC2520

El CC2520 es generalmente controlado por un microcontrolador conectado a la SPI y algunos a los pines de entrada y salida (*General Purpose Input/Output* ó GPIO). El microcontrolador enviará instrucciones al CC2520 y es responsabilidad de un codificador de instrucciones ejecutarlas o pasarlas a otros módulos. La ejecución de una instrucción o eventos externos (por ejemplo, la recepción de una trama) puede dar lugar a una o más excepciones. Las excepciones constituyen un mecanismo muy flexible para la automatización de tareas. Por ejemplo, puede ser utilizado para activar la ejecución

de otras interrupciones o pueden ser enviados a pines GPIO y usados como señales que interrumpen al microcontrolador. El controlador de excepción (*exception controller*) es responsable del manejo de las excepciones.

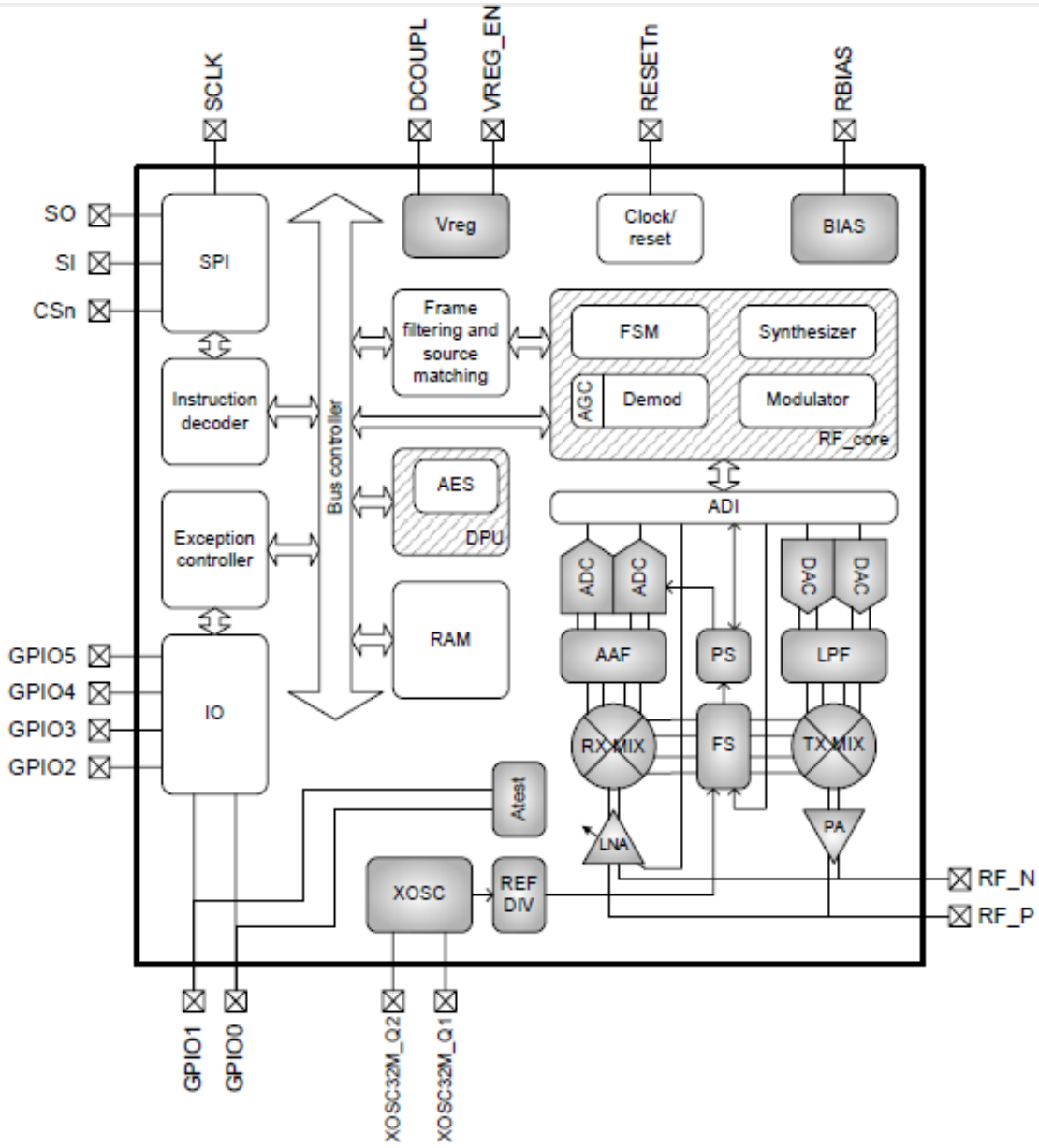


Figura 2.14: Diagrama de bloques del CC2520

El microcontrolador típicamente estará conectado a uno o más pines GPIO. La función de cada pin es controlado inmediatamente por el módulo IO basado en el registro de configuraciones. Es posible observar un gran número de señales internas en los pines GPIO. Además, estos también se pueden configurar como entrada y se utiliza para desencadenar la ejecución de ciertas instrucciones. Esto normalmente se usa cuando el microcontrolador necesita controlar con precisión la temporización de una instrucción.

El módulo de memoria RAM contiene la memoria que se utiliza para recibir y transmitir FIFOs (*First in, First out*), en intervalos de direcciones fijas, y almacenamiento temporal para otros datos.

La unidad de procesamiento de datos (*Data Processing Unit* ó DPU,) es responsable de la ejecución de las instrucciones más avanzadas. El DPU incluye un núcleo AES, que se utiliza durante la ejecución de las instrucciones de seguridad. La gestión de memoria (copiar, incrementando, ect) se realiza también por la DPU.

El módulo de '*Clock/Reset'* genera los relojes internos y las señales de restablecimiento.

El núcleo de RF contiene varios submódulos que dan soporte y controlan los módulos de radio analógicos.

El controlador del estado del transceptor de RF es el submódulo FSM, el cual controla los FIFOs de transmisor y receptor y la mayoría de las señales analógicas dinámicamente controladas tal como '*power up/down'* de módulos analógicos**.** El FSM se utiliza para proporcionar la secuencia correcta de eventos (por ejemplo, realizar una calibración de FS antes de activar al receptor). También, proporciona paso a paso el procesamiento de las tramas entrantes del demodulador; leyendo la longitud de las tramas, contando el número de bytes recibidos, comprueba el FCS y opcionalmente maneja la transmisión automática de tramas ACK después de la correcta recepción de las tramas. Realiza tareas similares en TX, incluyendo la realización de una CCA opcional antes de la transmisión y automáticamente va a RX después del final de la transmisión para recibir una trama ACK. Finalmente, la FSM controla la transferencia de datos entre el modulador / demodulador y el TXFIFO / RXFIFO en la RAM.

El modulador transforma datos puros al interior de las señales de entrada/salida para el transmisor DAC. Esto se hace de acuerdo con el estándar IEEE 802.15.4.

El demodulador es responsable de recuperar los datos enviados desde la señal recibida.

La información de amplitud del demodulador es utilizado por el control automático de ganancia (AGC). El AGC ajusta la ganancia del amplificador de bajo ruido (*Low Noise Amplifier* ó LNA) analógico para que el nivel de señal en el receptor sea aproximadamente constante.

El marco de filtrado y adaptación de origen admite la FSM en RF_core mediante la realización de todas las operaciones necesarias para realizar el filtrado de trama y la coincidencia de la dirección de origen, tal como se define en IEEE 802.15.4.

El módulo XOSC interactúa con el cristal que está conectada a los pines XOSC32M_Q1 y XOSC32M_Q2. El módulo XOSC genera un reloj para la parte digital y el sistema de RF, e implementa la sintonización de cristal de frecuencia programable.

El módulo de BIAS genera las referencias de tensión y corriente. Se basa en una resistencia externa de alta precisión (1%) 56kΩ que se muestra en el circuito de aplicación en la Figura 2.15.

Los DACs TX (convertidores digital-analógicos) convierten la señal digital de banda base a señales analógicas.

Después de la señal de campo potencial local (*Local Potential Field* ó LPF) se alimenta al módulo de TXMIX, que es un mezclador complejo de conversión creciente.

El amplificador de potencia (PA) amplifica la señal de RF, hasta un máximo de ~ 5dBm durante TX.

El LNA (amplificador de bajo ruido) amplifica la señal de RF recibida. La ganancia es controlada por el módulo de AGC digital, de modo que se logra una sensibilidad óptima y un rechazo de interferencia.

El módulo RXMIX es un complejo mezclador que convierte la señal de RF a una señal de banda base.

Un filtro anti ruido pasivo(AAF) filtra la señal después del mezclado.

El paso bajo filtra señales I y Q y digitalizadas por el ADC (convertidor analógico-digital).

El sintetizador de frecuencia (FS) genera la onda portadora para la señal de RF.

El regulador de voltaje (VREG) proporciona una tensión de alimentación de 1,8 V para el núcleo digital. Contiene un limitador de corriente, que está habilitado para corrientes por encima de aproximadamente 32 mA.

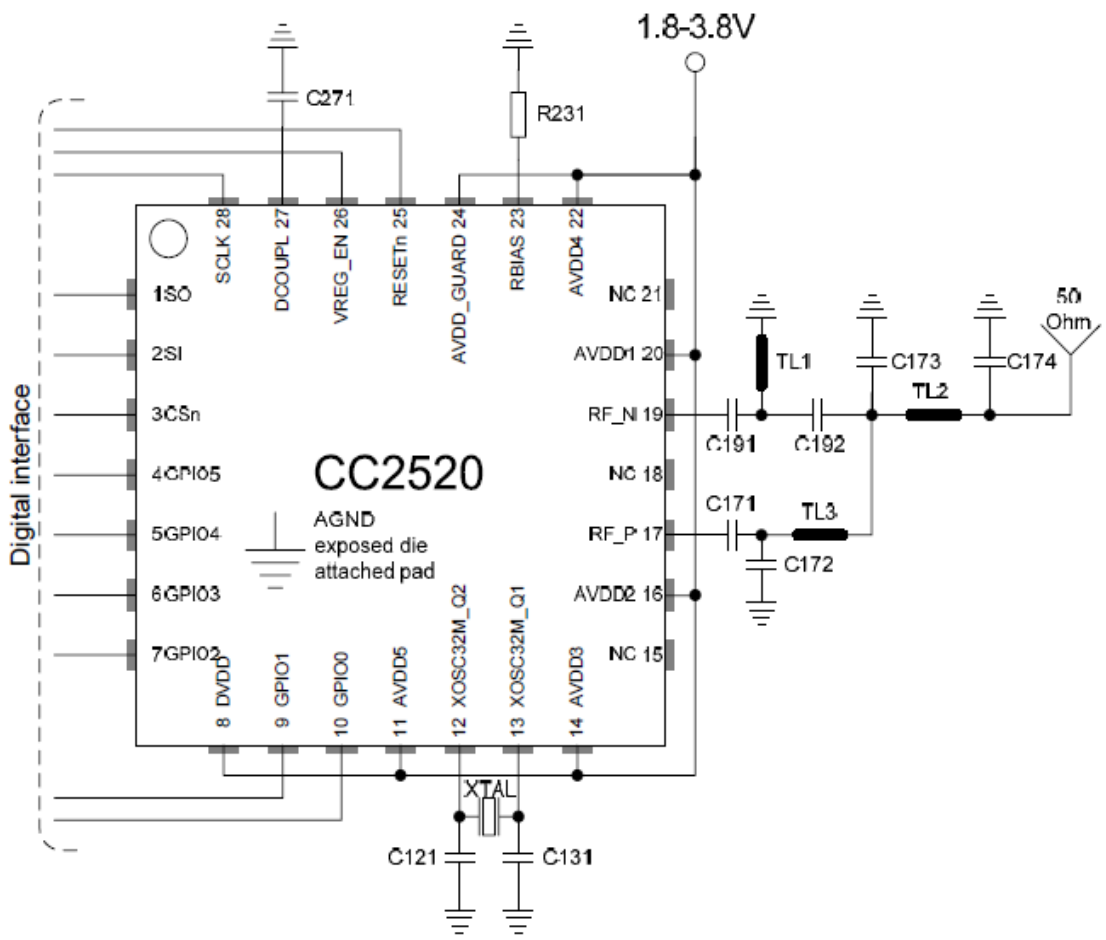


Figura 2.15: Circuito de aplicación del CC2520

3.3. MSP430F2618

Los microcontroladores de la familia MSP430 de Texas Instruments de potencia ultra baja constan de varios dispositivos que ofrecen diferentes conjuntos de periféricos específicos para diversas aplicaciones. La arquitectura, junto con cinco modos de bajo consumo, se ha optimizado para lograr una vida prolongada de la batería en aplicaciones de medición portátiles. El dispositivo cuenta con un poderoso CPU con conjunto de instrucciones reducidas (*Reduced Intruction Set Computer* ó RISC) de 16-bit, los registros de 16-bits, y generadores de constantes que contribuyen a la eficiencia máxima del código. El oscilador calibrado controlado digitalmente (*Digital Calibrate Oscilator* ó DCO) permite despertar de modos de bajo consumo al modo activo en menos de 1 ms.

Las aplicaciones típicas incluyen sistemas de sensores, aplicaciones industriales de control, medidores de mano, etc.

Presenta las siguientes características:

- Suministro de Voltaje: 1,8 V a 3,6 V.
- Tres canales DMA interno.
- Arquitectura RISC de16 Bit, el tiempo de ciclo de instrucción es de 62.5 ns.
- Ultra-Bajo Consumo de Energía:

✓ Modo activo: 365 μA a 1 MHz, 2,2 V
✓ Modo de espera (VLO): 0,5 μA
✓ *Off Mode* (Retención RAM): 0,1 μA

Figura 2.16: MSP430F2628

En la figura 2.17 se pueden observar los bloques funcionales de un MSP430 cualquiera.

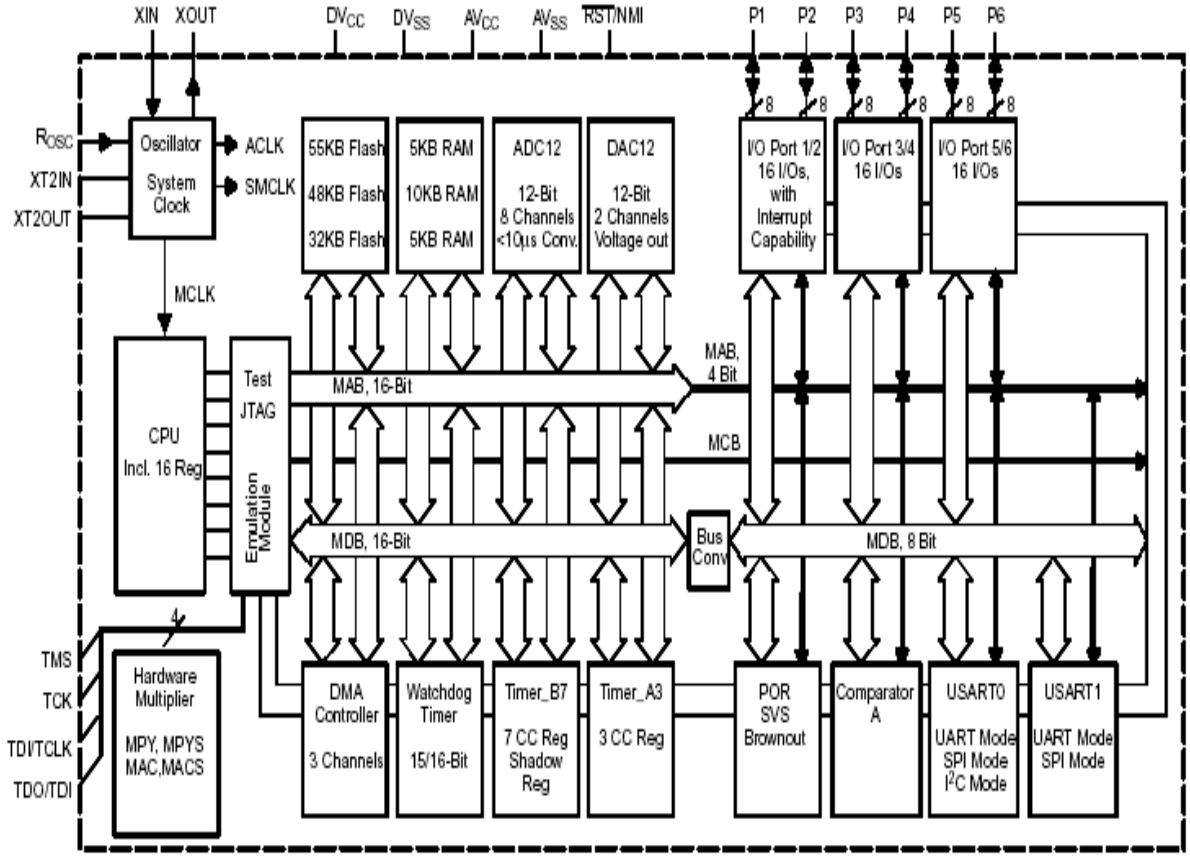


Figura 2.17: Diagrama de bloques para la familia MSP430

A continuación se presenta una breve descripción de éstos:

RISC-CPU: Es la unidad central de procesamiento. Aquí realiza el proceso de las instrucciones contenidas en la memoria de programa (*Flash*).

