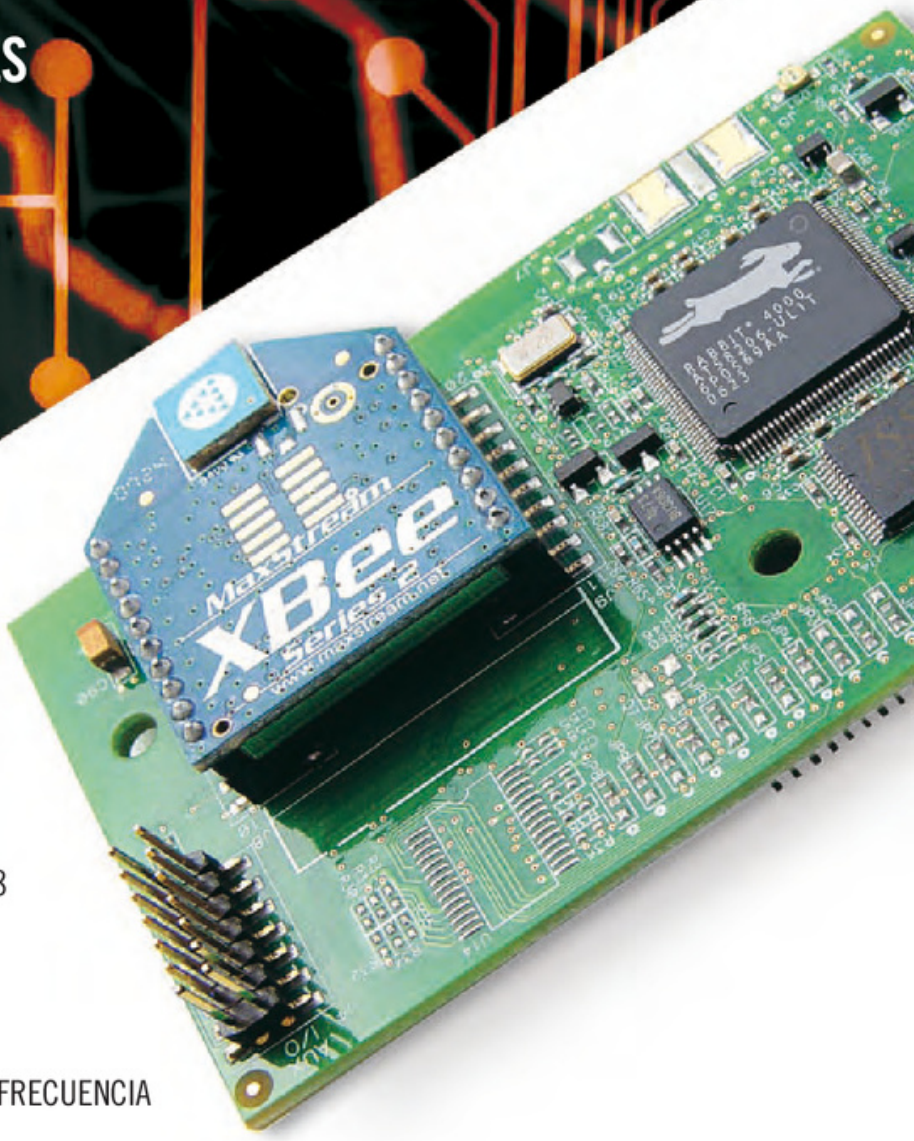
USERS

¡CONTIENE EJEMPLOS PRÁCTICOS PARA PONER MANOS A LA OBRA!

PROYECTOS CON MICROCONTROLADORES

APRENDA A DESARROLLAR SUS PROPIAS APLICACIONES



ADEMÁS

APLICACIONES CON MICROCONTROLADORES PIC18

DISPLAYS LCD ALFANUMÉRICOS Y GRÁFICOS

MÓDULOS RABBIT, ZIGBEE Y BLUETOOTH

CONTROL REMOTO POR INFRARROJO Y POR RADIOFRECUENCIA

COLECCIÓN USERS ELECTRÓNICA

DESCUBRA CÓMO ACCEDER REMOTAMENTE A SUS EQUIPOS

» HARDWARE
» 192 PÁGINAS
» ISBN 9978-987-1773-24-4

En esta obra nos introducimos en los conceptos relacionados con el networking y los protocolos de comunicación, y los pondremos en relación con los microcontroladores, para así conectarlos en red. Desde la programación con RabbitWeb hasta el control remoto de un motor a través de Wi-Fi, y mucho más.

SOBRE LA COLECCIÓN: USERS ELECTRÓNICA

- Aprendizaje guiado mediante explicaciones claras y concisas
- Proyectos prácticos basados en necesidades reales
- Consejos de los profesionales
- Producciones fotográficas profesionales
- Infografías y procedimientos paso a paso

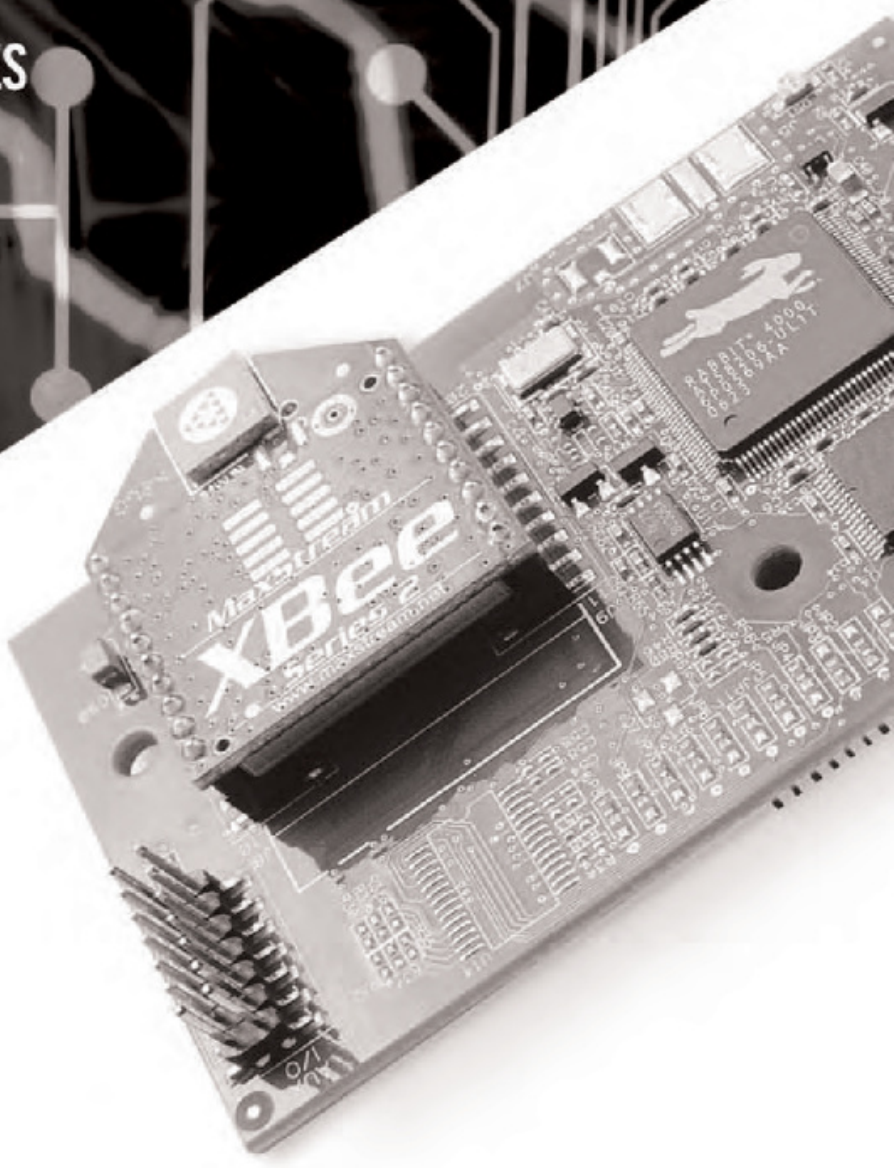
LLEGAMOS A TODO EL MUNDO VÍA OCA* Y DHL**

* SÓLO VÁLIDO EN LA REPÚBLICA ARGENTINA // ** VÁLIDO EN TODO EL MUNDO EXCEPTO ARGENTINA

usershop.redusers.com // usershop@redusers.com

PROYECTOS CON MICROCONTROLADORES

APRENDA A DESARROLLAR SUS PROPIAS APLICACIONES



TÍTULO: Proyectos con microcontroladores
COLECCIÓN: desde Cero
FORMATO: 15 X 19 cm
PÁGINAS: 192

ISBN 978-987-1773-23-7

Proyectos con microcontroladores / coordinado por Daniel Benchimol.
Buenos Aires: Fox Andina; Dalaga, 2011.
v. 20, 192 p. ; 19x15 cm. - (Desde cero)

ISBN 978-987-1773-23-7

1. Informática. I. Daniel Benchimol, coord.
CDD 005.3

RedUSERS.com

Prólogo **al contenido**

Los microprocesadores y los microcontroladores han cambiado la forma de pensar y diseñar los circuitos electrónicos. Desde que Intel lanzó en 1971 el 8080, el primer microprocesador exitoso, estos dispositivos no han dejado de evolucionar. Las microcomputadoras o microcontroladores nacieron a mediados de los '80 y rápidamente ganaron mercado, al desplazar a los sistemas mínimos desarrollados con microprocesadores en el campo del control industrial.

Los microcontroladores, debido a su muy bajo costo, alta inmunidad al ruido eléctrico y pequeño tamaño, produjeron la revolución microcontrolada, que desplazó a toda la lógica cableada (utilizada en la electrónica industrial) y a la lógica programada (realizada con microprocesadores). Es en este campo donde se los bautizó con el nombre de microcontroladores y se desechó el de microcomputadores.

A partir de los '90 los microcontroladores invadieron la electrónica de consumo, brindando a los electrodomésticos y a todo tipo de sistema electrónico de consumo la capacidad de inteligencia y conectividad. El mundo actual está rodeado de microcontroladores; desde nuestros celulares, sistemas de alarmas y lavarropas, hasta las computadoras de abordo de los automóviles. Sin ellos, nuestro mundo actual no existiría.

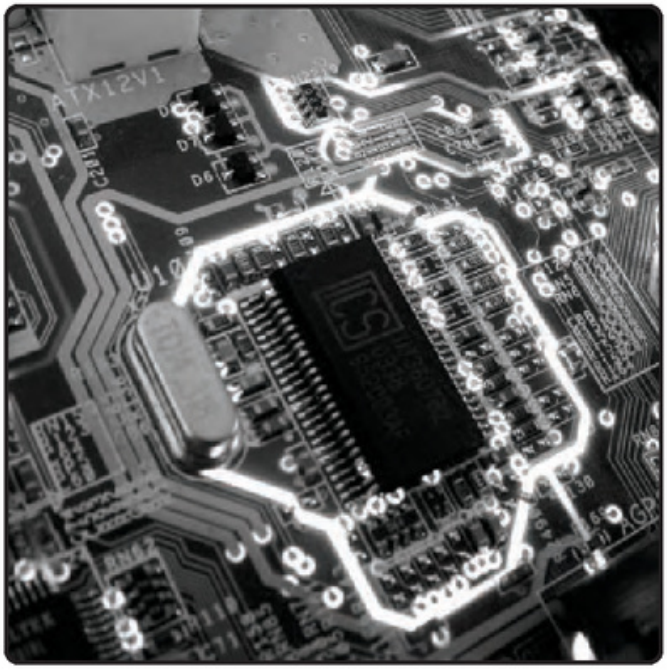
En este libro, nos dedicaremos a realizar proyectos con los microcontroladores, a partir de la teoría expuesta en cada capítulo.

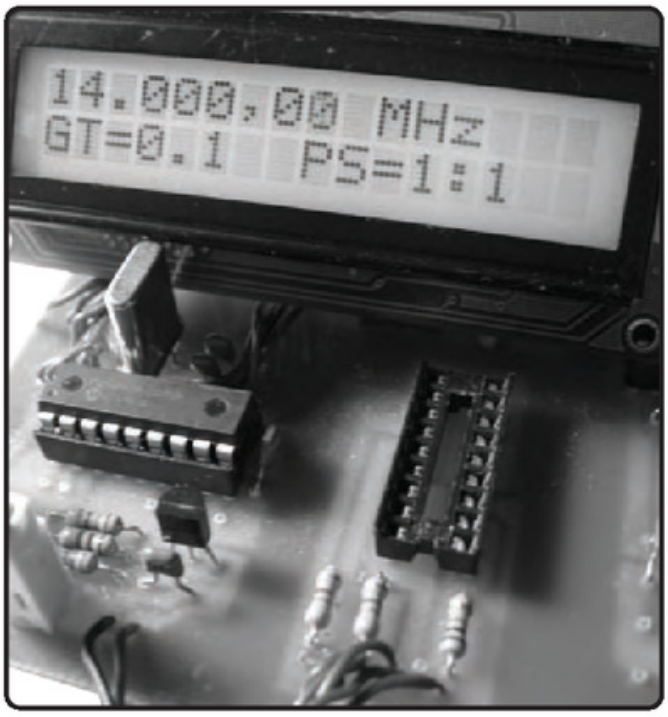
El libro de un vistazo

Este libro está enfocado en aquellas personas que quieran estudiar en profundidad el mundo de los microcontroladores. Empezamos con una introducción a las señales digitales y a la electrónica digital. Analizamos el desarrollo de los microprocesadores hasta llegar a los diferentes tipos de microcontroladores.

CAPÍTULO 1
ARMADO DE UNA PLACA EXPERIMENTAL

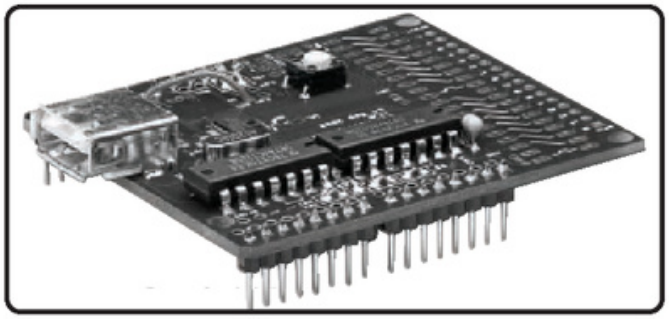
Realizaremos la construcción de una placa experimental para el PIC18LF46 con los componentes necesario para llevar a cabo una simulación de código de programación. Conoceremos algunos de los periféricos básicos que podemos conectar al microcontrolador para interactuar con él. Aprenderemos a utilizar el módulo interno del PIC para generar señales con ancho de pulsos variables (PWM). Haremos un repaso teórico de cada una de las tecnologías en conversores A/D y veremos cuáles son los usos de cada uno de ellos.





CAPÍTULO 2
PERIFÉRICOS PIC

Aprenderemos a utilizar algunos de los periféricos internos del PIC que nos serán muy útiles a la hora de programarlo. Conoceremos el conversor A/D integrado en el PIC, veremos su funcionamiento y con?guración para poder utilizarlo. Analizaremos todo lo relacionado con las pantallas LCD alfanuméricas y aprenderemos a controlarlas con nuestro PIC18LF4620. Con todos los recursos vistos con el PIC, realizaremos un proyecto real que tiene muchas aplicaciones en electrónica.



CAPÍTULO 3
CONECTIVIDAD NO INALÁMBRICA

Veremos las diferentes opciones disponibles para conectar un sistema embebido con la PC. Conoceremos las características más relevantes del puerto serie y del RS-232. Construiremos una placa de expansión para la placa de experimentación que nos permitirá comunicar un microcontrolador con la PC mediante RS-232. Estudiaremos la utilidad de este software a la hora de veri?car, analizar y simular diferentes protocolos de comunicación serie.

CAPÍTULO 4
CONECTIVIDAD INALÁMBRICA

Estudiaremos algunas de las opciones disponibles para establecer una conexión sin cables entre dos o más dispositivos. Mediante una PDA, un celular o una PC, Bluetooth nos permite estar conectados con nuestro proyecto, sin recurrir a cables. ZigBee y 802.15.4: más económico en muchos aspectos que Bluetooth, ZigBee se basa en 802.15.4 para proveer de conectividad inalámbrica entre equipos sencillos. Con módulos ZigBee, vamos a armar la base de una red de sensores que podemos utilizar para tomar mediciones remotas.

CAPÍTULO 5
MÓDULOS RABBIT Y DISPLAYS LCD GRÁFICOS

Conoceremos los módulos Rabbit con todos sus elementos a bordo, cuya vinculación con el circuito por controlar se realiza medianteun conector. Veremos las distintas alternativas disponibles del mundo Rabbit, sus campos de aplicación y características generales. Utilizaremos los displays LCD gráficos y desarrollaremos algunos algoritmos para realizar grá?cos simples. No nos quedaremos en la teoría, sino que veremos ejemplos prácticos acerca de cómo mostrar grá?cos y funciones sobre el display.

SERVICIOS
AL LECTOR

En este último apartado, encontraremos un índice temático que nos ayudará a encontrar de forma más rápida y precisa los principales conceptos de la obra.

Contenido del libro

RedUSERS

MEJORA TU PC

Desarrollos temáticos en profundidad

Libros.

Coleccionables.

Cursos intensivos con multimedia

Capacitación dinámica

Revistas.

Sitios Web.

Noticias al día, downloads, comunidad

Información actualizada al instante

Newsletters.

La red de productos sobre tecnología más importante del mundo de habla hispana.

redusers.com

Introducción

Presentamos diferentes actividades relacionados con el uso de los microcontroladores. Para comenzar, realizaremos la construcción de una placa experimental para el PIC18LF4620 con los componentes necesarios para llevar a cabo una simulación de código de programación. Además, aprenderemos a utilizar el módulo interno del PIC para generar señales con ancho de pulsos variables (PWM).Con todos los recursos que veremos sobre el PIC, realizaremos un proyecto real que tiene muchas aplicaciones en electrónica: la alarma técnica, que nos servirá para indicar la temperatura en el LCD. Más adelante, nos adentraremos en el tema de la conectividad. Primero, estudiaremos la conectividad no inalámbrica y construiremos una placa de expansión para la placa de experimentación que nos permitirá comunicar un microcontrolador con la PC mediante RS-232. En el siguiente capítulo, aprenderemos a utilizar tecnologías inalámbricas para nuestros proyectos de una forma simple y efectiva. Con la utilización de módulos ZigBee, vamos a armar la base de una red de sensores que podemos utilizar para tomar mediciones remotas.Para finalizar los proyectos, nos adentraremos en la tecnología LCD y desarrollaremos algunos algoritmos para realizar gráficos simples. Además, veremos ejemplosprácticos acerca de cómo mostrar gráficos y funciones.

Capítulo 1

Armado de una placa experimental

Aprenderemos a realizar una placa experimental para el PIC18LF4620 con los componentes necesarios.

RedUSERS.com

Placa experimental **para PIC18LF4620**

En las siguientes páginas veremos cómo armar una placa experimental para el **PIC18LF4620** y, así, realizar las prácticas correspondientes. Para realizar los ejercicios de esta obra con el **PIC**, sugerimos armar una pequeña placa experimental que contenga **pulsadores**, **diodos LED y** algún **relay** que permita manejar cargas de mayores potencias. La placa en cuestión es sólo una posibilidad entre las infinitas configuraciones que cada uno puede implementar.

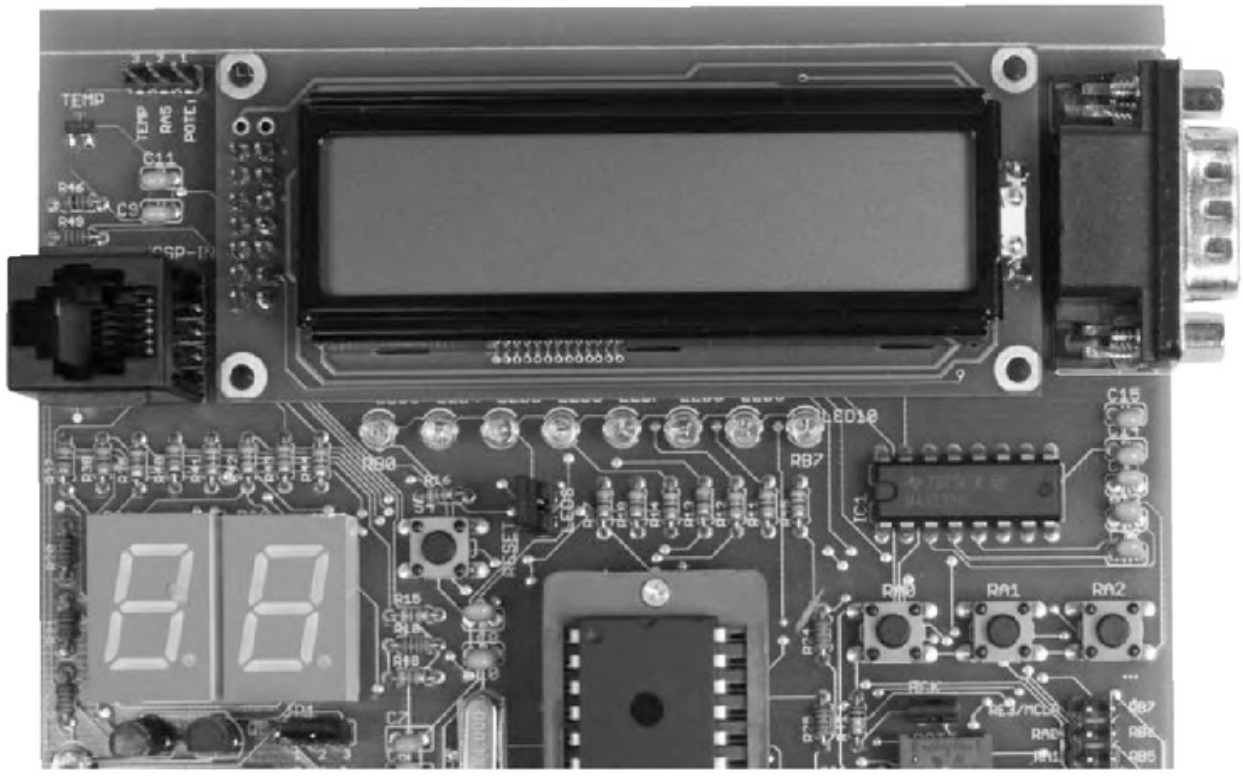
ESQUEMÁTICO

En la **Figura 1** vemos el esquemático de la placa que armaremos. Podemos ver que el centro del esquema circuital es el microcontrolador **PIC18LF4620**, al que se le acoplan distintos dispositivos, como **pulsadores**, **diodos LED** y la **interfaz** para el control de un **relay** (**Figura 2**).

A un costado, distinguimos el circuito de alimentación, que utiliza un regulador de **5 V**, y el **LM7805** con sus capacitores de filtrado.

A esta placa se le agrega un conector de cinco pines para realizar una programación **ICSP** (*In-Circuit Serial Programming* o programación serie en circuito), ya que será utilizada para experimentar con diferentes códigos. Este conector nos resultará de gran utilidad para programar el PIC sin necesidad de retirarlo de la placa.

Los dispositivos externos que agreguemos en la placa serán usados para entrenarnos en la programación



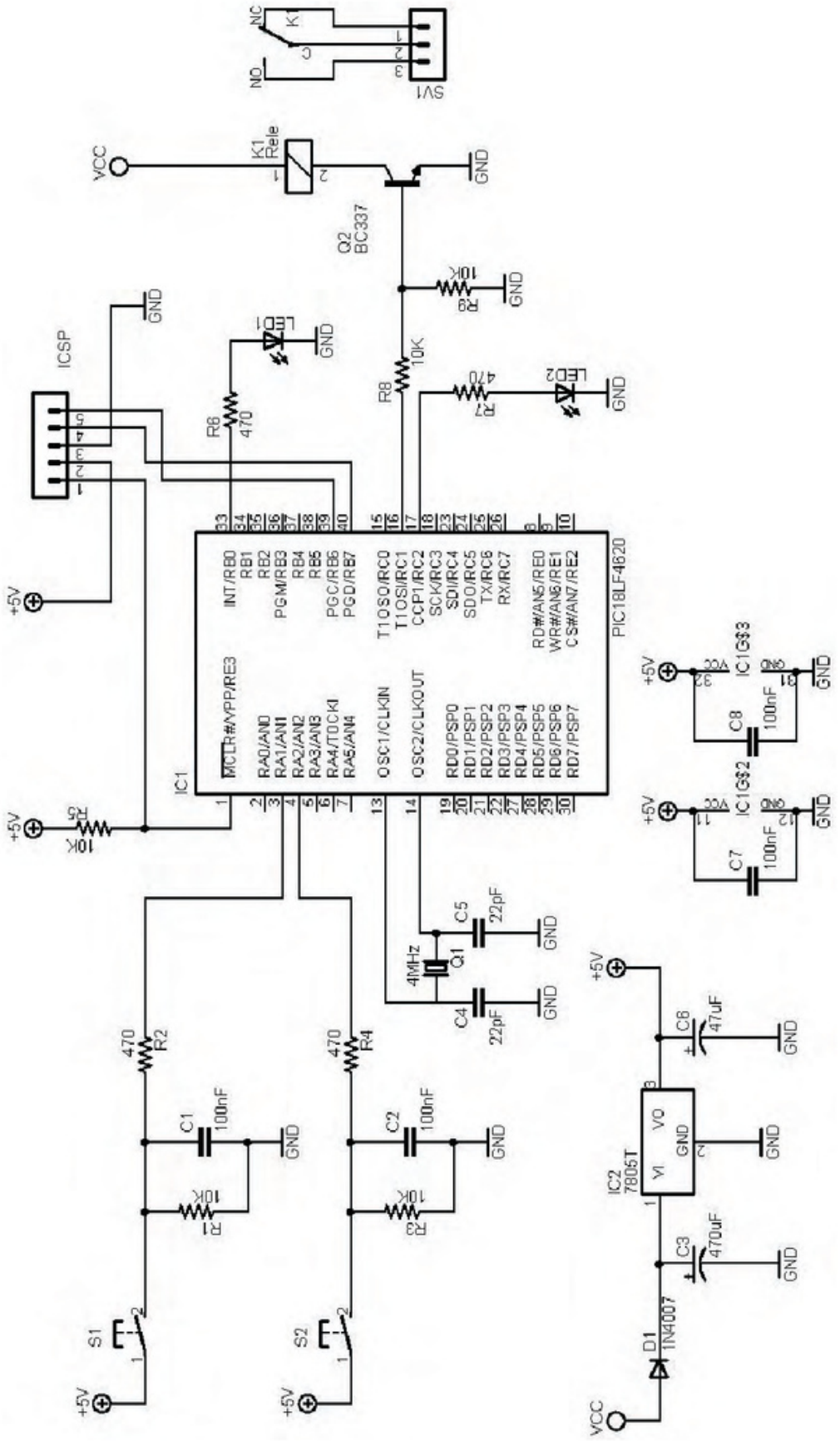


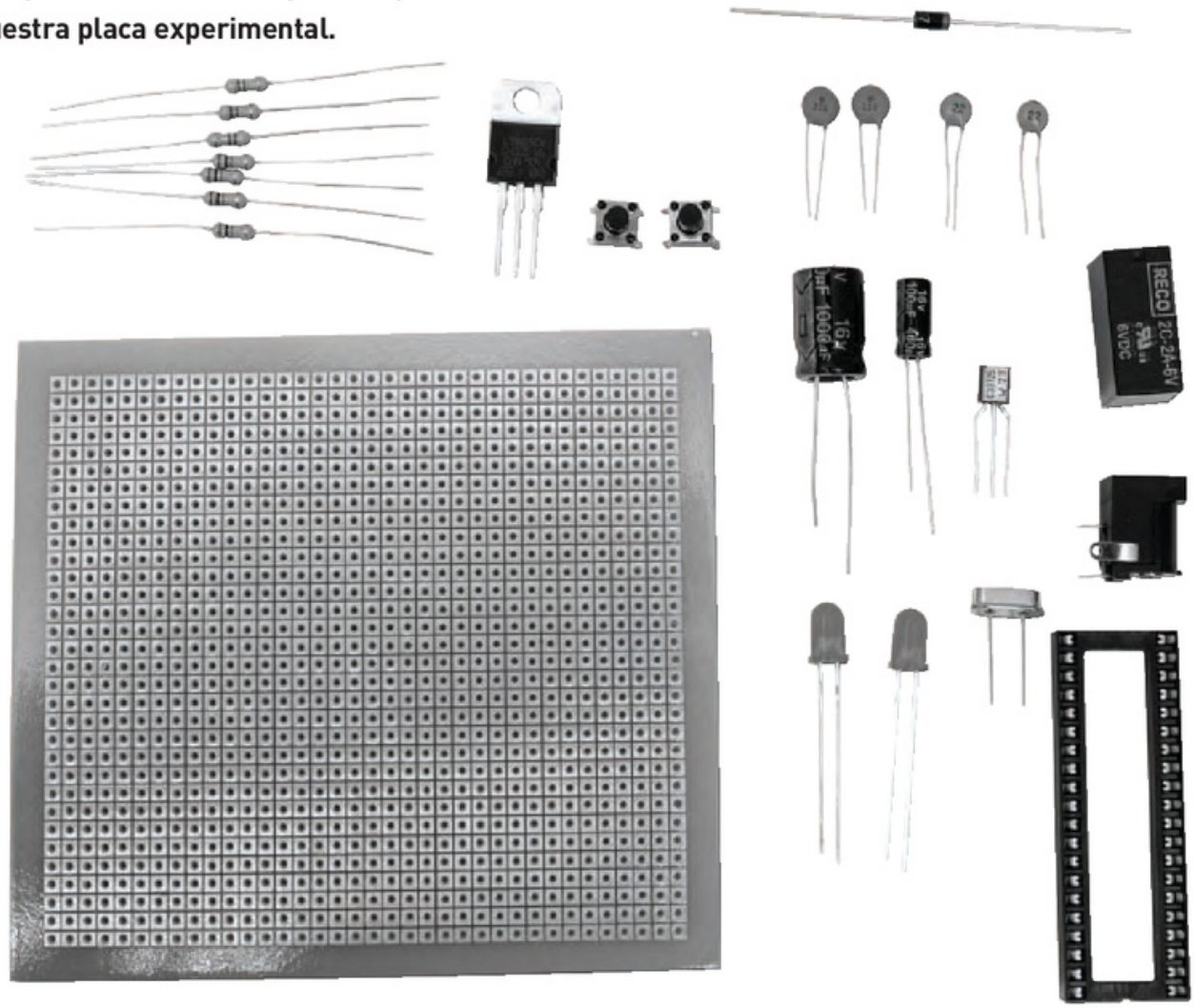
FIGURA 1. Aquí vemos la representación esquemática de la placa experimental con la que realizaremos las prácticas de este capítulo.

del PIC, en tanto que los pulsadores servirán como entradas de datos. En el esquemático observamos también que, cuando el pulsador se encuentra abierto, el circuito impone un 0 lógico, y cuando lo oprimimos, pasa al estado lógico alto. Colocamos diodos LED para indicar el estado de los pines **RB0** y **RC2** del PIC. Los LEDs son muy útiles para esta función, ya que se encenderán cuando coloquemos ese pin en estado alto. El del pin **RC2** será utilizado en las prácticas de **PWM** (modulación de ancho de pulso), donde podremos variar su intensidad por medio de la técnica de modulación.

