Estado actual

PID_00251491

Màrius Montón Macián

Tiempo mínimo de dedicación recomendado: 2 horas

Universitat Oberta de Catalunya

Índice

Introducción

En este material haremos un «estado del arte» o repaso de actualidad de cómo está en estos momentos el mercado y la industria de los sistema empotrados.

1. Microcontroladores actuales

1.1. 8 y 32 bits

Históricamente, los microcontroladores más utilizados en todo el mundo han sido los de 8 bits. Esto venía siendo así por diversas razones, pero principalmente por las siguientes:

- Disponibilidad y precio.
- Historia.
- Necesidad de cómputo.

La primera de ellas, disponibilidad y precio, viene dada por que los fabricantes, al producir masivamente este tipo de dispositivos, los comercializaban (y siguen haciéndolo) con precios muy competitivos. Así, una pieza de un microcontrolador de 8 bits común tiene un precio de menos de un dólar americano. Eso fue así durante muchos años y hasta bien entrados los primeros años del siglo XXI. Fue entonces cuando la bajada de precios de los microprocesadores de 32 bits fue progresiva y poco a poco se han ido equiparando precios.

Así, a día de hoy es posible encontrar microcontroladores de 32 bits de última generación también por menos de un dólar americano.

La segunda razón ha sido la histórica. Debido a que el mercado de los sistemas empotrados es un mercado muy conservador, históricamente con costes de desarrollo muy elevados, ha sido bastante reacio a los cambios. Así, era (y sigue siendo) habitual encontrar equipos de desarrollo que están acostumbrados y tienen grandes conocimientos de una familia o arquitectura antigua y siguen reacios a cambiar a tecnologías más novedosas.

También hay razones de costes de herramientas o de librerías y código desarrollado que se reutiliza en cada proyecto que no se quiere abandonar, que dificultan o han dificultado la transición.

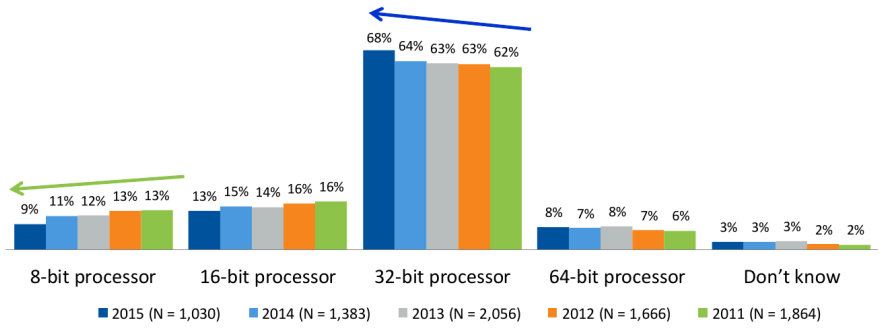
Por último, para muchas aplicaciones empotradas la capacidad de cálculo y de memoria que ofrecen los microcontroladores de 8 bits era y sigue siendo suficiente. En estos casos, usar una tecnología más compleja no aporta ningún beneficio.

Sin embargo, han ido apareciendo nuevas demandas, como nuevas tecnologías de conectividad o pantallas para la interfaz de usuario grandes y en color, que están requiriendo procesadores más potentes y con más memoria para su manejo.

Pese a todas estas razones, el uso de microcontroladores de 8 bits está en claro retroceso y son sustituidos por microcontroladores avanzados de 32 bits. Según la encuesta [Quinnell (2015)], en el año 2015 el 68 % de los procesadores usados eran de 32 bits, frente al 9 % que eran de 8 bits y el 13 %, de 16 bits (ver figura 1).

Cabe mencionar que hay microcontroladores de 64 bits y que su uso va en aumento, siendo en el año 2015 del 8 %.

Figura 1. Resultados encuesta [Quinnell (2015)]



Obtenida de [Quinnell (2015)].

También es destacable que la velocidad de reloj de los microcontroladores ha ido aumentando; en 8 bits una velocidad máxima típica es de unos 16 MHz, pero se ha pasado a velocidades habituales de 32, 48 o incluso 100 MHz en los microcontroladores de 32 bits. Este incremento de velocidad, unido a la mayor capacidad de cálculo, ha abierto nuevas oportunidades en las aplicaciones que un sistema empotrado sencillo puede implementar.