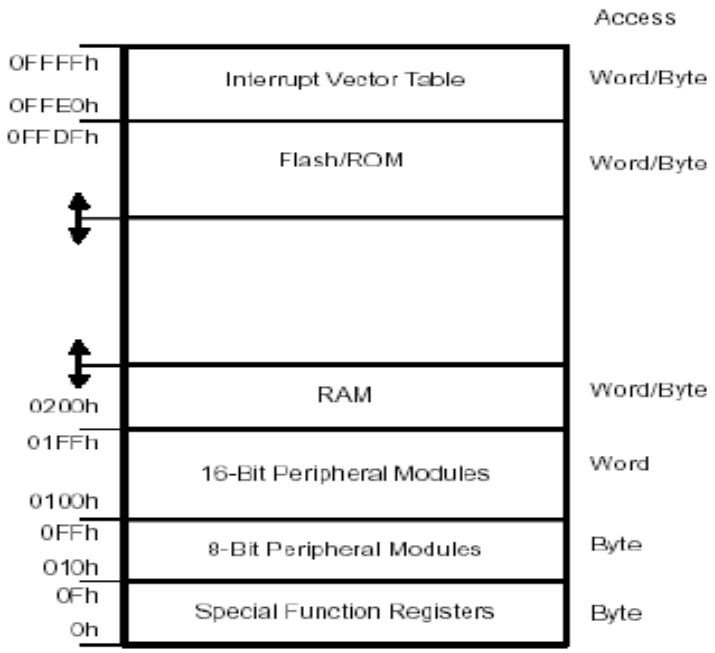


Figura 2.18: Organización del espacio de memoria

Clock System: El MSP430 cuenta con tres señales de reloj que son generadas a partir de tres fuentes distintas (ver figura 2.19). Un cristal de baja frecuencia, un cristal de alta frecuencia y una señal generada digitalmente a partir de circuitos R-C configurables. A partir de estas tres fuentes se generan las tres señales de reloj:

- *Main System Clock* (MCLK) que es utilizado por la CPU y el sistema.
- *Auxiliary Clock* (ACLK) que es utilizado por los periféricos.
- *Sub-System Clock* (SMCLK) que también sirve de fuente para los periféricos.

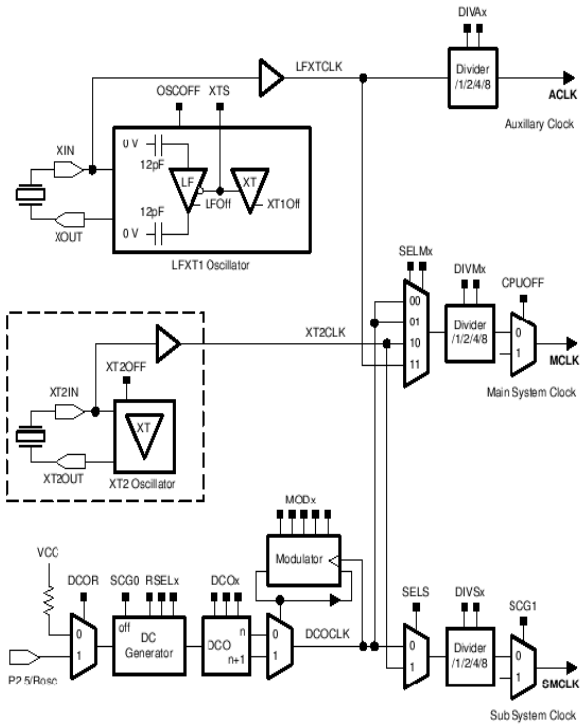


Figura 2.19: Sistema de generación de reloj

Es esta notable flexibilidad a la hora de seleccionar fuentes de reloj lo que le da al MSP430 su gran capacidad de ahorro de energía, al poder utilizar bajas frecuencias en periodos de ocio de la CPU y/o los periféricos. Una baja frecuencia de reloj implica un bajo consumo. En este diseño se optó por usar un cristal de baja frecuencia de 32KHz y uno de alta frecuencia de 8MHz para la CPU. De esta manera, el funcionamiento normal es lo suficientemente rápido y se puede utilizar un modo de bajo consumo.

*WatchDog Timer***:** la función principal del '*watch dog timer'* (WDT) es la de realizar un '*reset'* controlado del sistema en caso de que ocurra un problema de software. La detección del problema se logra por medio de sucesivos refrescos al WDT por parte del software. Si el WDT expira, entonces el sistema es reiniciado automáticamente. Adicionalmente, si el WDT no es utilizado para controlar la ejecución del software puede ser utilizado como un *timer* extra.

Memoria *Flash*: Esta memoria no volátil se divide en dos zonas: la zona de código y la zona de información. La zona de código es el lugar donde se almacena el programa a ejecutar por la CPU. La zona de información es una zona de libre propósito para uso del programador.

Memoria *RAM***:** Es una memoria volátil que se utiliza para las variables del programa.

Multiplicador por *Hardware*: El Multiplicador por Hardware (HM) es un periférico opcional que no es parte de la CPU, esto significa que su actividad no interfiere con el procesador. El HM se comunica con el CPU vía registros. Este soporta multiplicación sin signo, multiplicación con signo, multiplicación sin signo acumulada y multiplicación con signo acumulada. Puede funcionar en 16x16, 16x8, 8x16 y 8x8 bit.

Por otro lado, no está presente en todos los modelos de MSP430, pero si en la mayoría, es por eso que no se puede apreciar en la figura.

Direct Memory Access (DMA): El controlador de DMA se encarga de transferir información desde una dirección de memoria a otra sin la intervención de la CPU. Este periférico no está disponible en todos los microcontroladores de la familia. El controlador DMA no fue utilizado en el desarrollo de este trabajo.

Timer A y B**:** Este periférico puede funcionar como un timer o un contador de 16 bit, con tres registros de captura o comparación, salidas de PWM (*Pulse Width Modulation*) y medición de intervalos. El timer posee amplias capacidades de interrupción, las que pueden ser generadas por un *overflow* del contador o por los registros de captura y comparación.

USART 0 y 1**:** Los MSP430 generalmente incorporan 2 USART, dependiendo del modelo, estas pueden funcionar hasta en tres modos: modo UART, modo I2C y modo SPI.

- EL modo UART es un modo de transmisión asincrónico que se comunica por medio de dos pines, UTXD y URXD. Este modo es compatible con el estándar de transmisión RS-232.
- El modo I2C provee comunicación con cualquier dispositivo que implemente este protocolo. Este modo utiliza 2 cables para transmitir y recibir información.
- El modo SPI conecta el microcontrolador con otro dispositivo por medio de tres o cuatro líneas. Este modo de comunicación no sería abordado en este trabajo.

Convertidor Analógico-Digital: El modulo ADC (*Analog Digital Converter*) implementa un ADC de alta velocidad que funciona sin la intervención de la CPU. Posee buffers de conversión y control para mayor independencia. El convertidor analógico-digital no fue utilizado en este trabajo.

Convertidor Digital-Analógico: El modulo DAC (*Digital Analog Converter*) es un convertidor de 8 o 12 bit que utiliza como referencia los voltajes Vref+ y Vref−. El convertidor digital-análogico no fue utilizado en este trabajo.

Un ejemplo de placa comercial que utiliza un microcontrolador de la familia MSP430 es la placa de desarrollo MSP430-P1611 de la marca Olimex y cuyo precio aproximado es de 33 $. Sus características más destacables son [Olimex]:

- Programación mediante conector JTAG.
- RS232 + conector de botón de usuario DB9.
- LED de encendido.
- LED de estado.
- Frecuencia de oscilación del cristal de 32-768 Hz.
- Botón RESET.

Figura 2.20: Placa de desarrollo MSP430-P1611

- RST/NMI pin pull-up.
- Fuente de alimentación externa jack para AC o DC.
- Regulador de voltaje + fuente de alimentación filtrada por condensador.
- Cabecera de extensión de los pines del microcontrolador.
- Area prototipo de 0’1” de paso, Vcc + bus GND.
- PCB: FR-4, 1.5 mm (0,062"), marcas de soldaduras verdes, impresión de componentes en serigrafía blanca.

4. Revisión de nodos sensores

4.1. Soluciones de laboratorio

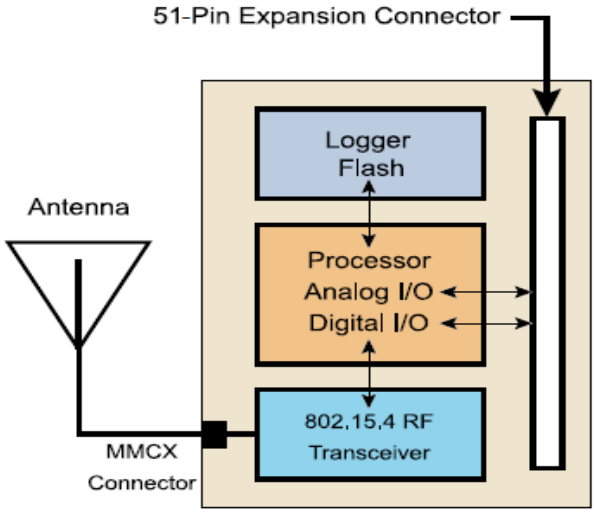
4.1.1. Micaz

La plataforma de sensores MICAz está comercializada por Crossbow [Crossbow]. Son una de las últimas generaciones de nodos sensores que trabaja en la banda de frecuencias de 2400 MHz a 2483.5 MHz. Utilizan una plataforma de procesador y radio del tipo “MPR2400”.

La familia MICAz usa el Chipcon CC2420 que es un transmisor-receptor que cumple la norma IEEE 802.15.4 y tiene un transmisor de radio frecuencia Zigbee y un microcontrolador Atmega 128L.

Figura 2.21: Dipositivo Micaz

Presenta las siguientes características:



- Proporciona una velocidad de transmisión de hasta 250 kbps.
- Dispone de una memoria flash.
- Tiene 51 pines de conexión, contando entradas, salidas e interfaces UART.
- Utilizada en redes de sensores de gran escala.
- El vendedor Crossbow proporciona además diferentes sensores y placas que se adhieren a los pines de entrada.
- Diseñado específicamente para redes de sensores Deeply Embedded.

Figura 2.22: Diagrama de bloques

Para la programación de estos nodos sensores se emplea la Placa de desarrollo *MIB510*. Esta placa actúa como interfaz entre el PC y los nodos sensores a través del puerto serie, permitiendo la programación de los dispositivos acoplados.

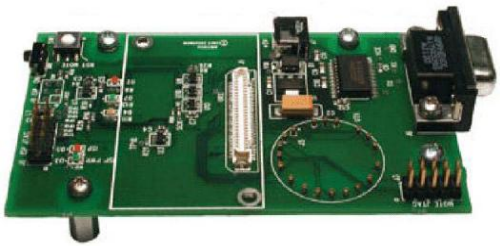


Figura 2.23: Placa de desarrollo MIB510

La placa MIB510 tiene un procesador de programación
es sistema (*In System Programing* ó ISP) Atmega16l mediante el cual se programan los motes. El código se descarga al ISP a través del puerto serie RS-232, y es el ISP el que programa el código en el nodo sensor. Esta placa tiene conectores para nodos sensores MICAz, MICA2 y MICA2DOT.

4.1.2 Mica2 y Mica2dot

Diseñados por investigadores de la Universidad de Berkeley, en California, existen dos modelos llamados Mica2 y Mica2dot. Su funcionalidad es similar pero son muy diferentes respecto a su forma, como se puede ver en la figura 2.24.

En cuanto a sus características tenemos que ambos funcionan a 4MHz con un microprocesador de 8 bits de la marca Atmel, 128KB de memoria para el programa y 4KB de memoria RAM, su velocidad de transmisión radio es de 19.2Kb/s sobre un canal CSMA/CA y utiliza el estándar IEEE 802.15.4. También están equipados con una memoria externa de tipo Flash no volátil con 512KB que puede ser usada para guardar otros datos [Mica2, Mica2dot].

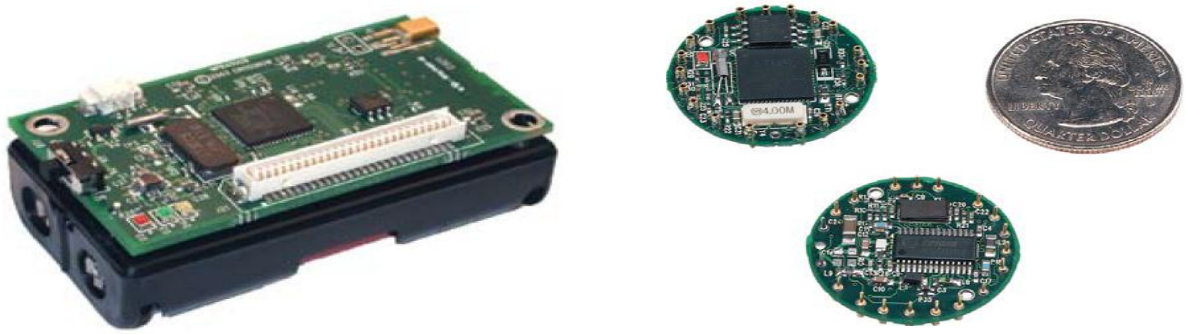


Figura 2.24: Mica2 y Mica2dot

A la placa principal del procesador se le pueden adherir placas con sensores/actuadores por medio de unos pines, en el caso del Mica2, o por medio de unos conectores, para el modelo Mica. Existe ya una placa con sensores de temperatura, luminosidad, un micrófono acelerómetro y un sensor magnético, entre otros.

El diagrama de bloques del nodo sensor Mica2 se muestra en la figura 2.24 y del Mica2dot se muestra en la figura 2.25:

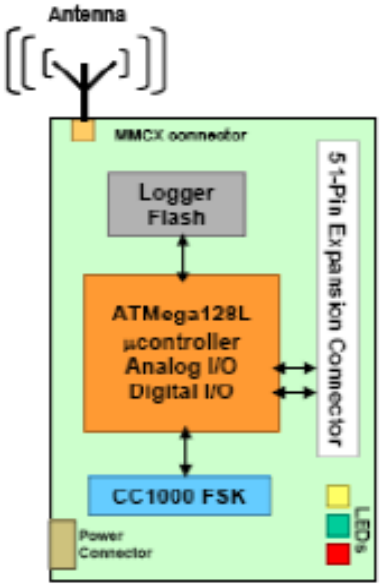


Figura 2.25: Diagrama de bloques Mica2

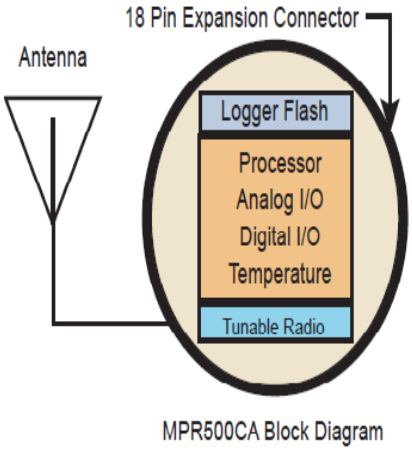


Figura 2.26: Diagrama de bloques Mica2dot

En cuanto a los kits de desarrollo, Crossbow proporciona diferentes versiones dependiendo del tipo de desarrollador:

- *Started Kits*: Proporciona los elementos imprescindibles para poner en marcha una red de sensores inalámbrica.
- *Profesional Kits*: Incluye todo lo necesario para crear una red real de sensores con un mayor número de Motes, diversas placas de sensores.
- *OEM Desing Kits*: Está enfocado a la generación de redes de sensores comerciales y específicos con gran volumen de producción.
- *Classroom Kits*: Ideal para su uso docente.

4.1.3 Intel Mote 2

Intel Mote surge de una colaboración entre los laboratorios de Intel en Berkeley, la Universidad de Berkeley y otros centros de investigación del resto de USA. Los Motes de Intel son pequeños, autónomos, alimentados por medio de baterías y con comunicación vía radio, de forma que son capaces de compartir información con otros y organizarse automáticamente dentro de una red AdHoc [INTEL MOTE 2].
Uno de los principales objetivos del grupo de trabajo de los Intel Motes es colaborar con la comunidad investigadora en la exploración de las potenciales aplicaciones de los nodos sensores y de las redes de sensores. Con este objetivo en mente, se han diseñado los nodos sensores con una total compatibilidad con el sistema operativo TinyOS.

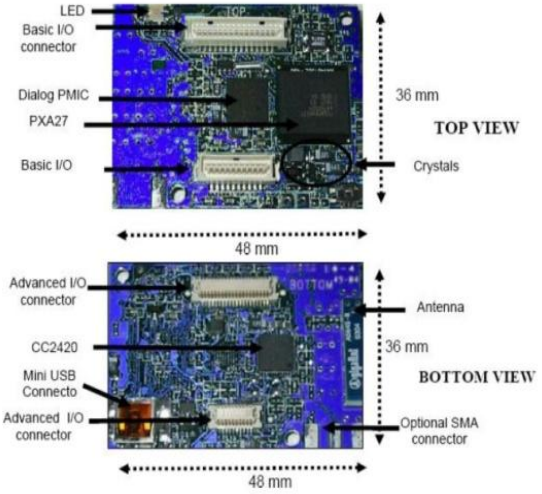


Figura 2.27: Intel Mote 2

Las investigaciones de mejora se están centrando en tres aspectos:

- Trabajo con consumos de energía ultra bajos.
- Integración del sistema.
- Reconfiguración hardware.

En cuanto a las características destacan:

- Procesador de bajo consumo XScale PXA27x.
- Antena integrada de 2.4GHz.
- Radio Integrado 802.15.4., utilizado para la comunicación con la placa principal.
- 2 interfaces de sensores básicos en uno de los lados de la placa central y otros 2 interfaces avanzados para sensores en el otro lado.

El Intel Mote 2 es una plataforma modular y escalable. La placa principal contiene un procesador y un módulo radio. Es posible agregar a dicho mote diferentes módulos sensores dependiendo de la aplicación final de la red de sensores.

El procesador, contiene una placa principal, la cual, puede funcionar a bajo voltaje (0.85V) y una frecuencia también baja (13MHz), cuando se habilita el modo de bajo consumo, mientras que el valor máximo de operación es de 416MHz. También integra una memoria SRAM de 256KB dividida en 4 bancos de 64KB y diferentes opciones de E/S, lo que lo hace extremadamente flexible para soportar diferentes sensores A/D. Estas E/S incluye I2C, 3 puertos serie síncronos y 3 puertos UART, E/S digitales. Además, el propio procesador incluye diversos temporizados y un reloj en tiempo real.

El Intel Mote 2 ofrece diversas opciones para su alimentación: desde aprovechar los 5V que proporciona el puerto USB hasta el uso de baterías recargables.

4.1.4 TelosB

Los nodos sensores TelosB son una plataforma para aplicaciones en redes de sensores de muy bajo consumo y alta recolección de datos. Llevan integrados tanto los sensores como la radio, antena y microcontrolador, además puede ser fácilmente programado [TelosB].