CONSIDERACIONES DE ARMADO

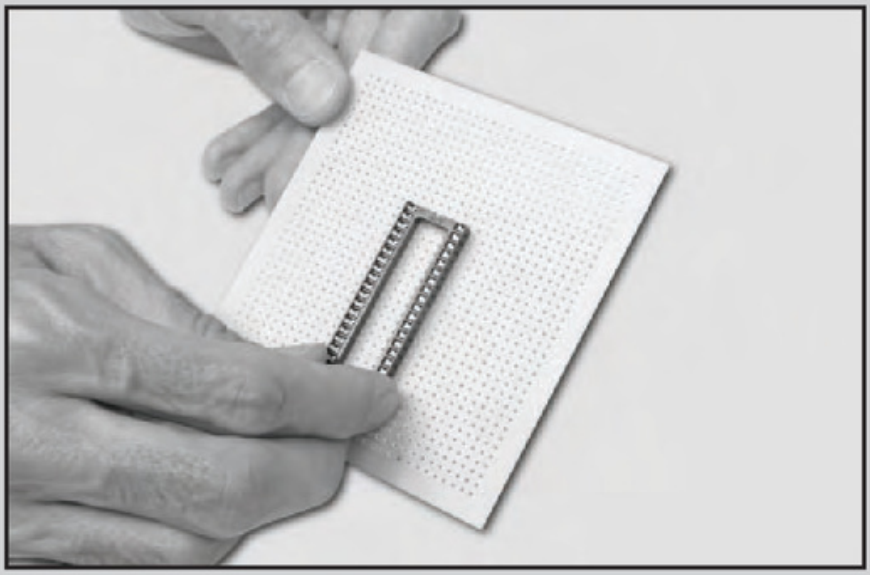
Al diseñar y armar una placa experimental, debemos tener presentes algunos consejos que daremos en estas páginas para asegurar un trabajo exitoso. En el **Paso a paso 1** vemos las recomendaciones que también nos servirán en el armado de futuras placas que cuenten con algún microcontrolador.

FIGURA 2. **Éstos son todos los componentes con los que debemos contar para empezar a armar nuestra placa experimental.**



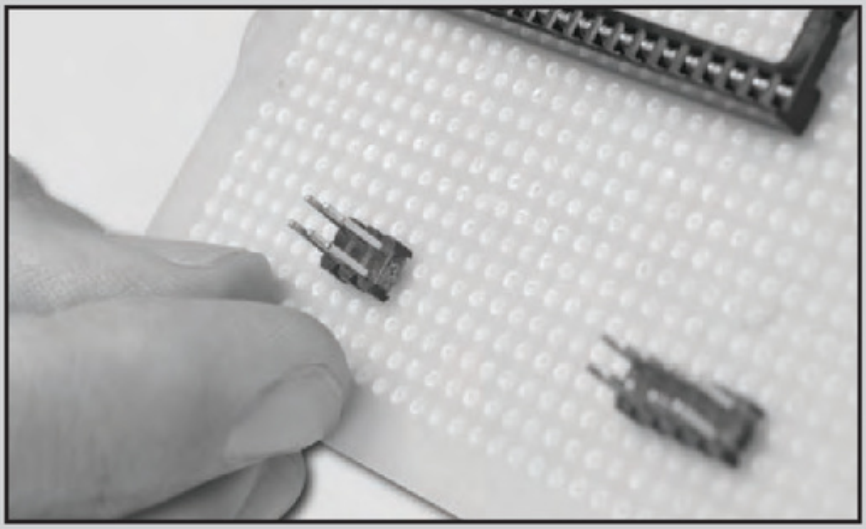
PASO A PASO /1
Consideraciones de armado

1



Una vez que tiene todos los materiales necesarios, Puede empezar a armar la placa. Comience colocando el zócalo DIP de 40 pines, para, luego, poder orientarse en el momento de instalar los demás componentes.

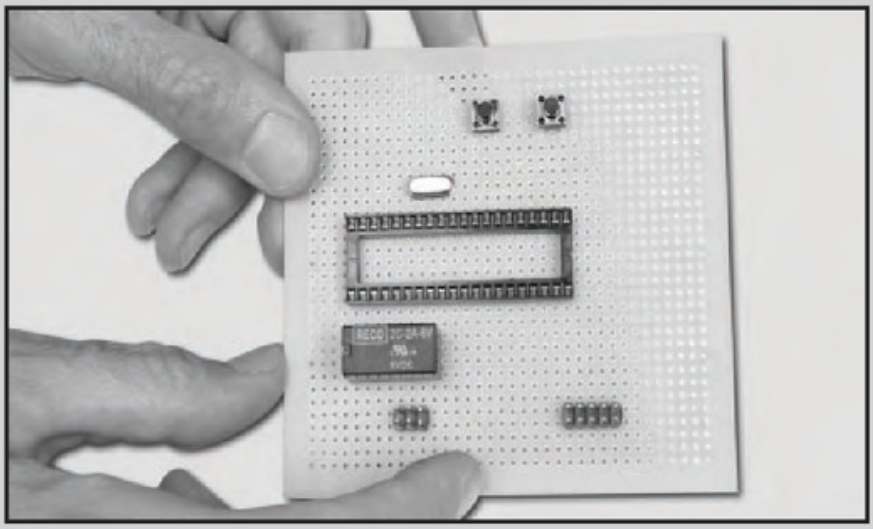
2



Luego de insertar el zócalo, ponga los demás componentes, guiándose con el esquemático del circuito. Es importante colocar los componentes más chicos, como conectores y pulsadores, para que el armado no resulte incómodo. Si empieza con los más grandes, le será difícil manipular la placa.

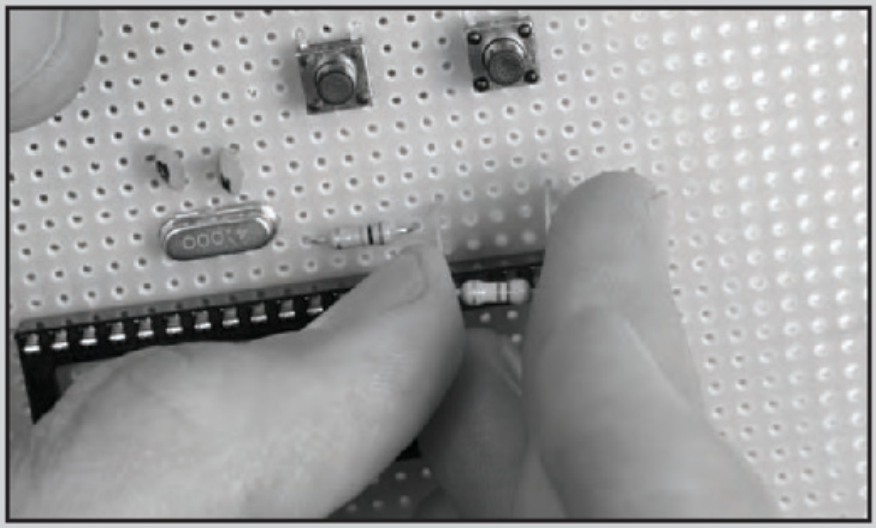
PASO A PASO /1 (cont.)

3



En el momento de elegir en dónde colocar el cristal resonador, ubíquelo lo más cerca posible del PIC, ya que el circuito oscilador es la parte de la placa que trabaja a mayor frecuencia. Si mantiene las pistas del circuito oscilador lo más chicas posible, evitará muchos problemas, considerando que esta zona es muy sensible a interferencias externas.

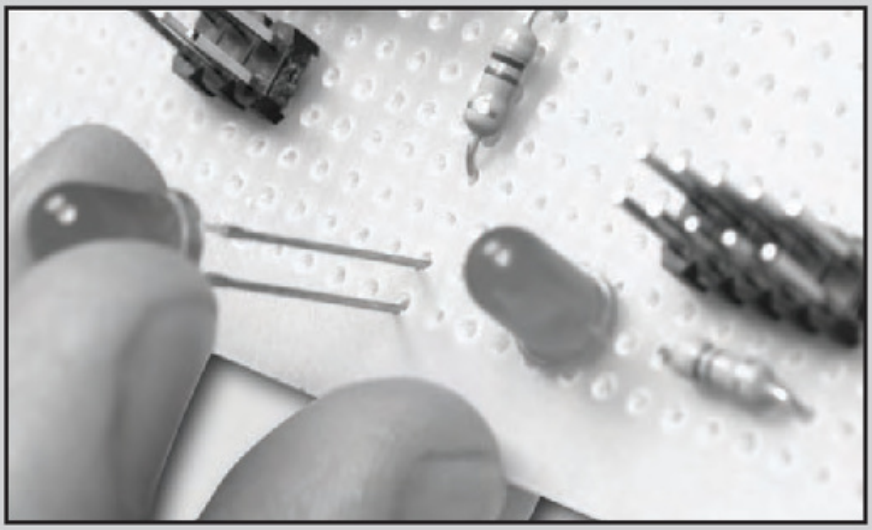
4



Coloque las resistencias y los capacitores. Para el oscilador, además, utilice capacitores de **22 pF** del tipo **NPO** (**Negative-Positive zero**), por ser más estables a variaciones de temperatura, y muy útiles en circuitos oscilantes y filtros. Se reconocen porque poseen una marca negra en la parte superior.

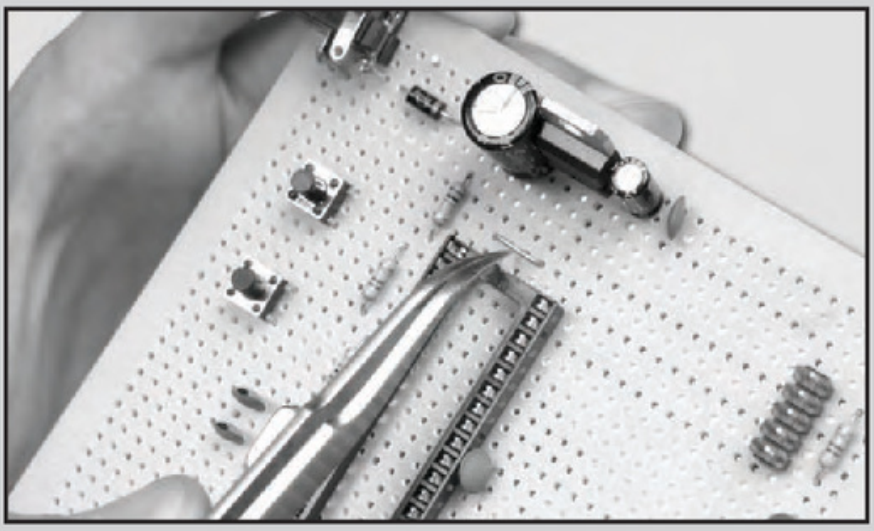
PASO A PASO /1 (cont.)

5



Debe tener especial cuidado al colocar **los LEDs indicadores**, dado que tienen polaridad y, si los pone al revés, no encenderán. Recuerde que el borde achatado del cuerpo marca el cátodo del dispositivo, que debe ser conectado, en este caso, a la masa del circuito.

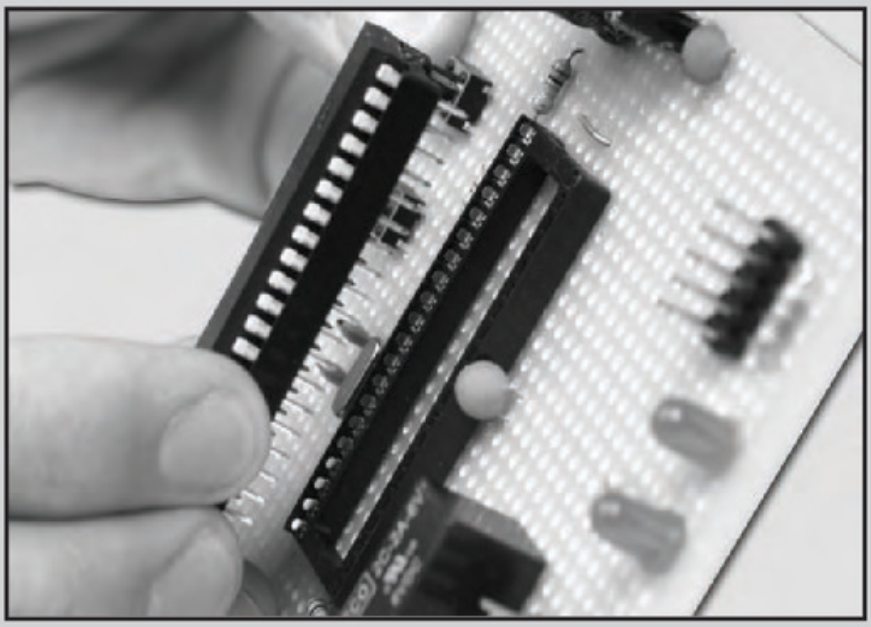
6



En el momento de comenzar a armar el circuito de alimentación, hay que colocar primero el regulador LM7805. Posiciónelo en un extremo de la plaqueta, para poder agregarle un disipador de calor más tarde. Ubique los capacitores electrolíticos a los lados del regulador y verifique que su polaridad sea correcta.

PASO A PASO /1 (cont.)

7



Cuando termine de soldar y probar la placa, puede colocar el PIC en el zócalo DIP de 40 pines. También puede utilizar un zócalo adicional para evitar romper alguna patita cuando tenga que maniobrar con él, por ejemplo, cuando lo saque del zócalo para grabarlo en el programador.

Conexión al PIC de **periféricos externos**

Los LEDs son los indicadores luminosos más simples que podemos conectar al PIC. En la **Figura 2** podemos ver dos modos de conectar el LED al pin del microcontrolador: una de las configuraciones excita al LED con una salida en alto en el pin, y la otra, con un nivel bajo. Para dimensionar la resistencia utilizamos la fórmula: **R = (Vf – Vled) / Iled,** donde **Vf** puede ser la tensión de salida del pin o la tensión de la fuente; **Vled** es la tensión del LED que el fabricante nos da como dato; e **Iled** es la corriente del LED, que puede estar entre **5 mA** y **20 mA**, según la intensidad que deseamos tener (colocando una resistencia de **470 Ω**, obtenemos un indicador lumínico aceptable).

Los LEDs son los indicadores luminosos más simples que podemos conectar al PIC

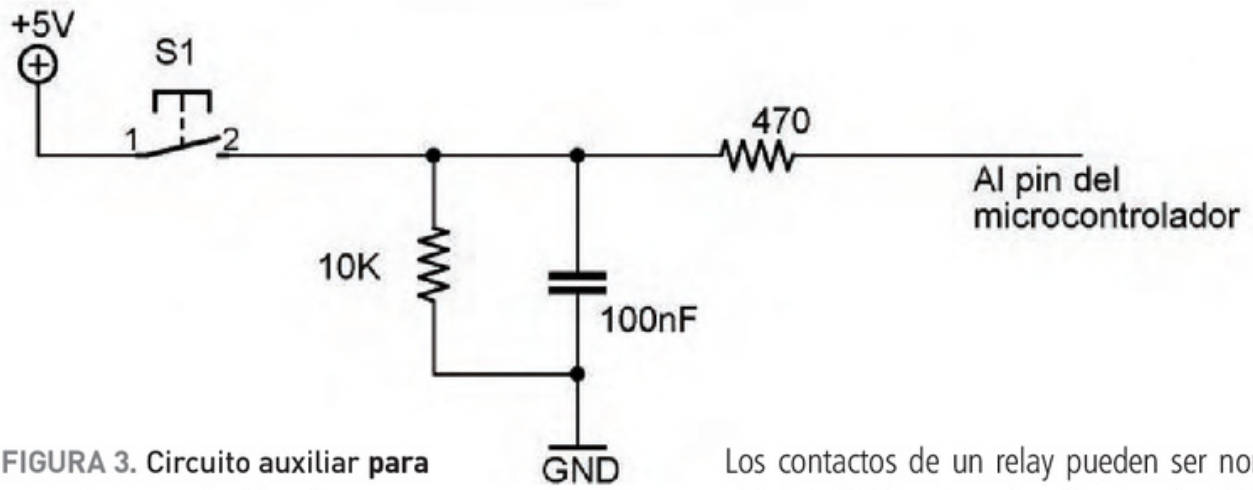


FIGURA 3. Circuito auxiliar para conectar un pulsador al microcontrolador.

PULSADORES

La manera más sencilla de enviar información hacia nuestro microcontrolador es a través de los pulsadores. Podemos enviar un **1** o un **0** al cerrar o abrir el interruptor o el pulsador (**Figura 3**).

RELAY

El relay (o relé) es un dispositivo electromecánico que actúa como un interruptor, controlado por una bobina que acciona un juego de uno o varios contactos. Éstos permiten abrir o cerrar circuitos electrónicos independientes a nuestro microcontrolador.

Como los contactos de salida del relay pueden conectarse a circuitos que son independientes del circuito de excitación, podemos controlar circuitos de salida con mayor potencia que el de entrada.

Los contactos de un relay pueden ser normalmente abiertos (**NA**) o normalmente cerrados (**NC**). Los NA se cierran cuando se excita la bobina del relay, en tanto que los NC se abren en ese caso. En la **Figura 4** vemos cómo conectar el relay a nuestro PIC. Para hacerlo, utilizamos un transistor **BC337,** porque puede drenar mayor corriente entre su colector y emisor que las salidas del microcontrolador. También hay que colocar un diodo de protección –llamado de **rueda libre–** entre la bobina del relay, para evitar que los transitorios de corriente destruyan el transistor.

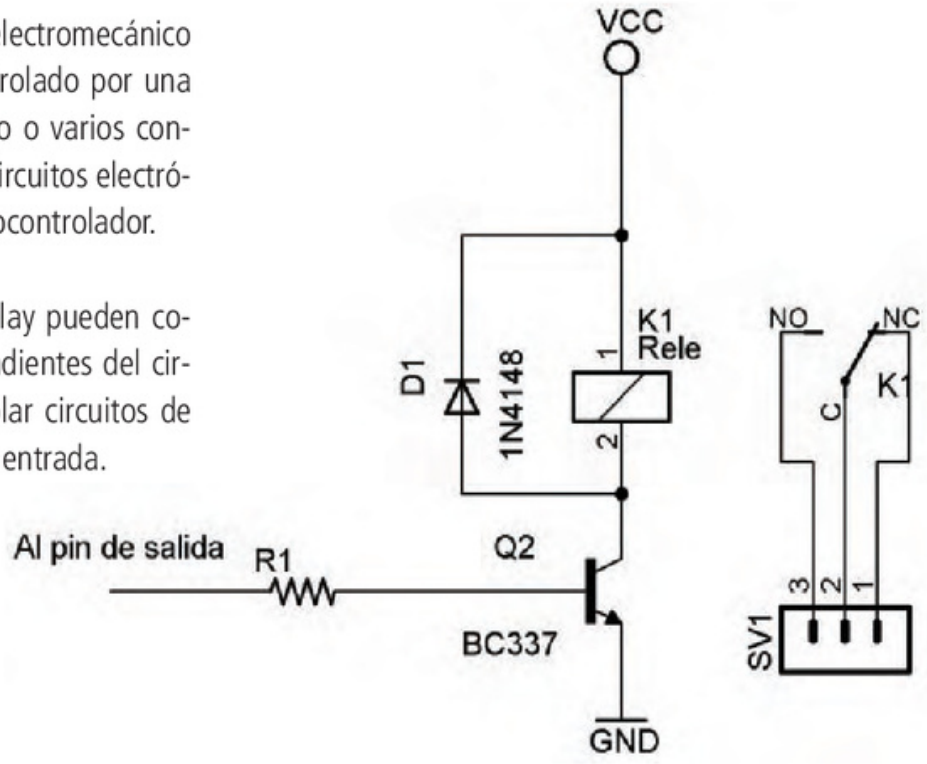
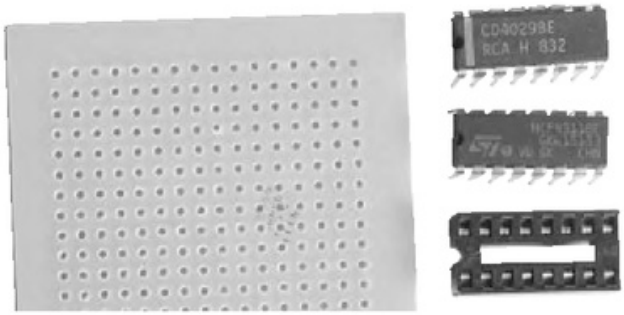


FIGURA 4. Circuito auxiliar para excitar un relay con un microcontrolador.

DISPLAY DE 7 SEGMENTOS

Los displays de 7 segmentos serán de utilidad en circuitos donde haya que mostrar números (**Figura 5**). En el mercado podemos conseguir displays de ánodo común y de cátodo común, según la conexión de los LEDs internos. Dependiendo de la configuración de estos LEDs internos, la barras se encenderán con un **1** (cátodo común) o con un **0** (ánodo común).

Para colocar una hilera de varios displays –con el fin de representar números de más de un dígito–, podemos utilizar la técnica de **multiplexado**. Ésta enciende los displays de manera secuencial a una velocidad tan rápida que el ojo humano no puede percibir su apagado; a su vez, nos permite ahorrar pines del PIC.



Los displays de 7 segmentos serán de utilidad en circuitos donde haya que mostrar números

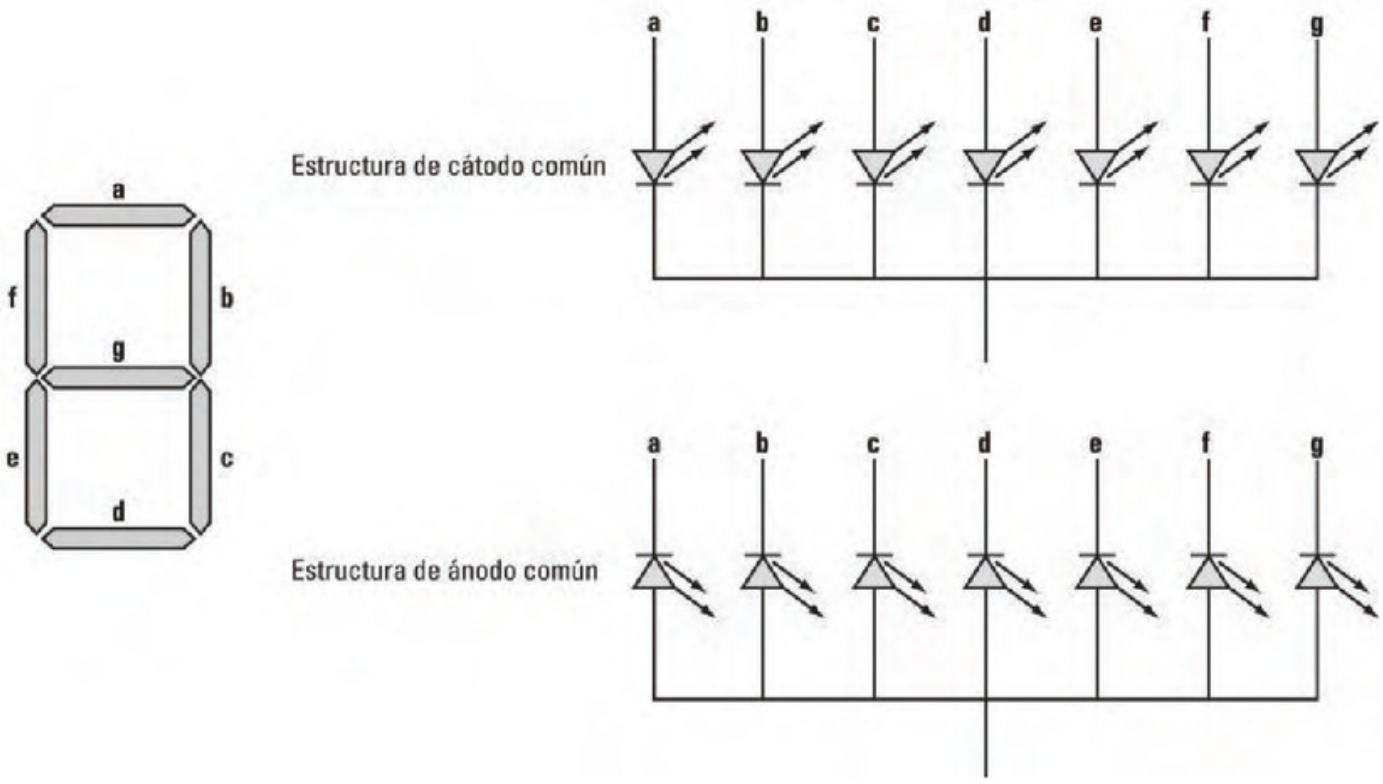


FIGURA 5. **Los displays de 7 segmentos son simplemente barras de LEDs que encendemos de manera selectiva para formar un número. Existen los de cátodo común y los de ánodo común, según la disposición de los LEDs.**

PUERTOS I/O

Los dispositivos expansores de puertos sirven para agregar uno o más puertos de **8 bits** de manera externa. Podemos recurrir a un dispositivo como éste cuando los puertos de nuestro PIC son insuficientes. Uno de los expansores más utilizados es el **integrado PCF8574,** que contiene un puerto de entrada/salida de 8 líneas. Se conecta al PIC a través del protocolo de comunicación I2C, el cual estudiaremos en los próximos capítulos.

PROGRAMA DE UN CONTADOR CON DISPLAY DE 7 SEGMENTOS

Veamos ahora un pequeño programa en **lenguaje C** en el que implementaremos una lectura de pulsador y el incremento de un contador de un dígito. Presentamos a continuación el código en C.

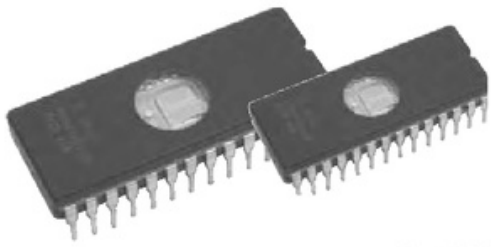
Con los relays podemos controlar circuitos de salida con mayor potencia que los de entrada

```
#include <p18f4620.h>  //este archivo
denominado de //cabecera incluye las
//de todos los SFRs del PIC18F4620
#include <p18f4620.h>
// este archivo incluye librerias de tiempo
#pragma config OSC = XT
//definimos que usaremos un
//oscilador a cristal estandar
#pragma config WDT = OFF
//definimos que no se usará el
//Watchdog Timer
#pragma config MCLRE = ON
//definimos que el pin MCLR se
//usará como RESET y no como I/O
void main (void)
//definimos la función principal
{
unsigned int contador = 0;
ADCON1 = 7;
//desactivando las funciones analógicas
LATB = 0;
//ponemos el LATB en 0
TRISAbits.TRISA1 = 1;
//configuramos RA1 como entrada
TRISB = 0;
// configuramos todo
el PORTB como salida
```

DETALLES DEL PROGRAMA

Su función principal será explorar el estado del pulsador colocado en el puerto RA1. Si éste se activa, entonces incrementaremos el valor de un contador. Este será decodificado a 7 segmentos y enviado al PORTB para presentarlo en un display de cátodo común.

```
while(1)
// creamos un bucle que se repite siempre
{
if (PORTAbits.RA1==1)
// este archivo incluye librerias de tiempo
{
//Si es así entonces...
contador = contador + 1;
//incrementa el contador
if (contador > 9)
// chequeamos si paso de 9
{
contador = 0;
// si paso reseteamos el contador
}
                    }
switch (contador)
//leemos el valor del contador
{
// y lo decodificamos a 7 segmentos
case 0:
PORTB = 0b00111111;
//escribe 0 en PORTB
Break ;
case 1:
PORTB = 0b00000110;
//escribe 1 en PORTB
Break ;
case 2:
PORTB = 0b01011011;
//escribe 2 en PORTB
Break ;
case 3:
PORTB = 0b01001111;
//escribe 3 en PORTB
Break ;
case 4:
PORTB = 0b01100110;
//escribe 4 en PORTB
Break ;
case 5:
PORTB = 0b01101101;
//escribe 5 en PORTB
Break ;
case 6:
PORTB = 0b01111100;
//escribe 6 en PORTB
Break ;
case 7:
PORTB = 0b00000111;
//escribe 7 en PORTB
Break ;
case 8:
PORTB = 0b01111111;
//escribe 8 en PORTB
Break ;
case 9:
PORTB = 0b01100111;
//escribe 9 en PORTB
Break ;
}
Delay1KTCYx(200);
//espera un tiempo
}
}
```

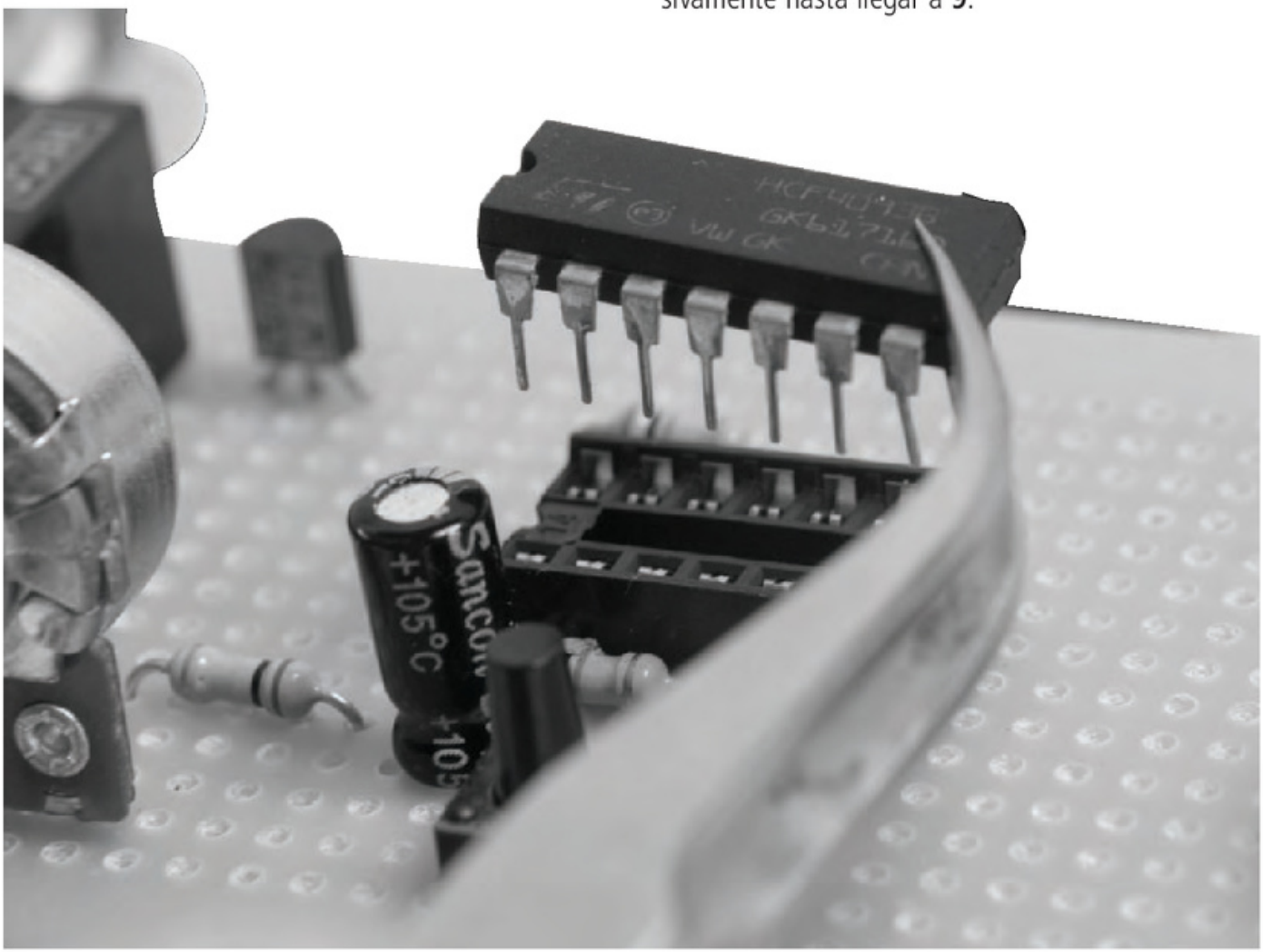


CONTADOR PMW

Comenzamos por inicializar las variables del programa y los puertos del microcontrolador. En el esquemático de la **Figura 6** vemos que el pulsador de entrada se conecta al pin **RA1,** y controlamos el display de 7 segmentos con el **puerto B**. Entonces, la configuración inicial será indicarle al PIC que el pin RA1 es una entrada de datos, y que el PORTB es un puerto de salida.