1.2. ARM y la familia Cortex-M

Históricamente, ha habido un conjunto relativamente pequeño de empresas fabricantes de microcontroladores. Estas compañías tenían sus familias de microcontroladores propias, cada una con una arquitectura y dispositivos distinta. Esto provocaba que cada fabricante dispusiera de sus herramientas, compiladores, librerías, etc., distintas a las de otros fabricantes.

Por ello, había cierta tradición dentro de una empresa a seguir ligado a un fabricante concreto, ya que sus equipos de ingenieros, desarrolladores, herramientas, etc., ya estaban habituados a trabajar con las herramientas y particularidades de dicho fabricante.

Así, empresas habituales en el mundo de los microprocesadores eran Microchip, Motorola (luego Freescale, luego NXP), Renesas (fundada en parte por Hitachi), Texas Instruments, Atmel (ahora comprada por Microchip), Intel, IBM o Toshiba.

Este mercado cambió con la aparición de ARM y sus CPU de 32 bits. Esta compañía, en lugar de desarrollar una arquitectura, las herramientas asociadas (compilador, *debugger*, etc.), fabricar los dispositivos y ponerlos a la venta, lo que hizo fue diseñar la arquitectura y licenciarla para que fuesen otros quienes realizasen la fabricación y venta.

Así, la compañía ARM se centraba en diseñar una buena arquitectura y una buena especificación de cómo implementarse y fabricarse, y en escribir buenas herramientas para los desarrolladores.

Con la llegada de ARM en el mercado, hubo una irrupción en este, y compañías históricas en la fabricación de microprocesadores casi han desaparecido y han aparecido nuevas centradas casi exclusivamente en la arquitectura ARM.

1.2.1. Cortex-M

La arquitectura mayoritaria en microcontroladores de 32 bits es la de ARM, llamada Cortex-M. Como ya se ha comentado, con este mismo *core* los distintos fabricantes ofrecen distintas familias y dispositivos, todos ellos con varias opciones de periféricos, encapsulado y conectividad.

Así, la familia Cortex-M incluye distintos *cores* según su potencia y complejidad (véase figura 2). Pese a sus diferencias, los distintos *cores* comparten un juego de instrucciones común y unas especificaciones similares, de modo que es muy parecido trabajar con ellos y pueden compartirse las herramientas de diseño.

Los *cores* disponibles actualmente dentro la familia Cortex-M son [ARM (2017b)]:

- Cortex-M0: *core* más sencillo y optimizado para muy bajo consumo. Carece de instrucciones complejas o de punto flotante.

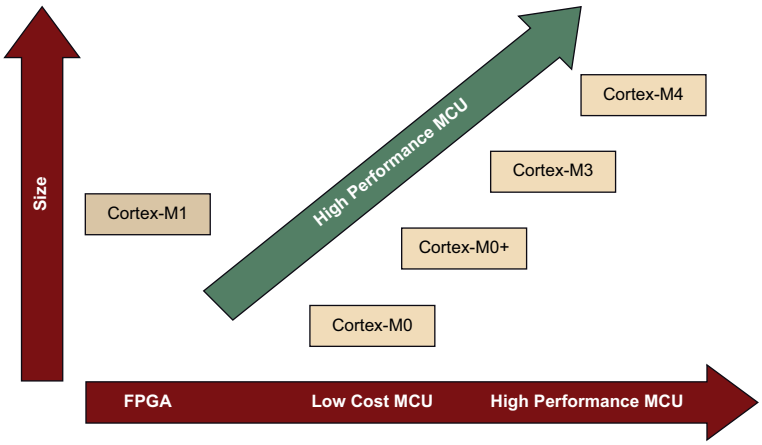
- Cortex-M0+: este *core* es una evolución del M9, añadiendo mejoras en cuanto a rendimiento y mejores herramientas internas para *debug*.

- Cortex-M3: *core* de alto rendimiento, con módulo hardware para la multiplicación de números enteros, mejor soporte de interrupciones y mejoras de rendimiento generales.