Las operaciones de baja energía en las Tmote Sky son realizadas gracias al microcontrolador MSP430 F1611. Este procesador RISC de 16 bits consume muy poca energía tanto en el estado activo como durante el estado de sueño o hibernación. Para reducir al máximo este consumo, permanece en modo sueño durante la mayoría del tiempo, se despierta tan rápido como puede para procesar, enviar y entonces vuelve a estado hibernación. Utiliza un controlador USB de Future Technology Devices International más conocida como FTDI para comunicarse con el ordenador y maneja una radio Chipcon CC2420, la cual es un estándar IEEE 802.15.4 que provee una fiable comunicación inalámbrica para redes de sensores inalámbricas. La radio tiene la posibilidad de enviar datos a muy alta frecuencia. El microcontrolador se comunica con la antena a través del puerto SPI y puede apagarlo para las operaciones de baja energía. La antena interna *'Invertid –F micro strip*' es una antena pseudo-omnidireccional que puede alcanzar los 50 metros dentro de un edificio y los 125 en el exterior.

Las características esenciales de TelosB son:

- Transmisor Chipcon inalámbrico de 250Kbps 2.4GHz IEEE 802.15.4.
- Interacción con otros dispositivos IEEE 802.15.4.
- Microcontrolador MSP430 de 8MHz (10Kb de
- RAM y 48 Kb de Flash).
- ADC, DAC, supervisor de voltaje y controladora DMA integrada.
- Antena, sensores de humedad, temperatura y luz.
- Muy bajo consumo.
- Rápido en despertar del modosueño (<6 μs).
- Hardware para encriptación y autentificación de la capa de enlace.
- Programación y recogida de datos por USB.
- 16 pines para soportar una expansión y conector de antena opcional.
- Conector SMA.
- Ayuda de TinyOS: enrutamiento de malla e implementación de las comunicaciones.

Figura 2.28: TelosB

Los sensores de humedad/temperatura están manufacturados por Sensiron AG, producidos con procesadores CMOS y emparejados con un dispositivo ADC de 14 bits. Los coeficientes para estos sensores se guardan en la EEPROM. La precisión del sensor es de 0.5 º C.

Esta plataforma consigue un bajo consumo de potencia permitiendo una larga vida a las baterías además de tener un tiempo mínimo en el estado de despertado, otro de los objetivos dentro de las estrategias de bajo consumo.

Los TelosB se alimentan de 2 baterías AA, aunque si está conectado mediante el puerto USB para programación o comunicación, la alimentación la proporciona el ordenador. También proporcionan la capacidad de añadir dispositivos adicionales. Los dos conectores de expansión de los que dispone pueden ser configurados para controlar sensores analógicos, periféricos digitales y displays LCD.

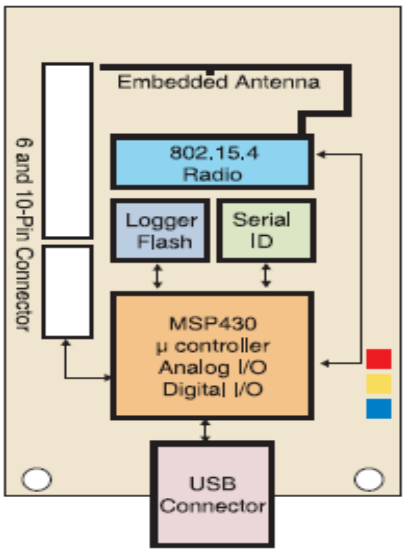


Figura 2.29: Diagrama de bloques

4.2. Soluciones de desarrollo

4.2.1 Soluciones basadas en el CC2530

El kit de desarrollo de CC2530DK soporta la segunda generación de Texas Instruments de 2.4 GHz IEEE 802.15.4, compatible con System-on-Chip, CC2530. Contiene todo el hardware, el software y las herramientas necesarias para construir la 802.15.4 compatible con el producto. El CC2530DK incluye módulos de evaluación, placas de desarrollo, una interfaz de candado USB, cables, antenas y documentación para empezar enseguida el kit de desarrollo. Los módulos de evaluación CC2530EM puede colocarse en las placas SmartRF05EB, que se incluyen en el kit de desarrollo.

Los módulos de evaluación pueden ser utilizados como un módulo de referencia para la creación de prototipos y para verificar el rendimiento del circuito integrado de radio frecuencia CC2530. El Sistema en chip de radio frecuencia CC2530 permite aplicaciones domesticas e industriales de calidad, ofreciendo selectividad en el estado del arte / co-existencia, excelente balance de enlace, una temperatura operativa alrededor de los 125 ° C y tensión operativa baja. El CC2530DK viene con varios ejemplos de software, incluyendo el IEEE 802.15.4 compatible con *TIMAC*, *Z-Stack* compatible con *ZigBee*, propiedad del protocolo de creación de redes *SimpliciTI*, test tipo de paquetes de errores, pruebas de enlace simples, librerías USB '*firmware'* y ejemplos para el CC2531.

El kit del CC2530DK contiene lo siguiente:

- 2 x módulos de evaluación CC2530EM.
- 2 x Antenas 2,4 GHz.
- 2 x Placas de evaluación SmartRF05EB.
- 1 x *Dongle* USB CC2531.
- Cables USB y documentación (manuales de uso).

Figura 2.30: Kit de desarrollo del CC2530

El CC2530DK está apoyado por el compilador en C *IAR EW8051*, que se utiliza con el *TIMAC*, *ZStack*, *SimpliciTI*, y ejemplos de aplicación. El depurador C-SPY se utiliza como interfaz en el emulador. Tanto el C-compilador y depurador C-SPY se incluyen en el kit de desarrollo como licencias de evaluación de 30 días.

4.2.2 Soluciones basadas en el CC2520

El kit de desarrollo CC2520 incluye hardware y software que permite probar rápidamente el rendimiento del CC2520 RF y ofrece una plataforma completa para el desarrollo de prototipos de sistemas avanzados de radio frecuencia. El kit puede ser utilizado para una serie de pruebas usando el comprobador pre-programado PER que se ejecuta en el MSP430F2618.

Todas las entradas/salidas del CC2520 están disponibles en conectores de pin en el SmartRF ® 05EB y en el CCMSP-EM, lo que permite una fácil interconexión con otros dispositivos o controladores. Estos conectores también son compatibles con las sondas del analizador lógico para la depuración fácil.

El SmartRF05 + CC2520EM puede ser utilizado como un dispositivo de captura para el analizador de paquetes SmartRF.

El kit del CC2530DK contiene lo siguiente:

- 3 x SmartRF05EB.
- 3 x módulos de evaluación CC2520EM.
- 3 x Antenas.
- 2 x CCMSP-EM430F2618.
- 1 x interfaz de depuración MSP-FET430UIF.
- 3 x cable USB.

- Documentación (manuales de uso).

Figura 2.31: Kit de desarrollo del CC2520

4.2.3 Soluciones basadas en el JN5148

El JN5148 es el más reciente microcontrolador inalámbrico de bajo coste de consumo de energía ultra-bajo, para aplicaciones de redes inalámbricas de sensores basadas en el estándar IEEE802.15.4 incluyendo *JenNet*, *JenNet-IP* y aplicaciones *ZigBee PRO*.

Estos tienen un procesador RISC de 32-bit con una mayor eficiencia de codificación y un rendimiento de hasta 32MIPs, un transceptor de 2.4GHz, 128kB de ROM y 128kB de RAM para apoyar tanto la creación de redes con protocolo Stack y la aplicación de usuario en el chip como una generosa combinación de periféricos de usuario.

El nivel más alto de integración on-chip también significa que el dispositivo ofrece nuevas e innovadoras mejoras como:

- La interfaz digital de audio de 4 cables que será conectada directamente al conjunto de codecs de audio.

- Contadores de bajo consumo diseñados para contar pulsos en aplicaciones AMR.

Figura 2.32: JN5148

El JN5148 establece nuevos puntos de referencia para el consumo de energía más bajo, la más alta capacidad de procesamiento y los más altos niveles de integración en un microcontrolador inalámbrica de un solo chip.

El kit de desarrollo JN5148-EK010 proporciona un entorno completo para el desarrollo de aplicaciones IEEE 802.15.4, *JenNet* y *ZigBee PRO* basadas en el controlador inalámbrico JN5148. El kit contiene un paquete completo de hardware y software, incluyendo 5 nodos sensores inalámbricos y una aplicación de demostración preinstalada. El kit de desarrollo software (SDK) proporciona un completo set de herramientas de desarrollo, librerías software e interfaces de programación de aplicaciones (API) para programar y controlar los periféricos del chip del JN5148, para configurar y gestionar la red y para la comunicación de datos.

El kit del JN5148 contiene lo siguiente:

- Módulos JN5148:
- 2 antenas *power-standard* PCB
- 3 conectores *power-standard* Ufi
- Una placa, sensores de nivel de luz y humedad.
- Puerto de expansión OI JN5148.
- 2 cables USB para conexión con el PC.
- Baterías o suministro de energía externa.
- 1 nodo o *'mote'* con LCD

Figura 2.33: Kit de desarrollo del JN5148

5. Agricultura de precisión

La agricultura de precisión comenzó a desarrollarse en EEUU a principios de los años 80 gracias a los avances de las Tecnologías de la Información y las Comunicaciones (sistemas de localización de bajo coste, potentes ordenadores, sistemas de información geográfica, etc.) [Lowenberg-DeBoer, 2000].

Hacia finales de los años 80 y gracias a las extracciones realizadas mediante muestras, aparecieron los primeros mapas de preconización para las aportaciones moduladas de elementos fertilizados y para las correcciones de pH. La evolución de las tecnologías permitió el desarrollo de sensores de rendimiento y su uso, unido a la aparición del GPS, no ha dejado de crecer hasta alcanzar en la actualidad varios millones de hectáreas cubiertos por estos sistemas. A través del mundo, la agricultura de precisión se desarrolla a ritmos diferentes en función de los países.

La agricultura de precisión es un concepto agronómico de gestión de parcelas agrícolas, basado en la existencia de variabilidad en campo. Requiere el uso de las tecnologías de Sistemas de Posicionamiento Global (GPS), sensores, satélites e imágenes aéreas junto con Sistemas de Información Geográfico (SIG) para estimar, evaluar y entender dichas variaciones. La información recolectada puede ser usada para evaluar con mayor precisión la densidad óptima de siembra, estimar fertilizantes y otras entradas necesarias, y predecir con más exactitud la producción de los cultivos.

La agricultura de precisión tiene como objeto optimizar la gestión de una parcela desde el punto de vista:

- Agronómica: ajuste de las prácticas de cultivo a las necesidades de la planta (ej.: satisfacción de las necesidades de nitrógeno).
- Medioambiental: reducción del impacto vinculado a la actividad agrícola (ej.: limitaciones de la dispersión del nitrógeno).
- Económico: aumento de la competitividad a través de una mayor eficacia de las prácticas (ej.: mejora de la gestión del coste del estiércol nitrogenado).

Además, la agricultura de precisión pone a disposición del agricultor numerosas informaciones que pueden:

- Constituir una memoria real del campo.
- Ayudar a la toma de decisiones.
- Ir en la dirección de las necesidades de rastreabilidad.

A su vez, el agricultor debe llevar a cabo una serie de fases para poder llegar a la toma de decisiones de una zona homogénea concreta de la explotación agrícola. Estas fases son las siguientes [Anurag, 2008]:

- Recolección de datos.
- Análisis de los datos.
- Toma de decisiones.
- Actuación.

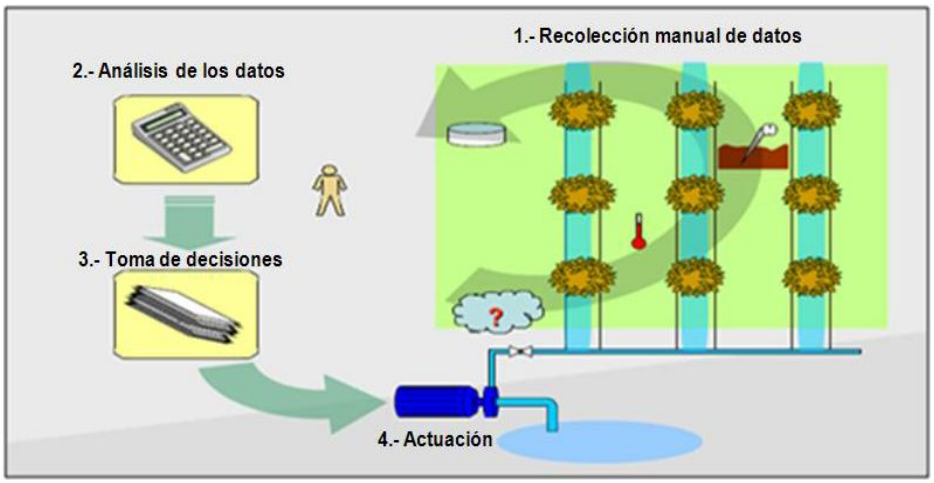


Figura 2.34: Fases que sigue la agricultura de precisión

La primera fase consiste en realizar la recogida de datos del cultivo, del ambiente, de las plantas, etc. Esta primera fase se puede realizar de forma manual o bien de manera automática, haciendo uso de una o varias determinadas tecnologías. La segunda fase consiste en analizar los datos para poder llevar a cabo la toma de decisiones en la tercera fase. La cuarta y última fase, es la encargada de cerrar el lazo del sistema. En esta fase se controlan los diferentes actuadores y/o maquinaría agrícola.

Existen tres cuestiones clave que es necesario tener en cuenta antes de emplear la agricultura de precisión [Basso, 2006]:

- Determinar la variabilidad de las características del terreno y de las plantas.
- Análisis de la influencia de las variaciones anteriores en la cantidad y la calidad de la producción.
- Selección de la tecnología a emplear para tratar adecuadamente la variabilidad existente.

5.1. Tecnologías de la información y la comunicación (TIC)

Los sistemas de producción agrícolas se han beneficiado de la incorporación de avances tecnológicos desarrollados inicialmente para otras industrias. La era industrial trajo la mecanización y fertilizantes sintetizados a la agricultura. La era tecnológica ofreció las ventajas de la aplicación de la ingeniería genética y la automatización. La era de la información trajo el potencial necesario para integrar los avances tecnológicos en la agricultura [Zhang, 2002].

Las tecnologías de la información y la comunicación (TIC) agrupan los elementos y las técnicas usadas en el tratamiento y la transmisión de la información, principalmente la informática, Internet y las telecomunicaciones.

El principal aporte del uso de las nuevas tecnologías en la agricultura es la generación de mapas del terreno, que representan gráficamente la variabilidad de parámetros de los cultivos, como por ejemplo, el rendimiento de la producción de los cultivos. La generación de mapas óptimos, así como de modelos de los campos agrícolas, ha sido posible gracias a los sistemas remotos y de posicionamiento global. A su vez, el muestreo se realiza generalmente a través de sensores electrónicos, tales como sondas de suelo y escáneres remotos ópticos localizados en satélites. La recopilación de esos datos en forma de bases de datos informáticas electrónicas dio lugar a los sistemas de información geográfica. Las tecnologías mencionadas, además de no tener características de tiempo real, conllevan el uso de tecnologías costosas como el satélite de detección, e implicó el uso de mano de obra intensiva, ya que, en la mayoría de los casos, los mapas de los campos agrícolas se realizan de forma manual.

Hoy en día, no es extraño disponer de un servidor web que ofrezca información sobre el estado de las plantas, principalmente sobre plagas y advertencia de posibles enfermedades, así como de pronósticos del tiempo.

El uso de las TIC es beneficioso para la agricultura, no sólo a nivel de obtener información, sino también permitiendo acceso remoto a los cultivos con objeto de monitorizarlos y poder controlarlos en tiempo real.

A continuación veremos algunos ejemplos de TICs.

5.1.1. Sistema de posicionamiento global (GPS)

El sistema de posicionamiento global es un sistema global de navegación por satélite (GNSS) que permite determinar en todo el mundo la posición de un objeto, una persona o un vehículo con una precisión de centímetros (si se utiliza GPS diferencial), aunque lo habitual son unos pocos metros de precisión. El sistema fue desarrollado, instalado y actualmente operado por el Departamento de Defensa de los Estados Unidos [Wang, 2001].

Los sistemas disponibles hoy en día son los siguientes: sistema de posicionamiento global y navegación estadounidense (NAVSTAR-GPS), sistema ruso global de navegación por satélite (GLO-NASS) y servicio europeo geoestacionario de navegación (EG-NOS).

El GPS funciona mediante una red de 24 satélites en órbita sobre el planeta tierra, a 20.200 km, con trayectorias sincronizadas para cubrir toda la superficie de la Tierra. Cuando se desea determinar la posición, el receptor que se utiliza para ello localiza automáticamente como mínimo tres satélites de la red, de los que recibe unas señales indicando la identificación y la hora del reloj de cada uno de ellos. Con base en estas señales, el aparato sincroniza el reloj del GPS y calcula el tiempo que tardan en llegar las señales al equipo, y de tal modo mide la distancia al satélite mediante "triangulación, la cual se basa en determinar la distancia de cada satélite respecto al punto de medición. Conocidas las distancias, se determina fácilmente la propia posición relativa respecto a los tres satélites. Conociendo además las coordenadas o posición de cada uno de ellos por la señal que emiten, se obtiene las posiciones absolutas o coordenadas reales del punto de medición.