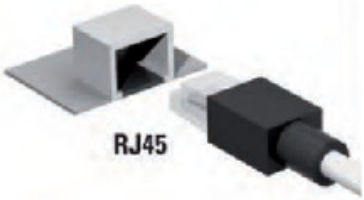
Una vez que configuramos el microcontrolador, entramos en un bucle infinito generado por la sentencia **`While(1)`**; donde constantemente preguntamos cuál es el estado del pin RA1. Si se oprimió el pulsador, el programa incrementará la variable contador, y si éste es mayor a 9, la cuenta se reseteará. Luego, según la cuenta del contador, convertimos ese valor para que sea visualizado en el display de 7 segmentos.

En el diagrama de flujo de la **Figura 7** se presenta, de manera lógica, cuál es el procedimiento de la conversión. Primero, preguntamos si el contador es igual a 0; si lo es, ponemos en el puerto de salida el valor que coloca un 0 en el display. Si no lo es, pasamos a interrogar si el contador vale **1** y, así, sucesivamente hasta llegar a **9**.



INFOGRAFÍA 1: PERIFÉRICOS EN MICROCONTROLADORES

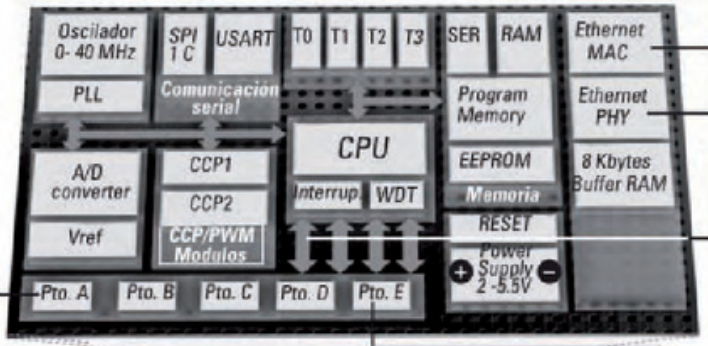
En esta infografía les mostramos los módulos de hardware internos de un PIC16F876A y una idea de conexión de algunos de los periféricos más utilizados en aplicaciones con microcontroladores.



Módulos Ethernet (PIC18F97J60):
Algunos microcontroladores como el PIC18F97J60 de Microchip, ofrecen controladores Ethernet embebidos. Se encuentran completamente implementados en el HW del micro. Los módulos "Media Access Control" (MAC) y "Physical Layer Transmitter" (PHY) se encargan de ello. Para su interfaz directa a una red Ethernet, sólo se requiere la utilización de 2 transformadores de pulso y algunos componentes pasivos que ayudan a la reducción de EMI (Interferencia Electromagnética).

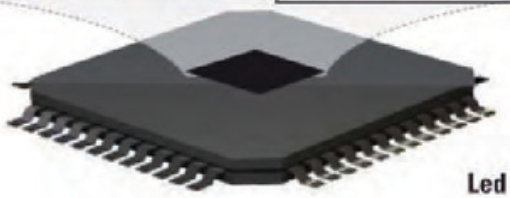
Circuito interfaz adaptador de líneas y reducción EMI

Display Alfanumérico



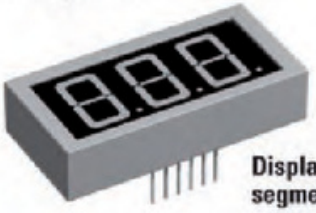
Módulos PWM:
Utilizados para control de motores pequeños de CC ó para medir ciclos de trabajo de señales como la salida de un acelerómetro. Utilizan los recursos de los módulos "Contadores/Timers" que forman parte de cualquier arquitectura micrcontrolador. Características PIC 16F876A de Microchip: Resolución Captura 12,5ns máx. y de 16bits Resolución Comparador 200ns máx y de 16bits Resolución del mod. Pulse Width Modulation es de 10 bits.

Puertos Paralelos esclavo de 8 bits:
Puertos paralelos de propósito general. Pueden conectarse típicamente a los mismos: leds, switches, botones, displays alfanuméricos, displays 7 segmentos y teclados a membrana. Dependiendo del microcontrolador utilizado, son capaces de manejar corrientes desde los 40uA a 25mA por línea.



Led

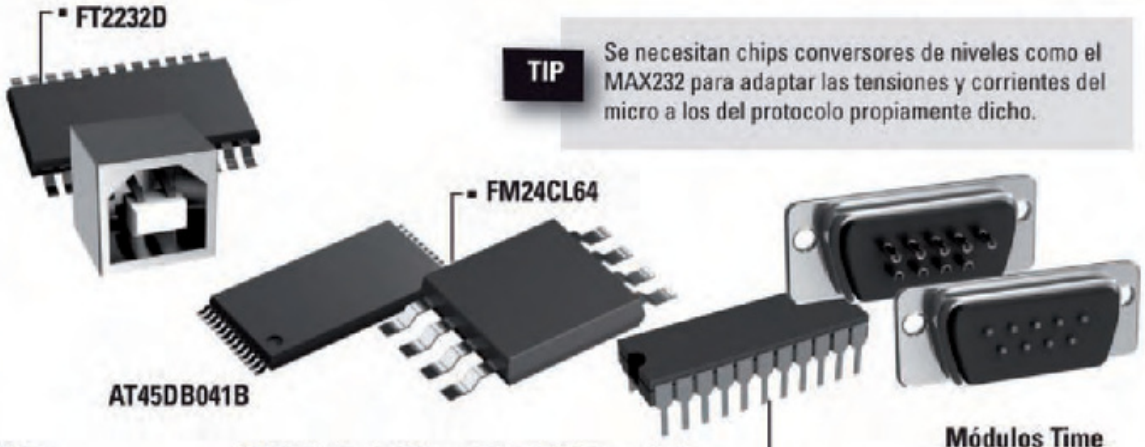
Teclado a membrana



Display 7 segmentos

TIP

El chip FT2232D de FTDI para conectividad USB 2.0 Full-Speed, permite interfaces con nuestro microcontrolador del tipo SPI, I2C ó mismo con un módulo asincrónico como RS232.



TIP

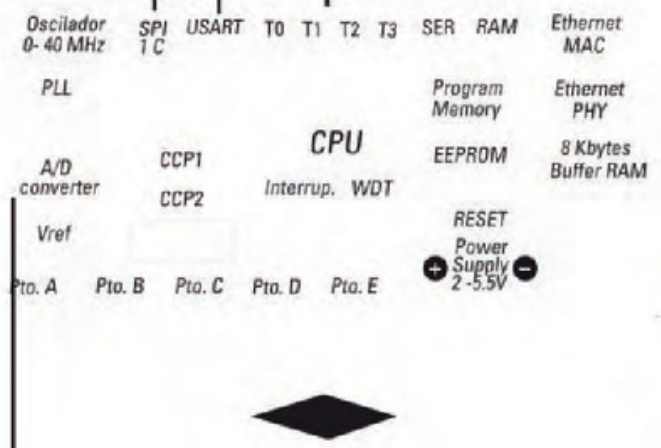
Se necesitan chips conversores de niveles como el MAX232 para adaptar las tensiones y corrientes del micro a los del protocolo propiamente dicho.

Módulos Time /Temporizadores

MAX 232 Driver

Universal Synchronous Asynchronous Receiver/Transmitter (USART/SCI en Microchip):
Para implementación de comunicaciones asincrónicas seriales como RS232. Generador de Baud-rate configurable: ej.: 9600 ó 19200 baudios etc. Manejan 8 ó 9 bits de datos, configuración de bits de parada, flags de errores, etc.

Módulo Serial Sincrónico Multipropósito (MSSP en Microchip): SPI / I2C
Módulo HW para interfaces sincrónicas seriales, encontrada en diversas arquitecturas. Permite configurar interfaces SPI e I2C en modos Master y Slave. Ejemplos: AT45DB041B Flash Eeprom SPI de Atmel y FRAM FM24CL64 I2C de Ramtrom.



Módulos Timer/Temporizadores:
Son utilizados para diversas aplicaciones que necesiten trabajar con marcas de tiempo, implementar retrasos (delays) y analizar eventos en forma temporal generando interrupciones. Pueden alimentarse con clk's externos o internos.
Características PIC16F876A de Microchip:
Timer 0: Contador/Temporizador de 8 bits con preescaler de 8 bits
Timer 1: Contador/Temporizador de 16 bits con preescaler. Puede ser incrementado durante "sleep", mediante un cristal o reloj externo.
Timer 2: Contador/Temporizador de 8 bits. Posee preescaler y postcaler.

Sensor de temperatura

Conversor A/D de 10 bits multicanal
Conversores analógico-digitales. Digitalización de señales de origen analógico como el sensor de temperatura.

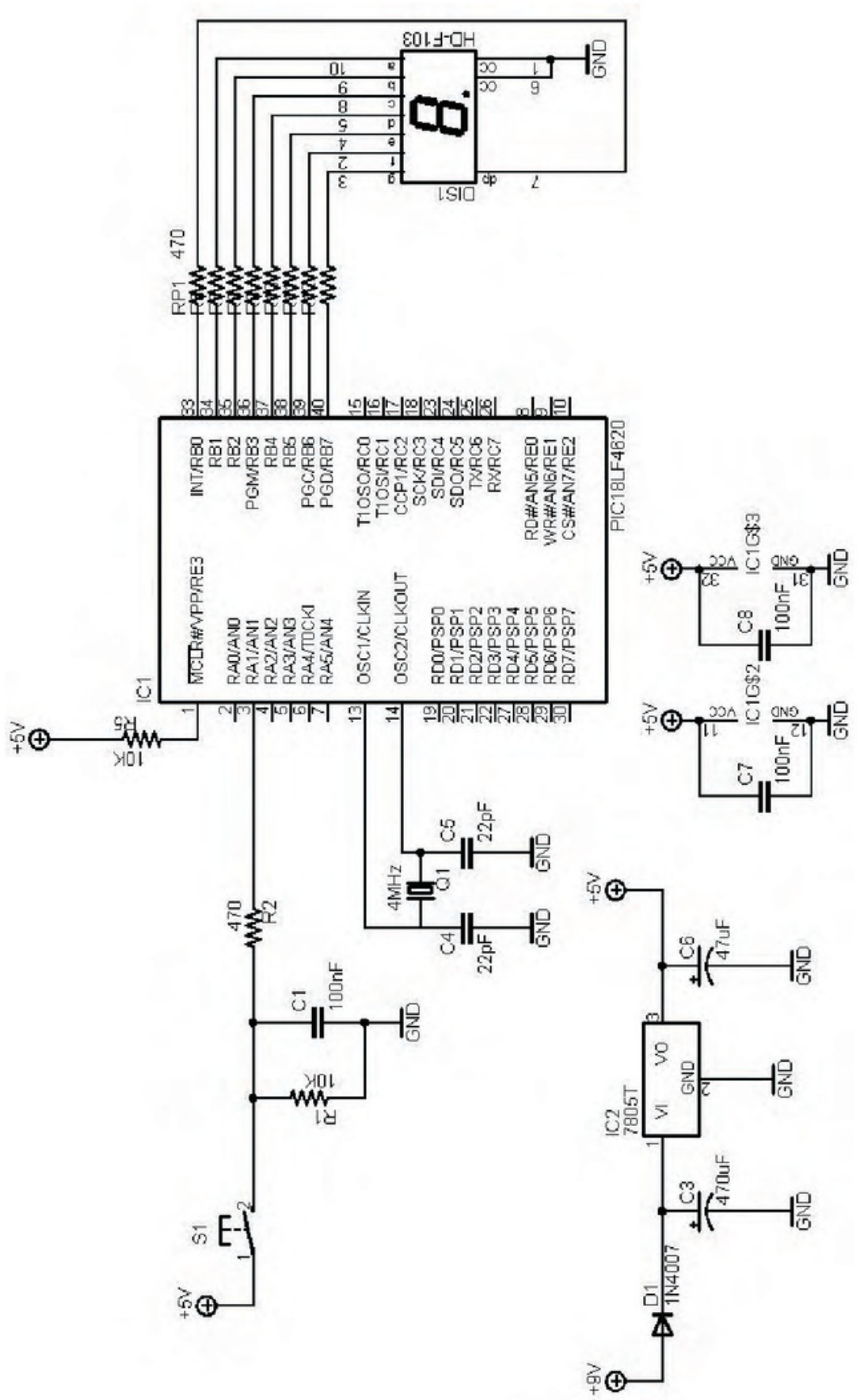


FIGURA 6. Éste es el esquemático de la práctica correspondiente al contador con display de 7 segmentos.

En el diagrama de flujo, sólo colocamos dos valores en la interrogación, para que el dibujo no se haga complicado. Luego, cuando pasamos estas interrogaciones al código, lo hacemos a través de la sentencia **switch-case,** donde, dependiendo del valor de la variable **contador**, escribimos el puerto **B** con el dígito en 7 segmentos.

Observemos que en el encabezado del código, además de incluir el archivo que define los registros de nuestros PICs, utilizamos la directiva

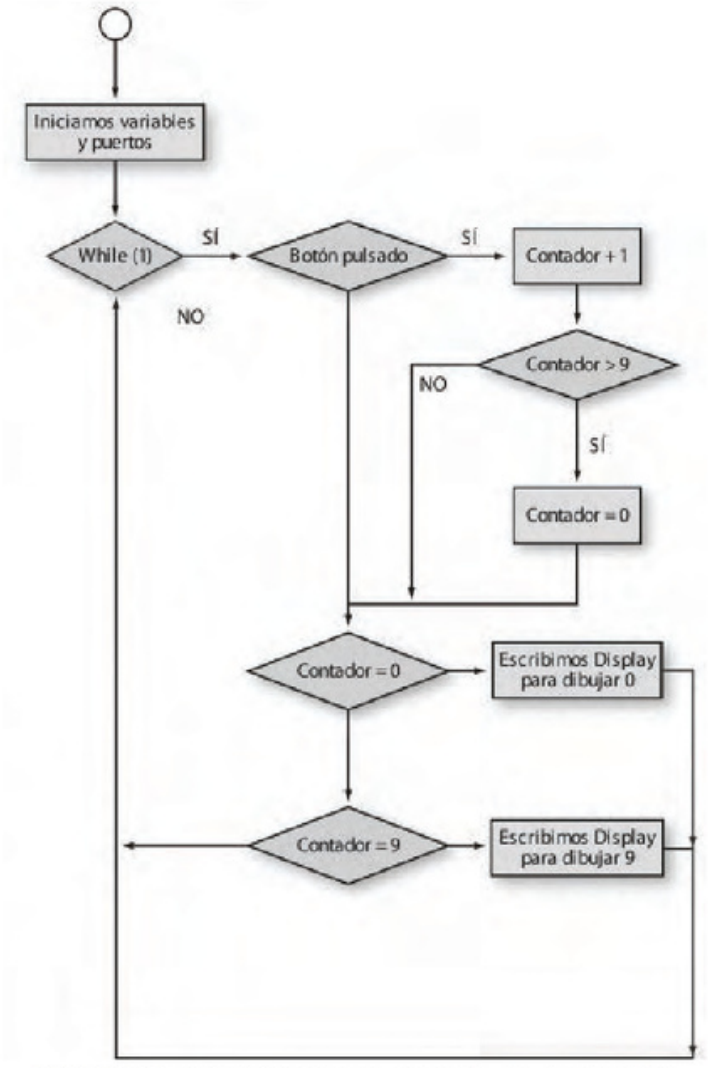


FIGURA 7. En el diagrama de flujo de la función principal podemos seguir la lógica del programa.

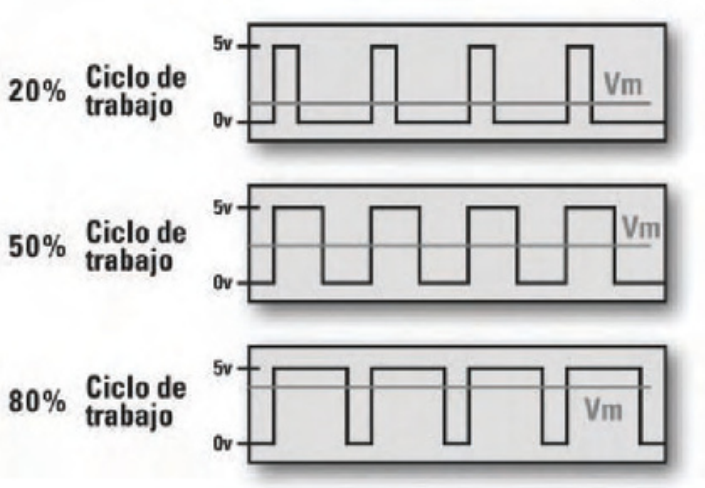


FIGURA 8. Podemos variar el ciclo de trabajo de una señal y entonces veremos cambios en la tensión media.

La directiva #pragma config permite configurar algunos recursos del PIC, como elegir el tipo de cristal

`#pragma config` para configurar algunos recursos del PIC. En esas líneas le indicamos al microcontrolador que emplearemos un cristal externo del tipo **XT** para el PIC, desactivamos el "**perro guardián"** (WDT) y activamos el pin de **`Reset`**.

Práctica PWM

Los microcontroladores PIC son capaces de producir pulsos de anchos variables en el tiempo por medio de un módulo interno llamado **CCP** (*Compare/Cap-*

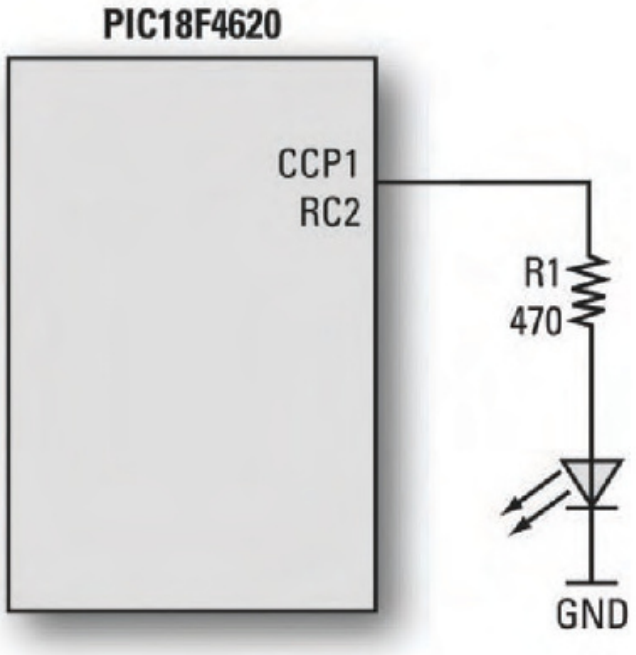


FIGURA 9. **Vemos cómo se conecta el LED cuya intensidad controlaremos. El pin del PIC es el RC2, el cual es la salida del módulo CCP que utilizaremos como PWM.**

ture/PWM, Comparación/Captura/Modulación de ancho de pulso). Las señales **PWM** permiten realizar diversas tareas, como **dimmers** para LEDs (variación de la intensidad del LED) hasta controlar la velocidad de un motor eléctrico de corriente continua.

Todas estas aplicaciones están basadas en un principio básico de señales llamado **PWM**, donde las señales modulan su ancho de pulso alterando su ciclo de trabajo. El ciclo de trabajo, o **Duty Cycle**, es la relación entre el tiempo que la señal se mantiene a nivel alto y el período de la señal; veremos que es común que sea expresado en forma porcentual. A medida que el ciclo de trabajo aumenta, también se incrementa la potencia entregada por la señal. En la **Figura 8** vemos cómo se incrementa el valor medio de la señal cuando aumentamos el ciclo de trabajo; realiza de manera lineal con la siguiente fórmula: **Vm = CT (Ciclo de trabajo) x Vmax**.

EJERCICIO

Realizaremos un **dimmer para LED**, es decir que modificaremos la intensidad del LED con la técnica de **PWM**. El LED se encuentra conectado al pin **RC2** del PIC por medio de una resistencia de **470Ω**, cuyo objetivo es limitar la corriente que pasa por el LED. Entonces, modulando la señal del pin donde se encuentra el LED, se modificará la corriente promedio que lo atraviesa, y cambiará la intensidad. Dos pulsadores conectados en **RA1** y **RA2** actuarán como el control de luminosidad; usaremos uno de ellos para aumentar el brillo y el otro para disminuirlo. Para evitar que el parpadeo que se produce en el LED sea visible al ojo humano, la señal que module la intensidad de éste debe tener una frecuencia mayor a los **100 Hz** (**Figura 9**).

MÓDULO PWM

El PWM es una de las tres funcionalidades que posee el módulo CCP (Comparación/Captura/PWM) de los microcontroladores PIC. Este módulo es controlado por uno de los registros especiales (SFR) del PIC, el registro CCP1CON. Los bits de este registro configuran al módulo para poder utilizarlo como PWM, haciendo que el pin RC2 actúe como la salida de la señal modu-

Las señales PWM permiten realizar tareas como dimmers para LEDs y controlar la velocidad de un motor

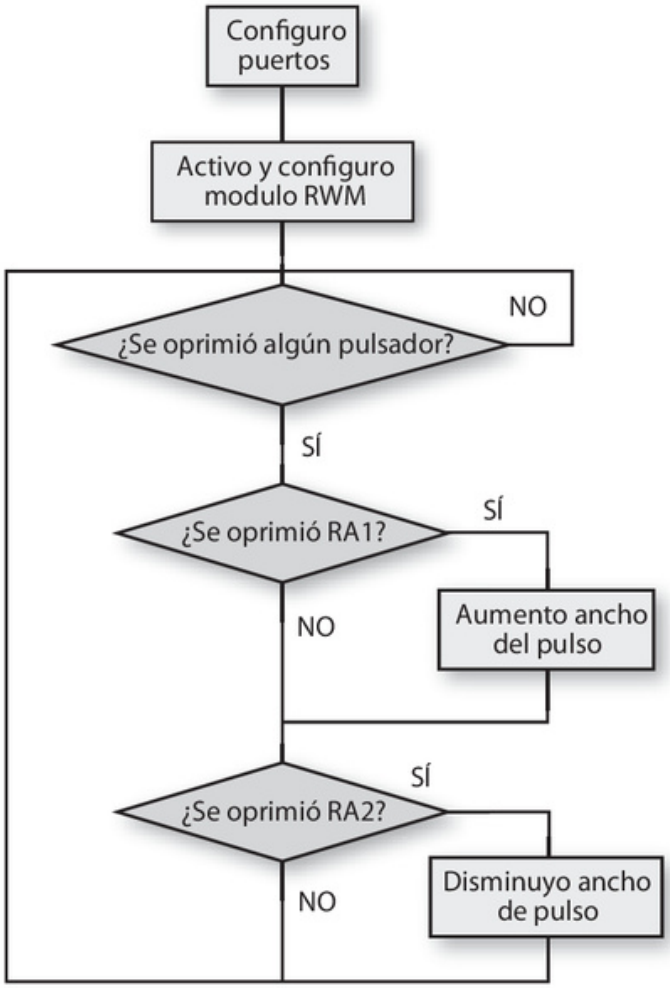


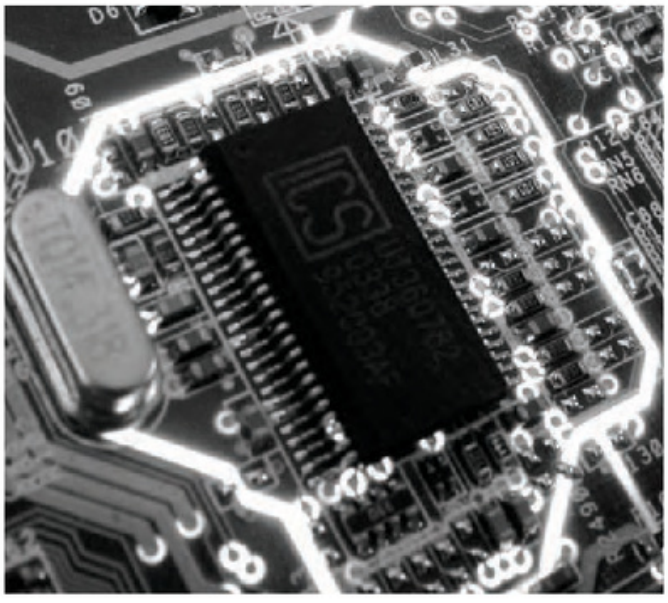
FIGURA 10. Éste es el diagrama de flujo de la práctica.

lada; por esta razón colocamos un LED allí. No vamos a detallar cómo es que este módulo genera una señal de ancho de pulso variable, pero podemos decir que a través de dos registros podemos configurar el período y el ciclo de trabajo de la señal. Cuando inicializamos el módulo y cargamos con un valor en el registro PR2, se realiza una comparación entre el valor del PR2 y un contador interno del PIC llamado Timer2. Los valores son iguales, la señal se pone a 1; esta operación marca el período de la señal. Luego, con el registro CCPR1L se establece el ciclo de trabajo de la señal cuando la comparación con el contador es la misma.

CÓDIGO

En la **Figura 10** tenemos el diagrama de flujo que resuelve el ejercicio, donde se comienza configurando los puertos del PIC. El pin **RC2** se configura como salida (allí encontramos el LED a controlar) y los pines **RA1** y **RA2** serán entradas digitales. El próximo paso es activar y configurar el módulo PWM. No entraremos en detalle sobre este tema y recomendamos leer la hoja de datos del PIC para configurar este módulo. Con el registro **PR2** definimos el período de la señal en **4 mseg** y modificamos el ancho de pulso por medio del registro **CCPR1L**.

Cuando terminamos de configurar los recursos del PIC, el programa se queda interrogando si se produce algún cambio en los pulsadores. Si se detecta ese cambio, se pasa a interrogar cuál es el pulsador que se oprimió: si fue **RA1**, se aumenta el ancho de pulso de la señal modificando el registro **CCPR1L** con la variable brillo. En cambio, si fue **RA2** el que se oprimió, se hace disminuir el ancho de pulso y, luego, se vuelve a realizar todo de nuevo (**Figura 11**).