- Cortex-M4: evolución del Cortex-M3 añadiéndole instrucciones tipo DSP y una unidad de procesado para números en punto flotante. En el caso de *cores* Cortex-M4 con la unidad de punto flotante se llaman Cortex-M4F.

- Cortex-M7: *core* de muy alto rendimiento, con ejecución de instrucciones fuera de orden (superescalar) e integración de una unidad de punto flotante mejorada. Integra un primer paso hacia los microcontroladores de 64 bits, usando buses de sistema con este ancho. En el caso de *cores* Cortex-M7 con la unidad de punto flotante se llaman Cortex-M7F.

Figura 2. Familia Cortex-M



Los fabricantes ofrecen estos *cores* integrados con los periféricos de su elección para tener un catálogo de dispositivos extenso y que cubra las necesidades de la mayoría de las aplicaciones.

Así, los fabricantes eligen una serie de periféricos según las necesidades que detectan en el mercado. De esta manera, para cada *core* ofrecido (Cortex-M0, M0+, M3, etc.) hay distintas opciones de qué periféricos hay y su número en distintos dispositivos.

Un solo fabricante puede tener cientos de dispositivos distintos, cada uno con un número de periféricos, velocidad de funcionamiento y tamaño de memoria ajustado a una necesidad concreta.

Por ejemplo, NXP (antigua Frescale, antes Motorola) presenta un *core* Cortex-M4 que puede funcionar a 180 MHz, con 2 MB de memoria FLASH y 256 kB de RAM, con memoria caché, 6 UART, 3 SPI, 4 I2C y 2 USB, 25 *timers* y 2 ADC de 16 canales cada uno por 8,5 dólares americanos y tiene también un Cortex-M0+ con el que puede funcionar a 20 MHz, con 16 kB de FLASH, 2 kB de RAM, 2 UART, 2 SPI y 1 I2C, un solo canal de ADC de 12 bits y 10 *timers* por 0,64 dólares.

Como puede verse en el ejemplo de NXP (otros fabricantes tienen dispositivos familiares y la misma diversidad), existe una gran variedad de configuraciones de periféricos y *cores*. También podemos ver que los precios son realmente económicos según los casos.

Hay que remarcar que, aunque el *core* sea el mismo entre distintos fabricantes, al usar periféricos diferentes puede suceder que código escrito para un dispositivo o familia

de un fabricante concreto no sea totalmente compatible con el mismo *core* de otro fabricante.

Por ello, propuestas como la librería CMSIS (véase el apartado «CMSIS» dentro del material titulado «Programación» para más detalles) tienen cada vez más sentido, ya que usando este tipo de librerías, sí que sería posible intercambiar de manera casi transparente un microcontrolador de un fabricante por otro sin tener que hacer grandes cambios en el código.

1.3. Otras arquitecturas

Aunque la arquitectura Cortex-M se ha popularizado en estos últimos años y es, quizá, la que más dispositivos del mercado usan, aún hay muchos fabricantes que mantienen arquitecturas desarrolladas por ellos.

La ventaja de este tipo de arquitecturas es que siendo antiguas tienen un buen historial, que se han reducido al mínimo errores que hubiera, tanto en la implementación de la arquitectura como en las herramientas de desarrollo, y que los ingenieros tienen amplia experiencia con ellas.

Algunas de las compañías que mantienen una o varias familias de arquitectura propia son:

- Atmel y su arquitectura AVR de 8 y 32 bits. La familia AVR32 UC3 está basada en esta arquitectura. Esta familia incluye dispositivos con unidad de punto flotante y tamaño de FLASH grandes (512 kB). Las herramientas son propias de Atmel, incluyendo editor de texto, compilador y herramientas de *debug*.

- NXP. La antigua Freescale (y aún más antigua Motorola) todavía mantiene vigentes microcontroladores muy antiguos (¡años ochenta!) en su familia ColdFire. Aún mantiene una compatibilidad parcial con los microcontroladores antiguos. Gracias a las mejoras tecnológicas ha ido aumentando su capacidad y su velocidad de funcionamiento, hasta llegar actualmente a los 80 MHz (las primeras versiones funcionaban a 8 MHz como máximo).