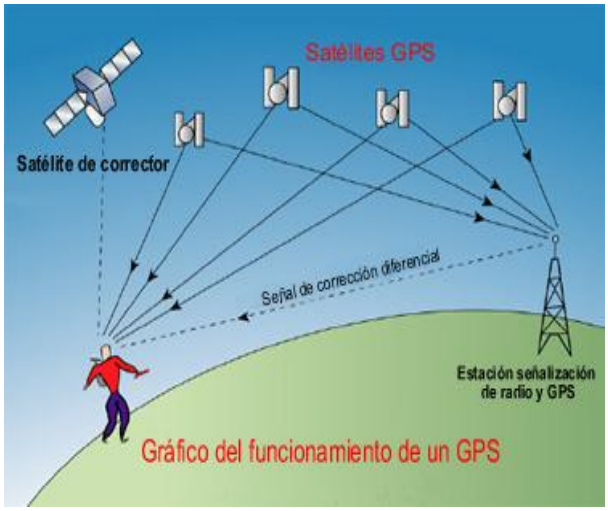


Figura 2.35: Funcionamiento del GPS

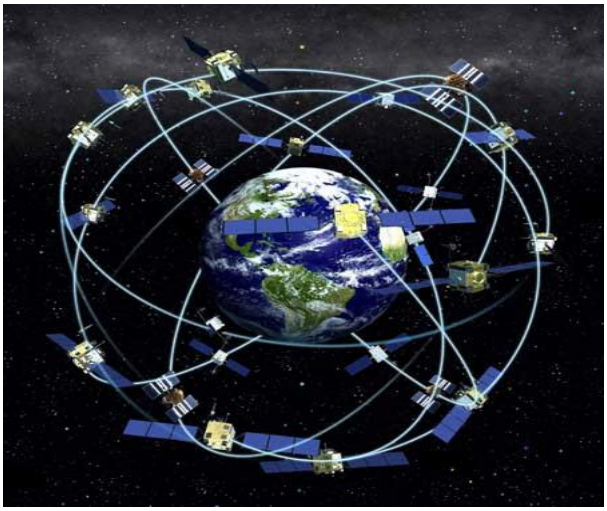


Figura 2.36: Posición de los satélites del sistema GPS

Las aplicaciones de los sistemas de posicionamiento global y de navegación en la agricultura pueden ser muy diversas. No obstante, las más comunes que se pueden citar son la determinación de los límites de la finca, guiado automático de maquinaría agrícola, asignación de las coordenadas a las muestras tomadas con objeto de elaborar los mapas de producción u otra característica y determinar la actuación en cada punto.

Un ejemplo de esto último, sería distribuir la dosis de fertilizantes y fitosanitarios en función de las coordenadas [Zhang, 2002].

En la misma línea, los sistemas de navegación también resultan muy útiles en la agricultura. Estos sistemas son muy necesarios en explotaciones grandes a la hora de aplicar los tratamientos fitosanitarios y los fertilizantes. En estas tareas se emplea maquinaría agrícola pesada, y pequeños errores en la conducción pueden provocar que una franja del cultivo quede sin tratar (pérdidas de producción), que se aplique una sobredosis (perjuicios medioambientales y económicos), etc. Los sistemas de navegación también resultan de gran utilidad en casos concretos, como la aplicación de los herbicidas sistémicos que se deben aplicar durante el crespúsculo, así como en el guiado del vehículo cuando las condiciones atmosféricas son adversas. Por ello, no es de extrañar que numerosos investigadores trabajen en este campo.

5.1.2. Sistema de Información Geográfica (SIG)

Al aplicar los métodos de la agricultura de precisión en una explotación agrícola se recogen gran cantidad de datos del cultivo. A su vez, los datos están referidos a unas determinadas posiciones geográficas unívocas, gracias al uso de los sistemas de posicionamiento global. Por lo tanto, para que los datos recogidos sean de utilidad, es necesario realizar una representación gráfica adecuada, y es en este punto donde intervienen los sistemas de información geográfica**.**

Un Sistema de Información Geográfica es una integración organizada de hardware, software y datos geográficos diseñada para capturar, almacenar, manipular, analizar y desplegar, en todas sus formas, la información geográficamente referenciada con el fin de resolver problemas complejos de planificación y gestión geográfica. También puede definirse como un modelo de una parte de la realidad referido a un sistema de coordenadas terrestre y construido para satisfacer unas necesidades concretas de información [Earl, 2000].

Como resultado, se obtiene un mapa en formato digital sobre el que se sitúan elementos de los datos espaciales o atributos, que pueden estar clasificados por capas. Además, existe una base de datos asociada a los atributos con una descripción de los mismos. Al final, un SIG puede combinar datos geográficos con datos de otro tipo para generar mapas y esquemas, permitiendo al usuario recoger, gestionar e interpretar de forma planificada y sistemática la información relativa a lugares específicos.

5.1.3. Teledetección

La teledetección o detección remota es la adquisición de información a pequeña o gran escala de un objeto o fenómeno, ya sea usando instrumentos de grabación o instrumentos de escaneo en tiempo real inalámbricos o que no están en contacto directo con el objeto.

Hay dos clases de teledetección principalmente: teledetección pasiva y teledetección activa:

- Los teledetectores pasivos detectan radiación natural emitida o reflejada por el objeto o área circundante que está siendo observada. La luz solar reflejada es uno de los tipos de radiación más comunes medidos por esta clase de teledetección. Algunos ejemplos pueden ser la fotografía, los infrarrojos, los sensores CCD (dispositivo de cargas eléctricas interconectadas) y los radiómetros.

- Los teledetectores activos por otra parte emiten energía para poder escanear objetos y áreas con lo que el teledetector mide la radiación reflejada del objetivo. Un radar es un ejemplo de teledetector activo, el cual mide el tiempo que tarda una onda entre la emisión y la recepción a un punto, estableciendo así la localización, altura, velocidad y dirección de un objeto determinado. La teledetección remota hace posible recoger información de áreas peligrosas o inaccesibles.

La teledetección unida con los SIG tiene múltiples aplicaciones en la agricultura, entre las que se pueden citar la identificación de las parcelas y los tipos de cultivo, valorar los daños por heladas, granizos, fuertes lluvias y otros efectos meteorológicos adversos para los cultivos, así como determinar el estado de la producción. Son, por lo tanto, métodos y técnicas para monitorizar cultivos a gran escala, y que no están pensados para pequeñas parcelas.
En los últimos años, los avances en la detección y las comunicaciones así como la previsible reducción de los costes de implementación y funcionamiento de estos sistemas hacen prever que serán utilizados masivamente en la agricultura de precisión. En esta línea, se han desarrollado nuevas tecnologías inalámbricas con bajas necesidades de potencia y de capacidad de cómputo de datos, que pueden ser aplicables en la agricultura de precisión. Esta tecnología, surgida gracias a los avances en los micro-sensores, la miniaturización de dispositivos, las comunicaciones inalámbricas y las redes ad-hoc se conoce como las redes de sensores inalámbricas (WSN, Wireless Sensor Networks) [López, 2011].

CAPÍTULO 3

DESCRIPCIÓN DEL SISTEMA

1. Introducción

En el capitulo anterior, se ha descrito todo el estado técnico de los nodos sensores y su funcionamiento con el fin de entender algo más acerca de los nodos sensores y todo lo que les rodea.

A partir de aquí, se explicarán todos los detalles relativos al sistema utilizado en este proyecto, incluyendo elementos tanto de software como de hardware y donde se exponen sus características, métodos de funcionamiento e instalación, tanto de los componentes como de los programas necesarios para su programación.

2. Descripción Hardware

2.1. Kit de Desarrollo CC2530

Como se ha explicado en el capítulo anterior, el CC2530 es un microcontrolador que permite construir nodos de red robustos a un coste muy bajo. Es muy utilizado en aplicaciones como controles remotos de Radio Frecuencia (RF), control industrial, electrónica de consumo. Cuenta con estándares como IEEE 802.15.4, Zigbee.

2.1.1. Descripción de la placa de desarrollo SmartRF05EB

Para la programación del CC2530 se emplea la Placa de Evaluación *SmartRF05EB*. Esta placa actúa como interfaz entre el PC y los motes a través del puerto serie, permitiendo la programación de los dispositivos acoplados.

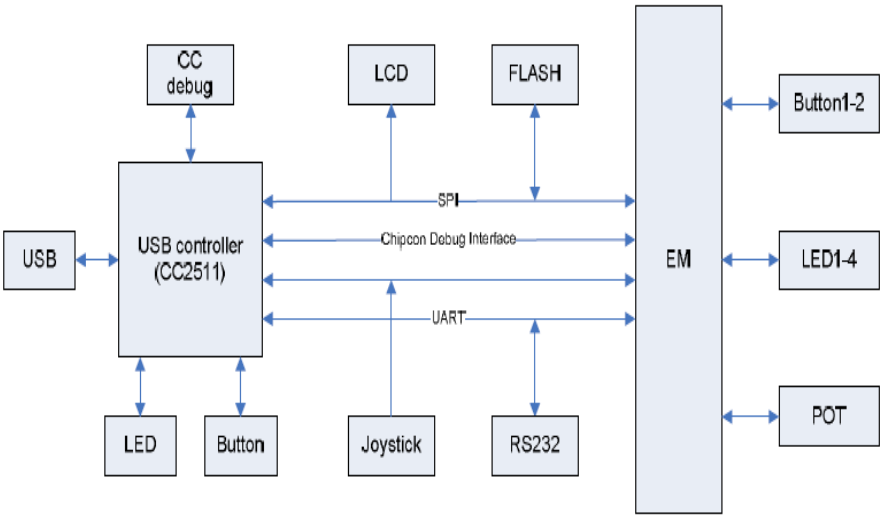


Figura 3.1: Arquitectura *SmartRF05EB*

El principal componente de la placa es el controlador USB. Se comunica con el PC a través del USB y traduce las peticiones de los distintos instrumentos del PC a varias acciones.

El controlador USB se comunica con el módulo de evaluación usando SPI, UART y / o la interfaz de depuración.

El módulo conectado al conector EM tiene un acceso potencial a todos los dispositivos periféricos EB. Cuenta con acceso completo a la pantalla LCD, flash de serie, cuatro LEDs, 2 botones, joystick y la interfaz UART RS232.

2.1.1.1. MCU USB

El controlador USB es programado con un gestor de arranque y el estándar firmware *SmartRF05EB*. Cuando el gestor de arranque empieza a funcionar, se comprobará para una aplicación válida en el CC2511. Si la detección es satisfactoria, la aplicación se inicia y la placa puede ser usada con normalidad. Si la aplicación no es detectada, el LED USB empezará a parpadear rápidamente indicando fallo.

La aplicación estándar *firmware* es usada para controlar el dispositivo RF en el adjunto módulo de Evaluación y para comunicarse con las aplicaciones que se ejecutan en la PC a través de USB.

2.1.1.2. Fuentes de Alimentación

Hay pocas soluciones posibles para alimentar una *SmartRF05EB*.

La fuente de alimentación puede ser seleccionada usando el conector de selección en las cabeceras de P11.

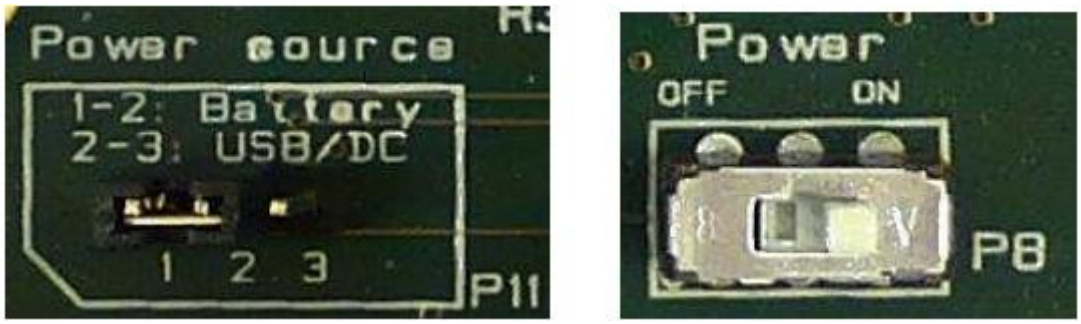


Figura 3.2: Puente principal de selección de energía (P11) y el interruptor de encendido (P8)

El interruptor de alimentación principal (P8) apaga todas las fuentes de energía, a menos que una fuente de alimentación externa está conectada a la placa, anulando los reguladores de tensión de la placa.

2.1.1.3. Batería de Alimentación

La *SmartRF05EB* incluye un zócalo para 2 baterías de 1.5V AA en el reverso. La selección de la fuente de alimentación debe hacerse cortocircuitando con el jumper el pin 1 y el 2 de la cabecera P11. Un LED de la placa se encenderá cuando el voltaje de esta oscile por debajo de 1.56 V. Téngase en cuenta que esta función solo se activa cuando la placa es alimentada por baterías.

2.1.1.4. Conector DC (DC *Jack*)

SmartRF05EB tiene un conector estándar conectores DC de alimentación con un pin central de 2.5mm. El pin central es usado para el voltaje positivo. El regulador de voltaje integrado aplica aproximadamente 3.3 V a la placa. La selección de la fuente de alimentación debe hacerse cortocircuitando con el jumper el pin 2 y el 3 de la cabecera P11.

2.1.1.5. Alimentación USB

Cuando *SmartRF05EB* es conectado al PC vía cable USB, se puede extraer energía del mismo. El regulador de tensión integrado suministra aproximadamente 3,3 V a la placa.

2.1.2. CC2530EM

El CC22530EM es un completo módulo de Radio Frecuencia (RF) basado en uno de los diseños de referencia recomendados para la radio CC2530. El módulo está dotado con un oscilador de 32 MHz, un oscilador de 32.768 KHz , componentes pasivos externos para la bobina y la antena de filtro de encuentro, un conector SMA para la antena u cualquier instrumento de conexión de Radio Frecuencia (RF) y general entrada salida de cabeceras/conectores.

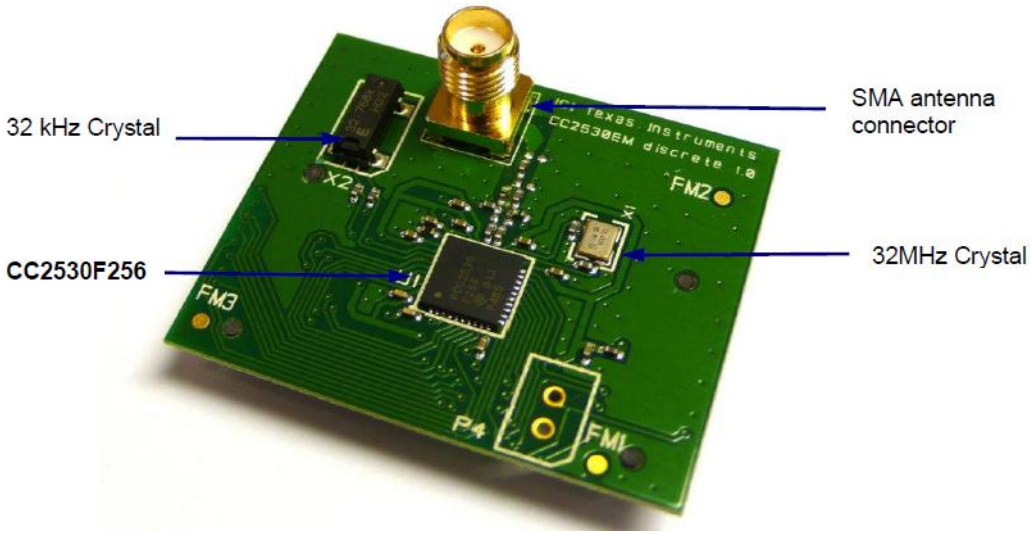


Figura 3.3: CC2530EM

La siguiente tabla muestra el pin de salida del CC2530 para los dos conectores de la parte posterior del modulo de evaluación.

CC2530 Signal	P1	P1	CC2530 Signal
GND	1	2	NC
P0.4	3	4	P1.3
P0.1	5	6	P1.0
P0.2	7	8	NC
P0.3	9	10	P2.1
P0.0	11	12	P2.2
P1.1	13	14	P1.4
P0.6	15	16	P1.5
P0.7	17	18	P1.6
GND	19	20	P1.7

CC2530 Signal	P2	P2	CC2530 Signal
NC	1	2	NC
NC	3	4	NC
NC	5	6	NC
VDD	7	8	NC
VDD	9	10	NC
NC	11	12	NC
NC	13	14	NC
RESET	15	16	NC
P1.2	17	18	P0.5
P2.0	19	20	NC

Tabla 3.1: Pin de salida del CC2530EM

2.1.3. Adaptador USB

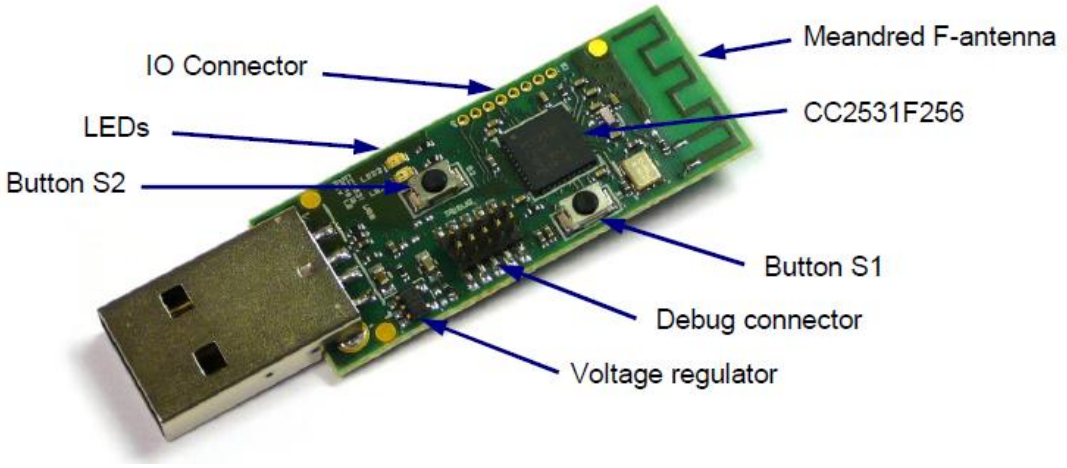


Figura 3.4: Adaptador USB

El adaptador USB que es incluido en el kit viene programado de tal manera que puede ser usado junto con el *SmartRF Packet Sniffer* para captar paquetes que vayan por el aire. Para usar el adaptador como un rastreador solo se instala la aplicación *Packet Sniffer* (Analizador de Paquetes) en el PC, se conecta el adaptador USB y comienza la captura de paquetes.

El adaptador USB también se puede utilizar como una tarjeta de desarrollo general para USB y el software de RF.

Con el fin de depurar y programar el CC2530, el adaptador USB puede ser conectado a la *SmartRF05EB* como se muestra en la figura 3.6. La pequeña placa de adaptador y el cable plano están incluidos en el kit de desarrollo.

Figura 3.5: Adaptador USB conectado con *SmartRF05EB*

2.2. Kit de Desarrollo CC2520

2.2.1. CCMSP-EM430F2618

Esta es una placa de desarrollo del microcontrolador *MSP430F2618* (*CCMSP-EM430F2618*). La placa MCU puede estar conectada en el *SmartRF05EB* y es compatible con la mayoría de los módulos de evaluación TI LPW RF-IC.