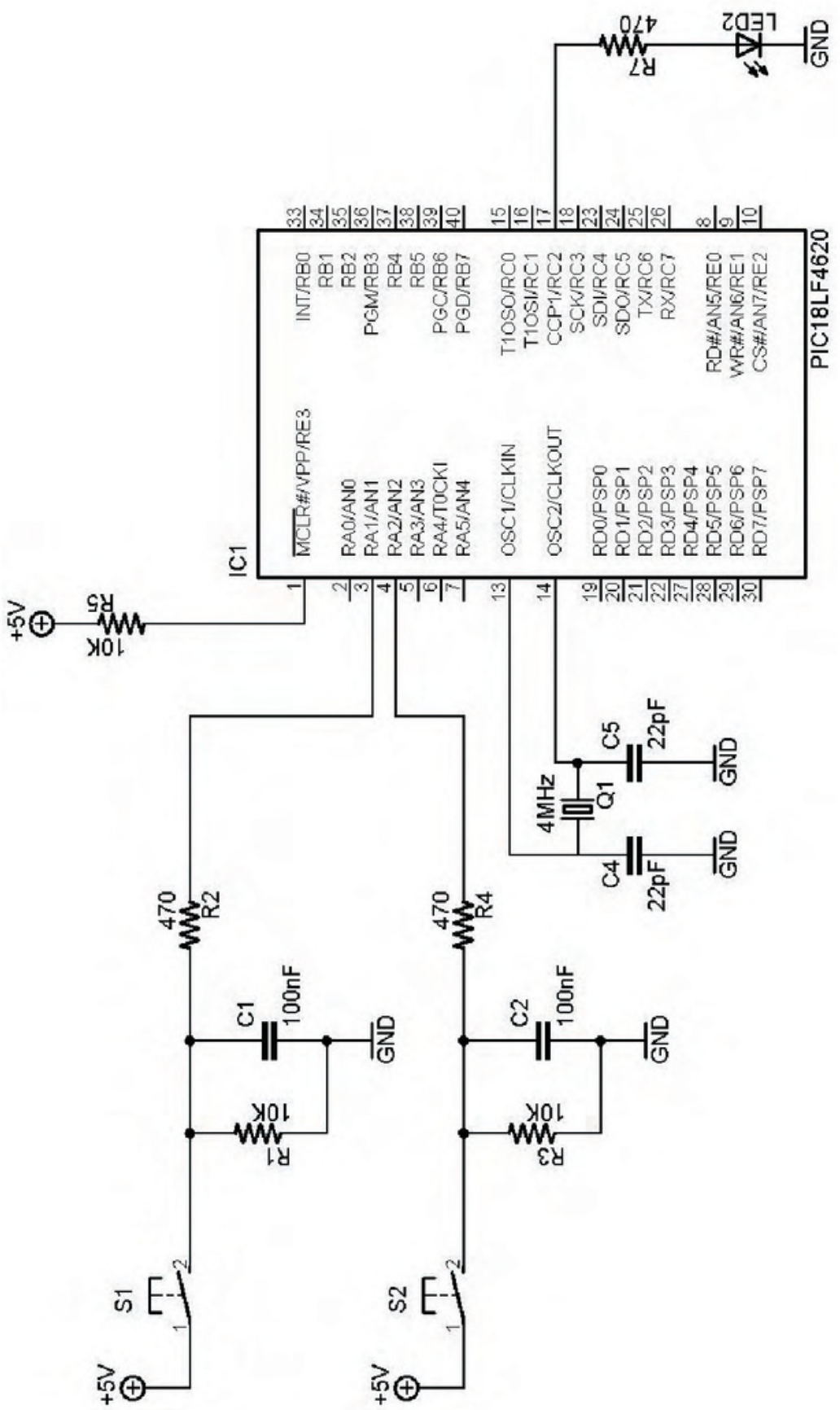


FIGURA 11. Éste es el esquemático de la práctica, una sección de la placa experimental que construimos donde encontramos los pulsadores y el LED.

A partir del diagrama de flujo obtenemos el siguiente **código en C** para el enunciado de esta práctica. En los comentarios se explica cada línea de la configuración del PIC y el proceso de monitoreo de los pulsadores para aumentar o disminuir la intensidad del LED. Observemos que la variable **brillo** controla el ancho de pulso de la señal y, de ese modo, la intensidad en el LED. La configuración del temporizador del PIC se verá en detalle en el próximo capítulo, pero el módulo **PWM** lo necesita para su operación.

```
/* Includes */
#include "p18f4620.h"
#pragma code     // Zona de memoria de programa
void main (void)
{
unsigned char brillo = 0;
TRISCbits.TRISC2 = 0;          //RC2 salida
ADCON1=0x0F;
//Configuramos todos los pines como digitales
   TRISA = 0b00000110;
// PORTA bit 1 y 2 entrada(Aqui estan
// conectados los pulsadores)
//Configuracion Timer 2
// La configuracion de los temporizadores
se vera en detalle en la proxima
// clase
T2CON = 0b00000111;// Prescale = 1:16, timer on
// Periodo de la señal
PR2 = 249;
// Con esta configuracion generamos un
periodo de 4mS, este periodo no
// se modificara en todo el programa
//Duty Cycle o ciclo de trabajo
CCPR1L=brillo;
// La variable brillo se encargara de
modificar el ancho de pulso
// de la señal, observar que tenemos un
ciclo de trabajo que tiene
// una resolucion de 8 bits.
// Configuramos el módulo CCP del PIC para
que trabaje en modo PWM
// modificamos los bits del Registro CCP1CON
CCP1CON = 0b00001100;
while(1)
{
// Nos quedamos esperando a que se
presionen los pulsadores
while(PORTAbits.RA1 != 1 &&
PORTAbits.RA2 != 1);
if(PORTAbits.RA1 == 1)
// Se presiono RA1?
{
brillo = brillo + 2;// Aumentamos
brillo en multiplo de 2
CCPR1L = brillo;
// modificamos en ancho de pulso
}
if(PORTAbits.RA2 == 1)
// Se presionó RA2?
{
brillo = brillo - 2;
// Decrementamos brillo en multiplo de 2
CCPR1L = brillo;
// modificamos en ancho de pulso
        }
    }
}
```

Una vez que tenemos el código fuente, generamos un nuevo proyecto en el **MPLAB** y simulamos el código con el **MPLAB SIM** para verificar que el programa no tiene errores y resuelve el enunciado del ejercicio. Luego, podemos grabarlo en el PIC y correr el programa en nuestra placa experimental (**Figura 12**).

MEDICIÓN DEL ANCHO DE PULSO

En nuestra placa experimental podemos apreciar el funcionamiento de la señal **PWM** por medio del LED de prueba, donde se ve cómo varía la intensidad cada vez que oprimimos los pulsadores. También podemos visualizar la forma de onda de la señal que excita al LED en un osciloscopio, el cual nos permite calcular la frecuencia y el ciclo de trabajo de la señal.

Si no disponemos de un osciloscopio, por su elevado costo, tenemos herramientas alternativas para poder visualizar la señal en la computadora. Una de ellas es el programador de **PICs MCE PDX USB**. Este programador puede ser utilizado, en su función como analizador lógico, para visualizar los cambios de estado de una señal. También nos será de utilidad para visualizar la señal PWM que genera nuestro PIC en la práctica que acabamos de hacer.

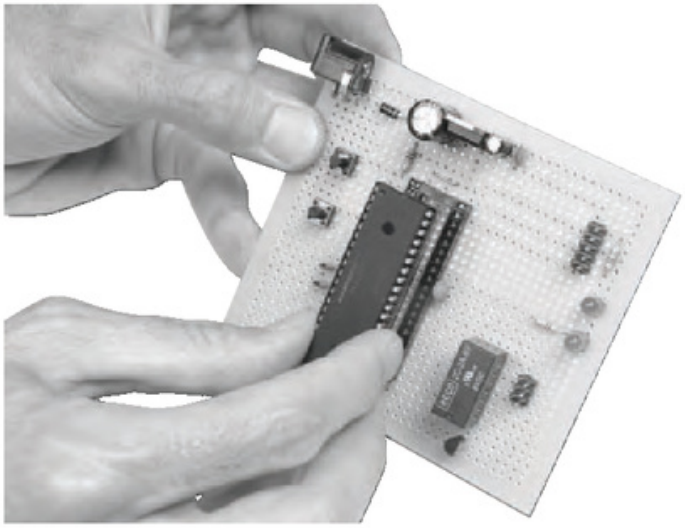


FIGURA 12. **Una vez que compilamos el proyecto, podemos probarlo en nuestra placa experimental.**

Si no disponemos de un osciloscopio, tenemos herramientas alternativas para poder visualizar una señal en la computadora

El **MCE PDX USB** se comporta como un analizador lógico de tres canales, con una frecuencia máxima de muestreo de **1 MHz.** Es decir, es capaz de tomar **1 millón** de muestras por segundo en cada canal. Las conexiones de los canales del analizador están multiplexadas con el conector de programación **ICSP** que tiene esta herramienta; desde allí podemos conectar cables a nuestra placa experimental y realizar las mediciones.

Para utilizar el programador como analizador lógico debemos ejecutar el software **PicKit 2 programmer**, que viene incluido en el programador **PDX**, y activar la herramienta lógica desde **Tool/Logic Tool.** Debemos seleccionar el modo **Analizador** para poder visualizar la forma de onda de la señal que genera el PIC. Allí, tendremos una pantalla como se muestra en la **Figura 13**. Desde ella podemos ver la disposición de los canales **CH1**, **CH2** y **CH3** en el conector **ICSP** (**Figura 14**).

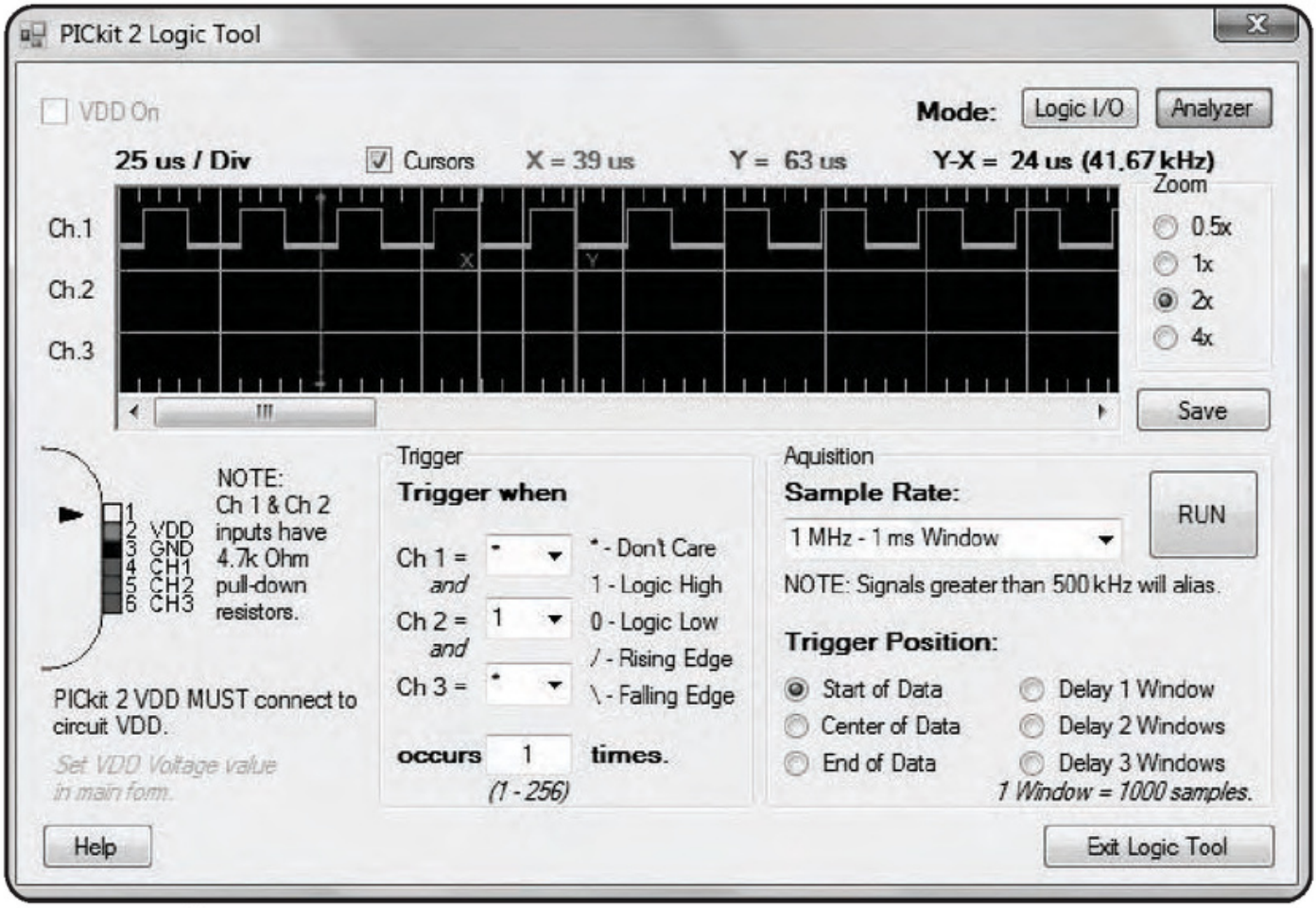


FIGURA 13. Ésta es la pantalla típica del analizador lógico del programador, donde podemos configurar la medición. También tenemos la posibilidad de utilizar cursores para medir el ancho de pulso de nuestra señal PWM.

Conversores Analógico-Digitales

Los conversores A/D son dispositivos que transforman una variación analógica a formato digital. Son fundamentales en los lazos de control cerrados digitales; podríamos decir son los "ojos" de los microprocesadores y de los microcontroladores. En todo sistema digital de control existe siempre un **conversor A/D** que se encarga de "traducir" la señal entregada por un sensor de temperatura, presión o fuerza a un valor digital equivalente que pueda procesar el sistema de control digital, ya sea que éste se encuentre implementado con microprocesadores o con microcontroladores.

RESOLUCIÓN DE LOS CONVERSORES A/D

La resolución de un conversor A/D es un parámetro muy importante, ya que nos indica la precisión que tendrá el dispositivo al realizar la conversión. La se-

FIGURA 14. **En esta imagen del programador MCE PDX USB podemos ver el conector de programación ICSP (la tira de postes) que pueden actuar como los canales de entrada del Analizador Lógico.**

ñal analógica que ingresa al conversor será dividida en una serie de pequeñas fracciones. Cuanto mayor sea la resolución del conversor, más pequeñas serán las fracciones y, por lo tanto, más aproximada será la conversión al valor real.

Dicha resolución queda determinada por el número de bits que puede procesar el conversor; por ejemplo, en un conversor de 8 bits, la señal de entrada se divide en **256 fracciones.** Sin embargo, en un conversor de **16 bits** la señal de entrada se dividirá en **65536 fracciones**, y la precisión para el mismo nivel de señal será mayor.

A la fracción se la denomina **escalón de conversión** o **rate de cambio.** Cuanto más pequeña sea, más precisa será la conversión analógico-digital. Los conversores más comunes son de **8 bits** de resolución, pero también existen de **10 bits**, **12 bits**, **16 bits**, **20 bits** y **24 bits**. Es importante destacar que cuanta mayor precisión tiene el conversor, más lento es y, por lo tanto, si la señal de entrada varía muy rápidamente, se producirán errores. (**Figura 15**).

CONVERSOR DIGITAL-ANALÓGICO

Existen varios métodos para obtener la conversión digital-analógica. Para conseguir pasar un valor **binario** a un valor de tensión **analógico** equivalente, se recurre a una red de resistencias. Ésta genera una tensión en función de los niveles binarios que se encuentren, en un momento dado, en cada uno de los bits que intervienen en la conversión.

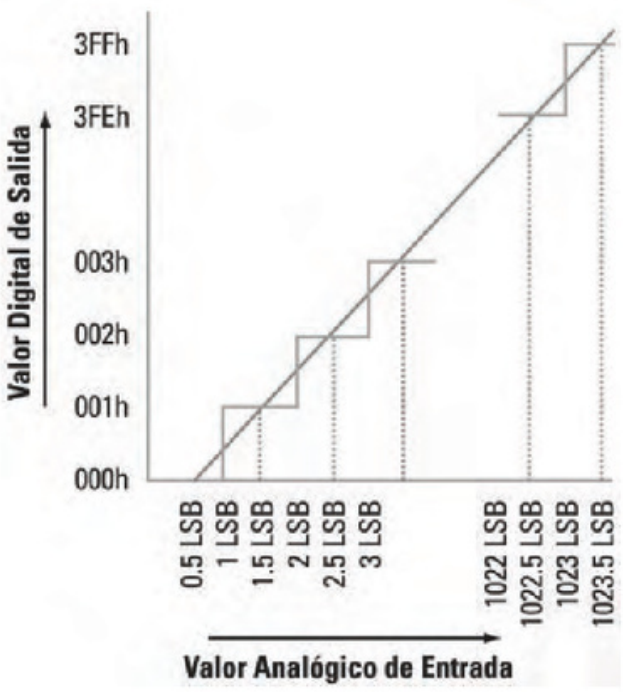


FIGURA 15. **En la figura observamos la función de transferencia de un conversor de 10 bits.**

La mencionada **red resistiva** es, por lo tanto, la clave de la conversión. Existen dos redes perfectamente diferenciadas que cumplen este cometido: **la red de resistores ponderados** (**Figura 16**) y la **red R-2R** (**Figura 17**). Para explicar este concepto, supongamos que disponemos de un registro de **8 bits**, el cual puede ser cargado con cualquier valor desde **00** hasta **FF**.

La resolución de un conversor A/D nos indica la precisión que tendrá el dispositivo al realizar la conversión

Por su parte, una **red ponderada de resistores** consistirá en conectar un resistor en cada salida del registro (de allí su nombre: "ponderada"), de manera que los valores de los resistores a conectar desde el bit de mayor peso al de menor peso son R, 2R, 4R, 8R, 16R, 32R, 64R y 128R.

FIGURA 16. **En la figura podemos ver un conversor D/A por resistores ponderadores. La dificultad que presenta este método radica en conseguir comercialmente los valores de los resistores.**

LOS INICIOS DE LOS A/D

Los conversores A/D que se usaron en las primeras PCs para las placas de sonido eran de 8 bits. Su resolución limitada generaba sonidos sintéticos. Luego fueron superados por los conversores de 16 bits, que generaron las tarjetas de sonido como SOUND BLASTER.

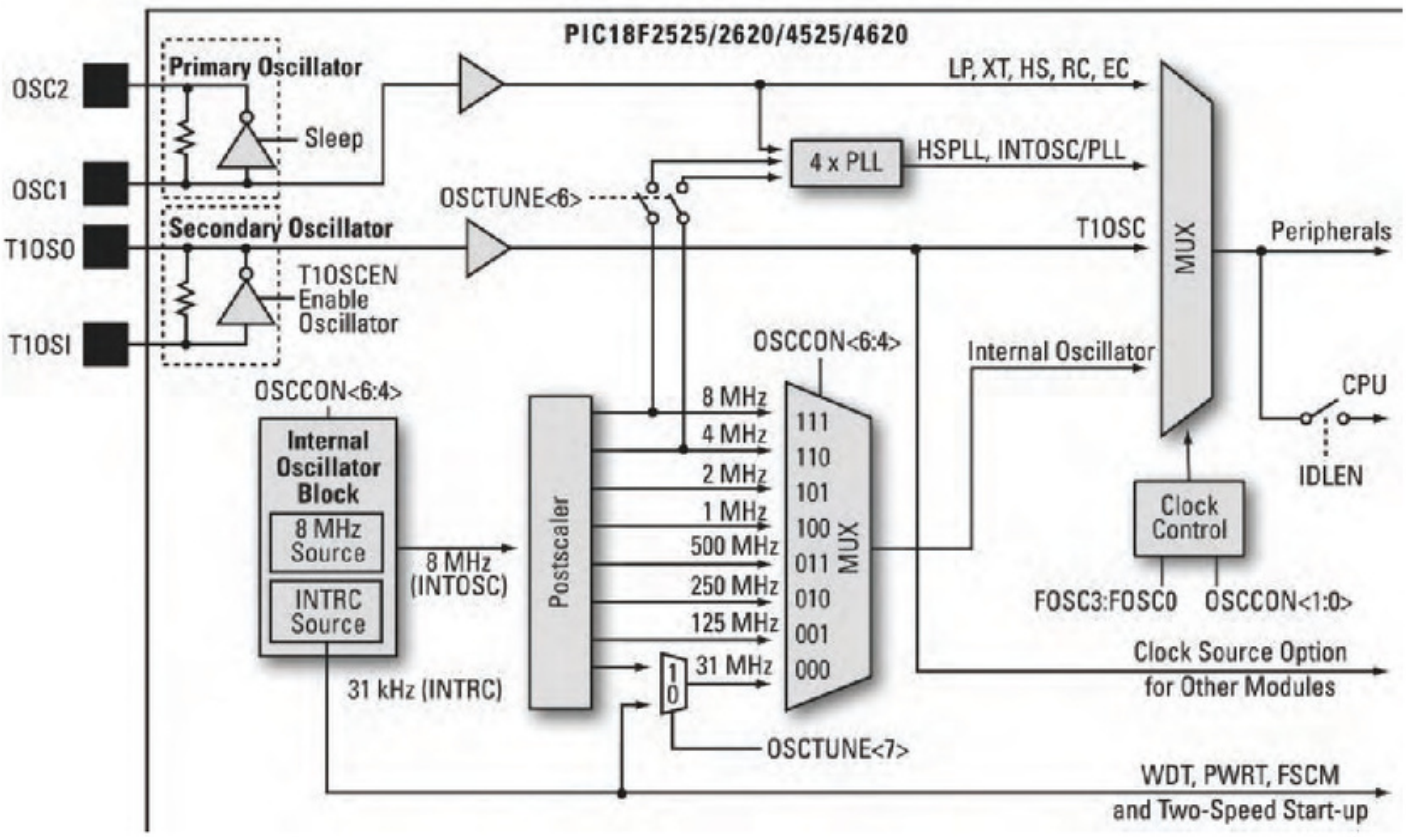


FIGURA 17. **En la figura podemos ver un conversor D/A del tipo** R-2R, **el cual utiliza sólo dos tipos distintos de resistores; sin embargo, su construcción necesita de más resistores que el método ponderado.**

Si una vez conectados estos resistores, se aplican valores binarios de **00** a **FF** consecutivamente, se generará una rampa ascendente de **255 escalones**, donde cada escalón corresponde a un valor binario.

Para conseguir pasar un valor binario a un valor de tensión analógico equivalente, se recurre a una red de resistencias

La conversión D/A también puede realizarse mediante otro procedimiento denominado **red de resistores R-2R**. En este sistema, a cada bit que sale del registro se le conecta un resistor de valor **2R** y, en el extremo del mismo, se coloca un resistor de **valor R**. Por ejemplo, si **R** vale **10 K**, **2R** vale **20 K**. Como se puede apreciar, en esta red el valor óhmico de los resistores R determina el valor de 2R, que es el doble de R (2R = 2 x R). Este sistema tiene la ventaja, respecto al sistema de resistores ponderados, de que emplea sólo dos valores de resistores distintos. La salida de señal analógica se toma después del resistor en serie del bit de mayor peso.

CONVERSOR ANALÓGICO-DIGITAL (ADC)

La conversión analógica-digital consiste en convertir valores analógicos en formato digital. Es el proceso inverso a la conversión **D/A**.

En la conversión **A/D** a cada valor analógico aplicado en la entrada del conversor le corresponde un valor digital de salida, dentro de los valores que puede generar el conversor. La precisión del conversor estará dada por la cantidad de bits que formarán al resultado de la conversión. Para obtener la conversión A/D existen varios métodos; sin embargo, trataremos aquí los más representativos que son:

- Conversor Estático o Flash
- Conversor Dinámico o de Rampa
- Conversor de Doble Rampa
- Conversor SAR o por Aproximación Sucesiva

CONVERSOR ESTÁTICO O FLASH

Este tipo de conversor es el más rápido de todos. Está formado por una cadena de comparadores que toman su voltaje de referencia desde un array de resistores. Como los comparadores se encuentran todos en paralelo, al sistema se lo conoce como **conversor paralelo**. La salida de los comparadores se aplica a un codificador de prioridad digital, el cual genera un número binario, según la entrada que se haya activado. De esta forma, cada comparador dispara una entrada, la que, a su vez, genera un código binario en la salida del codificador.

La conversión analógica-digital consiste en convertir valores analógicos en formato digital

El método de conversión es muy sencillo. La tensión de entrada se aplica a la entrada de comparación de todos los comparadores. Esto genera que los mismos activen su salida cuando la tensión de comparación supera a la de referencia. Pero del sistema, sólo sale un código binario, pues el codificador de prioridad sólo genera una salida equivalente a la entrada de mayor peso que esté activa, y el resto es ignorado.

Si bien este método es rápido y eficiente, sufre del inconveniente del tamaño y del costo del conversor, ya que se necesitan 255 comparadores para construir un conversor de 8 bits (**Figura 18**).

CONVERSORES A/D Y D/A

Los conversores A/D y D/A son esenciales en todo sistema de control digital, ya que nos permiten conectar un sistema digital al mundo real. Los microcontroladores incorporan generalmente el módulo A/D que puede ser de 8, 10 o 12 bits.

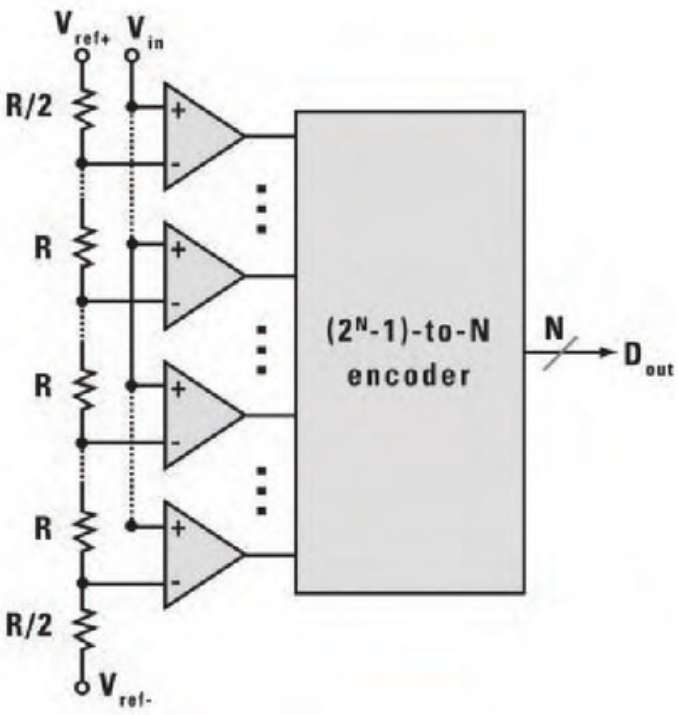


FIGURA 18. **En la figura podemos ver un conversor estático de 4 bits mediante el cual ilustramos su constitución interna.**

La principal desventaja del conversor de rampa simple es su inestabilidad en la generación de la rampa

CONVERSOR DE RAMPA SIMPLE O DINÁMICO

Este tipo de conversor es más económico y eficiente que el anterior. Se lo suele denominar conversor A/D de rampa simple o dinámico. Se construye a partir de un contador digital y un comparador.

Cuando el conversor arranca, tanto el contador como el comparador inician en 0. Como la salida del comparador tiene un nivel lógico 0, una compuerta **AND** inhibe el paso de la señal de **clock** hacia el contador. Al aplicarse una señal en la entrada del comparador, como la entrada de referencia de éste vale 0, la salida del comparador pasa a 1. Esto habilita la compuerta **AND**, la cual deja pasar los pulsos de clock al contador.

De esta forma, el comparador comienza a incrementar su estado de cuenta. La salida del contador se envía hacia un registro tipo **LATCH** y, al mismo tiempo, a un conversor **D/A**, mediante el cual se genera la tensión de referencia. Es así que, la referencia comienza a incrementarse y, cuando ésta llega al valor de la tensión de entrada, la salida del comparador pasa a 0. Esto desactiva la **AND** y ésta inhibe el paso de la señal de clock. Así se detiene el contador y se genera la señal para que el registro tipo **LATCH** capture el estado de cuenta y lo presente en su salida como el valor binario de la conversión de la señal de entrada.

VELOCIDAD DE CONVERSIÓN

La principal ventaja que tienen los conversores estáticos o tipo Flash es su velocidad de conversión. Como la señal de entrada se aplica simultáneamente, el único tiempo de retardo que existe es de propagación del comparador y el del codificador binario.

El tiempo de conversión depende del nivel de la señal de entrada; cuanto más grande sea, mayor será el tiempo de conversión. La principal desventaja del conversor de rampa simple es su inestabilidad en la generación de la rampa. Por otra parte, como no existe una sincronización entre la señal de clock y la generación de la rampa, cualquier corrimiento afectará el resultado de la conversión. Este problema se compensa en el sistema de conversión de doble rampa, el cual es más lento pero mucho más estable (**Figura 19**).

CONVERSOR DE DOBLE RAMPA

Este tipo de conversor subsana las deficiencias del de **Rampa simple**, pero es más lento que su antecesor. Este sistema elimina el efecto del corrimiento del voltaje de rampa a lo largo del tiempo y también utiliza un integrador de entrada.

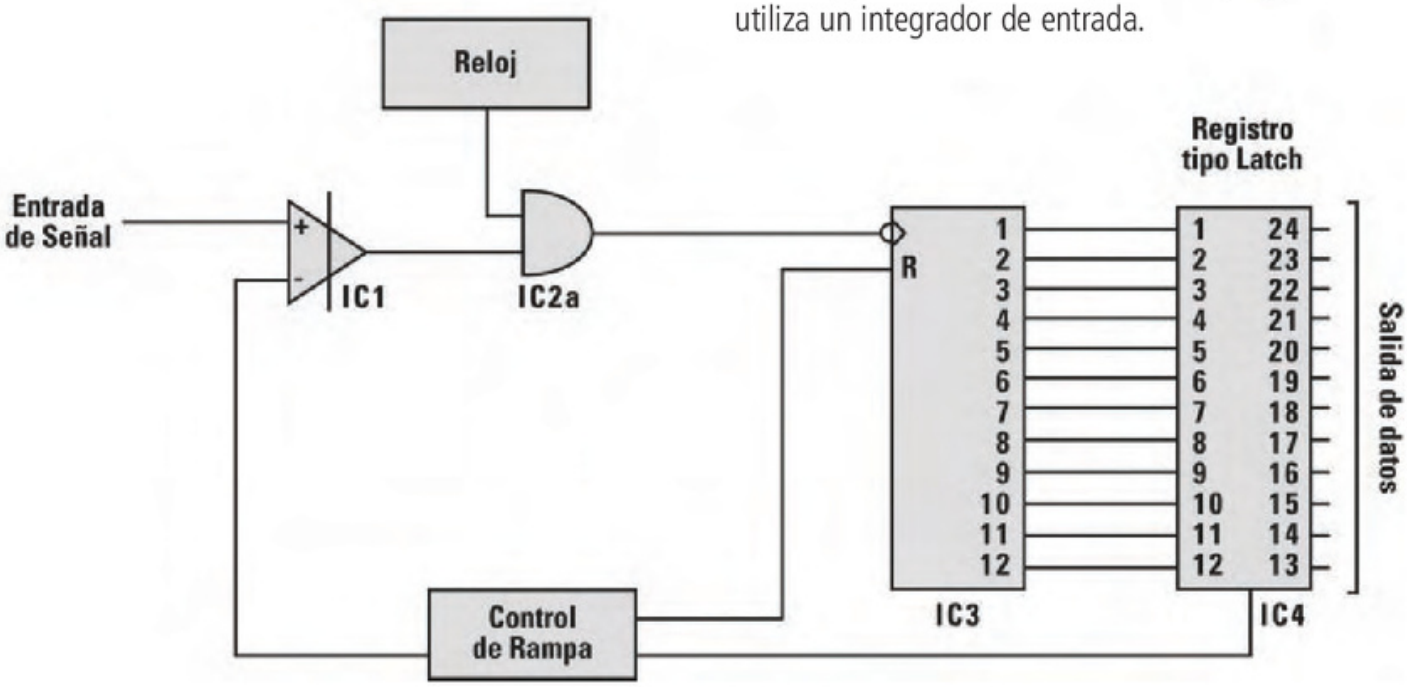


FIGURA 19. **En la figura vemos un conversor dinámico o de rampa simple, este sistema es más económico que el convertidor estático. La salida de la conversión se lee desde la salida del registro tipo** latch.