- Texas Instruments mantiene su arquitectura, llamada MSP430. Esta arquitectura es de 16 bits y está diseñada para ser de muy bajo consumo y bajo coste. Estos dispositivos tienen modos de bajo consumo, donde se desconectan la mayoría de los periféricos y la CPU deja de funcionar, lo que baja el consumo a 0.1 μA.

- Renesas. Esta compañía japonesa mantiene aún una gran variedad de familias de microcontroladores distintas. Tiene familias de 8, 16 y 32 bits, la mayoría con una arquitectura propia y algunas con arquitectura ARM.

2. Módulos integrados

Se entiende por módulos integrados los dispositivos que integran una funcionalidad completa y están pensados para ser fácilmente integrados en un sistema. Con el abaratamiento de los microcontroladores han ido apareciendo cada vez más módulos integrados de todo tipo.

Por ejemplo, con la creciente complejidad de los sistemas de comunicación *wireless* (Wi-Fi, Bluetooth, ZigBee, etc.) han aparecido módulos que integran esos protocolos junto con el *transceiver* de radio. El protocolo se ejecuta de forma opaca por el microcontrolador y ha sido diseñado por el fabricante del módulo. El módulo tiene una interfaz sencilla para que el microcontrolador principal pueda controlarlo.

En estos módulos habitualmente es posible actualizar el código que corre el microcontrolador interno para poder arreglar *bugs*, adaptarse a nuevos protocolos, añadir nuevas funcionalidades, etc.

Estos módulos permiten a los desarrolladores añadir una complejidad compleja de forma sencilla al diseño. Por ejemplo, añadir un módulo Wi-Fi a un diseño con un microcontrolador básico (tipo Cortex-M0+) le añade esta conectividad sin hilos sin tener que estar pendiente de los protocolos Wi-Fi. El módulo nos permitirá usarlo con una interfaz sencilla y una API clara y rápida de implementar en nuestro microcontrolador.

Siguiendo con el ejemplo, si el módulo está certificado Wi-Fi (la mayoría lo están), se podrá comercializar el dispositivo que lo use como certificado Wi-Fi, sin tener que pasar las certificaciones necesarias.

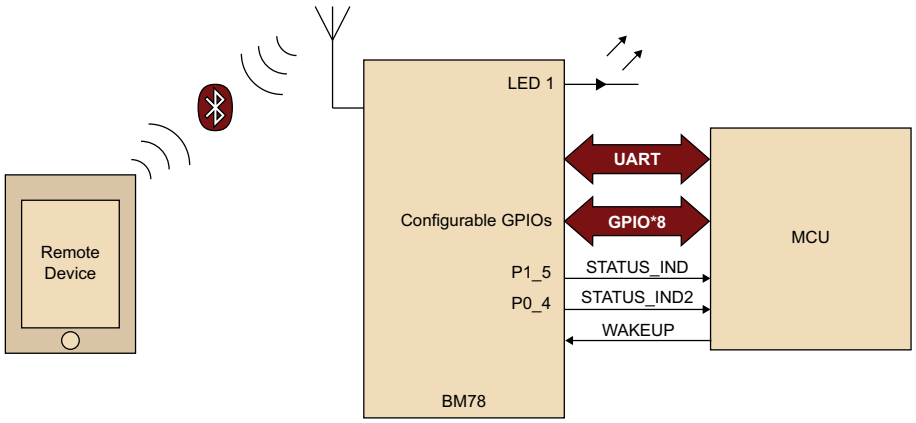
Esta característica es una de las más importantes del uso de módulos integrados. Las certificaciones de ciertos estándares y protocolos pueden llegar a ser muy complejas y costosas y, por tanto, el uso de módulos integrados que ya estén precertificados va a simplificar y abaratar mucho el proceso, siendo en algunos casos totalmente innecesaria dicha certificación. Este es el caso para la mayoría de los módulos que integran Bluetooth, Wi-Fi, ZigBee, LoRa, etc.

Debido, en parte, a las certificaciones que cumplir y a decisiones estratégicas de las compañías, la mayoría de los módulos integrados no permiten (o lo permiten de manera muy limitada) la programación del microcontrolador integrado.