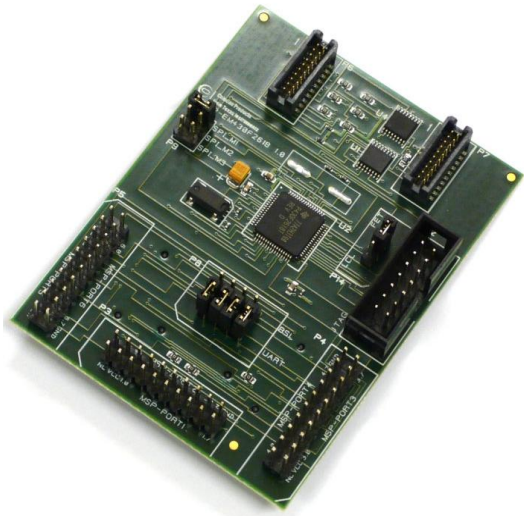


Figura 3.6: CCMSP-EM430F2618

2.2.2. CC2520EM

Este es el módulo de evaluación CC2520 con un circuito integrado de radio frecuencia y componentes externos necesarios y combinar filtros para sacar el máximo partido de la radio. El módulo puede ser conectado en el *SmartRF05EB* para el control de *SmartRF Studio* o en el *CCMSP-EM430F2618* para el control del MSP430.

Figura 3.7: CC2520EM

2.2.3. MSP-FET430UIF

Este es la interfaz de depuración USB MSP430 para programar y depurar aplicaciones que se ejecutan en el MSP430F2618. Se conecta al PC vía USB y usa JTAG para comunicarse con el microcontrolador.

Figura 3.8: MSP-FET430UIF

La siguiente figura 3.9 muestra las tres placas diferentes ensambladas y conectadas al programador, que constituye un nodo sensor completo con el microcontrolador MSP430F2618 y el módulo de radio CC2520.

Figura 3.9: Nodo sensor completo conectado al MSP-FET430UIF

Una vez que se ha explicado a nivel hardware todos los componentes utilizados para encontrar una solución adecuada al proyecto, se procede a describir a nivel software dicha solución. Dado que el problema se plantea en una parcela donde se encuentra una plantación de almendros y el desplazamiento continuado para realizar las pruebas es imposible, se ha simulado la situación en el laboratorio. En la figura 3.10 se puede observar la situación en el laboratorio. A continuación explicaremos el software utilizado en el desarrollo de la solución.

Figura 3.10: Conjunto de placas conectadas

3. Función de cada nodo sensor en la red inalámbrica

La función que cumple cada nodo sensor es importante para que se establezca correctamente la red inalámbrica entre ellos. Mediante el esquema representado en la figura 3.11 se explicará la función de cada nodo sensor. En cuando al "Nodo Coordinador", éste recibe los datos registrados por los dendrómetros del "Nodo Sensor 1" y del "Nodo Sensor 2", además recibirá el estado de la batería de los dos nodos sensores. Otra función que tiene el "Nodo Coordinador" es que mediante el joystick permite ajustar la frecuencia con que los datos serán recibidos. En cuanto al "Nodo Sensor 1" y el "Nodo Sensor 2", los dos realizan las mismas funciones por lo que se explicará el "Nodo Sensor 1". Éste enviará los datos y el estado de la batería al "Nodo Coordinador" y sólo los datos al "Nodo de Configuración". En cuanto al "Nodo de Configuración", se corresponde con el dispositivo que ajusta la posición de los dendrómetros. La función de éste es recibir los datos de los dos nodos sensores, previamente seleccionado con el joystick, y mediante éste mismo se ajusta la frecuencia con la que los datos serán recibidos. Otra función del "Nodo de Configuración" es ajustar el modo de los dos nodos sensores, ya sea modo normal o modo ajuste.

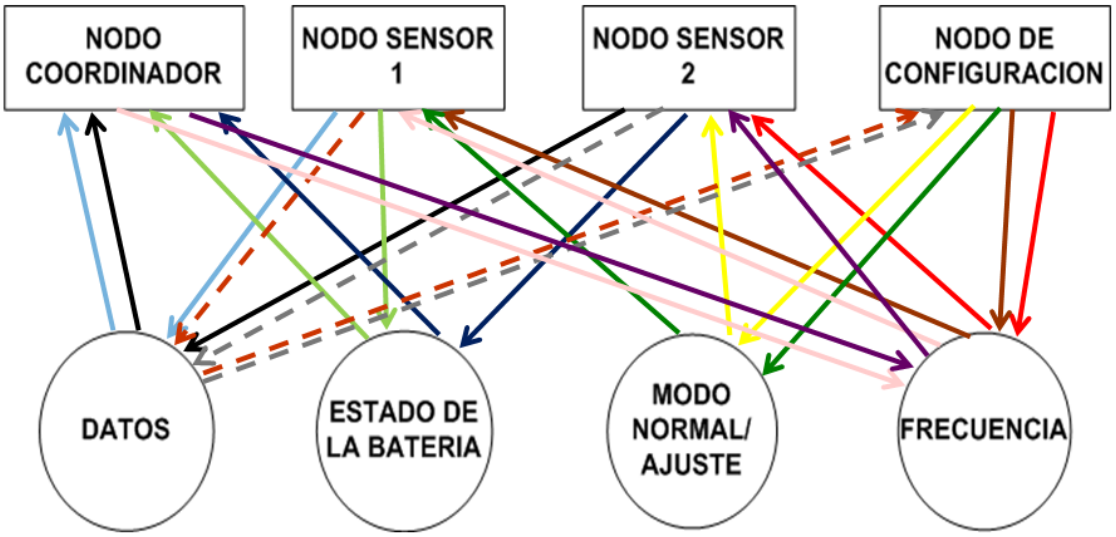


Figura 3.11: Función de cada nodo sensor en la red inalámbrica

4. Descripción Software

4.1 Introducción

Habiéndose diseñado y comprobado los componentes utilizados en el desarrollo del proyecto, queda el apartado más importante, que es la programación de los nodos sensores. En esta parte se desarrollarán y se explicarán las partes más importantes del código en el sistema operativo IAR Embedded Workbench que se cargarán en los diferentes nodos sensores que forman el sistema completo. Se empleará para ello los códigos de los cuatro motes usados y descritos en capítulos anteriores.

4.2. IAR Embedded Workbench

La biblioteca Z-Stack está desarrollada en lenguaje C con el entorno IAR Embedded Workbench, desarrollado y comercializado por IAR Systems Se trata de un entorno de desarrollo de software que permite edición, compilación, descarga y depuración del código desarrollado en lenguaje C. El entorno no incluye ningún simulador. Sin embargo, el código se puede verificar realizando una simulación real en el micro-controlador (depuración). El entorno está disponible para diferentes familias de micro-controladores. Existen diferentes versiones compatibles con el 8051, la familia Atmel AVR y la serie MSP430, entre otros.

La figura 3.10 presenta la vista de la interfaz de este entorno de desarrollo de software. En ella, se puede observar en la parte superior izquierda el árbol de archivos que compone el proyecto, la ventana de edición a la derecha y la ventana de salida en la parte inferior.

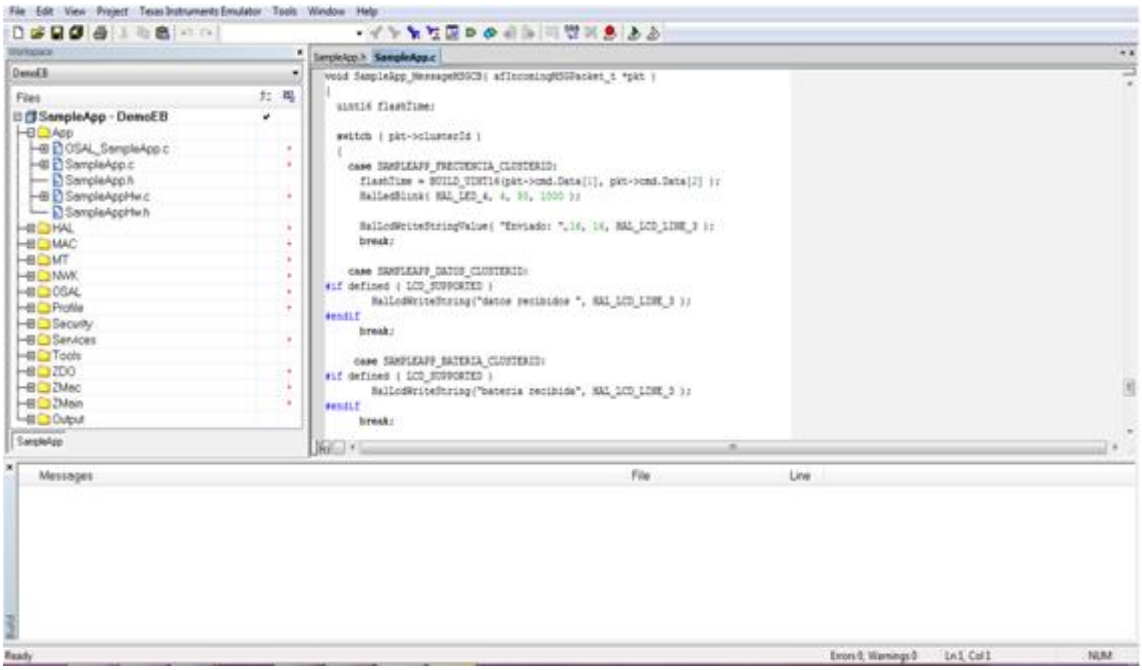


Figura 3.12: Entorno de compilación y depuración IAR Embedded Workbench

Una vez explicado el entorno que se va a utilizar para desarrollar el software de los nodos sensores, se explicará la función de cada uno y las principales estructuras que se usan en su código.

Nuestro sistema contará con cuatro nodos sensores. El primero, el "Nodo Coordinador", recibe los datos y el estado de la batería de los dos nodos sensores que están conectados a los dendrómetros. Los nodos sensores, "Nodo Sensor 1" y "Nodo Sensor 2", mandan los datos y el estado de la batería al "Nodo Coordinador". Además los datos se envían paralelamente al "Nodo de Configuración". Este último ajusta la frecuencia con la que se quiere recibir los datos y de que nodo sensor. La programación de estos dispositivos se expondrá más adelante.

4.3. Programación del Nodo Coordinador

Para empezar la descripción de la programación del "Nodo Coordinador" se debe tener en cuenta que en el entorno en el que se desarrollará el código habrá que programar dos archivos importantes. Por un lado, "SampleApp.h", donde simplemente se declaran todos los "clúster" de entrada y de salida y los eventos. Por otro lado, "SampeApp.c" donde se desarrolla todo el código.

Partiendo con el "SampleApp.h", en el listado 3.1 se observa la declaración de los "clúster" de entrada: "`SAMPLEAPP_DATOS_CLUSTERID`" (línea 11) y "`SAMPLEAPP_BATERIA_CLUSTERID`" (línea 12), y de salida: "`SAMPLEAPP_FRECUENCIA_CLUSTERID`" (línea 14). En este caso la función del "Nodo Coordinador" es recibir los datos, el estado de la batería y enviar la frecuencia.

```
1  * CONSTANTS
2   */
3  // These constants are only for example and should be changed to the
4  // device's needs
5  #define SAMPLEAPP_ENDPOINT           20
6  #define SAMPLEAPP_PROFID             0x0F08
7  #define SAMPLEAPP_DEVICEID           0x0001
8  #define SAMPLEAPP_DEVICE_VERSION     0
9  #define SAMPLEAPP_FLAGS              0
10 #define SAMPLEAPP_MAX_IN_CLUSTERS     2
11 #define SAMPLEAPP_DATOS_CLUSTERID      1
12 #define SAMPLEAPP_BATERIA_CLUSTERID    2
13 #define SAMPLEAPP_MAX_OUT_CLUSTERS        1
14 #define SAMPLEAPP_FRECUENCIA_CLUSTERID    3
```

Listado 3. 1: SampleApp.h

Ahora analizando "SampleApp.c" en el listado 3.2 se describen todos los "#define" y las bibliotecas que se incluyen en el archivo, la librería "`stdio.h`" (línea 13) y "`string.h`" en la (línea 14) para el uso de las cadenas de caracteres tipo string.

El resto de las líneas corresponden a definiciones globales de variables, en este caso suele tratarse de variables relacionadas con las direcciones que se utilizarán posteriormente.

```
1  #include "OSAL.h"
2  #include "ZGlobals.h"
3  #include "AF.h"
4  #include "aps_groups.h"
5  #include "ZDApp.h"
6  #include "SampleApp.h"
7  #include "SampleAppHw.h"
8  #include "OnBoard.h"
9  /* HAL */
10 #include "hal_lcd.h"
11 #include "hal_led.h"
12 #include "hal_key.h"
13 #include <stdio.h>
14 #include <string.h>
```

Listado 3. 2: Librerías incluidas

Como especifica Zigbee, los mensajes que envía y recibe un nodo se definen en un "clúster". El "Nodo Coordinador" recibe los datos y el estado de la batería que le envían el "Nodo Sensor 1" y el "Nodo Sensor 2".Ademas el "Nodo Coordinador" envía la frecuencia. En el listado 3.3 se observa los "clúster" de entrada y de salida (líneas 4,5 y 9).

```
1   // This list should be filled with Application specific Cluster IDs.
2   const cId_t SampleApp_ClusterListIn[SAMPLEAPP_MAX_IN_CLUSTERS] =
3   {
4     SAMPLEAPP_DATOS_CLUSTERID,
5     SAMPLEAPP_BATERIA_CLUSTERID,
6   };
7   const cId_t SampleApp_ClusterListOut[SAMPLEAPP_MAX_OUT_CLUSTERS] =
8   {
9     SAMPLEAPP_FRECUENCIA_CLUSTERID
10  };
11  const SimpleDescriptionFormat_t SampleApp_SimpleDesc =
12  {
13    SAMPLEAPP_ENDPOINT,              //  int Endpoint;
14    SAMPLEAPP_PROFID,                //  uint16 AppProfId[2];
15    SAMPLEAPP_DEVICEID,              //  uint16 AppDeviceId[2];
16    SAMPLEAPP_DEVICE_VERSION,        //  int   AppDevVer:4;
17    SAMPLEAPP_FLAGS,                 //  int   AppFlags:4;
18    SAMPLEAPP_MAX_IN_CLUSTERS,          //  uint8  AppNumInClusters;
19    (cId_t *)SampleApp_ClusterListIn,  //  uint8 *pAppInClusterList;
20    SAMPLEAPP_MAX_OUT_CLUSTERS,          //  uint8  AppNumInClusters;
21    (cId_t *)SampleApp_ClusterListOut   //  uint8 *pAppInClusterList;
22  };
```

Listado 3. 3: Declaración de los "clúster" en SampleApp.c

En el listado 3.4, por un lado el comando "`afAddrType_t`" utiliza variables de salida, el cual indica los datos que se enviarán, más no el valor. Por el otro, dos estructuras globales las cuales se utilizarán posteriormente.

La primera sería la estructura "Ddendrometros" (la línea 3), utilizada posteriormente para declarar los valores obtenidos por el "Nodo Sensor 1" y el "Nodo Sensor 2", de ahí que se utilice un formato uint_16 y los valores que tomará serán: "d1,d2,d3 y d4" haciendo referencia a los dendrómetros (la línea 5). Posteriormente se usa un formato uint_8 (línea 6) y los valores que tomará serán:

"Nodo" haciendo referencia al nodo sensor seleccionado. Tendrá como tipo "Dendrometros" la estructura descrita anteriormente (la línea 8).

También es utilizada la estructura "Bat" (línea 14), la cual será de tipo "Bateria" (línea 14) y tendrá una variable de tipo uint_16 (la línea 11) etiquetada como "bateria" y otra uint_8 (línea 12) etiquetada como "Nodo" y que simplemente darán el valor de la batería del nodo sensor correspondiente.

```
1   afAddrType_t SampleApp_FRECUENCIA_DstAddr;
2   afAddrType_t SampleApp_MODO_DstAddr;
3   typedef struct
4   {
5    uint16 d1, d2, d3, d4;
6    uint8 Nodo;
7   }Dendrometros;
8   Dendrometros Ddendrometros;
9   typedef struct
10  {
11   uint16 bateria;
12   uint8 Nodo;
13  }Bateria;
14  Bateria Bat;
```

Listado 3. 4: Variables globales, declaración y comienzo de proceso

En el listado 3.5 se observa los mensajes que se envían del "Nodo Coordinador" a los dos nodos sensores. En este caso, el "Nodo Coordinador" envía "`SampleApp_SendFRECUENCIAMessage`" (línea 6).

```
1  /*********************************************************************
2   * LOCAL FUNCTIONS
3   */
4  void SampleApp_HandleKeys( uint8 shift, uint8 keys );
5  void SampleApp_MessageMSGCB( afIncomingMSGPacket_t *pckt );
6  void SampleApp_SendFRECUENCIAMessage( uint8 );
7  /*********************************************************************
```

Listado 3. 5: Declaración de funciones locales

En el siguiente listado, se observa la configuración de la dirección de destino de los mensajes que serán enviados por el "Nodo Coordinador", los cuales serán "`SampleApp_FRECUENCIA_DstAddr.addrMode`" (líneas 3, 4 y 5). En este caso, el "Nodo Coordinador" envía el mensaje a los dos nodos sensores, por lo que se usa el comando "AddrBroadcast" (línea 3) y la dirección "`0xFFFF`" (línea 5).

```
1  // Setup for the periodic message's destination address
2    // Broadcast to everyone
3    SampleApp_FRECUENCIA_DstAddr.addrMode = (afAddrMode_t)AddrBroadcast;
4    SampleApp_FRECUENCIA_DstAddr.endPoint = SAMPLEAPP_ENDPOINT;
5    SampleApp_FRECUENCIA_DstAddr.addr.shortAddr = 0xFFFF;
```

Listado 3. 6: Configuración de la dirección de destino de los mensajes

El evento "`SYS_EVENT_MSG`" es propio del sistema y permite gestionar eventos de la pila ZigBee y del hardware de bajo nivel. El método gestionador de eventos de la tarea de la aplicación puede requerir el procesamiento del evento "`SYS_EVENT_MSG`" por diferentes motivos. Uno de ellos, es que el "Nodo Coordinador" haya establecido correctamente la conexión con los nodos sensores.