APLICAR LOS CONVERSORES

Los procesadores de efecto para guitarras eléctricas utilizan el conversor de rama simple o dinámica, con ligeras modificaciones, para generar la conversión, con la variante que la salida del comparador se aplica directamente a la entrada de un Procesador de Señales Digital (DSP).

Este circuito está formado por un amplificador operacional y un capacitor en el lazo de realimentación. Cuando aplicamos una tensión positiva a la entrada del integrador, la salida crece, pero en sentido negativo. Dicha tensión provoca que la salida del comparador pase a 1, lo que activa la **AND,** que permite el paso de los pulsos de clock que hacen avanzar al contador. La rampa negativa generada por el integrador tiene un tiempo fijo, determinado por el **RC** del integrador. Después de este tiempo, el circuito de control pone a 0 el contador y también pone la entrada del integrador a una tensión de referencia negativa. En estas condiciones, el integrador generará ahora una rampa positiva. El contador iniciará su cuenta hasta que la salida del integrador llegue a 0, lo que provocará que el comparador entregue en su salida 0.

El circuito de control detecta el flanco negativo generado por la salida del comparador y memoriza en el **LATCH** de salida el valor del contador. Este número binario es el valor digitalizado de la señal de entrada. Cuando se aplica la referencia negativa en la entrada del integrador, el tiempo requerido por el integrador para retornar a 0 depende de la magnitud de la señal de entrada. Cualquier variación en el circuito integrador generador de la rampa se cancela automáticamente en este retorno a 0.

El conversor de doble rampa subsana las deficiencias del de rampa simple, aunque es más lento que este

El gran problema que presenta este tipo de conversor es su extrema lentitud: generalmente se necesitan unos **100 ms** para efectuar un ciclo completo de conversión. Este tipo de conversor se encuentra en el voltímetro **TC7106/7107** diseñado por Microchip y otras compañías (**Figura 20**).

CONVERSOR POR APROXIMACIONES SUCESIVAS (SAR)

Estos tipos de conversores son muy usados en sistemas de bajo costo y se encuentran incorporados en la mayoría de los microcontroladores que tienen en su interior un conversor A/D. Por ejemplo, el **PIC16F887** incorpora un conversor **SAR de 10 bits**.

Se caracterizan por presentar una resolución media y ser muy rápidos, pero no tanto como los Flash. El núcleo principal de este tipo de conversores es un dispositivo denominado registro de aproximaciones sucesi-

VOLTÍMETRO DE 3 1/2

A mediados de los 80´ la empresa Intersilpresentó su voltímetro de $3^{1/2}$ dígitos basado en un conversor doble rampa. Este circuito integrado denominado ICL7106/7 fue fabricado, luego, por otros fabricantes como Microchip (TC7106/7).

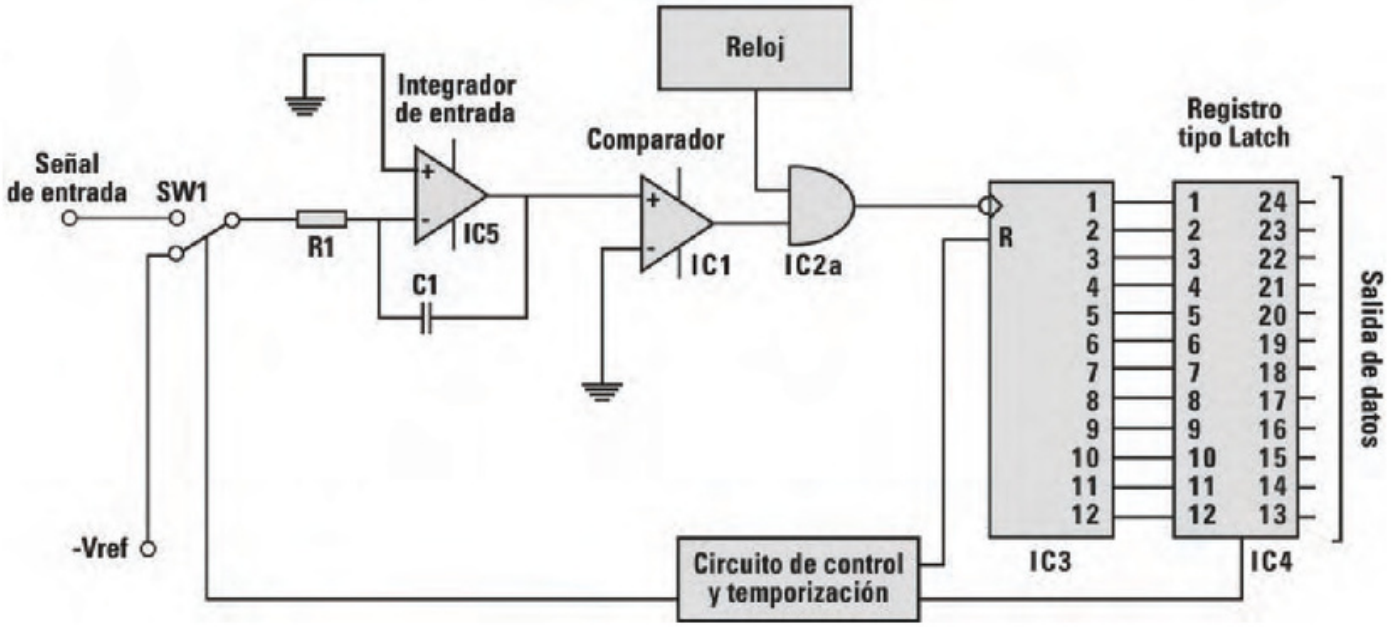


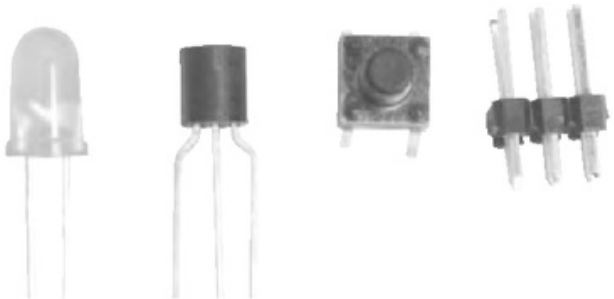
FIGURA 20. En el diagrama vemos el pin out del voltímetro TC7106/7 basado en el conversor de doble rampa.

vas o SAR, desde donde el conversor hereda su nombre. Este tipo de registro realiza un trabajo similar al que realizan los conversores tipo rampa. El circuito de un conversor SAR está compuesto por un SAR, un conversor DAC, un registro de salida y un comparador.

El proceso de conversión se inicia cuando se aplica una señal analógica a la entrada del conversor y se aplica un pulso de **START** o arranque al registro **SAR**. Dicho pulso hace que el SAR genere un **1** en el bit de mayor peso de su salida (MSB=1). Esta salida binaria hace que el **DAC** genere en su salida una tensión correspondiente a la mitad del valor máximo que puede generar (aproximadamente 2,5 V pues se alimentan con 5 V).

Entonces, el SAR mira la salida del comparador, para determinar si la salida del DAC es mayor o menor que la señal de entrada, pues ésta es usada como tensión de referencia para el comparador.

Si el voltaje del **DAC** es mayor, la salida del comparador se pone en 0, provocando que el **SAR** apague el bit **MSB.** En caso contrario, si el comparador entrega un 1, el SAR mantiene el estado del bit **MSB**. En el siguiente pulso de clock, el SAR pone en 1 el bit anterior al MSB y repite el chequeo de la salida del comparador. El SAR repite así una y otra vez la operatoria hasta que se genera el último bit, entonces, envía una señal de final de conversión (EOC). Este tipo de conversores tienen tiempos de conversión menores a 12 microsegundos (**Figura 21**).



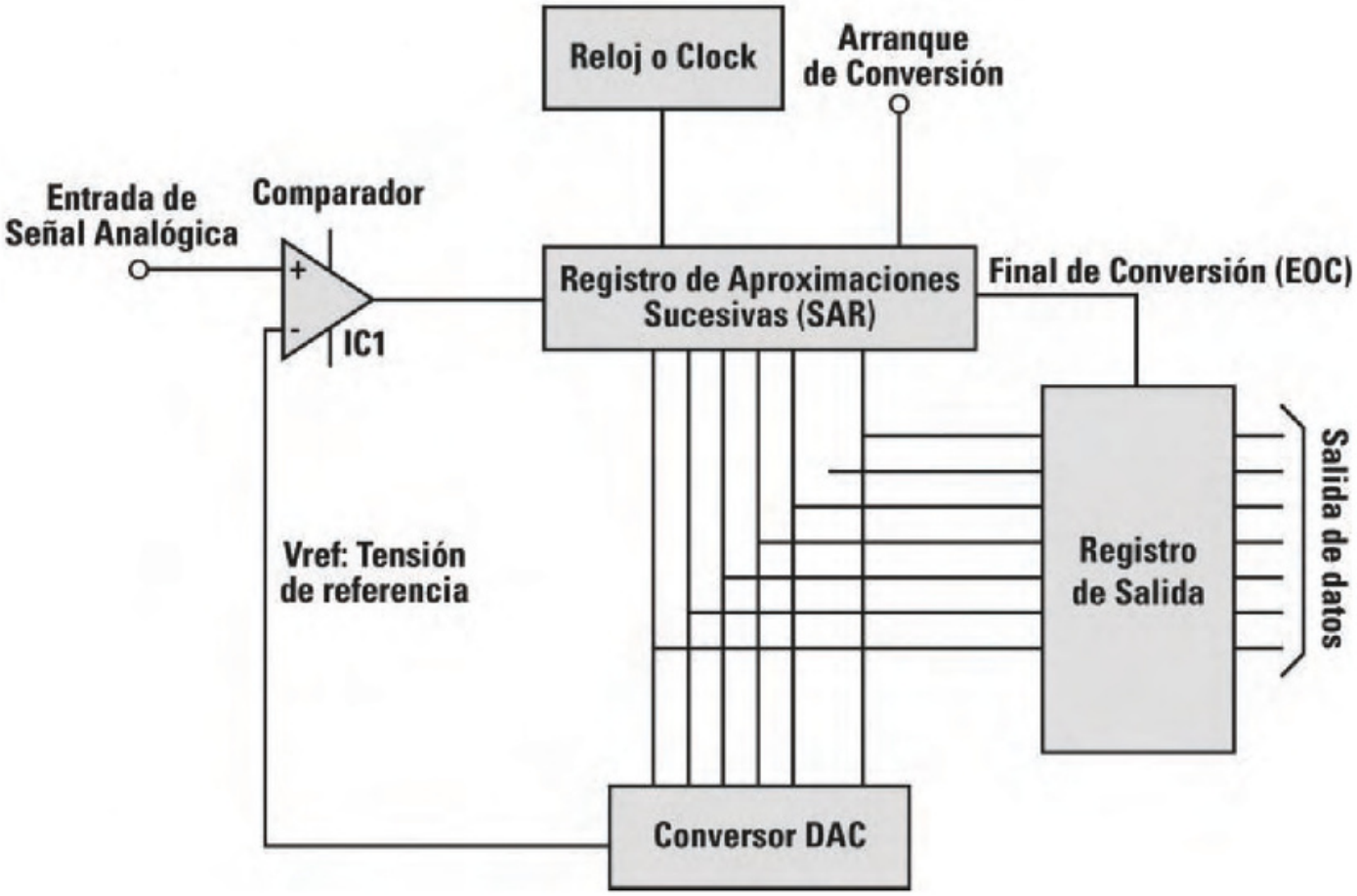


FIGURA 21. **En la figura vemos el diagrama interno esquematizado de un conversor tipo** SAR.

CONVERSOR COMERCIAL ADC0808

El ADC0808 es un conversor de **8 bits** del tipo SAR, el cual pertenece a una familia de conversores formada por el **ADC0808** y el **ADC0809**. Éstos fueron diseñados por National Semiconductors para trabajar con su familia de microprocesadores de 8 bits NSC800 (un clon del Z80).

El ADC0808 incorpora un multiplexor de 8 bits mediante el cual se crean 8 canales de conversión, lo que permite realizar hasta 8 mediciones diferentes, multiplexando las mismas.

El conversor se caracteriza por presentar dos terminales externos para alojar los voltajes de referencia denominados **+Vref** y **–Vref**. El canal de conversión es seleccionable mediante una palabra binaria de **3 bits** que se aplican a los terminales **AD0**, **AD1** y **AD2.** Mediante el terminal **ALE** se captura en un **LATCH** interno el valor del canal analógico seleccionado.

Para iniciar el proceso de conversión se debe aplicar un pulso en el terminal **START** y un oscilador de unos **500 KHz** en el terminal **CLK**. Las señales a medir se colocarán en los terminales **Vin1** a **Vin8**. Cuando el proceso de conversión finalice, el

conversor generará un pulso por el terminal **EOC** (final de conversión). Para poder leer el valor binario de la señal de entrada, aplicaremos un 1 en el terminal **OE** (habilitación de salida) y podremos leerlo desde las salidas **DB7** a **DB0**. La alimentación del mismo debe estar estabilizada en 5 V.

Si bien este conversor ADC ha sido fabricado para trabajar con los microprocesadores de National serie NSC800, también puede ser usado en otras familias, como por ejemplo en el **Z80**, el **8085** o el **6800**. En estos casos suele conectarse la salida

EOC a la entrada de interrupciones del microprocesador. De esta forma, el microprocesador puede disparar el inicio de conversión (**señal START**) e ir a atender otra tarea. Cuando la conversión finalice, se generará la interrupción y entonces el microprocesador leerá el resultado (**Figura 22**).

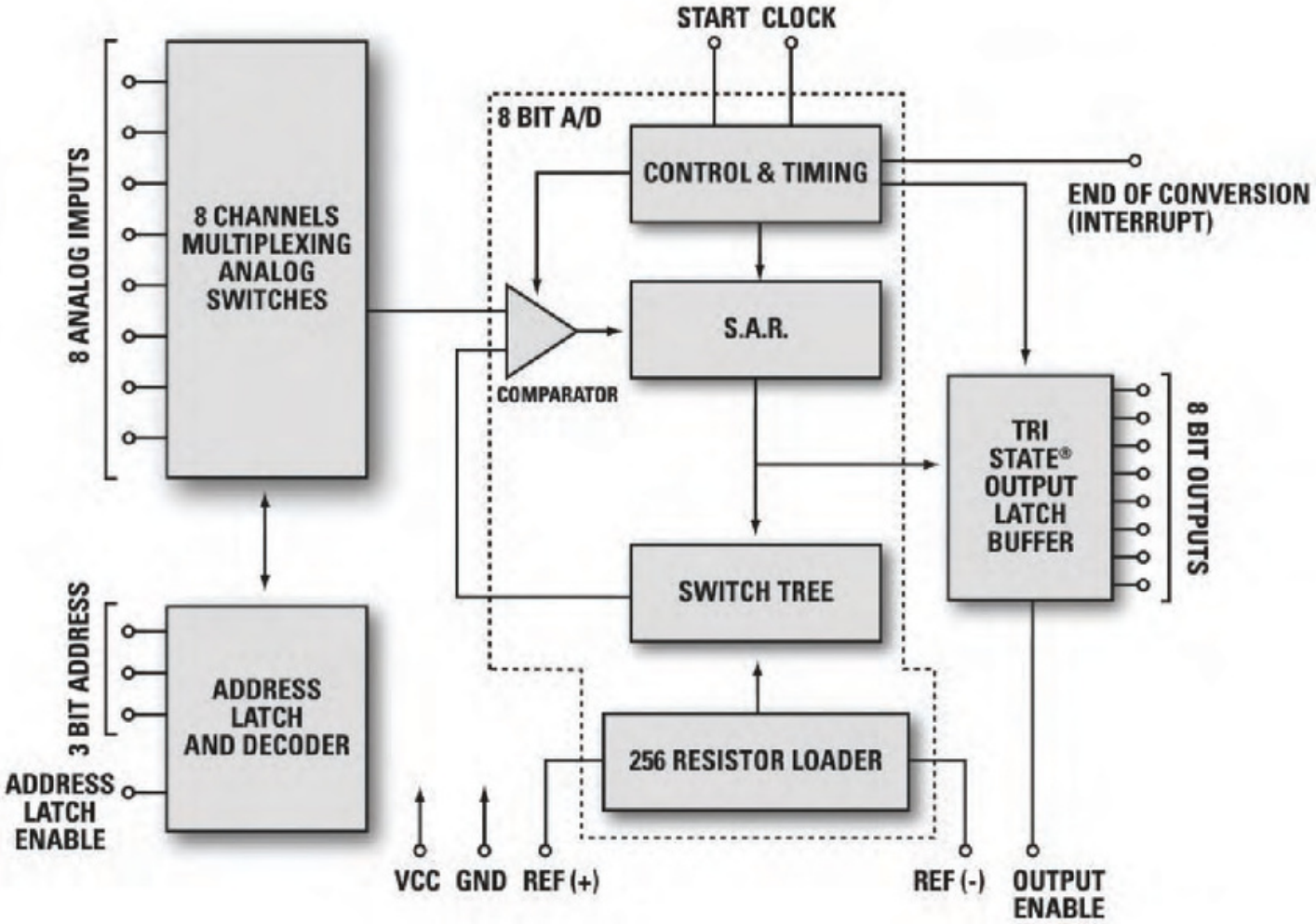


FIGURA 22. **En el diagrama vemos el pin out del voltímetro TC7106/7 basado en el conversor de doble rampa.**

Multiple **choice**

1 ¿Qué es el relay?

a- Un dispositivo expansor de puertos.
b- Un indicador luminoso.
c- Un dispositivo electromecánico que actúa como un interruptor.
d- Un dispositivo que envía información hacia el microcontrolador.

2 ¿Qué es el puerto I/O?

a- Un dispositivo expansor de puertos.
b- Un indicador luminoso.
c- Un dispositivo electromecánico que actúa como un interruptor.
d- Un dispositivo que envía información hacia el microcontrolador.

3 ¿Qué es un LED?

a- Un dispositivo expansor de puertos.
b- Un indicador luminoso.
c- Un dispositivo electromecánico que actúa como un interruptor.
d- Un dispositivo que envía información hacia el microcontrolador.

4 ¿Qué es un pulsador?

a- Un dispositivo expansor de puertos.
b- Un indicador luminoso.
c- Un dispositivo electromecánico que actúa como un interruptor.
d- Un dispositivo que envía información hacia el microcontrolador.

5 ¿Qué tipo de conversor se caracteriza por presentar una resolución media y ser muy rápidos, pero no tanto como el Flash?

a- Conversor comercial ADC0808.
b- Conversor por aproximaciones sucesivas.
c- Conversor Digital-Analógico.
d- Conversor de Rampa Simple o Dinámico.

6 ¿Cuál de los siguientes es un conversor de 8 bits del tipo SAR?

a- Conversor comercial ADC0808.
b- Conversor por aproximaciones sucesivas.
c- Conversor Digital-Analógico.
d- Conversor de Rampa Simple o Dinámico.

Respuestas: 1 d, 2 a, 3 b, 4 c, 5 b, 6 a.

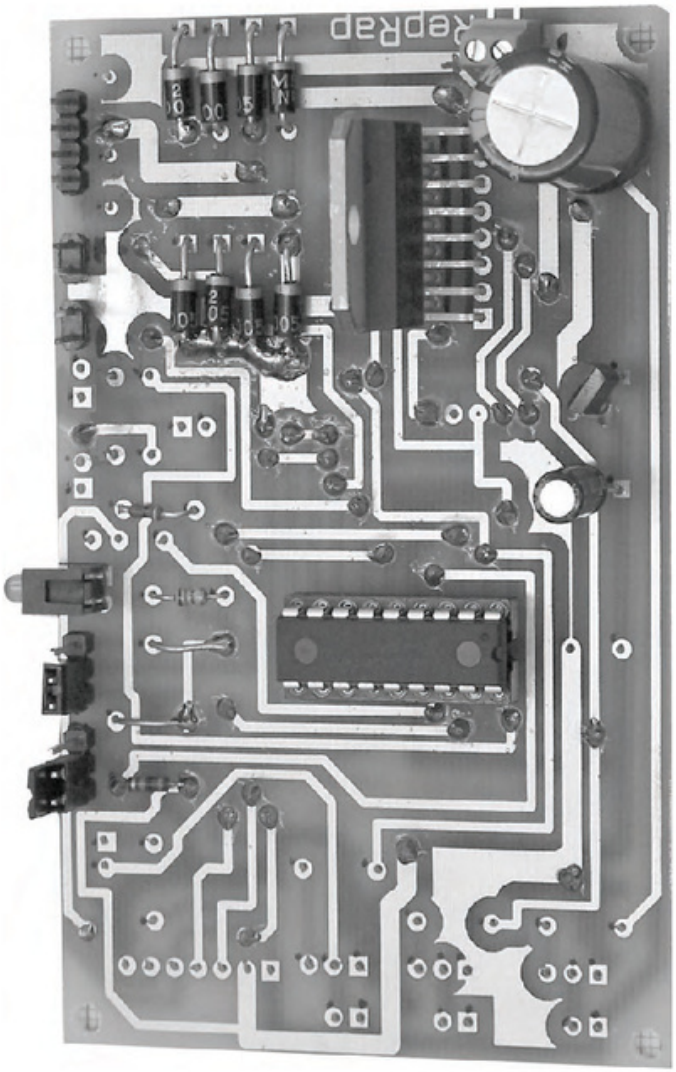
Capítulo 2

Periféricos PIC

En este capítulo, conoceremos cuáles son los periféricos internos que integran los microcontroladores.

Periféricos PIC

Los microcontroladores poseen algunos periféricos internos que son de gran utilidad cuando necesitamos realizar una tarea o alguna otra operación. En este capítulo veremos cuáles son los que integran los PICs y estudiaremos los temporizadores internos, también llamados timers, sumamente prácticos cuando debemos contar pulsos externos o efectuar operaciones temporizadas. Además estudiaremos en detalle el conversor **A/D integrado** y sus registros de control asociados.



Luego, pasaremos a ver qué son las pantallas LCD alfanuméricas, cómo funcionan, cuáles son sus características y cómo conectarlas al PIC. Por último, un proyecto que involucra todo lo aprendido: una alarma térmica.

Periféricos del PIC18LF4620

Microchip dotó a su microcontrolador **PIC18LF4620** de todos los periféricos necesarios para efectuar cualquier sistema de control de forma embebida (**Figura 1**). De esta manera, encontramos dentro de él todos los que hallaríamos en un **PIC16F877**, pero mejorados. El PIC18LF4620, además de los **puertos I/O**, cuenta con los siguientes elementos:

- **Timer 0**, que puede trabajar en modo de 8 o 16 bits.
- **Timer 1**, de 16 bits.
- **Timer 2**, de 8 bits.
- **Timer 3**, de 16 bits.
- **Conversor A/D** tipo **SAR** de 10 bits y 13 canales.
- **Puerto serie sincrónico maestro** (**MSSP**) para comunicaciones **I2C** y **SPI**.
- **Unidad de comunicaciones serie asincrónicas mejorada** (**EUSART**) con autodetección de velocidad de comunicaciones (*Auto Baud Rate*) y soporte para RED LIN 2.0.
- **Módulo captura/comparación/PWM mejorado** (**ECCP**), que incorpora control PWM para puente completo, con control de estrangulación de PWM automática y control de tiempos muertos (*Dead Time Control*).

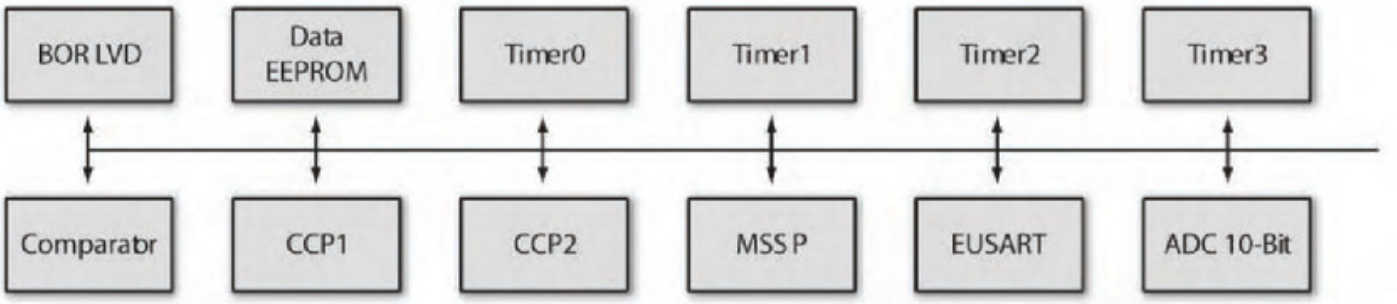


FIGURA 1. **Éstos son los periféricos que forman parte de la arquitectura interna del PIC18F4620.**

• **Puerto paralelo esclavo** (**PSP**), para comunicaciones con microprocesadores de 8 bits.
• **Memoria EEPROM** para datos con capacidad para almacenar hasta 1024 bytes.

Estos periféricos nos permiten realizar cualquier sistema de control de forma embebida, es decir, usando parte de los componentes internos y sin tener que recurrir a ninguno externo. Por ejemplo, podríamos realizar un control de velocidad para un motor de corriente continua. En este caso, usaríamos el módulo **ECCP** funcionando como **PWM** en modo **full bridge** (puente completo). Por medio del conversor A/D, en un canal colocaríamos un potenciómetro para fijar la velocidad, en otro un preset para setear el torque y, finalmente, en otros dos canales podríamos tener un monitoreo de la corriente del motor y la fuerza contra electromotriz que éste genera para conocer la velocidad del motor.

El PIC18LF4620 tiene los periféricos necesarios para realizar cualquier sistema de control

LOS TIMERS

Los temporizadores o timers son contadores binarios que pueden contar pulsos internos o externos. En el primer caso se los denomina temporizadores, mientras que en el segundo se los llama contadores de eventos. Estos timers pueden tener 8 o 16 bits. En el PIC18LF4620 se han integrado cuatro timers, conocidos como 0, 1, 2 y 3.

EL TIMER 0

Este temporizador puede trabajar a 8 o 16 bits, y está preparado para contar pulsos externos o internos. Cuando trabaja excitado por pulsos externos, cuenta aquellos que llegan al teminal T0CKI. A este timer se le puede agregar un módulo divisor de frecuencia programable (prescaler) para que el avance sea de forma más lenta. El resultado de la cuenta se almacena en los registros **TMR0L** y **TMR0H**. Cuando el timer opera en modo de 8 bits, sólo se usa el registro TMR0L. Sin embargo, cuando opera en 16 bits, se usan los dos registros (**Figura 2**).

TIMERS 1 Y 3

Ambos son de 16 bits y, estructuralmente, son iguales; sólo existe una leve diferencia entre ellos: el timer 3 tiene la posibilidad de un modo de disparo

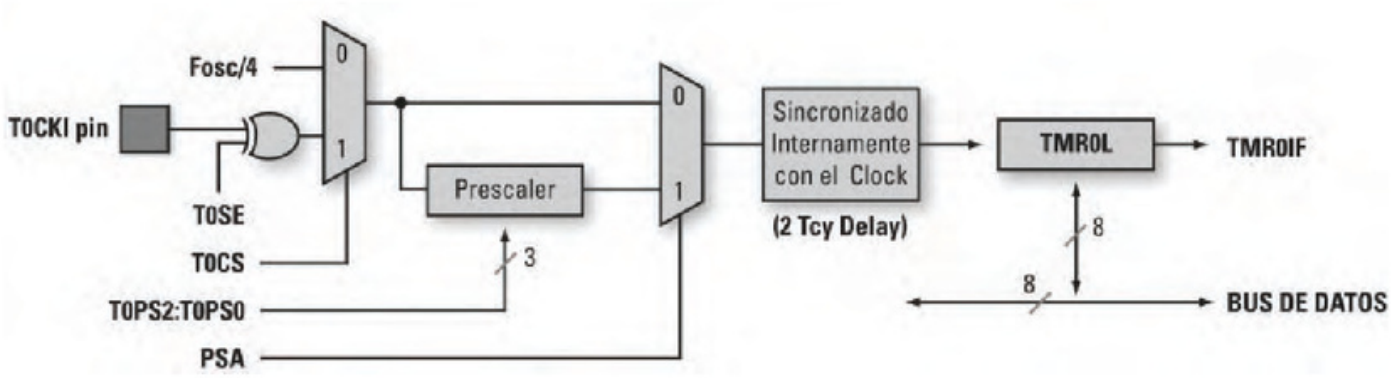


FIGURA 2. **Observamos el diagrama en bloque funcional del** timer 0.

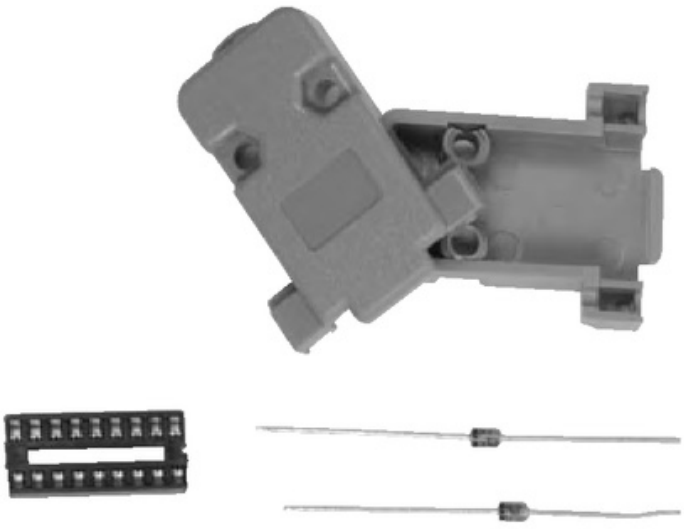
especial por eventos. En los dos casos, el contador está formado por dos registros de 8 bits, que pueden ser disparados por eventos externos o internos. Incluso, cuentan con la posibilidad de activar un driver inversor para configurar un oscilador de baja frecuencia realimentado externamente por medio de un cristal cuya frecuencia típica es de 32.768 KHz. Mediante este cristal, es posible crear una base de tiempos para armar un reloj de tiempo real dentro del mismo PIC (**Figura 3**).

TIMER 2

El timer 2 es de 8 bits, a diferencia de los anteriores, y no tiene la posibilidad de ser excitado por una señal externa, sino que sólo puede hacerlo internamente por la señal del oscilador principal **FOSC/4**. Es usado como base de tiempo para la generación de la señal PWM y, en muchas aplicaciones, se lo emplea como base de tiempo interna para medir un evento que ocurre en un puerto.