Esto se debe a que, si se tiene que cumplir una certificación, el fabricante debe asegurarse de que nadie cambia el comportamiento el módulo y se incumplen los requisitos de dicha certificación.

Este es el caso, por ejemplo, del módulo Bluetooth BM78 de Microchip, módulo que integra todos los elementos necesarios para tener una conexión Bluetooth versión 4.2. En este módulo, y para cumplir con las especificaciones del consorcio Bluetooth, el fabricante no permite introducir código dentro del módulo. El módulo se accede a través de un puerto serie y se maneja con comandos sencillos (ver figura 3).

Figura 3. Diagrama de bloques del módulo BM78



Estos comandos permiten el control de todo el *stack* Bluetooth empotrado dentro del módulo. Así, con un comando se configura el nombre que publicará el dispositivo, con otro comando se inicia la conexión con un controlador máster (este proceso se llama *pairing* en Bluetooth), con otro se inicia el envío de datos, etc. Todos los comandos, qué parámetros aceptan y cómo deben usarse está ampliamente documentado por el fabricante.

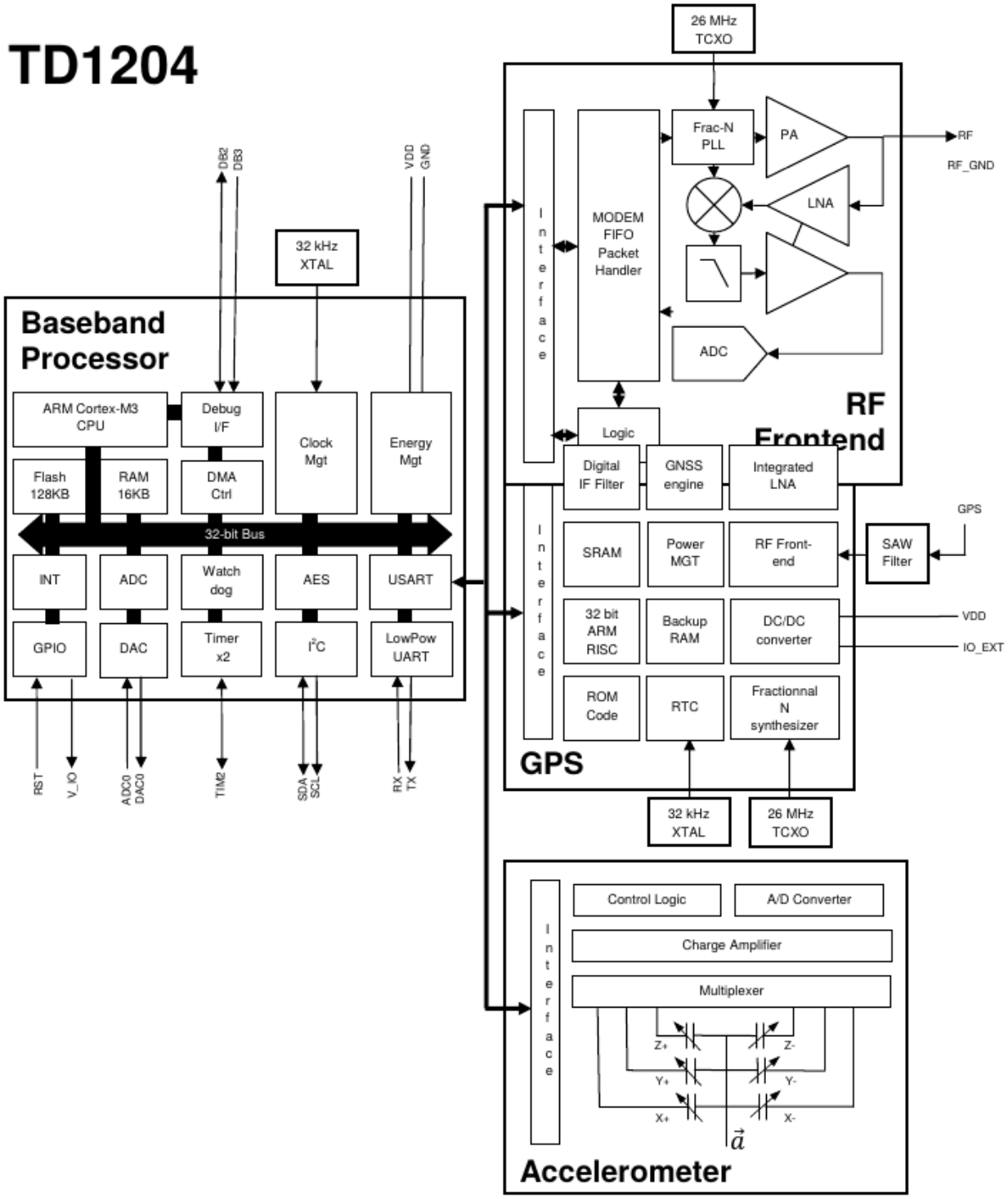
En algunos casos, los fabricantes dan cierto acceso limitado a poder programar cierto código que se ejecuta en el microcontrolador del módulo. En estos casos, lo más habitual es encontrarse con que el usuario del módulo puede escribir un código principal que estará controlado por el sistema integrado.