El listado 3.7 muestra el comando "HalLcdWriteString" (línea 30), que permite escribir en la pantalla LCD "ZC on" una vez encendido el "Nodo Coordinador".

```
1  uint16 SampleApp_ProcessEvent( uint8 task_id, uint16 events )
2  {
3    afIncomingMSGPacket_t *MSGpkt;
4    (void)task_id;  // Intentionally unreferenced parameter
5    if ( events & SYS_EVENT_MSG )
6    {
7      MSGpkt = (afIncomingMSGPacket_t *)osal_msg_receive( SampleApp_TaskID);
8      while ( MSGpkt )
9      {
10       switch ( MSGpkt->hdr.event )
11       {
12         // Received when a key is pressed
13         case KEY_CHANGE:
14           SampleApp_HandleKeys( ((keyChange_t *)MSGpkt)->state,
15 ((keyChange_t *)MSGpkt)->keys );
16           break;
17         // Received when a messages is received (OTA) for this endpoint
18         case AF_INCOMING_MSG_CMD:
19           SampleApp_MessageMSGCB( MSGpkt );
20           break;
21         // Received whenever the device changes state in the network
22         case ZDO_STATE_CHANGE:
23           SampleApp_NwkState = (devStates_t)(MSGpkt->hdr.status);
24           if ( (SampleApp_NwkState == DEV_ZB_COORD)
```

```
25                || (SampleApp_NwkState == DEV_ROUTER)
26                || (SampleApp_NwkState == DEV_END_DEVICE) )
27            {
28              // Start sending the FRECUENCIA message in a regular interval.
29  #if defined ( LCD_SUPPORTED )
30                HalLcdWriteString( "ZC on           ", HAL_LCD_LINE_3 );
31  #endif
32            }
33            else
34            {
35              // Device is no longer in the network
36            }
37            break;
```

Listado 3. 7: Evento "SYS_EVENT_MSG

Continuando con el software del "Nodo Coordinador", en el listado 3.8 se observa el comando "HalLcdWriteString", que se explicó anteriormente. El joystick tiene 5 posiciones: derecha, izquierda, arriba, abajo y el centro.

En este caso, se activa la posición 1 de joystick para que, una vez que se muestren los datos enviados por los nodos sensores en la LCD, puedan ser borrados al activar dicha posición.

En cuanto a la posición 2, se declara una variable tipo "char" de 16 elementos con nombre "aux" (línea 14) para ajustar el valor de la frecuencia con la que el "Nodo Coordinador" recibe los datos de los nodos sensores. Más adelante se observa el uso del comando sprintf (línea 17), que hace que los argumentos de "aux" se conviertan en salida posteriormente. Así, con esta posición se incrementa el valor y se muestra en la LCD. Con la posición 4, (línea 22) se hace lo mismo, pero el valor será decrementado. Por último, la posición 3 será utilizada para mandar la frecuencia a los nodos sensores y una vez hecho esto, con "`HalLcdWriteString`" aparecerá en la LCD "`enviando periodo`".

```
1   if ( keys & HAL_KEY_SW_1 )
2     {
3       /* This key sends the Flash Command is sent to Group 1.
4        * This device will not receive the Flash Command from this
5        * device (even if it belongs to group 1).
6        */
7    #if defined ( LCD_SUPPORTED )
8       HalLcdWriteString( "                ", HAL_LCD_LINE_1 );
9       HalLcdWriteString( "                ", HAL_LCD_LINE_2 );
10      HalLcdWriteString( "                ", HAL_LCD_LINE_3 );
11  #endif
12  if ( keys & HAL_KEY_SW_2 )
13    {
14      char aux[16];
15      if (SampleAppFRECUENCIACounter < 255) SampleAppFRECUENCIACounter++;
16  #if defined ( LCD_SUPPORTED )
17      sprintf(aux,"<  %d  >", SampleAppFRECUENCIACounter);
18      HalLcdWriteString(aux, HAL_LCD_LINE_3);
19  #endif
20    }
21
22    if ( keys & HAL_KEY_SW_4)
23    {
24      char aux[16];
25      if (SampleAppFRECUENCIACounter > 1) SampleAppFRECUENCIACounter--;
26  #if defined ( LCD_SUPPORTED )
```

```
27      sprintf(aux,"<  %d  >", SampleAppFRECUENCIACounter);
28      HalLcdWriteString(aux, HAL_LCD_LINE_3);
29  #endif
30    }
31    if ( keys & HAL_KEY_SW_3)
32    {
33      SampleApp_SendFRECUENCIAMessage(frecuencia);
34  #if defined ( LCD_SUPPORTED )
35      HalLcdWriteString("enviando periodo", HAL_LCD_LINE_3);
36  #endif
37    }
```

Listado 3. 8: Posiciones del joystick

En el listado 3.9 se detallan los "clúster" de entrada: "SAMPLEAPP_DATOS_CLUSTERID" y "SAMPLEAPP_BATERIA_CLUSTERID". Por un lado, el *array* llamado "auxArray" de tipo "char" de 16 elementos (línea 3), se declara para usarlo más adelante. También se tiene "*pkt" (línea 1), que es un puntero que contendrá un valor de la dirección de memoria.

El comando "memcpy" (línea 7), copia los caracteres que contenga "pkt->cmd.Data" en "Ddendrometros" y "pkt->cmd.DataLength" da cuantos bytes han sido detectados. También se tiene el comando "sprintf" (línea13), que retorna el número de caracteres escritos al "auxArray", sin contar el carácter nulo al final. Una vez hecho esto, con los comandos explicados anteriormente se muestra el valor obtenido en la LCD con "HalLcdWriteString".

Por otro lado, en "SAMPLEAPP_BATERIA_CLUSTERID" se cumple el mismo proceso, pero se usa "Bat", que es donde se copia a través de "memcpy" los caracteres.

```
1   void SampleApp_MessageMSGCB( afIncomingMSGPacket_t *pkt )
2   {
3     char auxArray [16];
4     switch ( pkt->clusterId )
5     {
6       case SAMPLEAPP_DATOS_CLUSTERID:
7         memcpy(&Ddendrometros, pkt->cmd.Data, pkt->cmd.DataLength);
8   #if defined ( LCD_SUPPORTED )
9         HalLcdWriteStringValue("Nodo: ", Ddendrometros.Nodo, 10,
10  HAL_LCD_LINE_1);
11          sprintf(auxArray,"D4:%d D3:%d", Ddendrometros.d4,Ddendrometros.d3);
12          HalLcdWriteString(auxArray, HAL_LCD_LINE_2 );
13          sprintf(auxArray,"D2:%d D1:%d", Ddendrometros.d2, Ddendrometros.d1);
14          HalLcdWriteString(auxArray, HAL_LCD_LINE_3 );
15  #endif
16        break;
17       case SAMPLEAPP_BATERIA_CLUSTERID:
18         memcpy(&Bat, pkt->cmd.Data, pkt->cmd.DataLength);
19  #if defined ( LCD_SUPPORTED )
20         HalLcdWriteStringValue("Nodo: ", Bat.Nodo, 10, HAL_LCD_LINE_1);
21         HalLcdWriteStringValue("Bateria: ",Bat.bateria, 10, HAL_LCD_LINE_2);
22  #endif
23        break;
24    }
25  }
```

Listado 3. 9: Clúster de Entrada

Continuando con el listado 3.10, es visible el mensaje enviado por el "Nodo Coordinador": "SampleApp_SendFRECUENCIAMessage". El comando "&SampleApp_FRECUENCIA_DstAddr" (línea 3), indica el destino al que va el mensaje enviado.

"SampleApp_epDesc" es un descriptor, "uint8*" (línea 6) que indica el valor de lo envíado y "&frec" indica el lugar a donde se envió. Los demás comandos son genéricos.

```
1  void SampleApp_SendFRECUENCIAMessage( uint8 frec )
2  {
3    if ( AF_DataRequest( &SampleApp_FRECUENCIA_DstAddr, &SampleApp_epDesc,
4                         SAMPLEAPP_FRECUENCIA_CLUSTERID,
5                         1,
6                         (uint8*)&frec,
7                         &SampleApp_TransID,
8                         AF_DISCV_ROUTE,
9                         AF_DEFAULT_RADIUS ) == afStatus_SUCCESS )
10   {
11   }
12   else
13   {
14     // Error occurred in request to send.
15   }
16 }
```

Listado 3. 10: Mensajes enviados por el "Nodo Coordinador"

4.4. Software Nodo Sensor 1 y Nodo Sensor 2

Al igual que el "Nodo Coordinador", ahora se explicará el código desarrollado para el "Nodo Sensor 1" y el "Nodo Sensor 2". Debido a que los dos nodos sensores cumplen la misma función en la red inalámbrica, tienen el mismo código pero con diferente número. Más adelante se indicará donde se configura esto último.

De la misma manera que el "Nodo Coordinador", existen dos archivos que se deben programar. Empezando por el "SampleApp.h", en el listado 3.11 se puede observar que el "Nodo Sensor 1" tiene como "clúster" de entrada:

"SAMPLEAPP_FRECUENCIA_CLUSTERID" (línea 14) y "SAMPLEAPP_MODO_CLUSTERID" (línea 15), y de salida: "SAMPLEAPP_DATOS_CLUSTERID" (línea 11) y "SAMPLEAPP_BATERIA_CLUSTERID" (línea 12). En cuanto a los eventos, en este archivo solo se define los eventos necesarios:

"SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT" (línea 17) que tiene por dirección de memoria "0x0001" y es el evento principal. También tenemos la definición de los otros eventos:

"SAMPLEAPP_SEND_BATERIA_MSG_EVT" (línea 19), el evento del estado de la batería y tiene la dirección de memoria "0x0002" y, "SAMPLEAPP_SEND_DATOS_MSG_EVT" (línea 21), que es el evento de los datos y tiene por dirección de memoria "0x0004". Para ajustar el número de nodo del nodo sensor, NUM_NODO" simplemente se pone el número de nodo correspondiente, en este caso 1 (línea 22).

```
1  * CONSTANTS
2   */
3  // These constants are only for example and should be changed to the
4  // device's needs
5  #define SAMPLEAPP_ENDPOINT           20
6  #define SAMPLEAPP_PROFID             0x0F08
7  #define SAMPLEAPP_DEVICEID           0x0001
8  #define SAMPLEAPP_DEVICE_VERSION     0
9  #define SAMPLEAPP_FLAGS              0
10 #define SAMPLEAPP_MAX_OUT_CLUSTERS     2
11 #define SAMPLEAPP_DATOS_CLUSTERID     1
```

```
12 #define SAMPLEAPP_BATERIA_CLUSTERID   2
13 #define SAMPLEAPP_MAX_IN_CLUSTERS     2
14 #define SAMPLEAPP_FRECUENCIA_CLUSTERID 3
15 #define SAMPLEAPP_MODO_CLUSTERID       4
16 // Application Events 1
17 #define SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT          0x0001
18 // Application Events 2
19 #define SAMPLEAPP_SEND_BATERIA_MSG_EVT               0x0002
20 // Application Events 3
21 #define SAMPLEAPP_SEND_DATOS_MSG_EVT                 0x0004
22 #define NUM_NODO 1
```

Listado 3. 11: SampleApp.h del "Nodo Sensor 1"

Ahora analizando el "SampleApp.c" del "Nodo Sensor 1", en el listado 3.12 se describen todos los "#define" y las bibliotecas que se incluyen en el archivo.

Las librerías añadidas en "Nodo Sensor 1" son: "`hal_adc.h`" (línea 15) para la identificación de algunos comandos, "`stdio.h`" (línea 16) que contiene comandos necesarios y "`string.h`" (línea 17) para el uso de las cadenas de caracteres tipo string.

Al igual que en el "Nodo Coordinador", el resto de las líneas corresponden a definiciones globales de variables, en este caso suele tratarse de variables relacionadas con las direcciones que se utilizarán posteriormente.

```
1  * INCLUDES
2   */
3  #include "OSAL.h"
4  #include "ZGlobals.h"
5  #include "AF.h"
6  #include "aps_groups.h"
7  #include "ZDApp.h"
8  #include "SampleApp.h"
9  #include "SampleAppHw.h"
10 #include "OnBoard.h"
11 /* HAL */
12 #include "hal_lcd.h"
13 #include "hal_led.h"
14 #include "hal_key.h"
15 #include "hal_adc.h"
16 #include <stdio.h>
17 #include <string.h>
```

Listado 3. 12: Librerías incluidas en "Nodo Sensor 1"

El "Nodo Sensor 1" envía los datos y el estado de la batería al "Nodo Coordinador" y tan solo los datos al "Nodo de Configuración" y recibe la frecuencia y el modo de este último, pero solo la frecuencia del "Nodo Coordinador". El "Nodo de Configuración" se explicará más adelante. En el listado 3.13 se observa los "clúster" de entrada y de salida (líneas 6,7, 11 y 12).

```
1  * GLOBAL VARIABLES
2   */
3  // This list should be filled with Application specific Cluster IDs.
4  const cId_t SampleApp_ClusterListOUT[SAMPLEAPP_MAX_OUT_CLUSTERS] =
5  {
6    SAMPLEAPP_DATOS_CLUSTERID,
7    SAMPLEAPP_BATERIA_CLUSTERID,
8  };
9  const cId_t SampleApp_ClusterListIN[SAMPLEAPP_MAX_IN_CLUSTERS] =
10 {
11   SAMPLEAPP_FRECUENCIA_CLUSTERID,
12   SAMPLEAPP_MODO_CLUSTERID
13 };
```

```
14 const SimpleDescriptionFormat_t SampleApp_SimpleDesc =
15 {
16   SAMPLEAPP_ENDPOINT,              //  int Endpoint;
17   SAMPLEAPP_PROFID,                //  uint16 AppProfId[2];
18   SAMPLEAPP_DEVICEID,              //  uint16 AppDeviceId[2];
19   SAMPLEAPP_DEVICE_VERSION,        //  int   AppDevVer:4;
20   SAMPLEAPP_FLAGS,                 //  int   AppFlags:4;
21   SAMPLEAPP_MAX_IN_CLUSTERS,          //  uint8  AppNumInClusters;
22   (cId_t *)SampleApp_ClusterListIN,  //  uint8 *pAppInClusterList;
23   SAMPLEAPP_MAX_OUT_CLUSTERS,          //  uint8  AppNumInClusters;
24   (cId_t *)SampleApp_ClusterListOUT   //  uint8 *pAppInClusterList;
25 };
```

Listado 3. 13: Declaración de los "Clúster" en SampleApp.c

En el listado 3.14, igual que en los apartados anteriores, se vuelven a declarar las variables de salida mediante el comando "`afAddrType_t`" explicado anteriormente.

En este caso figuran como salida "`SampleApp_DATOS_DstAddr`" (línea 1) y "`SampleApp_BATERIA_DstAddr`" (línea 2), además se observa que se utiliza un formato uint_8 para las variables auxiliares que se utilizarán posteriormente (línea 3): "`minutos, minutosLectura, nuevoPeriodo, minutosBateria, minutosLecturaBateria`".

También figuran las dos estructuras explicadas anteriormente. En la línea 17 se tiene dos variables auxiliares de tipo "`bool`", es decir que éstas pueden tomar solo dos valores que pueden ser verdadero o falso (true o false). En las líneas 19 y 20 se tiene dos variables de tipo uint8 que son inicializadas a la vez y que se utilizarán posteriormente.

```
1  afAddrType_t SampleApp_DATOS_DstAddr;
2  afAddrType_t SampleApp_BATERIA_DstAddr;
3  uint8 minutos, minutosLectura, nuevoPeriodo, minutosBateria,
4  minutosLecturaBateria;
5  typedef struct
6  {
7  uint16 d4,d3,d2,d1;
8  uint8 Nodo;
9  }Dendrometros;
10 Dendrometros Ddendrometros;
11 typedef struct
12 {
13  uint16 bateria;
14  uint8 Nodo;
15 }Bateria;
16 Bateria Bat;
17 bool reiniciarTimer, modoAjuste;
18 //aps_Group_t SampleApp_Group;
19 uint8 SampleAppFRECUENCIACounter = 0;
20 uint8 SampleAppDATOSCounter = 10;
```

Listado 3. 14: Variables globales, declaración y comienzo de proceso del "Nodo Sensor 1"

El listado 3.15, igual que en el "Nodo Coordinador", están declarados los mensajes enviados por el "Nodo Sensor 1" al "Nodo Coordinador" y al "Nodo de Configuración". Vemos "SampleApp_ENVIAR_DATOS_Message" (línea 5) y "SampleApp_SendBATERIAMessage" (línea 6).

```
1  * LOCAL FUNCTIONS
2   */
3  void SampleApp_HandleKeys( uint8 shift, uint8 keys );
4  void SampleApp_MessageMSGCB( afIncomingMSGPacket_t *pckt );
5  void SampleApp_ENVIAR_DATOS_Message( void );
6  void SampleApp_SendBATERIAMessage (uint8);
```

Listado 3. 15: Declaración de las funciones locales

En el listado que se aprecia a continuación, se ha inicializado algunas variables auxiliares, que se utilizarán más adelante y que para el correcto funcionamiento del software diseñado se han fijado dichos valores. Es visible el comando "`HalAdcInit`" (línea 14), que es una instrucción para inicializar los convertidores analógicos digitales (ADCs) del MSP430.

```
1  //MIS COSAS
2    minutos = 0;
3    minutosLectura = 1;
4    HalAdcInit ();
5    reiniciarTimer = false;
6
7    minutosBateria = 0;
8    minutosLecturaBateria = 3;
9    modoAjuste = false;
```

Listado 3. 16: Inicialización de algunas variables auxiliares

Continuando con el listado 3.17, al igual que en apartados anteriores, se configura la dirección de destino de los mensajes que serán enviados por el "Nodo Sensor 1", en este caso será "`SampleApp_DATOS_DstAddr.addrMode`" (línea 3), además, como el que recibe es el "Nodo Coordinador" se usa el comando "`afAddr16Bit`" (línea 3), que indica que se envía a un solo dispositivo, y la dirección "`0x0000`" (línea 5).