Desde el punto de vista estructural, posee un **prescaler** y un **postscaler**, es decir, un divisor de frecuencia de entrada y otro de salida. El **timer 2** trabaja comparándose siempre contra el contenido de un registro denominado **PR2**. De esta manera, cuando alcanza a tener el mismo valor que hay en el registro PR2, el módulo comparador, encargado de comparar ambos registros, genera un pulso de salida.

Los temporizadores o timers son contadores binarios que pueden contar pulsos internos o externos



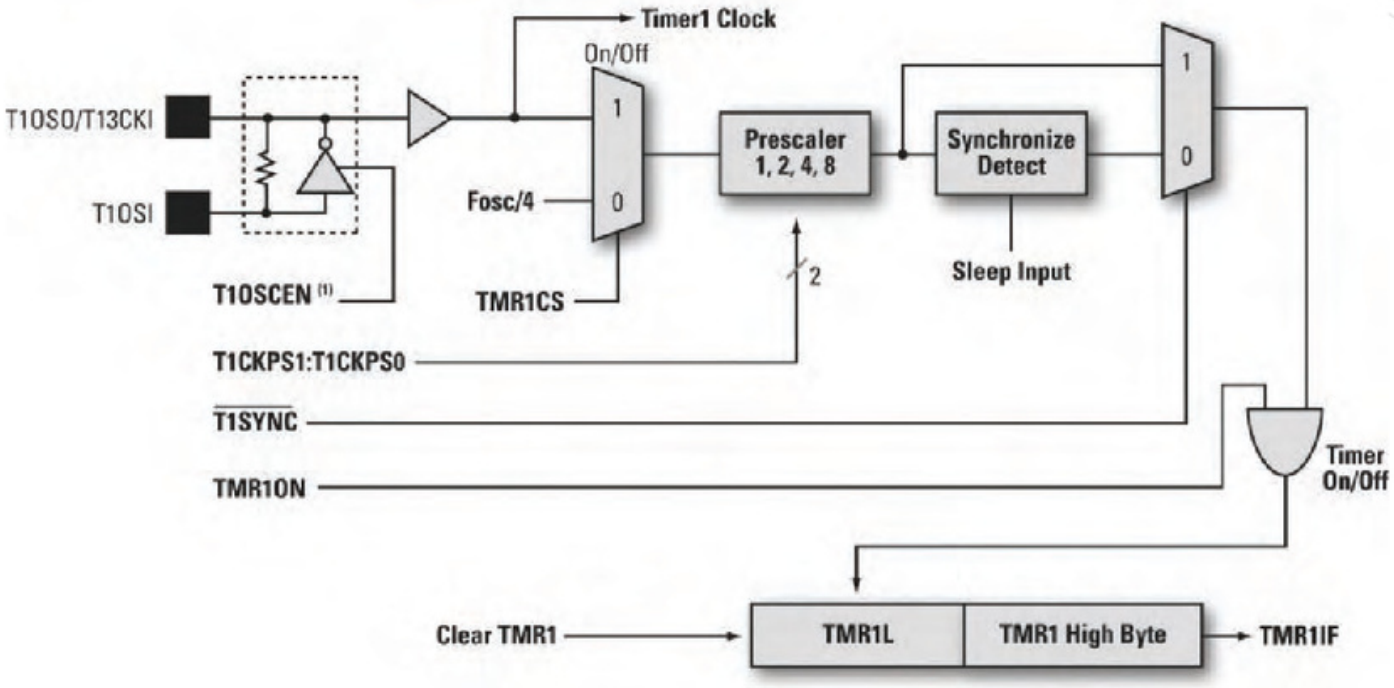


FIGURA 3. **Podemos apreciar el diagrama en bloques funcional del** timer 1.

Dicho pulso es usado para reiniciar el timer 2 a 0 y para ser enviado hacia el PWM, el MSSP. El **postscaler**, luego de dividir la señal, pondrá en 1 la bandera de interrupciones TMR2IF. El módulo PWM usará esta salida del comparador como base de tiempo para hacer funcionar el PWM y, en el caso del MSSP, se podrá usar como base de tiempo para generar la señal **SCK** o **SCL** (**Figura 4**).

EL CONVERSOR ANALÓGICO/DIGITAL

Los PIC18F, al igual que algunos modelos de PIC 16F, están equipados con un conversor A/D de 10 bits, cuya precisión es más que suficiente para realizar cualquier instrumento para control industrial. El conversor es del tipo **SAR**, por lo tanto, es muy rápido, y en el caso del PIC18F4620, posee

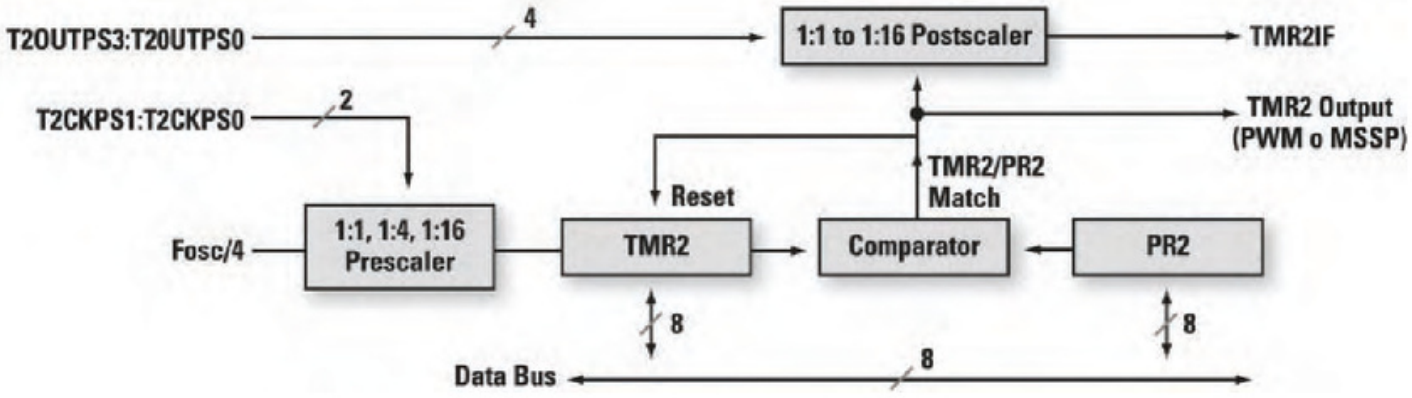


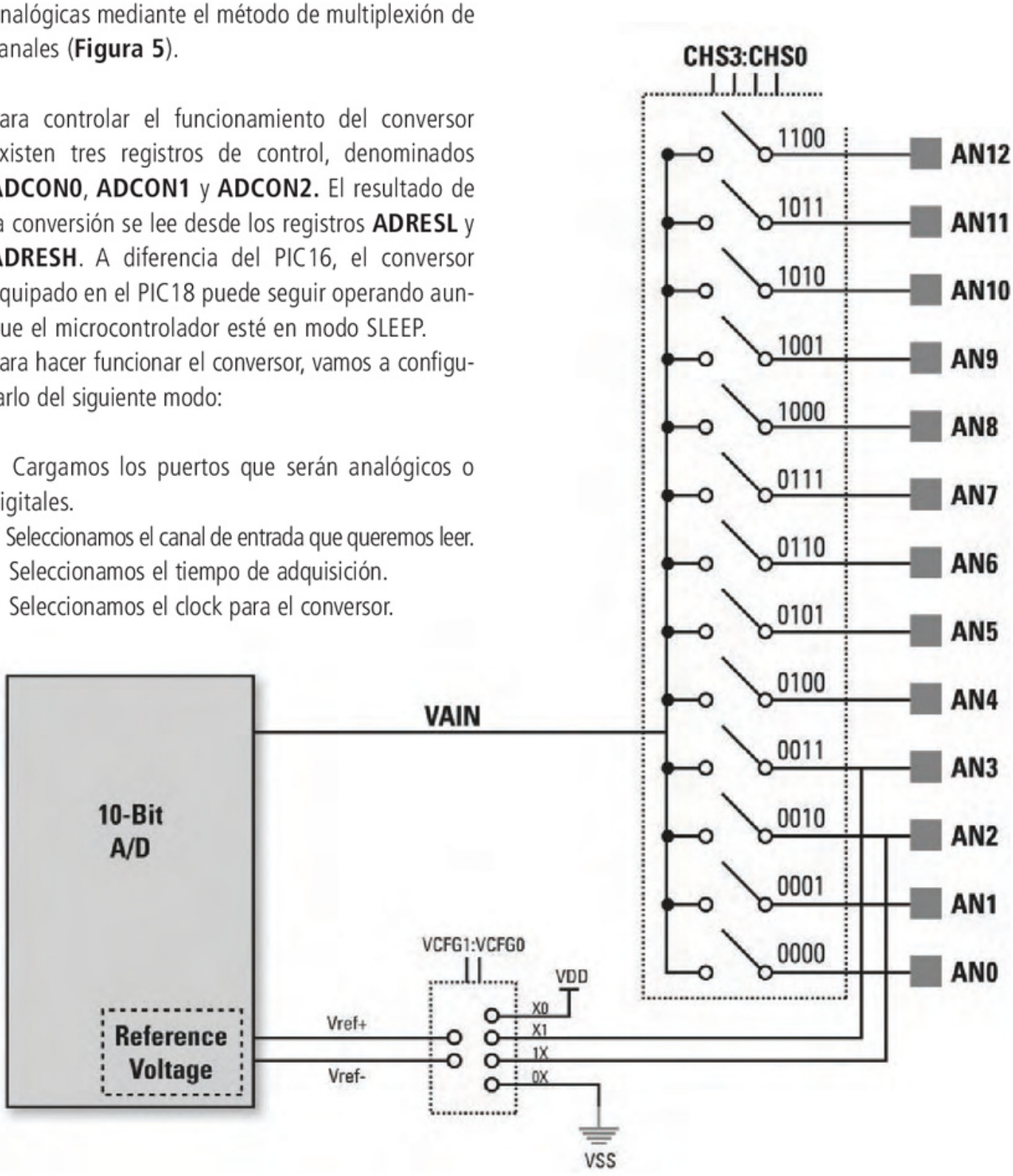
FIGURA 4. **Diagrama en bloques funcional del** timer 2.

13 canales de conversión (AN0-AN12), lo cual nos permite monitorear hasta esa cantidad de señales analógicas mediante el método de multiplexión de canales (**Figura 5**).

Para controlar el funcionamiento del conversor existen tres registros de control, denominados **ADCON0**, **ADCON1** y **ADCON2.** El resultado de la conversión se lee desde los registros **ADRESL** y **ADRESH**. A diferencia del PIC16, el conversor equipado en el PIC18 puede seguir operando aunque el microcontrolador esté en modo SLEEP.
Para hacer funcionar el conversor, vamos a configurarlo del siguiente modo:

- Cargamos los puertos que serán analógicos o digitales.
- Seleccionamos el canal de entrada que queremos leer.
- Seleccionamos el tiempo de adquisición.
- Seleccionamos el clock para el conversor.

FIGURA 5. Diagrama en bloques funcional del conversor A/D del PIC18LF4620.



Si trabajamos por interrupciones, deberemos realizar la siguiente configuración:

- Borrar la bandera de interrupciones del conversor A/D, bit ADIF.
- Poner en 1 el bit ADIE para habilitar la interrupción por final de conversión y, luego, el GIE mediante el cual habilitamos el sistema general de interrupciones.

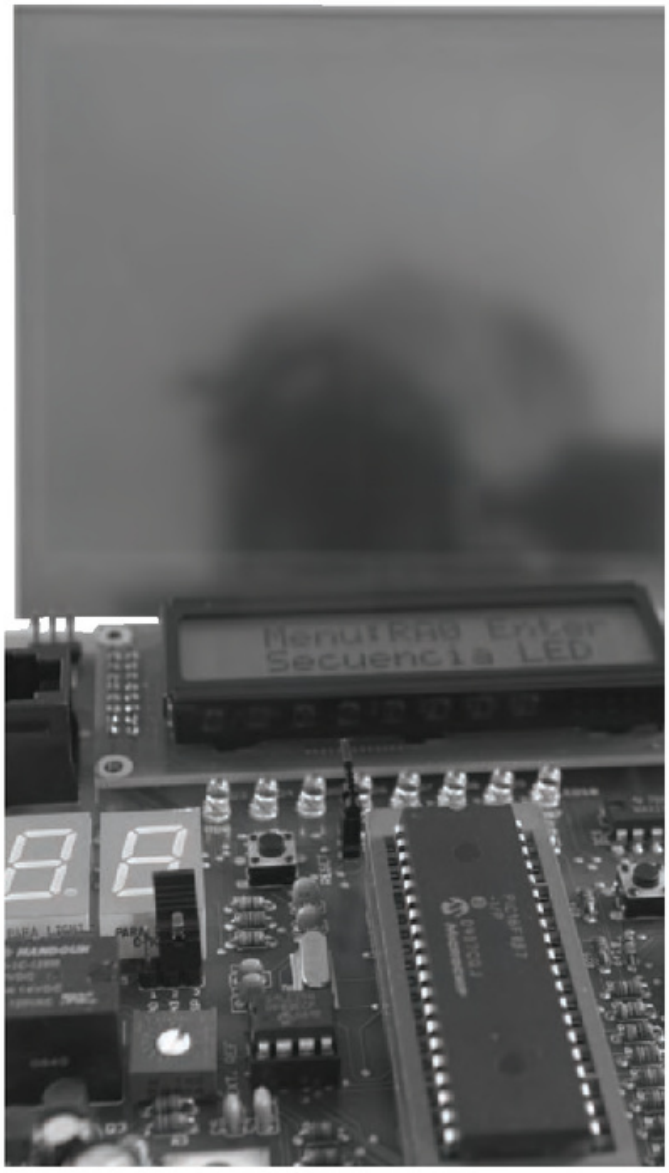


FIGURA 6. **La placa de entrenamiento nos permite manejar los periféricos básicos.**

El conversor equipado en el PIC18 puede seguir operando aunque el microcontrolador esté en modo SLEEP

- Esperar el tiempo de adquisición que se requiera.
- Iniciar la conversión poniendo en 1 el bit GO/DONE.
- Si no usamos la interrupción, revisar el estado que adquiere el bit GO/DONE, ya que, al finalizar el proceso de conversión, éste será puesto en 0 por el hardware del conversor, lo cual nos indica que el valor analógico ya fue convertido a formato digital
- Leer el valor convertido desde los registros ADRESL y ADRESH, y si se usó la interrupción, borrar el bit ADIF.
- Para reiniciar la conversión, esperar como mínimo dos tiempos de adquisición (Tad).

PLACAS ENTRENADORAS

Para entrenarnos en el uso de los microcontroladores PIC existen en el mercado placas específicas en las cuales aprender a utilizar todos los recursos disponibles que estuvimos viendo hasta el momento. Muchas de ellas cuentan con LEDs indicadores, pulsadores, potenciómetros, displays de 7 segmentos, pantallas LCD y bornes de salidas. En algunas ocasiones, es más económico comprar una placa entrenadora que armar una cuando ésta cuenta con integrados SMD, por ejemplo (**Figura 6**).

Entonces, estas herramientas, junto con un programador, son ideales para quienes desean iniciarse en el mundo de los microcontroladores. Una tarjeta de este tipo es la **MCE Starter Kit Student** de **MCElectronics** (**Figura 7**).

Manejo del display **LCD inteligente**

Los LCDs inteligentes son dispositivos periféricos usados para mostrar información. En ellos podemos representar todo tipo de mensajes conformados por letras, números e, incluso, caracteres gráficos programados por el usuario. Este tipo de display se denomina inteligente porque incluye sus propios controladores sobre el PCB que soporta a todo el sistema. Tiene un bus de datos capaz de operar en 8 o en 4 bits para comunicar la información, y cuenta con un bus de control formado por tres líneas. Son considerados periféricos en sí mismos.

Estos componentes surgieron a partir de mediados de los años '80, cuando la firma Hitachi desarrolló un controlador específico para tal fin denominado **HD44780**, el cual fue mejorado luego por Samsung. Este controlador permitió la contracción de distintos modelos de LCDs comerciales; tanto es así, que hoy podemos encontrar muchas ofertas de distintos fabricantes, como Winstar o Truly, por nombrar sólo dos de los muchos que hay. En el mercado nacional las ofertas disponibles son:

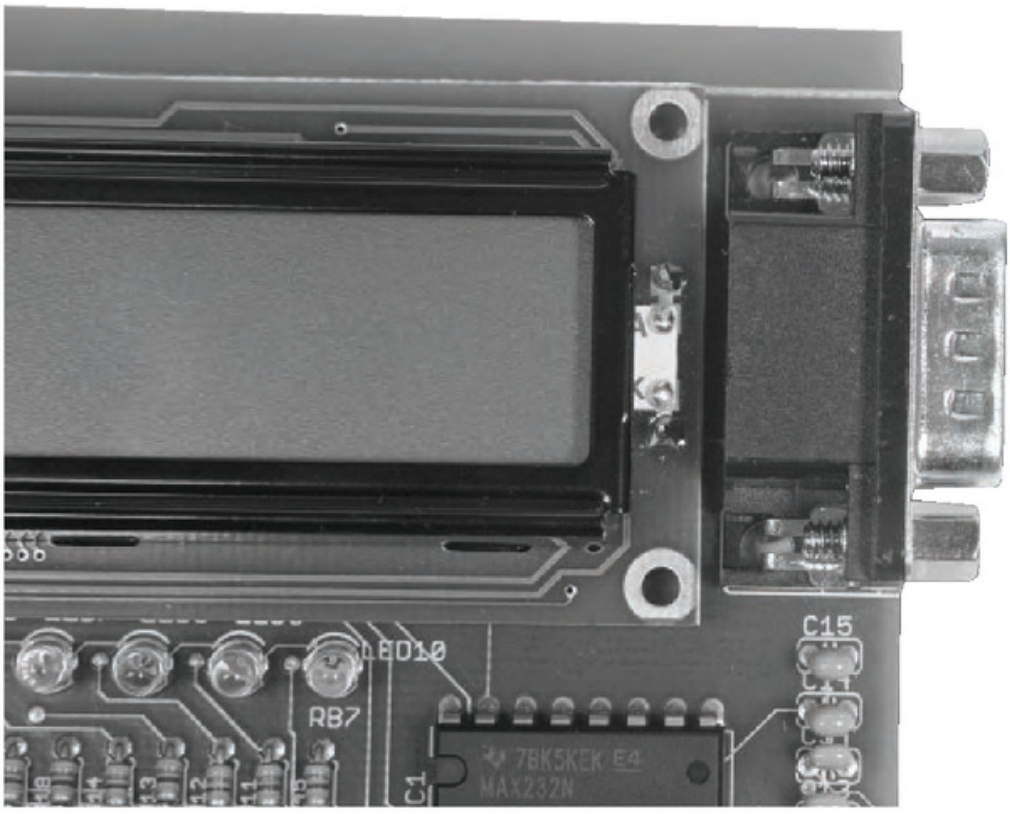


FIGURA 7. **La placa** MCE Starter Kit Student **cuenta con una conexión** RS232**, con la cual podemos comunicarnos con la PC desde nuestro PIC. Esta utilidad de los microcontroladores será analizada en el próximo capítulo.**

FIGURA 8. **Imagen de un LCD alfanumérico de 2 líneas por 40 caracteres.**

- 2 líneas x 8 caracteres en cada una, con back light (luz de fondo).
- 1 línea x 16 caracteres, con y sin back light.
- 2 líneas x 16 caracteres cada una, con y sin back light.
- 2 líneas x 40 caracteres cada una, con y sin back light (**Figura 8**).
- 2 líneas x 20 caracteres cada una, con back light.
- 4 líneas x 20 caracteres cada una, con back light.

Estos dispositivos se alimentan con **5 Volts**, tienen un control de contraste y el back light o luz de fondo puede tener distintos colores, siendo los más típicos los amarillos con letras en negro, los verdes con letras en negro, y los azules con letras en blanco. También los hay con back light inverso, es decir, con pantalla negra y letras en amarillo.

Según el estado lógico que apliquemos, el LCD interpretará si el dato enviado es un carácter por imprimir o un comando

COMANDOS DEL LCD

El LCD inteligente recibe por su bus de datos dos tipos de información: datos por imprimir en el display y comandos que le ordenan al display dónde imprimir esos datos o borrar el área visible del LCD. Para indicarle al LCD qué tipo de dato le estamos enviando, existe un terminal denominado **RS**. Según el estado lógico que apliquemos en él, el LCD interpretará si el dato enviado es un carácter por imprimir o un comando. De este modo, si queremos imprimir un carácter, habrá que poner en **1** el terminal RS; para indicarle que el dato enviado es un comando, debe estar en **0**.

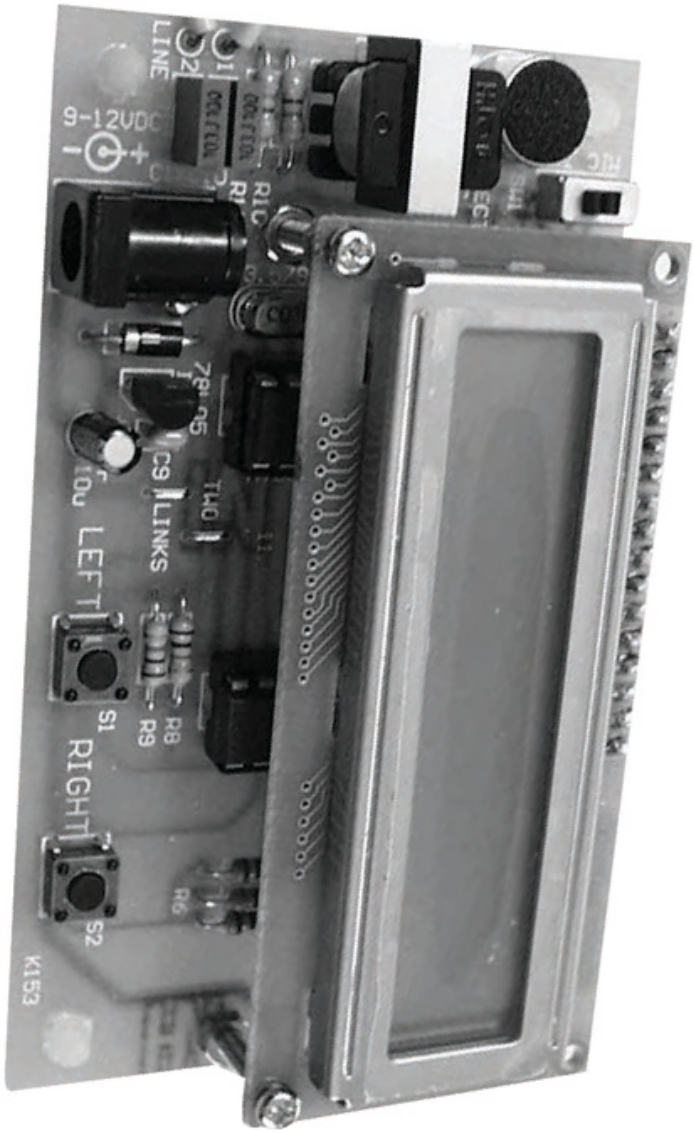
Los comandos más representativos que podemos enviarle al LCD se muestran en la **Tabla 1**. Además de ellos, existen otros que permiten controlar el modo de operación del display en 4 u 8 bits, así como también señalar si el cursor aparecerá titilante o no. Para que el display funcione, debe ser inicializado,

y esto se logra mandándole una secuencia de comandos, variable según el modo de operación de éste (es decir, si está conectado en 8 o en 4 bits). En ambos casos, debemos esperar inicialmente unos 15 ms al alimentar el circuito para, luego, activar la rutina de inicialización.

Si bien todo esto parece complejo de entender, los fabricantes de compiladores han desarrollado una serie de librerías en los lenguajes de alto nivel que nos permiten, de manera muy simple, inicializar el

FUNCIÓN	COMANDO
Borrar pantalla	(01H)
Posicionar cursor al principio	(02H)
Encender pantalla	(0EH)
Apagar pantalla	(0AH)
Desplazar el cursor a la izquierda	(10H)
Desplazar el cursor a la derecha	(14H)
Desplazar la pantalla a la izquierda	(18H)
Desplazar la pantalla a la derecha	(1CH)

TABLA 1. La letra H al final del número indica que el valor está expresado en hexadecimal.



LCD, y enviarle caracteres, mensajes completos y comandos de posicionamiento. En el caso, por ejemplo, del compilador C18, Microchip provee de una librería denominada **XLCD.h**, muy completa y fácil de manejar.

CARACTERES DEL LCD

Hemos mencionado anteriormente que al LCD podemos enviar comandos y caracteres imprimibles. En este caso, los caracteres se mandan en **formato ASCII** y todos los establecidos por el código ASCII estándar se imprimirán. Sin embargo, si bien ha existido un acuerdo entre todos los fabricantes en cuanto a los primeros 128 símbolos, no sucedió lo mismo con los códigos que van desde el 128 hasta el 255

Para construir un carácter, el controlador del LCD lo dibuja sobre una matriz de puntos

(conocidos como tabla ASCII extendida). En esta tabla, por ejemplo, los fabricantes orientales suelen colocar los símbolos de su alfabeto.

Para decirle al display que le enviamos un carácter imprimible, debemos poner en **0** el terminal **RS**. Los caracteres que enviamos se almacenan en la **memoria RAM** de video del controlador del LCD, denominada **DDRAM**. Ésta puede albergar un total de **80 caracteres**. Sin embargo, sólo serán visibles los que permita la ventana del LCD. Por ejemplo, en un LCD tipo 2 x 16 (2 líneas de 16 caracteres por línea), sólo veremos 32 caracteres ubicados en las direcciones de la memoria RAM, que van desde (**Figura 9**):

Primera línea: 00-0FH
Segunda línea: 40-4F

Esto sólo es cierto para los **LCD tipo 2 x 16**, ya que los **2 x 20** tienen una mayor área visible. Por

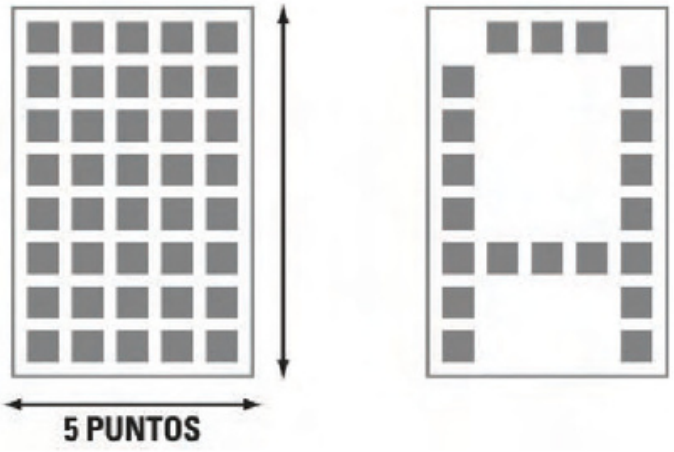


FIGURA 10. **Las dos matrices de caracteres que es posible configurar por medio del envío del comando apropiado.**

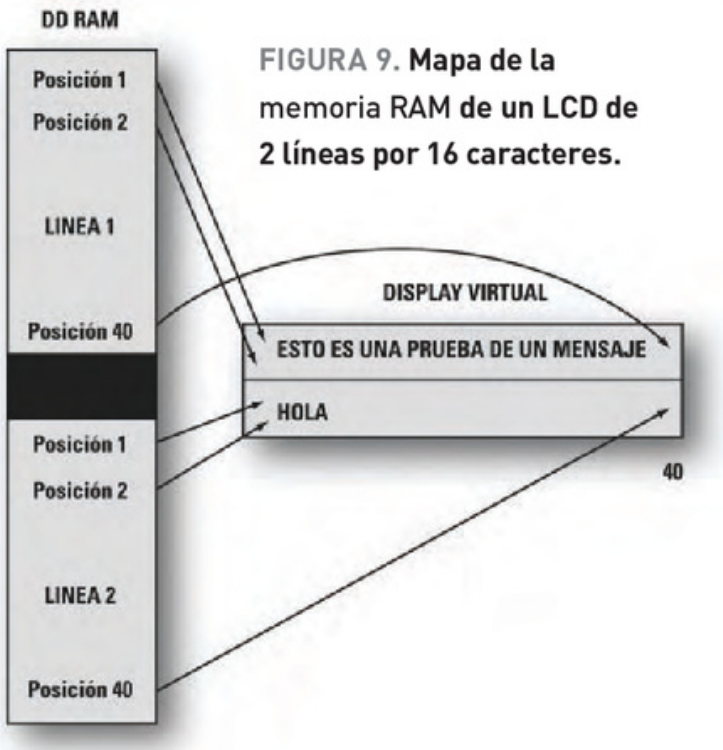


FIGURA 9. **Mapa de la memoria RAM de un LCD de 2 líneas por 16 caracteres.**

lo tanto, siempre deben consultarse las hojas de datos del controlador que tenga el LCD. Para construir un carácter, el controlador del LCD lo dibuja sobre una matriz de puntos, que puede tener dos tamaños: **5 x 7** o **5 x 10** (esto es programado por el usuario). Además, el usuario tiene la posibilidad de armar sus propios caracteres, por ejemplo, para introducir un logo propio en el LCD. En este caso, es posible programar hasta 8 caracteres de 5 x 7 o hasta 4 caracteres de 5 x 10. Éstos se almacenarán en una RAM distinta, denominada RAM del generador de caracteres o **CGRAM** (**Figura 10**).

CONEXIÓN DEL LCD EN 8 BITS

Éste es el modo normal de operación de los LCD, y donde obtenemos el mayor rendimiento en cuanto a velocidad de comunicaciones. En este caso, de-

bemos conectar el puerto del microcontrolador –por ejemplo, el **PORTD–** al bus de datos del **LCD**, haciendo corresponder el número de bits en la conexión. Además, necesitaremos un par de puertos auxiliares para controlar la comunicación, lo cual se consigue manejando las líneas de control del LCD (ver **Tabla 2**).

Existen dos maneras de enviar información al LCD. La primera es hacerlo de a un carácter por vez, esperando unos 10 milisegundos entre cada envío hasta terminar con el mensaje. Si bien este método, denominado de fuerza bruta, es lento, nos permite ahorrar un puerto, ya que nunca leemos el LCD. En este sistema, el terminal **R/W** debe ir conectado directamente a masa.

TERMINAL	DESCRIPCIÓN
E (Enable o habilitador)	Esta línea debe recibir un pulso que le permite capturar el dato colocado sobre el bus.
RS (Select Register o selector de registro)	Mediante esta línea le decimos al LCD si el dato que hemos puesto sobre el bus es un dato o un comando.
R/W (Read/Write o lectura/escritura)	Si estamos escribiendo un dato, este pin deberá ponerse en 0. Si queremos leer el estado interno del LCD para saber si está ocupado o no, lo ponemos en 1.