Este es el caso del módulo TD1208 de Telecom Design [Telecom Design (2017)], un módulo que integra un microcontrolador y un *transceiver* compatible con la red de largo alcance SIGFOX. En este caso, y para cumplir las especificaciones de la compañía SIGFOX, el fabricante proporciona una mezcla de código para el usuario y unas librerías cerradas. Con estas librerías, el usuario puede escribir su aplicación, manejar ciertos periféricos del microcontrolador integrado y, usando la librería cerrada, enviar datos usando la red SIGFOX.

Esta misma compañía tiene un módulo integrado muy parecido (el TD1207) que no permite la programación del microcontrolador integrado y ofrece a cambio una interfaz serie con comandos sencillos para enviar y recibir datos de la misma red. Por último, este mismo fabricante proporciona un módulo que integra también un receptor de GPS y un acelerómetro (TD1204, figura 4) y las librerías asociadas. Así, desarrollar un dispositivo que vaya enviando periódicamente su posición GPS usando la red

SIGFOX se convierte en un trabajo sencillo sin tener que conocer las particularidades ni de la red de comunicación ni de la red de posicionamiento global.

Figura 4. Diagrama de bloques interno del módulo integrado TD1204



© TD Next. Se pueden ver en el diagrama los cuatro bloques principales que componen el sistema: un microcontrolador (a la izquierda); el *transceiver* de radio (derecha arriba); receptor GPS (derecha medio); acelerómetro (derecha abajo).

Otra característica importante de este tipo de módulos, sobre todo en los que proporcionan conectividad *wirelesss*, es que llevan integrada, en la mayoría de los casos, toda la circuitería de adaptación de la antena. Estos circuitos permiten el trabajo óptimo de

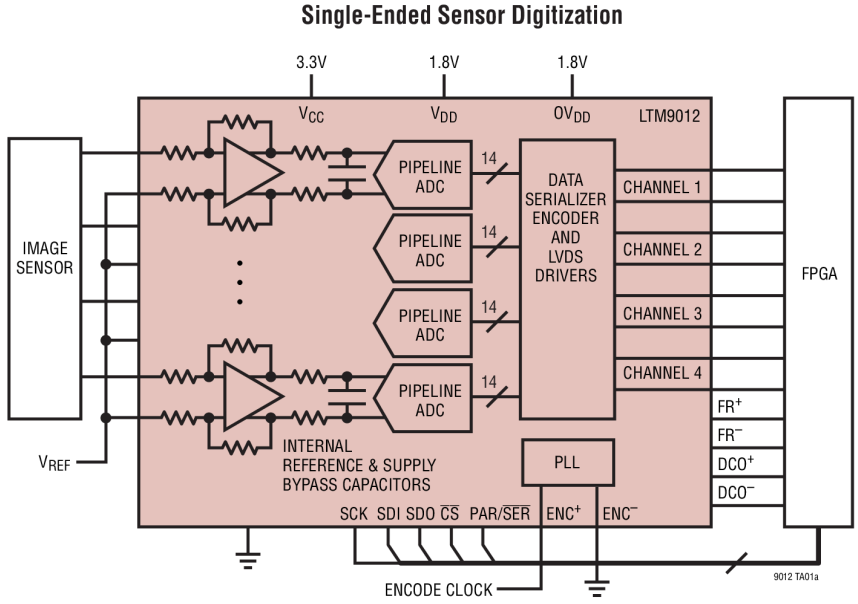
la antena de comunicación y normalmente son un complejo problema de ingeniería. Usando módulos que llevan integrado estos circuitos se ahorran tiempo y costes de desarrollo.