También se envía "`SampleApp_BATERIA_DstAddr.addrMode`" (línea 6), y de igual manera, se envía al "Nodo Coordinador" (línea 6) con dirección "`0x0000`" (línea 8).

```
1  // Setup for the periodic message's destination address
2    // Broadcast to everyone
3    SampleApp_DATOS_DstAddr.addrMode = (afAddrMode_t)afAddr16Bit;
4    SampleApp_DATOS_DstAddr.endPoint = SAMPLEAPP_ENDPOINT;
5    SampleApp_DATOS_DstAddr.addr.shortAddr = 0x0000;
6    SampleApp_BATERIA_DstAddr.addrMode = (afAddrMode_t)afAddr16Bit;
7    SampleApp_BATERIA_DstAddr.endPoint = SAMPLEAPP_ENDPOINT;
8    SampleApp_BATERIA_DstAddr.addr.shortAddr = 0x0000;
```

Listado 3. 17: Configuración de la dirección de destino de los mensajes en el "Nodo Sensor 1"

Anteriormente, se ha declarado una estructura de tipo "`Dendrometros`", en el listado 3.18 inicializamos los campos de la variable "`Ddendrometros`". En este caso, para simular los valores reales que se tendrían en los dendrómetros, se realiza una operación matemática; por ejemplo, se enviará un 1002 para el dendrómetro 2 del nodo 1 (línea 2) y para el "Nodo Sensor 2" se tendrá un 2002 en el segundo dendrómetro. En el caso de la batería, (línea 7) se tendrá el valor 100 para el "Nodo Sensor 1" y para el "Nodo Sensor 2" el valor 200.

```
1  Ddendrometros.d1 = 1000*NUM_NODO+1;
2  Ddendrometros.d2 = 1000*NUM_NODO+2;
3  Ddendrometros.d3 = 1000*NUM_NODO+3;
4  Ddendrometros.d4 = 1000*NUM_NODO+4;
5  Ddendrometros.Nodo = NUM_NODO;
6  Bat.Nodo = NUM_NODO;
7  Bat.bateria = 100*NUM_NODO;
```

Listado 3. 18: Equivalencias auxiliares

Igual que en apartados anteriores, en el listado 3.19 se observa el evento que viene por defecto en el sistema "`SYS_EVENT_MSG`". Una vez que se ha ejecutado este evento, se planifica un evento "`SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT`" (línea 26) cada 1000 milisegundos, este evento se explicará más adelante.

Posteriormente se declara un array llamado "arrayaux" (línea 29) de tipo "char" de 16 elementos y con el comando "sprintf", explicado anteriormente, se consigue que tras el encendido del LCD "Nodo Sensor 1" aparezca "ZED 1" y en el caso del "Nodo Sensor 2", "ZED 2".

```
1    if ( events & SYS_EVENT_MSG )
2    {
3      MSGpkt = (afIncomingMSGPacket_t *)osal_msg_receive( SampleApp_TaskID );
4      while ( MSGpkt )
5      {
6        switch ( MSGpkt->hdr.event )
7        {
8          // Received when a key is pressed
9          case KEY_CHANGE:
10           SampleApp_HandleKeys( ((keyChange_t *)MSGpkt)->state,
11 ((keyChange_t *)MSGpkt)->keys );
12           break;
13         // Received when a messages is received (OTA) for this endpoint
14         case AF_INCOMING_MSG_CMD:
15           SampleApp_MessageMSGCB( MSGpkt );
16           break;
17         // Received whenever the device changes state in the network
18         case ZDO_STATE_CHANGE:
19           SampleApp_NwkState = (devStates_t)(MSGpkt->hdr.status);
20           if ( (SampleApp_NwkState == DEV_ZB_COORD)
21               || (SampleApp_NwkState == DEV_ROUTER)
22               || (SampleApp_NwkState == DEV_END_DEVICE) )
23           {
24             // Start sending the periodic message in a regular interval.
25             osal_start_timerEx( SampleApp_TaskID,
26                                 SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT,
27                                 1000 );
28 #if defined ( LCD_SUPPORTED )
29             char arrayaux[16];
30             sprintf(arrayaux, "ZED%d On", NUM_NODO);
31             HalLcdWriteString(arrayaux, HAL_LCD_LINE_3 );
32 #endif
```

Listado 3. 19: Evento SYS_EVENT_MSG en el "Nodo Sensor 1"

En el listado 3.20 se observa el evento timer. Una vez se ha planificado este evento, se utiliza un tipo de sentencia if/else que es muy común en programación, en este caso si no se está en modo ajuste (línea 4) es decir, se está en modo normal, el evento "SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT" se ejecuta cada 60000 milisegundos, en caso contrario se ejecutará cada 1000 milisegundos. En el segundo caso, si se cumplen las tres condiciones (líneas 10, 11 y 12), la variable "minutos" se incrementa. Más adelante vemos otra estructura tipo if (línea 16), que si cumple la condición impuesta y se está en modo normal o "!modoAjuste" se ejecuta el evento "SAMPLEAPP_SEND_DATOS_MSG_EVT" cada 2000 milisegundos, y sino cada 500 milisegundos. A continuación, hecho todo el proceso explicado anteriormente, se muestra en la LCD "LEYENDO DATOS" mediante el comando correspondiente.

Posteriormente se tiene otro if (línea 26) para el caso en el que se esté en modo normal, los "minutosBateria" se incrementarán y si "minutosBateria >= minutosLecturaBateria" se cumple, el evento "SAMPLEAPP_SEND_BATERIA_MSG_EVT" se ejecutará cada 3000 milisegundos y en la LCD aparecerá "LEYENDO BATERIA". Finalmente este evento terminará (línea 39).

```
1  //evento de mi timer
2    if ( events & SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT)
3    {
4         if (!modoAjuste) osal_start_timerEx( SampleApp_TaskID,
```

```
5   SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT, 60000 );
6         else osal_start_timerEx( SampleApp_TaskID,
7   SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT, 1000 );
8         if (reiniciarTimer)
9         {
10          reiniciarTimer = false;
11          minutos = 0;
12          minutosLectura = nuevoPeriodo;
13        }
14        minutos++;
15        if (minutos >= minutosLectura)
16        {
17          minutos = 0;
18           if (!modoAjuste) osal_start_timerEx( SampleApp_TaskID,
19  SAMPLEAPP_SEND_DATOS_MSG_EVT, 2000 );
20           else osal_start_timerEx( SampleApp_TaskID,
21  SAMPLEAPP_SEND_DATOS_MSG_EVT, 500);
22  #if defined ( LCD_SUPPORTED )
23              HalLcdWriteString("LEYENDO DATOS", HAL_LCD_LINE_3 );
24  #endif
25        }
26  if (!modoAjuste)
27  {
28        minutosBateria++;
29        if (minutosBateria >= minutosLecturaBateria)
30        {
31          minutosBateria = 0;
32          if (!modoAjuste) osal_start_timerEx( SampleApp_TaskID,
33  SAMPLEAPP_SEND_BATERIA_MSG_EVT, 3000 );
34  #if defined ( LCD_SUPPORTED )
35              HalLcdWriteString("LEYENDO BATERIA", HAL_LCD_LINE_3 );
36  #endif
37        }
38  }
39        return (events ^ SAMPLEAPP_SEND_EVENTO_TIMER_MSG_EVT);
40     }
```

Listado 3. 20: SAMPLEAPP_SEND_EVENTO_TIMER_EVT

A continuación, en el listado 3.21, al ejecutar el evento "`SAMPLEAPP_SEND_DATOS_MSG_EVT`", (línea 1) la LCD escribirá "`ENVIANDO DATOS`", entonces el mensaje enviado será "`SampleApp_ENVIAR_DATOS_Message`". En el caso del evento "`SAMPLEAPP_SEND_BATERIA_MSG_EVT`", en la LCD aparecerá "`ENVIANDO BATERIA`" y el mensaje que se envía con este evento es "`SampleApp_SendBATERIAMessage(NUM_NODO*100)`". Finalmente, el evento "`SAMPLEAPP_CLEAR_LINE3`" (línea 23) únicamente escribe una línea en blanco en la LCD.

```
1   if ( events & SAMPLEAPP_SEND_DATOS_MSG_EVT )
2     {
3   #if defined ( LCD_SUPPORTED )
4       HalLcdWriteString("ENVIANDO DATOS ", HAL_LCD_LINE_3 );
5   #endif
6       // Send the periodic message
7       SampleApp_ENVIAR_DATOS_Message();
8       // return unprocessed events
9       return (events ^ SAMPLEAPP_SEND_DATOS_MSG_EVT);
10    }
11    // Send a message out - This event is generated by a timer
12    //  (setup in SampleApp_Init()).
13    if ( events & SAMPLEAPP_SEND_BATERIA_MSG_EVT )
14    {
15  #if defined ( LCD_SUPPORTED )
16      HalLcdWriteString("ENVIANDO BATERIA", HAL_LCD_LINE_3 );
17  #endif
18      // Send the periodic message
```

```
19      SampleApp_SendBATERIAMessage(NUM_NODO*100);
20      // return unprocessed events
21      return (events ^ SAMPLEAPP_SEND_BATERIA_MSG_EVT);
22    }
23    if ( events & SAMPLEAPP_CLEAR_LINE3 )
24    {
25  #if defined ( LCD_SUPPORTED )
26      HalLcdWriteString("                ", HAL_LCD_LINE_3 );
27  #endif
28      // return unprocessed events
29      return (events ^ SAMPLEAPP_CLEAR_LINE3);
30    }
```

Listado 3. 21: Ejecución de los eventos en el "Nodo Sensor 1"

En el caso del "Nodo Sensor 1" y del "Nodo Sensor 2", la única posición de joystick que estará activa será la 2 y solo servirá para borrar la LCD. Esto se puede observar en el listado 3.22, (líneas 4 y 5).

```
1   if ( keys & HAL_KEY_SW_2 )
2     {
3   #if defined ( LCD_SUPPORTED )
4       HalLcdWriteString("                ", HAL_LCD_LINE_2);
5       HalLcdWriteString("                ", HAL_LCD_LINE_3);
6   #endif
7     }
```

Listado 3. 22: Posiciones activas de Joystick en el "Nodo Sensor 1"

En el siguiente listado 3.23 tenemos los "clúster" de entrada: "`SAMPLEAPP_FRECUENCIA_CLUSTERID`" y "`SAMPLEAPP_MODO_CLUSTERID`". Por un lado, es visible el comando "`HalLedBlink`", (línea 2) que se utiliza para que un LED determinado parpadee un determinado tiempo, en este caso parpadea el LED 2. Posteriormente, la variable "`reiniciarTimer`" se pone a verdadero y "`nuevoPeriodo`" se iguala a la posición de memoria que apunta "`pkt`". Finalmente en la LCD se escribe "`periodo recibido`" con el comando explicado anteriormente.

En el caso de "`SAMPLEAPP_MODO_CLUSTERID`", es incluida una sentencia if/else explicada en apartados anteriores. En este caso, si se cumple la condición (línea 10), el LED 1 parpadea 250 milisegundos y dentro se tiene otro if, que de igual manera, si "`modoAjuste = false`", se envía al "Nodo Coordinador" el "clúster" (línea 17). Si "`modoAjuste = true`" se envía el "clúster" "`SampleApp_DATOS_DstAddr`" a la siguiente dirección: "`pkt->srcAddr.addr.shortAddr`".

```
1   case SAMPLEAPP_FRECUENCIA_CLUSTERID:
2       HalLedBlink (HAL_LED_2, 4, 50, 250);
3       reiniciarTimer = true;
4       nuevoPeriodo = pkt->cmd.Data[0];
5   #if defined ( LCD_SUPPORTED )
6       HalLcdWriteString("periodo recibido", HAL_LCD_LINE_2);
7   #endif
8       break;
9     case SAMPLEAPP_MODO_CLUSTERID:
10      if(pkt->cmd.Data[0] == NUM_NODO)
11      {
12        HalLedBlink (HAL_LED_1, 4, 50, 250);
13        if(modoAjuste)
14        {
15          modoAjuste = false;
16          //enviar al coordinador
17          SampleApp_DATOS_DstAddr.addr.shortAddr = 0x0000;
18        }
```

```
19        else
20        {
21          modoAjuste = true;
22          //enviar al nodo que realizo la petición
23          SampleApp_DATOS_DstAddr.addr.shortAddr = pkt-
24  >srcAddr.addr.shortAddr;
25        }
26      }
27      break;
28  }
    }
```

Listado 3. 23: Clúster de entrada en el "Nodo Sensor 1"

Continuando con el listado 3.24, se llega al desarrollo de los "clúster" de salida, con el cual el "Nodo Sensor 1" envía los datos al "Nodo Coordinador". En este caso, se utiliza la variable auxiliar (línea 3) y el comando "`sizeof`" (línea 7), que indica el tamaño la estructura "`Dendrometros`". Los demás comandos se han explicado anteriormente. A continuación se muestra otra sentencia if (línea 16), si se está en modo normal, es decir, "`!modoAjuste`", se ejecuta el evento "`SAMPLEAPP_CLEAR_LINE3`" cada 5000 milisegundos. En el caso de que todo lo anterior no se cumpliera, en la LCD aparecerá "`ERROR ENV. DATOS`".

```
1   void SampleApp_ENVIAR_DATOS_Message( void )
2   {
3     uint8 SampleAppDATOSCounter = 10;
4
5     if ( AF_DataRequest( &SampleApp_DATOS_DstAddr, &SampleApp_epDesc,
6                          SAMPLEAPP_DATOS_CLUSTERID,
7                          sizeof(Dendrometros),
8                           (uint8*)&Ddendrometros,
9                          &SampleApp_TransID,
10                          AF_DISCV_ROUTE,
11                         AF_DEFAULT_RADIUS ) == afStatus_SUCCESS )
12      {
13  #if defined ( LCD_SUPPORTED )
14        HalLcdWriteString("DATOS ENVIADOS", HAL_LCD_LINE_3 );
15         if (!modoAjuste) osal_start_timerEx( SampleApp_TaskID,
16  SAMPLEAPP_CLEAR_LINE3, 5000 );
17  #endif
18      }
19
20    else
21    {
22      // Error occurred in request to send.
23  #if defined ( LCD_SUPPORTED )
24      HalLcdWriteString("ERROR ENV. DATOS", HAL_LCD_LINE_3 );
25  #endif
26    }
27  }
```

Listado 3. 24: SampleApp_ENVIAR_DATOS_Message

En el listado 3.25 se tiene el otro "clúster" de salida: "`SampleApp_SendBATERIAMessage`". Este tiene la misma configuración que el anterior "clúster".

```
1   void SampleApp_SendBATERIAMessage( uint8 bat )
2   {
3
4     if ( AF_DataRequest( &SampleApp_DATOS_DstAddr, &SampleApp_epDesc,
5                          SAMPLEAPP_BATERIA_CLUSTERID,
6                          sizeof(Bateria),
7                          (uint8*)&Bat,
```

```
8                         &SampleApp_TransID,
9                         AF_DISCV_ROUTE,
10                        AF_DEFAULT_RADIUS ) == afStatus_SUCCESS )
11    {
12  #if defined ( LCD_SUPPORTED )
13        HalLcdWriteString("BATERIA ENVIADA ", HAL_LCD_LINE_3 );
14         if (!modoAjuste) osal_start_timerEx( SampleApp_TaskID,
15  SAMPLEAPP_CLEAR_LINE3, 5000 );
16  #endif
17    }
18     else
19    {
20      // Error occurred in request to send.
21
22  #if defined ( LCD_SUPPORTED )
23      HalLcdWriteString("ERROR ENV. BAT  ", HAL_LCD_LINE_3 );
24  #endif
25
26    }
27  }
```

Listado 3. 25: SampleApp_SendBATERIAMessage

4.5. Software Nodo de Configuración

De la misma manera que el "Nodo Sensor 1", este gestiona dos archivos que deben programarse. Igual que antes, se empieza por el "SampleApp.h", en el listado 3.26 se observa que el "Nodo de Configuración" tiene como "clúster" de entrada: "`SAMPLEAPP_DATOS_CLUSTERID`" (línea 5), y de salida: "`SAMPLEAPP_FRECUENCIA_CLUSTERID`" (línea 2) y "`SAMPLEAPP_MODO_CLUSTERID`" (línea 3).

```
1   #define SAMPLEAPP_MAX_OUT_CLUSTERS      2
2   #define SAMPLEAPP_FRECUENCIA_CLUSTERID  3
3   #define SAMPLEAPP_MODO_CLUSTERID        4
4   #define SAMPLEAPP_MAX_IN_CLUSTERS      1
5   #define SAMPLEAPP_DATOS_CLUSTERID      1
```

Listado 3. 26: SampleApp.h

Ahora analizando el "SampleApp.c" del "Nodo de Configuración", en el listado 3.27 se describen todos los "#define" y las bibliotecas que están incluidos en el archivo, que son los mismos utilizados en el "Nodo Coordinador" y en el "Nodo Senso1".

```
1   #include "OSAL.h"
2   #include "ZGlobals.h"
3   #include "AF.h"
4   #include "aps_groups.h"
5   #include "ZDApp.h"
6   #include "SampleApp.h"
7   #include "SampleAppHw.h"
8   #include "OnBoard.h"
9   /* HAL */
10  #include "hal_lcd.h"
11  #include "hal_led.h"
12  #include "hal_key.h"
13  #include <stdio.h>
14  #include <string.h>
```

Listado 3. 27: Librerías incluidas en SampleApp.c

De igual manera que en apartados anteriores, en el listado 3.28 se observa la declaración de los "clúster" de entrada y de salida en el archivo "SampleApp.c". El "clúster" de entrada (línea 3) y los de salida (líneas 7 y 8) son definidos. La demás líneas son definiciones genéricas del programa.

```
1  const cId_t SampleApp_ClusterListIn[SAMPLEAPP_MAX_IN_CLUSTERS] =
2  {
3    SAMPLEAPP_DATOS_CLUSTERID
4  };
5  const cId_t SampleApp_ClusterListOut[SAMPLEAPP_MAX_OUT_CLUSTERS] =
6  {
7    SAMPLEAPP_FRECUENCIA_CLUSTERID,
8    SAMPLEAPP_MODO_CLUSTERID
9  };
10 const SimpleDescriptionFormat_t SampleApp_SimpleDesc =
11 {
12   SAMPLEAPP_ENDPOINT,              //  int Endpoint;
13   SAMPLEAPP_PROFID,                //  uint16 AppProfId[2];
14   SAMPLEAPP_DEVICEID,              //  uint16 AppDeviceId[2];
15   SAMPLEAPP_DEVICE_VERSION,        //  int   AppDevVer:4;
16   SAMPLEAPP_FLAGS,                 //  int   AppFlags:4;
17   SAMPLEAPP_MAX_IN_CLUSTERS,          //  uint8  AppNumInClusters;
18   (cId_t *)SampleApp_ClusterListIn,  //  uint8 *pAppInClusterList;
19   SAMPLEAPP_MAX_OUT_CLUSTERS,          //  uint8  AppNumInClusters;
20   (cId_t *)SampleApp_ClusterListOut   //  uint8 *pAppInClusterList;
21 };
```