TABLA 2. **Podemos ver aquí qué función cumple cada uno de los terminales destinados a controlar la comunicación con el LCD.**

Para leer el estado interno, además de poner el terminar R/W en 1, también debemos poner en 1 el terminal RS

La otra manera es más inteligente, porque usamos el terminal **R/W** para leer el LCD antes de enviar un dato, para saber si está ocupado. Si no lo está, mandamos el dato; en caso contrario, esperamos un tiempo y repetimos la lectura.

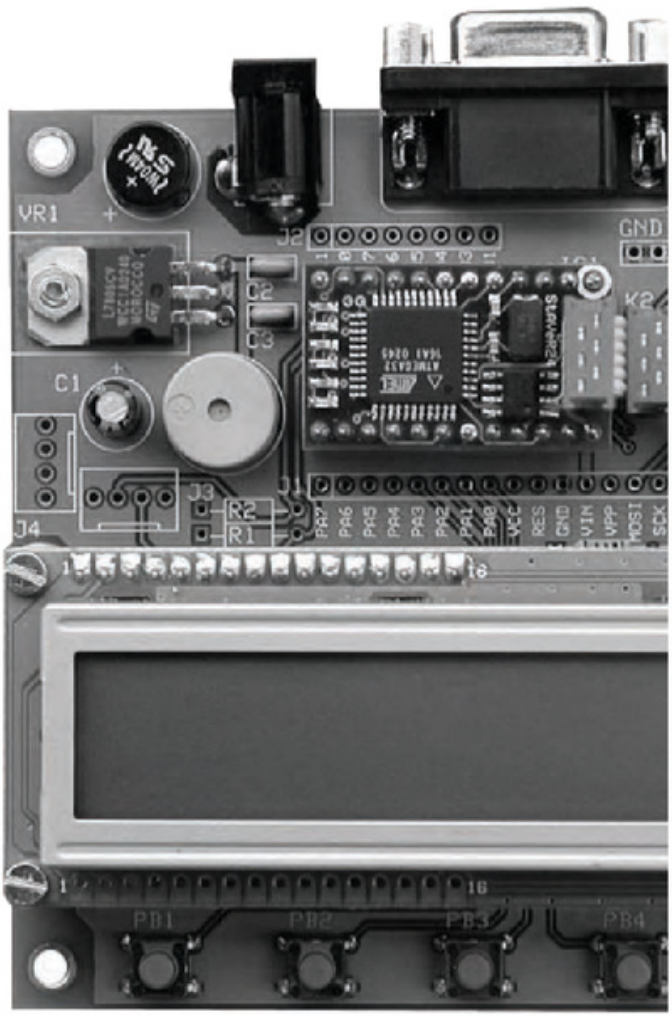
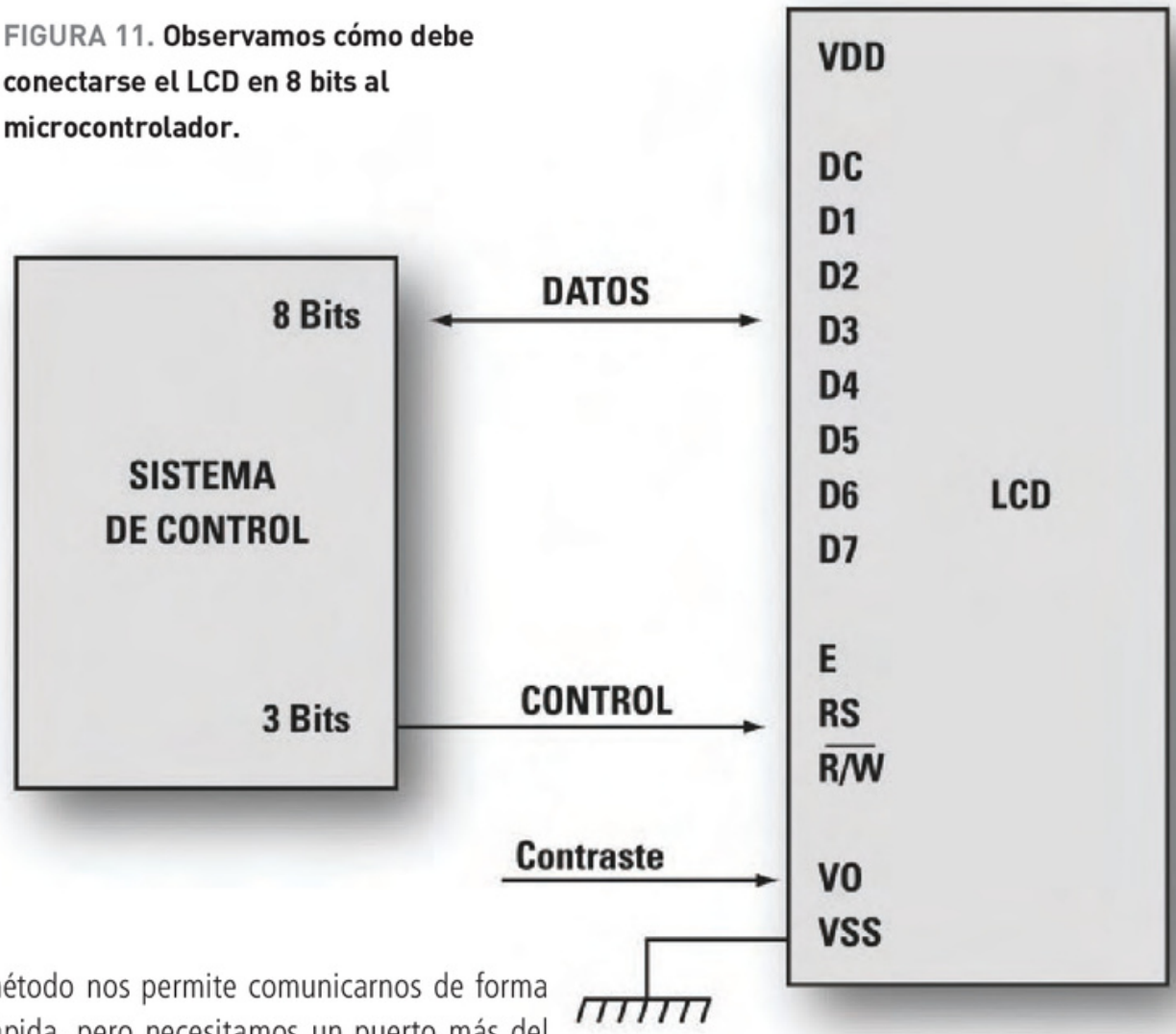


FIGURA 11. **Observamos cómo debe conectarse el LCD en 8 bits al microcontrolador.**



Este método nos permite comunicarnos de forma más rápida, pero necesitamos un puerto más del microcontrolador, por lo cual no es muy utilizado. En este caso, el puerto del microcontrolador debe reconfigurarse como entrada –cuando leemos el estado interno del LCD– y como salida –cuando escribimos un dato–, dado que la comunicación es bidireccional (**Figura 11**).

Para leer el estado interno, además de poner el terminar **R/W** en **1**, también debemos poner en **1** el terminal **RS**. El LCD enviará su estado interno sacando un byte, donde su bit de mayor peso (bit 7) nos indicará si está ocupado o no. Este bit se denomina **bit de busy**.

CONEXIÓN DEL LCD EN 4 BITS

Otra forma de conectar el LCD a un microcontrolador es la de 4 bits, en cuyo caso ahorramos muchos puertos (**Figura 12**). Para hacer la conexión, se usan los 4 bits de mayor peso del bus de datos del LCD. Como en el caso anterior, podemos controlar las transferencias de datos usando el terminal **R/W**, que en este último caso nos permitirá ahorrar otro pin más. Sin embargo, en la conexión de 4 bits es aconsejable leer el estado interno del LCD, ya que las transferencias son más lentas.

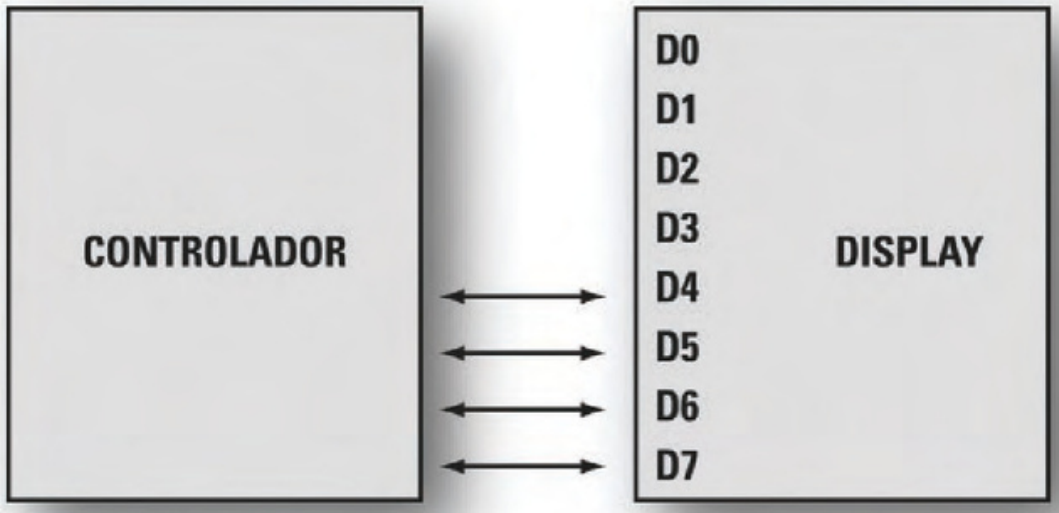


FIGURA 12. **Ésta es la manera de conectar el LCD en 4 bits al microcontrolador.**

En la conexión de **4 bits** las transferencias se duplican, puesto que para enviar cada dato, hay que dividirlo en dos transferencias de 4 bits cada una. Esto aumenta el tiempo del proceso y hace que la comunicación resulte más lenta. Sin embargo, las librerías de los lenguajes de alto nivel están preparadas para la operación tanto en 8 como en 4 bits.

El único cuidado que debemos tener es que los puertos elegidos para conectar el microcontrolador con el LCD deben pertenecer al mismo **PORT**. En este caso, si usáramos el **PORTB,** podríamos conectar el LCD a los terminales **RB0** a **RB3** o **RB4** a **RB7**. De ningún modo podemos usar puertos de distintos PORTS en lo que respecta al bus de datos. Sin embargo, es posible usar cualquiera para las líneas de control. Por lo general, en la práctica es común ver, por ejemplo, todo el LCD conectado al mismo PORT (PORTB o PORTD).

No debemos olvidar que también hay que tener cuidado en cuanto a la inicialización del LCD, porque en dicha rutina cambian los códigos que se le envían. Si no se están usando las librerías que trae el lenguaje de alto nivel, habrá que crear una nueva rutina de inicialización.

VENTAJA DE LA CONEXIÓN EN 4 BITS

La conexión del display en 4 bits es muy usada en los microcontroladores con poca cantidad de puertos. Esto nos permite tener una mayor libertad para aprovechar los escasos recursos de estos micros, como sucede con el PIC16F84A que cuenta con 13 puertos.

Para enviar los caracteres y comandos suele hacerse una rutina que divide cada byte por transferir en dos **nibbles (formación de 4 bits)** y realiza la transferencia correspondiente, comenzando por el nibble alto y, luego de haber verificado el estado del LCD, siguiendo con el bajo.

LIBRERÍA LCD DEL MPLAB C18

El programa **MPLAB C18** posee su propia librería para manejar LCDs inteligentes, basado en la normalización que impuso el controlador **Hitachi HD44780** (**Figura 13**). Esta librería se llama **xlcd** (*External LCD Functions*, funciones para LCD externo) y nos permite modificar su código para definir, por ejemplo, cuál es el puerto del PIC que se conectará al bus de datos del LCD; es decir, si será de 8 o 4 bits y cuáles son los pines del PIC que actuarán para el control del LCD (conexión de los pines **RW**, **RS** y **E** del LCD al PIC). Todas estas configuraciones deben efectuarse desde el archivo de cabecera **xlcd.h** de la librería. Más adelante veremos la manera de realizarlas.

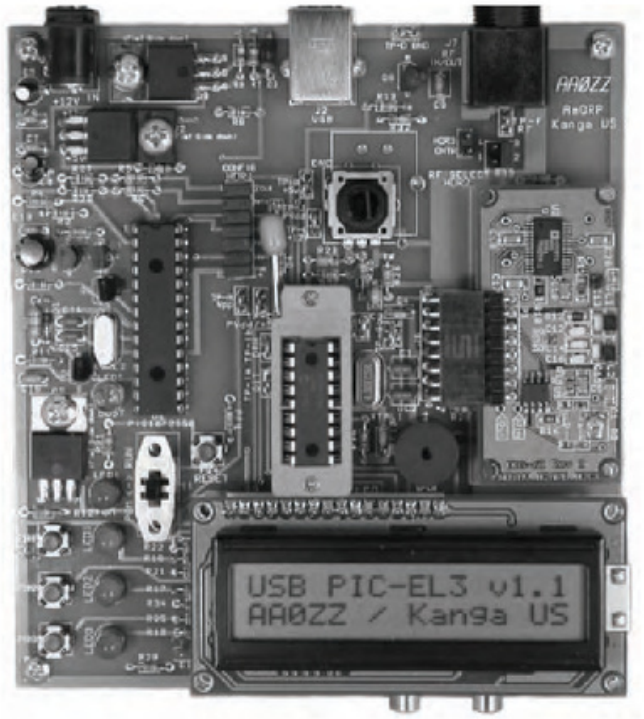


FIGURA 13. **Los displays LCD alfanuméricos poseen un controlador basado en el** Hitachi HD44780. **Es posible utilizar las librerías de los compiladores para controlarlos.**

FUNCIÓN	DESCRIPCIÓN
BusyXLCD	Interroga si el bus de datos está desocupado.
OpenXLCD	Inicializa el LCD.
putcXLCD	Escribe un byte en el LCD.
putsXLCD	Escribe un string de la memoria de datos en el LCD.
putrsXLCD	Escribe un string de la memoria de programa en el LCD.
ReadAddrXLCD	Lee la dirección de un byte del LCD.
ReadDataXLCD	Lee un byte del LCD.
SetCGRamAddr	Establece la dirección del generador de caracteres.
SetDDRamAddr	Establece la dirección de la visualización de caracteres.
WriteCmdXLC	Escribe un comando en el LCD.

TABLA 3. **Listado de funciones de la librería LCD del** MPLAB C18**, con una breve descripción de cada una.**

FUNCIONES DE LA LIBRERÍA XLCD

Esta librería posee un repertorio de funciones bastante amplio, que nos permite realizar diversas tareas con el LCD, desde enviar un comando, hasta escribir mensajes enteros en la pantalla. En la **Tabla 3** podemos observar las diferentes funciones que ofrece, y a continuación describimos las más utilizadas.

• **OpenXLCD**: inicializa el LCD, al cual debemos indicarle la cantidad de bits del bus de datos y el tipo de display que utilizaremos. También debemos señalar si es de una o de múltiples líneas, y el tamaño de la matriz para cada carácter (5 x 7 o 5 x 10).
• **WriteCmdXLCD**: con esta función enviamos comandos de control al LCD.
• **WriteDataXLCD**: envía caracteres al LCD, que deben ser de tipo **char de 8 bits.**
• **BusyXLCD**: interroga si el bus de datos se encuentra libre para poder enviar otro dato.
• **putsXLCD**: escribe una cadena de caracteres al LCD que se encuentra almacenada en la memoria de datos.
• **putrsXLCD**: escribe una cadena de caracteres al LCD que se encuentra almacenada en la memoria de programa.

AGREGAR LA LIBRERÍA XLCD A NUESTRO PROYECTO

Para utilizar las funciones de esta librería, primero tenemos que añadirla a nuestro proyecto, realizando los siguientes pasos.

En el entorno del MPLAB, vamos a **Projects/Source Files/Add Files** y agregamos todos los archivos fuente de la carpeta **XLCD** a nuestro proyecto (**Figura 14**). Éstos son los códigos fuente de las funciones de la librería.

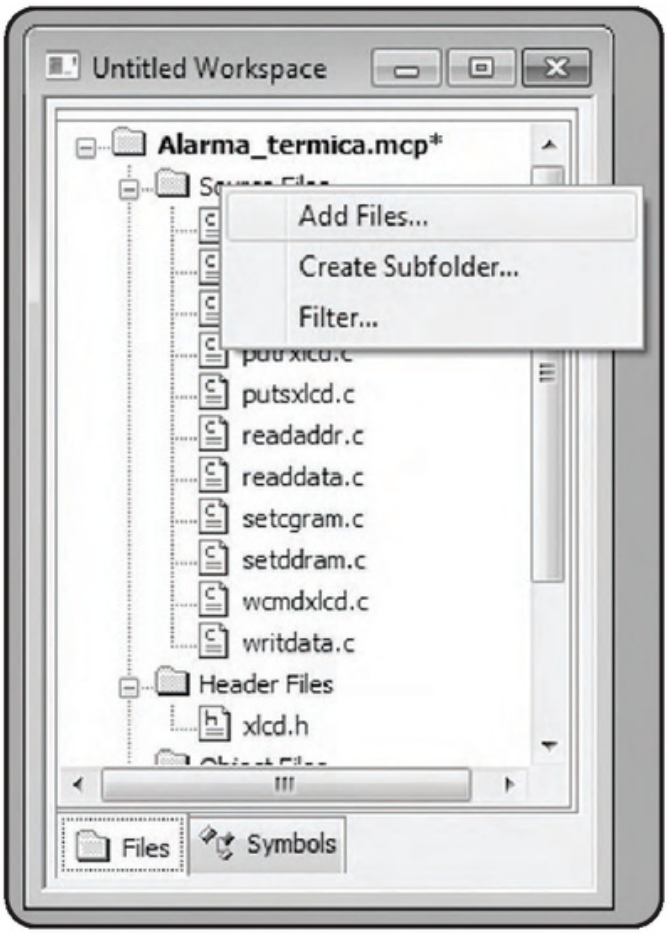
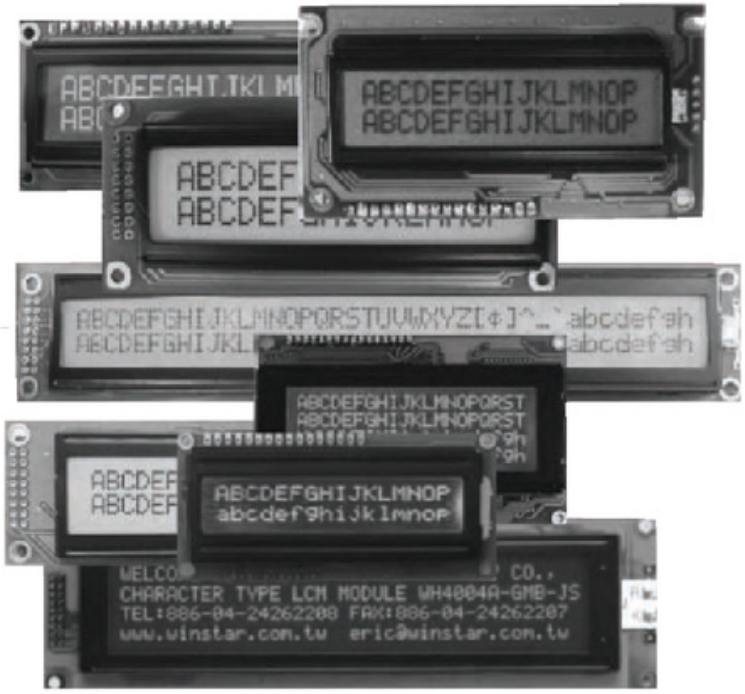


FIGURA 14. En la ventana Projects agregamos los códigos fuente de las funciones correspondientes a la librería del LCD.

Nos dirigimos a la ventana **Projects**, hacemos clic con el botón derecho del mouse en la carpeta **Header Files**, seleccionamos **Add Files** y agregamos el archivo **xlcd.h** del proyecto.

RETARDOS

La librería del LCD del MPLAB C18 requiere de tres retardos que debe definir el usuario y que dependerán de la frecuencia de trabajo del microcontrolador. Los tiempos que se indican deben ser: uno de 18 ciclos de máquina, otro de 15 mS y otro de 5 mS.




```
Para la conexión del bus de datos:
#define DATA_PORT  PORTD
#define TRIS_DATA_PORT TRISD

Para la conexión de las líneas de control:
#define RW_PIN LATEbits.LATE0 /* PORT for RW */
#define TRIS_RW TRISEbits.TRISE0 /* TRIS for RW */
#define RS_PIN LATEbits.LATE1 /* PORT for RS */
#define TRIS_RS TRISEbits.TRISE1 /* TRIS for RS */
#define E_PIN LATEbits.LATE2 /* PORT for E */
#define TRIS_E TRISEbits.TRISE2 /* TRIS for E */
```

A continuación tenemos que configurar el compilador. Para hacerlo, le indicamos dónde buscar el archivo **xlcd.h**. Esto se hace desde la barra de menús **Project/Build Options/Projects**. En la solapa **Directories** elegimos **Include Search Path**, hacemos clic en **New** y colocamos la dirección **C:\MCC18\h**.

MODIFICAR LA LIBRERÍA DEL LCD

Cuando trabajamos con la librería del LCD, debemos modificarla para adaptarla a la conexión de nuestra placa. Por defecto, la librería define la línea de datos en el puerto **B** de los PICs, y los pines de control **E**, **RS** y **RW** en los pines **RB4**, **RB5** y **RB6,** respectivamente.

Por ejemplo, si queremos colocar el LCD en el puerto D y las líneas de control en el E, escribimos lo siguiente en el archivo **xlcd.h** (**Figura 15**):

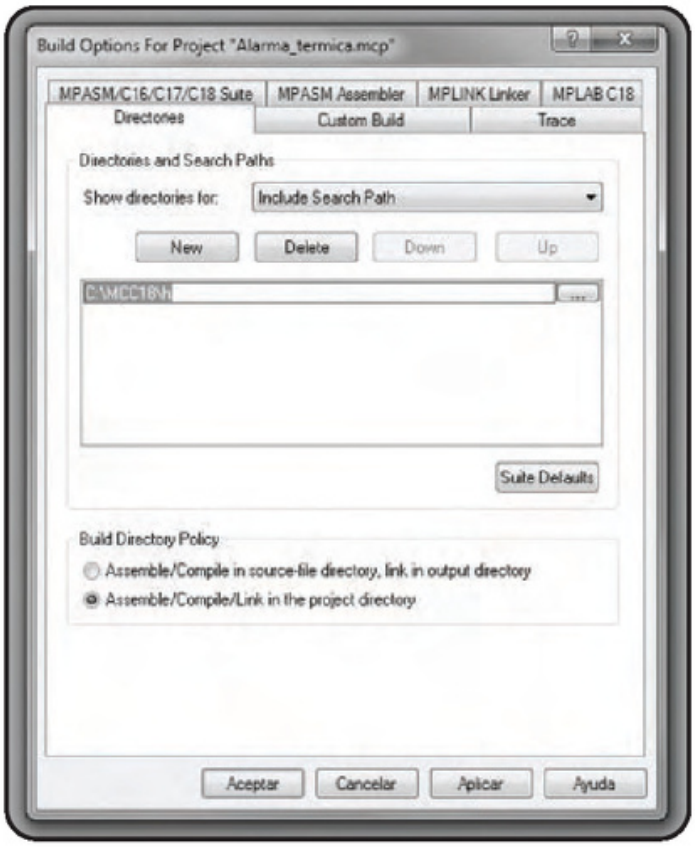


FIGURA 15. Debemos configurar el compilador indicando la dirección donde se encuentra el archivo de cabecera xlcd.h.

Proyecto de **alarma térmica**

Esta alarma se ocupará de monitorear la temperatura de un recinto cerrado, que puede ser una habitación, el gabinete de una PC o un horno industrial. Este valor debe ser visualizado en todo momento a través de una pantalla LCD, donde tiene que figurar con un dígito decimal de precisión.

A su vez, la alarma debe tener predefinida por el programador una temperatura crítica, de modo que si el recinto excede ese valor, se active un relay que controle un sistema de refrigeración, como un ventilador. También podemos definir que el programador se encuentre conectado a una bocina, para indicar que la temperatura está por encima del valor crítico. Además, si se acerca, por ejemplo, a 5° C (umbral de la temperatura crítica), la alarma tendrá que indicarnos sobre esta situación mediante un mensaje de atención en la pantalla LCD.

SENSOR DE TEMPERATURA

El sensor de temperatura que utilizaremos para este proyecto es el **TC1047** de Microchip, que convierte la temperatura en un nivel de tensión bastante preciso (**Figura 16**).

Es de tres pines solamente y puede ser alimentado con **2,5 V** hasta **5,5 V.** Tiene un encapsulado **SMD**, lo cual lo hace ideal para medir temperaturas en recintos muy pequeños, como en teléfonos celulares. El rango de temperatura de medición es de **-40° C** a **125° C**, como se observa en la **Figura 17**, en donde la tensión de salida se comporta de manera

El sensor de temperatura que utilizaremos es el TC1047, que resulta bastante preciso en la conversión

lineal. La pendiente de la recta es de **10 mV/°C,** es decir que por cada grado que se modifique la temperatura, la tensión variará unos **10 mV.**

Podemos obtener el valor de la temperatura sensada por el dispositivo si sabemos cuál es la tensión de salida (Vout). Sólo debemos despejar la temperatura

FIGURA 16. **En la imagen podemos observar el aspecto físico del sensor de temperatura que utilizaremos para este proyecto.**

de la ecuación de la recta que se muestra en la **Figura 17: Temperatura (ºC) = (Vout (Volts) – 500 mV) / 10 mV/ºC;** donde **Vout** es la tensión de salida que mediremos con el conversor, y **10 mV/ºC,** la pendiente de la recta (**Figura 17**).

ESQUEMÁTICO

En la **Figura 18** observamos el esquemático del proyecto con el que vamos a trabajar. La alimentación estará dada por un transformador de **9 Volts** a la entrada. Un circuito regulador con el **7805** se encargará de controlar y estabilizar la tensión a unos **5 V**. El centro de nuestro proyecto es el **PIC18LF4620,** al cual se le acoplarán los distintos periféricos elegidos para este desarrollo. A un costado vemos el sensor de temperatura, donde sólo tenemos que conectar la alimentación y el pin con la tensión de salida al pin **RA5** (canal 4 del conversor) del PIC. En el pin **RA3** del PIC colocamos un potenciómetro para calibrar la tensión de referencia del conversor A/D y lo llamamos **potenciómetro de referencia**.