Por último, cabe mencionar que no son todo ventajas en el uso de estos módulos, ya que normalmente su coste económico es mayor que el conjunto de los componentes que incorporan. Así que para casos en los que el producto en desarrollo tenga costes muy ajustados o se vaya a producir en grandes cantidades se puede estudiar el reemplazo de un módulo integrado por sus componentes directamente en el diseño.

Aparte de los módulos integrados para temas de conectividad (normalmente, conectividad sin hilos), hay módulos integrados para todo tipo de funcionalidades. Así, es también común encontrar módulos que integran un ADC de alto rendimiento junto a toda la circuitería de adaptación para un tipo de señal concreto y proporcionando una interfaz simple tanto de datos como de control. De esta manera se ahorran, de nuevo, costes en la circuitería delicada, como puede ser la que afecta al tratamiento de la señal antes de la digitalización.

Un ejemplo de este tipo de módulos integrados es el LTM9012 de Analog Devices [Analog Devices (2017)]. Este módulo integra un conjunto de ADC junto a la electrónica de adaptación de señales provenientes de sensores de imagen especiales y los circuitos de tratamiento y conversión de los datos a un tipo específico de interfaz serie (figura 5).

Figura 5. Diagrama de bloques interno del módulo integrado LTM9012



3. Bajo consumo

Gracias a la evolución tecnológica tanto de los dispositivos como de las baterías, están apareciendo más aplicaciones empotradas de bajo consumo y alimentadas por baterías.

Ya se ha visto en el apartado «Programación para bajo consumo» dentro del material titulado «Programación» el estado actual de los microcontroladores en este tema. La mayoría de los microcontroladores actuales tienen modos de bajo consumo donde diferentes partes de este están desconectadas o suspendidas.

Así, es posible tener un microcontrolador en un estado de muy bajo consumo (apenas un μA) mientras un periférico tipo *timer* o RTC mantiene el control del tiempo transcurrido. De esta manera, el periférico encargado de mantener el tiempo actualizado va despertando el microcontrolador cada cierto tiempo y realizando las acciones necesarias.

Lo mismo sucede con los sensores y dispositivos externos al microcontrolador. La mayoría de ellos tienen diversos modos de funcionamiento, que son controlados por el desarrollador y que permiten o bien poner a dormir el dispositivo para reducir su consumo, o bien activar un modo de bajo consumo a costa de bajar las prestaciones del dispositivo.

Algunos dispositivos tienen modos de funcionamiento para bajo consumo que pueden ser preconfigurados para trabajar con un consumo mínimo y que permiten despertar todo el sistema en caso de necesidad. Por ejemplo, existen acelerómetros que en un modo de bajo consumo, si detectan una vibración elevada y brusca (que puede ser provocada por un golpe), son capaces de detectar el suceso y generar una interrupción para despertar el microcontrolador y activar el código que sea necesario (por ejemplo, enviar una alarma de robo o de manipulación de un equipo sensible). Así, tener este sistema de monitorización puede realizarse con un consumo mínimo sin perder su capacidad y, por tanto, puede ser alimentado mediante baterías.

Bibliografía

Analog Devices (2017). «LTM9012 - Quad 14-Bit, 125Msps ADC with Integrated Drivers».
URL: http://www.linear.com/product/LTM9012.

ARM (2017b). «Cortex-M Series Family». URL: https://www.arm.com/products/ processors/cortex-m.

Quinnell, R. (2015). «2015 Embedded Markets Study». *Report técnico*, ARM, EDN network.

Telecom Design (2017). «TD1208». URL: http://rfmodules.td-next.com/modules/ td1208/.