Listado 3. 28: Declaración de los "clúster" de entada y salida

En el listado 3.29, igual que en apartados anteriores, se utiliza la misma estructura que en el "Nodo Coordinador" y en el "Nodo Sensor 1" pero en este caso, se utiliza una única estructura, la "`Ddendrometros`", ya que el "Nodo de Configuración" recibe los datos que envían los dos nodos sensores. Además, se declara una variable de tipo uint8 que se nombra como "`MODO`" (línea 8), También existe otra variable, que igual que la anterior es de tipo uint8 (línea 10). Y por último, otra variable de tipo "`bool`" (línea 11), cuyos valores pueden oscilar solo entre "true" o "false"

```
1  afAddrType_t SampleApp_ENVIOBROADCAST_DstAddr;
2  typedef struct
3  {
4   uint16 d1, d2, d3, d4;
5   uint8 Nodo;
6  }Dendrometros;
7  Dendrometros Ddendrometros;
8  uint8 MODO;
9  //aps_Group_t SampleApp_Group;
10 uint8 SampleAppFRECUENCIACounter = 0;
11 bool modo;
```

Listado 3. 29: Declaración de variables globales y comienzo del proceso

En el listado 3.30, igual que en el "Nodo Sensor 1", se declaran los mensajes enviados por el "Nodo de Configuración" a los dos nodos sensores. El mensaje de Modo es declarado con "`SampleApp_SendMODOMessage`" (linea 6) y el mensaje de Frecuencia con "`SampleApp_SendFrecuencyMessage`" (línea 7)

```
1  /*********************************************************************
2   * LOCAL FUNCTIONS
3   */
4  void SampleApp_HandleKeys( uint8 shift, uint8 keys );
5  void SampleApp_MessageMSGCB( afIncomingMSGPacket_t *pckt );
```

```
6  void SampleApp_SendMODOMessage( uint8 );
7  void SampleApp_SendFrecuencyMessage( uint8 );
8  /*********************************************************************
```

Listado 3. 30: Declaración de las funciones locales en el "Nodo de Configuración"

En el siguiente listado, el 3.31, tenemos la configuración de la dirección de destino de los mensajes que serán enviados por el "Nodo de Configuración" a los nodos sensores. Se realiza de la misma manera que se ha explicado en apartados anteriores, en este caso con "`SampleApp_ENVIOBROADCAST_DstAddr`".

```
1  // Setup for the periodic message's destination address
2    // Broadcast to everyone
3  SampleApp_ENVIOBROADCAST_DstAddr.addrMode = (afAddrMode_t)AddrBroadcast;
4  SampleApp_ENVIOBROADCAST_DstAddr.endPoint = SAMPLEAPP_ENDPOINT;
5  SampleApp_ENVIOBROADCAST_DstAddr.addr.shortAddr = 0xFFFF;
```

Listado 3. 31: Declaración de las funciones locales en el "Nodo de Configuración"

Una vez que se ha ejecutado el evento que viene por defecto en el sistema, "`SYS_EVENT_MSG`", igual que en el "Nodo Coordinador", se coloca el nombre del "Nodo de Configuración", entonces una vez que se ha conectado correctamente a red inalámbrica se muestra el mensaje "`ZED CC2530 on`" en la LCD.(listado 3.32)

```
1  #if defined ( LCD_SUPPORTED )
2                HalLcdWriteString( "ZED CC2530 on", HAL_LCD_LINE_3 );
3  #endif
```

Listado 3. 32: Configuración del nombre del "Nodo de Configuración"

El listado 3.33 contiene las diferentes posiciones activadas del joystick. Al igual que en el "Nodo Coordinador" para la posición 1, la LCD se borrará, para la posición 2 el valor de "`SampleAppFRECUENCIACounter`" incrementará, para la posición 4 decrementará "`SampleAppFRECUENCIACounter`" y para la posición 3, una vez activada, envía el valor que tiene "`SampleAppFRECUENCIACounter`" por lo que en el LCD aparecerá "`Frecuencia cambiad`".

En el "Nodo de Configuración" se ha utilizado la posición 5 y su función es:
si se está en modo normal o "`!modo`", la variable "`modo`" pasa a "true" y en la LCD aparecerá "`modo ajuste`", en caso contrario, la variable "`modo`" pasará a "false" y en la LCD aparecerá "`modo normal`".

```
1  if ( keys & HAL_KEY_SW_1 )
2    {
3  #if defined ( LCD_SUPPORTED )
4      HalLcdWriteString("                ", HAL_LCD_LINE_2);
5      HalLcdWriteString("                ", HAL_LCD_LINE_3);
6  #endif
7    }
8    if ( keys & HAL_KEY_SW_2 )
9    {
10     if (SampleAppFRECUENCIACounter < 255) SampleAppFRECUENCIACounter++;
```

```
#if defined ( LCD_SUPPORTED )
    HalLcdWriteStringValue("        ", SampleAppFRECUENCIACounter , 10,
HAL_LCD_LINE_3);
#endif
  }
   if ( keys & HAL_KEY_SW_4 )
  {
     if (SampleAppFRECUENCIACounter > 1) SampleAppFRECUENCIACounter--;
#if defined ( LCD_SUPPORTED )
    HalLcdWriteStringValue("        ", SampleAppFRECUENCIACounter , 10,
HAL_LCD_LINE_3);
#endif
  }
  if ( keys & HAL_KEY_SW_3)
  {
#if defined ( LCD_SUPPORTED )
    HalLcdWriteString("Frecuencia cambiad", HAL_LCD_LINE_3);
#endif
    SampleApp_SendFrecuencyMessage(SampleAppFRECUENCIACounter);
  }
  if ( keys & HAL_KEY_SW_5)
  {
    if (!modo)
    {
       modo = true;
#if defined ( LCD_SUPPORTED )
       HalLcdWriteString("modo ajuste", HAL_LCD_LINE_3);
#endif
    }
    else
    {
      modo = false;
#if defined ( LCD_SUPPORTED )
       HalLcdWriteString("modo normal", HAL_LCD_LINE_3);
#endif
    }
    SampleApp_SendMODOMessage(SampleAppFRECUENCIACounter);
  }
}
```

Listado 3. 33: Posiciones del joystick del "Nodo de Configuración"

Dado que el "Nodo de Configuración" recibe los datos que le envían los nodos sensores, la configuración del "clúster" de entrada en el listado 3.34 es la misma que se explicó en el "Nodo Coordinador".

```
void SampleApp_MessageMSGCB( afIncomingMSGPacket_t *pkt )
{
  char auxArray [16];
  switch ( pkt->clusterId )
  {
    case SAMPLEAPP_DATOS_CLUSTERID:
      memcpy(&Ddendrometros, pkt->cmd.Data, pkt->cmd.DataLength);
#if defined ( LCD_SUPPORTED )
      HalLcdWriteStringValue("Nodo: ", Ddendrometros.Nodo, 10, HAL_LCD_LINE_1);
        sprintf(auxArray,"D4:%d D3:%d", Ddendrometros.d4, Ddendrometros.d3);
        HalLcdWriteString(auxArray, HAL_LCD_LINE_2 );
        sprintf(auxArray,"D2:%d D1:%d", Ddendrometros.d2, Ddendrometros.d1);
        HalLcdWriteString(auxArray, HAL_LCD_LINE_3 );
#endif
      break;
  }
}
```

Listado 3. 34: "Clúster" de entrada

El listado 3.35 muestra los "clúster" de salida del "Nodo de Configuración", su configuración se hace de la misma manera que se ha explicado en apartados anteriores, en este caso se tiene como "clúster" de salida: "`SampleApp_SendMODOMessage`" (línea 1) y "`SampleApp_SendFrecuencyMessage`" (línea 17).

```
1   void SampleApp_SendMODOMessage( uint8 nodo )
2  {
3    if ( AF_DataRequest( &SampleApp_DATOS_DstAddr, &SampleApp_epDesc,
4                         SAMPLEAPP_MODO_CLUSTERID,
5                         1,
6                         (uint8*)&nodo,
7                         &SampleApp_TransID,
8                         AF_DISCV_ROUTE,
9                         AF_DEFAULT_RADIUS ) == afStatus_SUCCESS )
10   {
11   }
12   else
13   {
14     // Error occurred in request to send.
15   }
16 }
17 void SampleApp_SendFrecuencyMessage( uint8 tiempo)
18 {
19   if ( AF_DataRequest( &SampleApp_DATOS_DstAddr, &SampleApp_epDesc,
20                         SAMPLEAPP_FRECUENCIA_CLUSTERID,
21                         1,
22                         (uint8*)&tiempo,
23                         &SampleApp_TransID,
24                         AF_DISCV_ROUTE,
25                         AF_DEFAULT_RADIUS ) == afStatus_SUCCESS )
26   {
27   }
28   else
29   {
30     // Error occurred in request to send.
31   }
32 }
```

Listado 3. 35: "Clúster" de salida del "Nodo de Configuración"

CAPÍTULO 4

CONCLUSIONES Y TRABAJOS FUTUROS.

1. Conclusiones

Durante la realización de este proyecto, se ha llevado a cabo el estudio, diseño e implementación de un nodo sensor inalámbrico que permite reajustar la posición de los dendrómetros conectados a una red inalámbrica de sensores. Tanto la red inalámbrica, como los sensores conectados a la misma, entre ellos los dendrómetros, están ubicados en una parcela de almendros de la Estación Experimental Agroalimentaria "Tomás Ferro". Con este dispositivo se facilita enormemente la tarea de reajuste de los dendrómetros, ya que sólo es necesario encender el dispositivo, seleccionar el nodo al que está conectado el dendrómetro y habilitar el modo ajuste en dicho nodo. Una vez realizado este proceso, en el dispositivo desarrollado en este trabajo se empiezan a recibir datos en tiempo real de los dendrómetros conectados al nodo sensor configurado en "modo ajuste". Al finalizar con un nodo, se habilita de nuevo el "modo normal" y se realizar el proceso en otro nodo si es necesario. En definitiva este dispositivo permite ajustar de una forma más precisa los dendrómetros, así como aumentar los tiempos entre ajustes de los sensores.

El dispositivo inalámbrico se ha diseñado utilizando el SoC CC2530 de Texas Instruments. Se trata de un dispositivo que incorpora en un solo chip tanto el micro-controlador como el módulo de radio. Antes de seleccionar este chip, se realizó un estado del arte de las redes inalámbricas de sensores, el cual permitió elegir este SoC como solución óptima para la implementación del dispositivo inalámbrico de mano.

Con objeto de llevar a cabo el diseño del dispositivo en el laboratorio, se realizó la implementación de una red inalámbrica de sensores que simulaba el funcionamiento de la existente en la parcela de almendros. Para realizar dicha implementación se utilizaron placas de un kit de desarrollo, que presentaban los mismos elementos principales (micro-controlador y módulo de radio) que los nodos sensores desplegados en el cultivo de almendros. Concretamente, se utilizó el kit de desarrollo del módulo CC2520.

Para realizar la programación de los nodos sensores, así como del dispositivo inalámbrico utilizado en el ajuste de los dendrómetros, se ha estudiado el entorno de desarrollo IAR Embedded Workbench y la biblioteca Z-Stack. Esta última es la implementación de la especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica, conocida como ZigBee, realizada por Texas Instruments.

Específicamente, se llevó a cabo la implementación de los componentes software necesarios para obtener los siguientes dispositivos: "Nodo Coordinador", "Nodo Sensor 1", "Nodo Sensor 2" y el "Nodo de Configuración". El "Nodo Coordinador" se encarga de recibir los datos obtenidos por los dendrómetros a través del "Nodo Sensor 1"y del "Nodo Sensor 2". Además, recibe el estado de la batería de estos nodos. El "Nodo Sensor 1" y el "Nodo Sensor 2" realizan la misma función, la cual es enviar los datos registrados de los dendrómetros al "Nodo Coordinador" y al "Nodo de Configuración". Finalmente, el "Nodo de Configuración" recibe los datos del "Nodo Sensor 1" y del "Nodo Sensor 2", pero a una determinada frecuencia que será ajustada por el mismo.

En este trabajo también se ha demostrado que se puede establecer una comunicación inalámbrica óptima entre un nodo sensor basado en el módulo de radio de radio CC2520 y otro constituido por el SoC CC2530.

Finalmente, destacar que con el trabajo llevado a cabo en este proyecto se han adquirido conocimientos en el diseño electrónico, y más específicamente, en el diseño de redes inalámbricas de sensores en el ámbito de la agricultura de precisión.

2. Trabajos Futuros

Partiendo de la realización de este proyecto, se proponen varias mejoras que se podrían hacer para futuros trabajos:

- Actualmente el sistema está diseñado para el control de 2 dendrómetros a través de 2 nodos sensores. Con esta implementación solo se puede monitorizar y controlar el crecimiento del tallo central de dos almendros. Sería interesante extender el sistema a la monitorización y control de crecimiento de campos de almendros. Así de poder desplegar más nodos de configuración y así estimar con mayor exactitud el crecimiento del tallo. Para conseguir esto, sería necesario depurar el código para aumentar la cantidad de nodos sensores que puede monitorizar y controlar el sistema.

- Aplicando estos nodos sensores a otras variables medibles en los campos, como luz, presión, humedad, temperatura, etc, se podría hacer seguimientos en tiempo

real de las necesidades del campo sin necesidad de estar in situ en el campo. Esto, junto a la automatización del cultivo, permitiría controlar con mayor precisión las necesidades del cultivo, permitiendo el aumento de la capacidad de producción en dicho cultivos.

- Aprovechando la capacidad de los Smartphone y de las tecnologías de aplicaciones para estos, sería posible añadir funcionalidad a estos nodos sensores, creando una aplicación capaz de controlar y modificar las características de los nodos sensores según las necesidades reales del cultivo.

BIBLIOGRAFÍA

[Akyildiz, 2002] Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E., 2002. Wireless sensor networks: a survey, Computer Networks 38, 393-422.

[Anurag, 2008] Anurag D, Siuli Roy and Somprakash Banyopadhyay. ―Agro-sense: Precision agriculture using sensor-based wireless mesh networks. Innovations in NGN: Future Network and Services‖. Proceedings of the First ITU-T Kaleidoscope Academic Conference (K-INGN 2008), pp. 383-388.

[Basso, 2006] Basso B., Sartori L., Bertocco M. ―Manual de Agricultura de Precisión‖. Eumedia. 2006.

[Crossbow] Crossbow Technology, Inc. Disponible on-line en: http://bullseye.xbow.com:81/Products/productdetails.aspx?sid=164

[Earl, 2000] R. Earl, G. Thomas, B. S. Blackmore. ―The potential role of GIS in autonomous field operations‖. Computers and Electronics in Agriculture, Volume 25, Issues 1-2, January 2000, Pages 107-120.

[Handle, 2006] Hande, A., Polk, T., Walker W., Bhatia, D. ―Self-Powered Wireless Sensor Networks for remote Patient Monitoring in Hospitals‖. Sensors 2006, 6, pp. 1102, -1117.

[Huang, 2003] G. Huang. ―Casting the Wireless Sensor Net‖. July/August 2003. MIT Technology Review, pp. 51-56.

[Intel Mote 2] Disponible on-line http://cps.cse.wustl.edu/images/c/cb/Imote2-ds-rev2_2.pdf

[López, 2011] Juan Antonio López Riquelme, "Contribución a las redes de sensores inalámbricos. Estudio e implementación de soluciones hardware para agricultura de precisión", Tesis Doctoral, 2012.

[Lowenberg-DeBoer, 2000] Lowenberg-DeBoer, J. and K. Erickson. 2000. Precision Farming Profitability.Agricultural Research Programs, Purdue University.

[Mfbarcell] Disponible on-line http://www.mfbarcell.es/conferencias/wsn.pdf

[Mica2, Mica2dot]
Disponible on-line http://bullseye.xbow.com:81/Products/productdetails.aspx?sid=174
Disponible on-line http://www.willow.co.uk/html/mpr5x0-_mica2dot_series.html

[Olimex] Disponible on-line https://www.olimex.com/Products/MSP430/Proto/MSP430-P1611/

[Pérez, 2008] Miguel Ángel Pérez García y otros, "Instrumentación electrónica", Ed. Thomson Paraninfo, ISBN: 84-9732-166-9.

[Rabaey, 2000] J. Rabaey, J. Ammer, J.L. da Silva Jr., D. Patel. ―Pico-Radio: ad-hoc wireless networking of ubiquitous lowenergy sensor/monitor nodes‖. Proceedings of the IEEE Computer Society Annual Workshop on VLSI (WVLSI'00), Orlanda, Florida, April 2000, pp. 9–12.

[Rodríguez, 2011] Juan José Rodríguez Gil, "Diseño e implantación de un sistema de riego para planta basado en redes inalámbricas de sensores", Septiembre 2011.

[Song, 2008] Song G., Zhou Y., Ding F., Song A. ―A Mobile Sensor Network System for Monitoring of Unfriendly Environments‖. Sensors 2008, 8, pp. 7259-7274.

[TelosB] Disponible on-line http://www.willow.co.uk/TelosB_Datasheet.pdf

[TI] Texas Instruments Inc.. Disponible on-line en: http://www.ti.com/

[Wang, 2001] Wang Maohua. ―Possible adoption of precision agriculture for developing countries at the threshold of the new millennium‖. Millennium Special Issue. Computers and Electronics in Agriculture 30, 45, 50.

[Yick, 2008] Jennifer Yick, Biswanath Mukherjee, Dipak Ghosal. ―Wireless sensor network survey‖. Computer Networks, Volume 52, Issue 12, 22 August 2008, Pages 2292-2330.

[Zhang, 2002] Zhang, N.,Wang, M., Wang N. 2002. Precision agriculture-a worldwide overview, Computers and Electronics in Agriculture 36, 113-132.