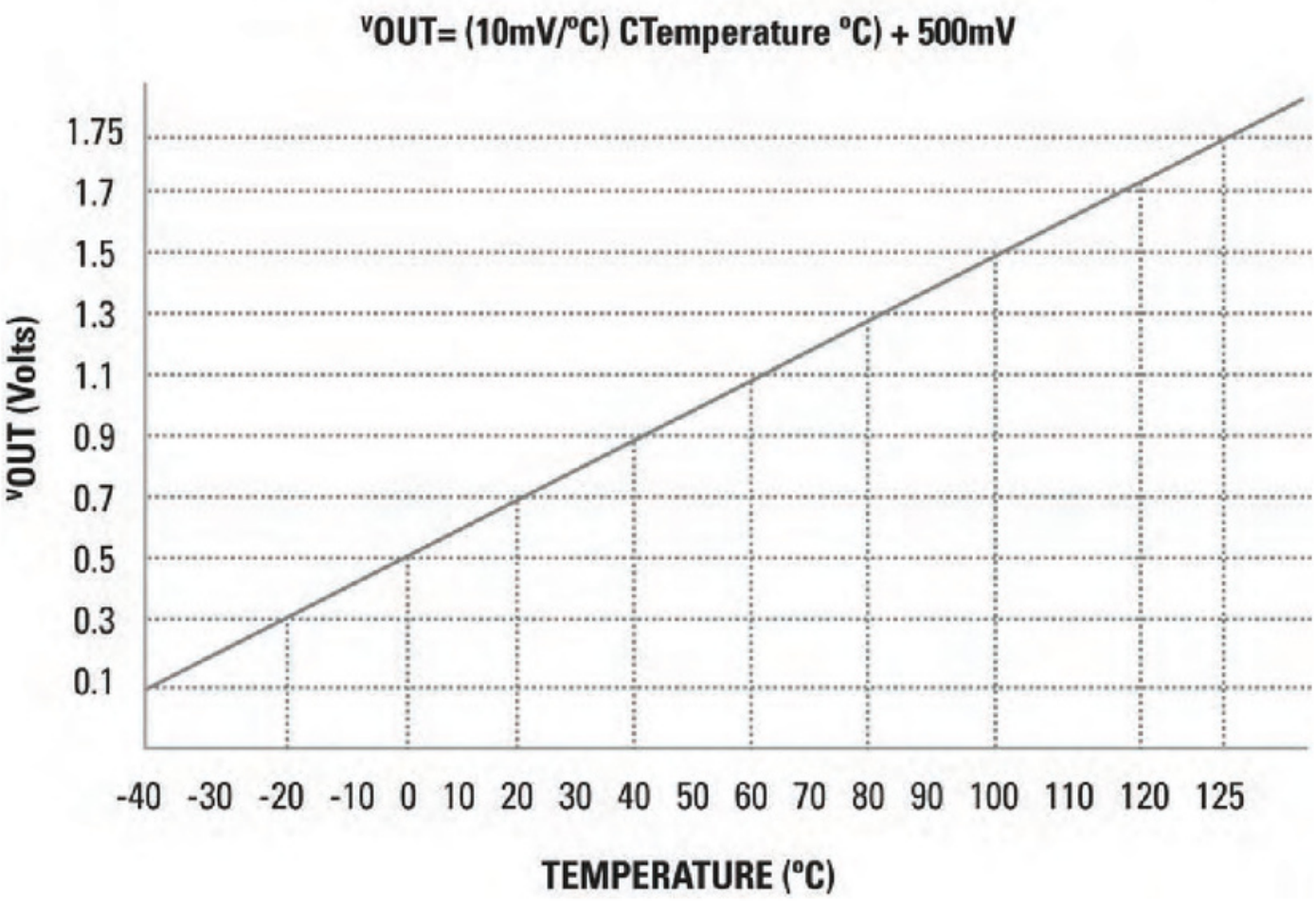


FIGURA 17. **Ésta es la gráfica que relaciona la temperatura leída por el sensor y la tensión de salida. Podemos notar que se comporta de manera lineal en el rango de temperatura especificado por el fabricante.**

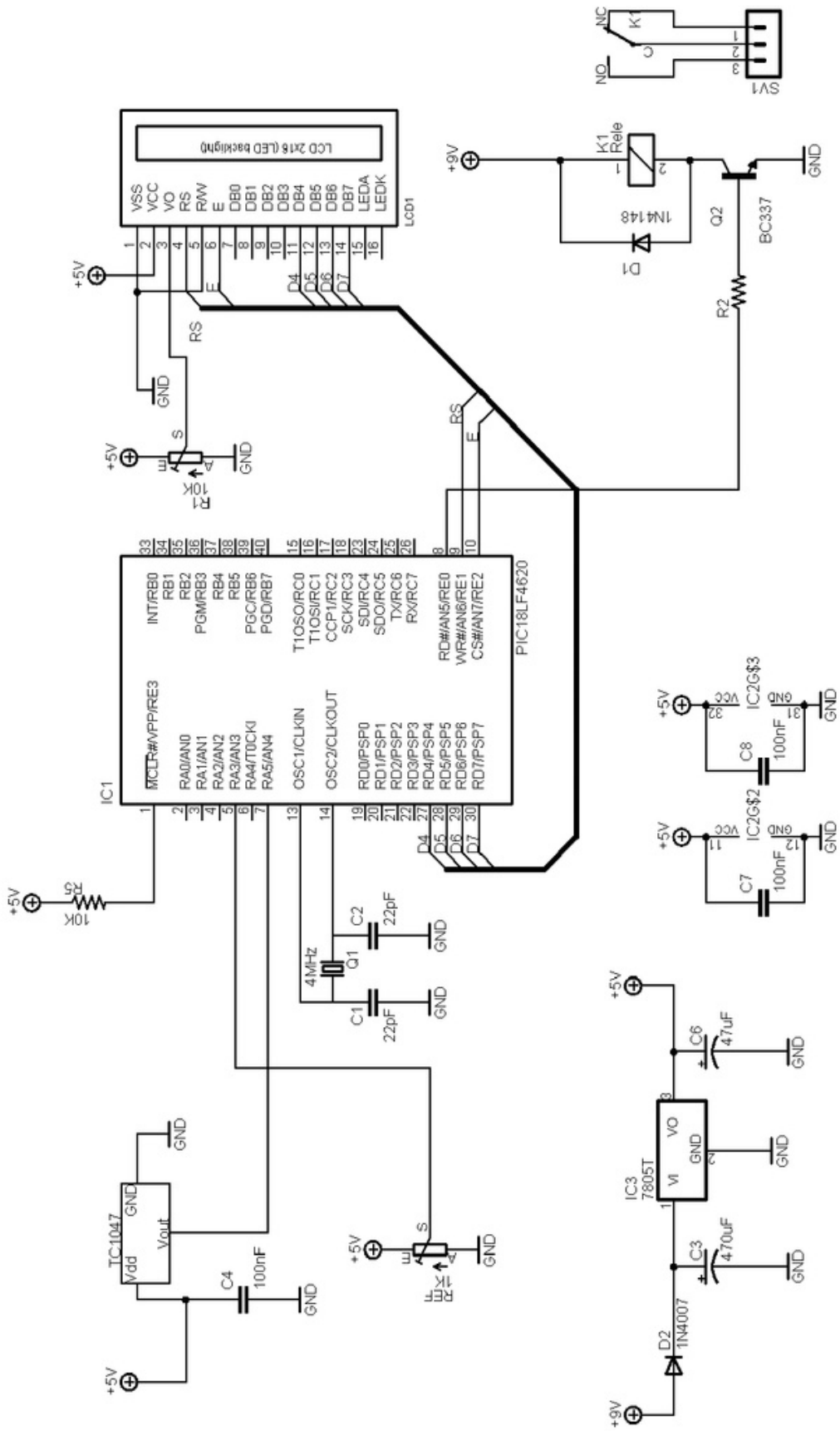
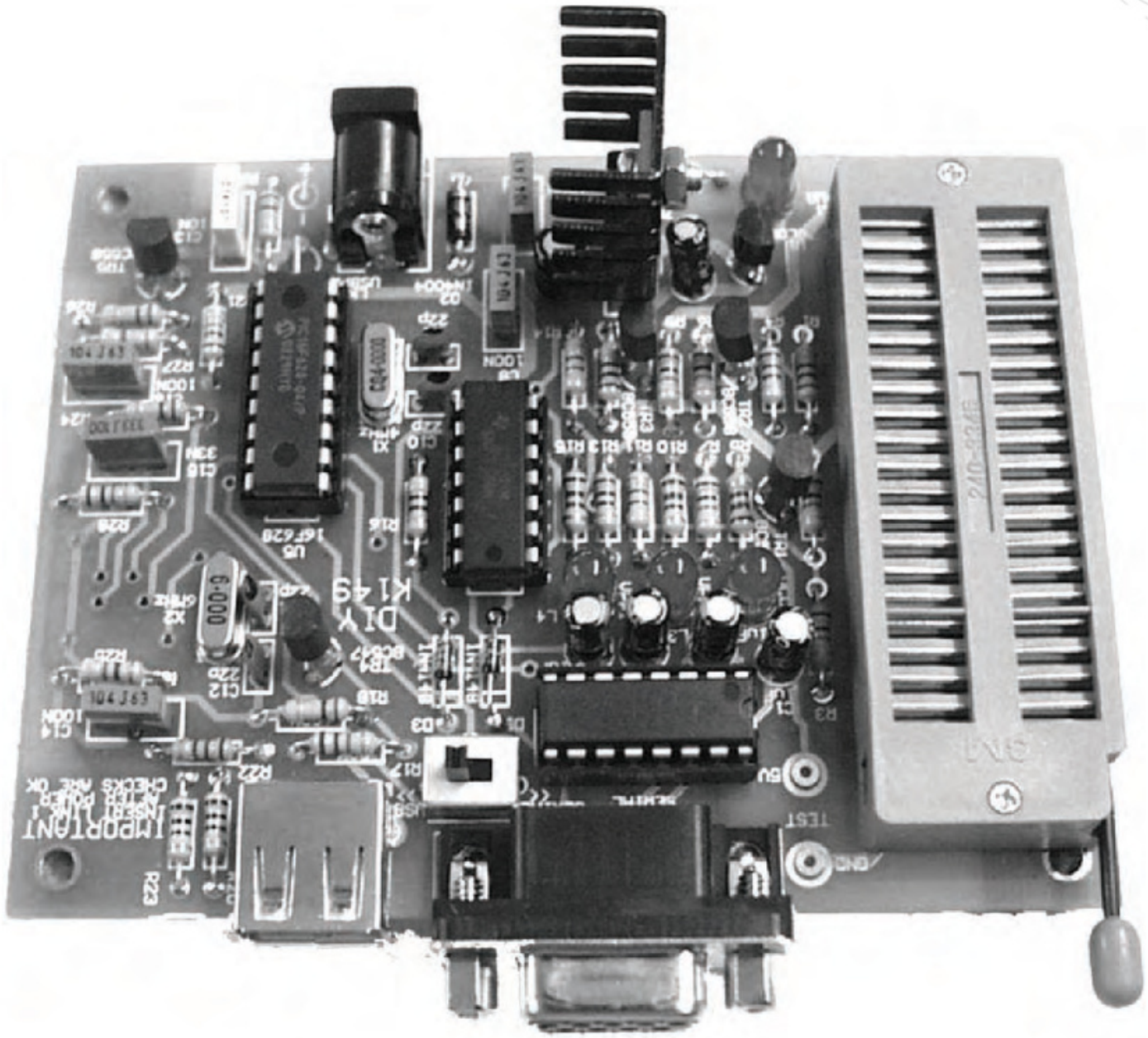


FIGURA 18. En la imagen vemos el esquemático del proyecto.



La pantalla LCD está conectada con un bus de 4 bits al **puerto D** del PIC, y los pines de control E y RS del LCD, a los pines RE2 y RE1 del PIC, respectivamente. La línea R/W del LCD se conecta a masa. Sólo vamos a escribir en el LCD, nunca leeremos un dato de él. Luego, la salida de nuestra alarma térmica es un relay controlado por el pin **RE0** del microcontrolador. Allí podemos conectar el control de un sistema de refrigeración o un simple ventilador para regular la temperatura del recinto, o agregar una bocina que nos indique el exceso de temperatura de manera sonora.

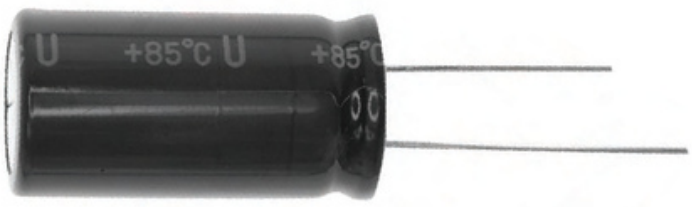
Cada usuario puede expandir la placa experimental para este proyecto, aplicando todo lo aprendido hasta el momento en esta obra. También existe la alternativa de utilizar una placa de entrenamiento comercial para simular el ejercicio.

El nivel de referencia positivo del conversor será la tensión en el pin RA3/Vref, en tanto que la otra referencia será GND

DIAGRAMA DE FLUJO

En la **Figura 19** se presenta el diagrama de flujo de este proyecto, que tomaremos como guía para programar el PIC. Para manejar el LCD, en esta práctica vamos a emplear las funciones de la librería de **MPAB C18**. Luego, continuamos con la configuración de los registros del conversor **A/D** del PIC para efectuar la conversión durante el programa.

Una vez que terminamos de configurar todos los recursos del PIC, comenzamos el programa realizando la conversión analógica-digital de la entrada analógica, donde está conectado el sensor de temperatura. Cuando tenemos el valor digital de la conversión, calculamos el valor de la temperatura mediante la ecuación descripta en la página anterior, con lo cual obtendremos un valor que es almacenado en una variable **float.** Si queremos visualizarlo en el LCD alfanumérico, debemos convertir ese valor en caracteres ASCII, para, luego sí, enviarlo a la pantalla.



Ahora, si la temperatura del recinto está por debajo de un cierto umbral crítico, no hacemos nada; pero si está dentro del rango del umbral, lo indicamos en la pantalla mediante un mensaje de atención. Cuando la temperatura del recinto supera el valor crítico, se activará el relay, lo cual se indica en el display LCD.

CONFIGURACIONES DEL PIC

Los primeros bloques del diagrama de flujo se encargan de configurar los recursos del PIC. Para configurar

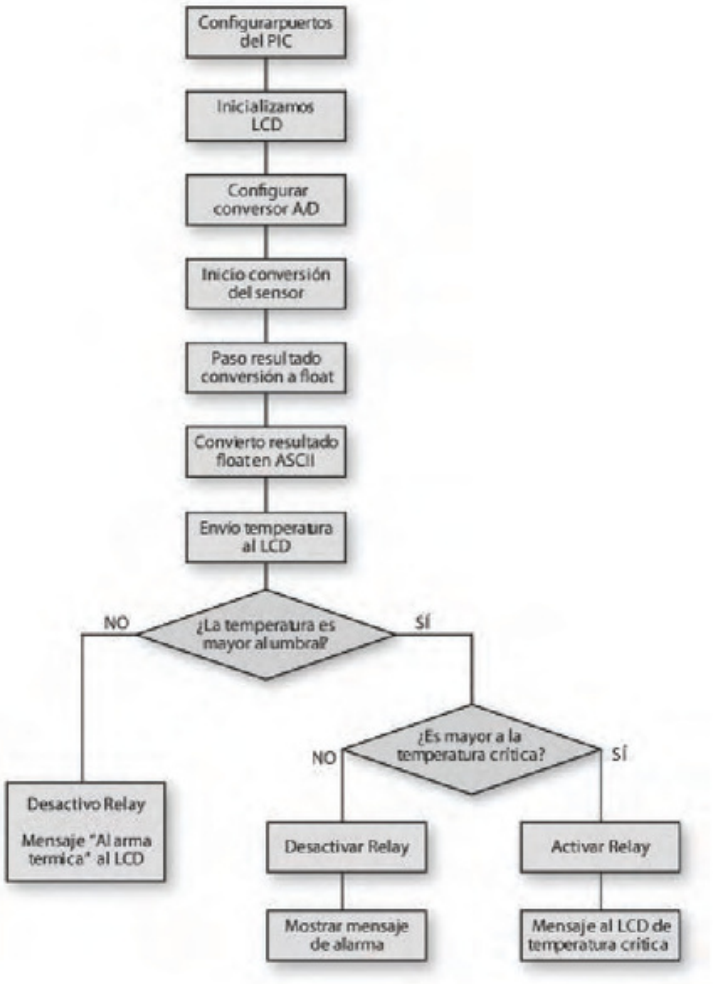


FIGURA 19. **Diagrama de flujo completo correspondiente al proyecto de la** alarma térmica.

los puertos, debemos identificar cuáles se utilizarán como entradas digitales, cuáles como salidas digitales y cuáles serán las entradas analógicas.

En el esquemático podemos ver que no contamos con entradas digitales, y que tenemos una salida digital en **RE0,** que controla el relay. No debemos preocuparnos por los puertos que maneja el LCD, porque la librería del MPLAB C18 se encarga de configurarlos (**Figura 20**).

Luego tenemos la configuración de la parte analógica del PIC, en la cual seteamos el **conversor A/D** y la entrada del canal donde se encuentra el sensor. Vamos a configurar el módulo A/D para tomar la

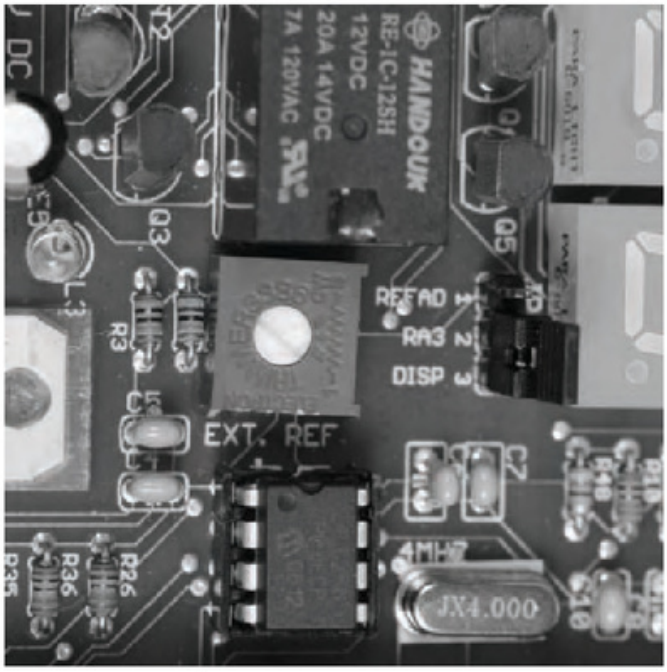
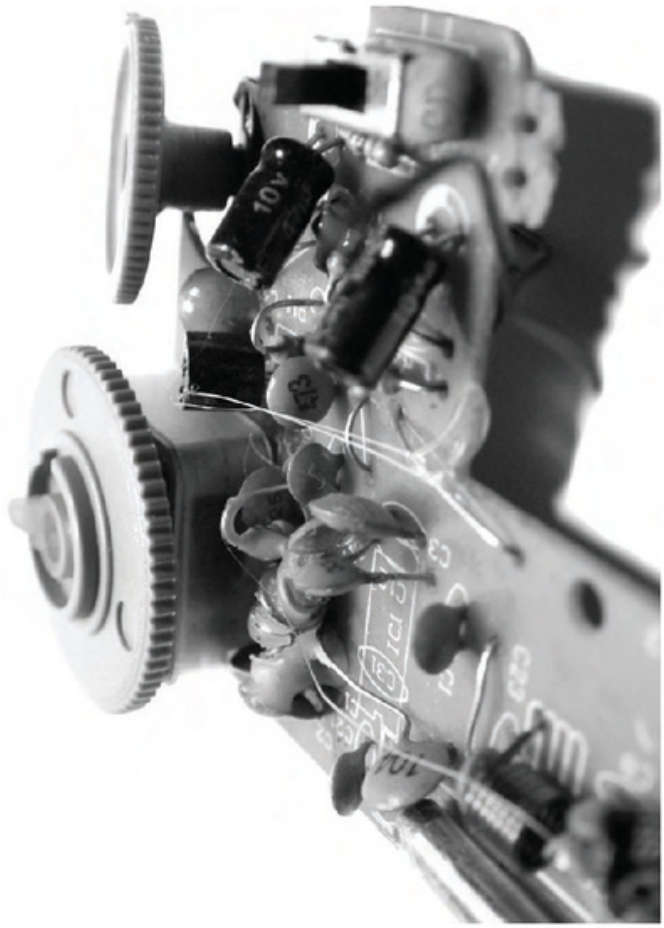


FIGURA 20. Éste es el relay conectado al pin RE0 que utilizaremos como salida para controlar algún sistema de refrigeración o alguna bocina que funcione como alarma.



señal analógica del canal 4 (RA5), realizar la conversión y hacer que la tensión de referencia sea el nivel de tensión en que se encuentre el pin **RA3/Vref.** Es por eso que allí colocamos un potenciómetro que nos permita calibrar dicha referencia (**Figura 21**).

Para configurar este módulo, utilizamos tres registros: **ADCON0**, **ADCON1** y **ADCON2**. Con **ADCON0** seleccionamos el canal de entrada para la conversión –en este caso, el 4 (RA5)– y habilitamos el módulo para poder utilizarlo. Con **ADCON1** configuramos los primeros cinco canales del PIC (AN0, AN1, AN2, AN3 y AN4), con el fin de usarlos como entradas analógicas (sólo utilizaremos el 4). El nivel de referencia positivo será la tensión en el pin **RA3/Vref**, en tanto que la otra referencia será **GND**.

Con **ADCON2** elegimos una justificación a izquierda del resultado (para que los 8 bits más significativos de la conversión queden en el registro ADRESH) y el tiempo de adquisición más alto.

CONFIGURACIÓN DEL LCD

Para usar el LCD modificamos la disposición de los pines que se conectarán a él, que están definidos en el archivo **`xlcd.h`**. Si utilizamos el esquemático de la **Figura 12**, los cambiamos tal como se mostró en paginas anteriores. Otro tema es la conexión del pin R/W: si lo colocamos a masa como en el esquemático, debemos modificar la función **`BusyXLCD`** para que sólo genere un retardo de 1 mseg, en vez de realizar el chequeo del busy flag. Esto lo hacemos así porque, al no estar conectado el R/W, no podemos leer datos del LCD; siempre vamos a escribir, por lo cual no podemos leer el estado del pin D7 para saber si el bus de datos está desocupado.

ALARMA TÉRMICA EM MPLAB C18

En el **Paso a paso 1** convertiremos en código toda la teoría vista para armar este proyecto. Estamos preparados para pasar todo lo que estuvimos estudiando sobre este proyecto a código fuente. Ya tenemos en claro cuál es la tarea que debe realizar la alarma térmica; tenemos el diagrama de flujo y el esquemático, y sabemos cuáles son los periféricos con los que el microcontrolador tendrá que interactuar. Ahora, volquemos toda esta teoría al código fuente.

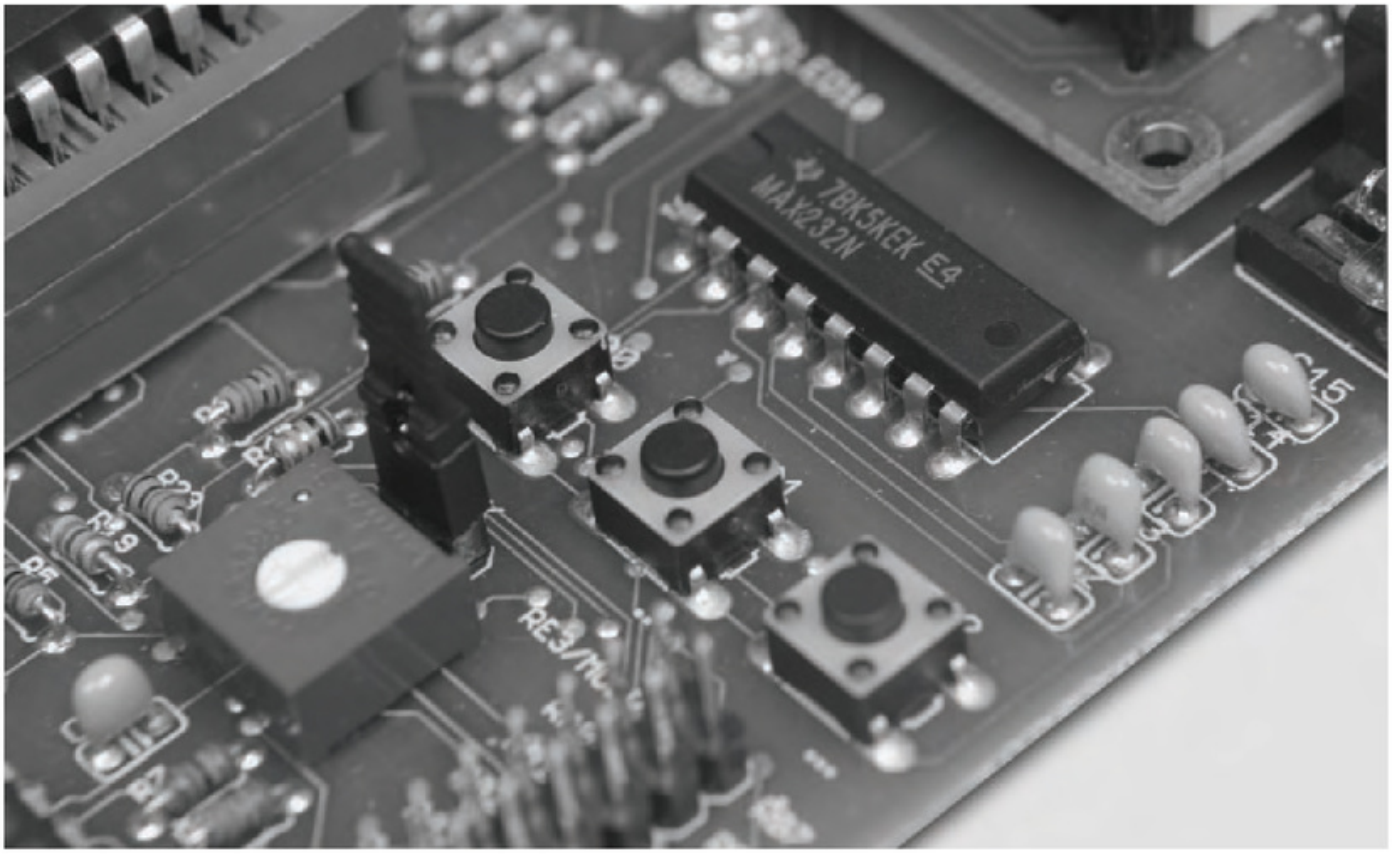


FIGURA 21. **Con el potenciómetro podemos emular una entrada analógica al** convesor A/D **del PIC.**

PASO A PASO /1
Alarma térmica en MPLAB C18

1

El encabezado del código incluye todos los archivos que necesitará el programa. Algunos contienen funciones útiles, como **`stdlib.h`** e **`itoa`**. Luego, defina algunas constantes y variables globales para almacenar el valor de la temperatura, tanto en float como en ASCII. Coloque los prototipos de las funciones que se definen en el código.

2

Entre en el programa principal y realice el código según lo indica el diagrama de flujo, inicializando los puertos utilizados, el LCD y la función **`ADC_Init()`**; para configurar el conversor. Luego, dentro del bucle **`while`**, la función **`ADC_Convert()`**; realiza la conversión A/D, y el resultado se lo asigna a **`Conversion`**, para después calcular la temperatura.

PASO A PASO /1 (cont.)

3

C:\Alarma Termica\Alarma termica.c

```
// La siguiente rutina convierte la variable float
// en ASCII y lo envia al LCD
Temp_LCD(Temp);

// Temp es mayor al umbral???
if (Temp >= (Temp_critica - Umbral))
{   // si, entonces preguntamos
    // Temp es mayor a la critica???
    if (Temp >= Temp_critica)
    {   // Si
        LATEbits.LATE0 = 1; // Activo Rele
        // Comando para colocarnos en la primera
        // linea del LCD
        WriteCmdXLCD(LINE_0);
        putrsXLCD(" Rele activado  ");
    }else
    {   // Temp es menor a la Temp_critica
        LATEbits.LATE0 = 0; // Desactivo Rele
        // Comando para colocarnos en la primera
        // linea del LCD
```

Una vez que tiene la temperatura, la función **`Temp_LCD()`**; la toma, la convierte en ASCII y la envía al LCD. Luego están los bloques condicionales, que actúan según el diagrama de flujo y, dependiendo del valor de la temperatura, envían un mensaje al LCD para indicar el estado y controlar la temperatura crítica definida en la cabecera.

4

C:\Alarma Termica\Alarma termica.c

```
// Retardos que debemos definir para la libreria LCD

void DelayFor18TCY( void )
{
  Delay100TCYx(1);  // Retardo de 100uSeg
}

void DelayPORXLCD(void)
{
    Delay1KTCYx(15);     // Delay de 15mS
                         // ciclos = (TimeDelay * Fosc)/4
                         // ciclos = (15mS * 4Mhz)/4
                         // ciclos = 15000
}

void DelayXLCD(void)
{
```

Éstos son los retardos que debe definir para que la librería del LCD del MPLAB C18 pueda funcionar. Fueron realizados con las funciones de la librería Delays, que generan retardos de 100 ciclos de máquina o 1000 ciclos de máquina, como en este caso. Si el cristal externo es de 4 MHz, cada ciclo de máquina equivale a 1 useg.

PASO A PASO /1 (cont.)

5

C:\Alarma Termica\Alarma termica.c

```
void ADC_Init(void)
{ // Inicializamos el conversor Analogico-Digital
    // Comenzamos configurando cuales van a ser los pines analogicos
    // y digitales.

    /*
    La siguiente configuracion coloca la referencia en VREF+ (AN3)
    para poder colocar la referencia con el pote a 1,76V
    y configura los 4 primeros canales como analogicos
    */
    ADCON1 = 0b00011010;

    /* Modificando las referencias */
    //Las tensiones de referencia seran VDD y VSS
    // y configuramos los 4 primeros canales como analogicos
    //ADCON1 = 0b00001010;

    // Justificamos a izquierda, elegimos un tiempo de adquisicion
    // de 20TAD y FOSC/2
    ADCON2 = 0b00111000;

    // Seleccionamos canal 4(RA5), allí encontramos el sensor
    ADCON0 = 0b00010001;
}
```

La función **`ADC_Init()`**; configura el conversor como se pidió en páginas anteriores. Observe que modifica los valores de los registros ADCON por configurar. Por su parte, **`ADC_Convert()`** realiza la conversión. Primero activa el módulo poniendo a 1 el bit **`GO_DONE`** de ADCON0, y espera a que se termine. Luego, devuelve el resultado del registro **`ADRESH`**.

6

C:\Alarma Termica\Alarma termica.c

```
void Temp_LCD(float fTemp)
{   // Para poder mostrarlo en el LCD se debe pasarlo a ASCII
    // Lo que realizamos es una conversion de float a int para
    // quedarnos primero con la parte entera
    Entero = (int)fTemp;
    // Enviamos ese valor al LCD
    WriteCmdXLCD(NEXT_LINE + 4);// Nos posicionamos en el LCD
    // en la sugnda fila y en la posicion 4

    itoa(Entero,Temperatura);   // Realizo conversion en ASCII

    putsXLCD(Temperatura);      // y lo enviamos al LCD
    // Luego escribimos el punto que separa el entero de las
    // decimas
    WriteDataXLCD('.');

    // y convertimos las decimas en ASCII
    fTemp = (fTemp - (int)fTemp) * Decimales_precision;
    Decimas=(int)fTemp;
```

Esta función toma el valor de la temperatura en float, lo convierte en ASCII y lo manda al LCD, del siguiente modo: primero, toma la parte entera de **`fTemp`** (float) y la convierte en ASCII con la función **`itoa`**, lo manda al LCD junto con el punto decimal y, después, despeja los decimales en una variable entera, lo convierte y lo envía al LCD.

PASO A PASO /1 (cont.)

7

C:\Alarma Termica\Alarma termica.c

```
/* Prototipos */
void ADC_Init(void);
unsigned char ADC_Convert(void);
void Temp_LCD(float fTemp);

/* Programa Principal ***/

void main(void)
{
    Temp = 0;   // Inicializo

    TRISEbits.TRISE0 = 0;   // RE0 salida

    OpenXLCD( FOUR_BIT & LINES_5X7); //Inicializamos LCD
    // Configuramos LCD con 4 bits de datos de varias lineas
    // con matriz de caracter de 5x7

    putrsXLCD(" Alarma Termica  "); // Enviamos mensaje al LCD

    // Inicializamos conversor ADC
    ADC_Init();

    while(1)
    {
        // comenzamos conviertendo el valor leido en el conversor
        // en grados dentro de una variable del tipo float
        Conversion = ADC_Convert();
```

Una vez finalizado el código fuente, cree el proyecto **Alarma térmica** en el MPLAB, desde **`Project Wizard`**. En él tiene que incluir los códigos fuente de la librería del LCD y el archivo **`xlcd.h`** que modificó, porque si lo compila sin tener estos archivos, surgirán errores. El código ya está listo para simularlo y grabarlo en el PIC.

Luego que terminamos de compilar el proyecto, podemos simularlo desde el MPLAB con la herramienta **MPLAB SIM**. Cuando llegamos al momento de la conversión, vemos que el conversor se toma un tiempo para entregar un resultado, que siempre será 0. Si queremos simular el resultado de una conversión, la manera más sencilla de hacerlo es modificar el registro **ADRESH**. Según sea el valor de la conversión, debemos verificar que los bloques condicionales realizan su tarea de manera perfecta. Cuando estamos seguros de que el código funciona, podemos poner en práctica el código en cuestión.

Grabamos el PIC, quitándolo de la placa experimental; luego, volvemos a colocarlo. En cambio, si contamos con un programador comercial que maneje in-circuit, la placa posee un conector **ICSP** para grabarlo sin necesidad de quitar el PIC.

A continuación, configuramos el potenciómetro **VREF** de la placa, que entregará el nivel de referencia positivo del conversor **A/D** del PIC. Para hacerlo, nos dirigimos al potenciómetro **de referencia del conversor** y, con un multímetro, medimos la tensión del pin **RA3** hasta que llegue a **1,75 Volts**. Con esto, ya tenemos la referencia externa calibrada (**Figuras 22** y **23**).

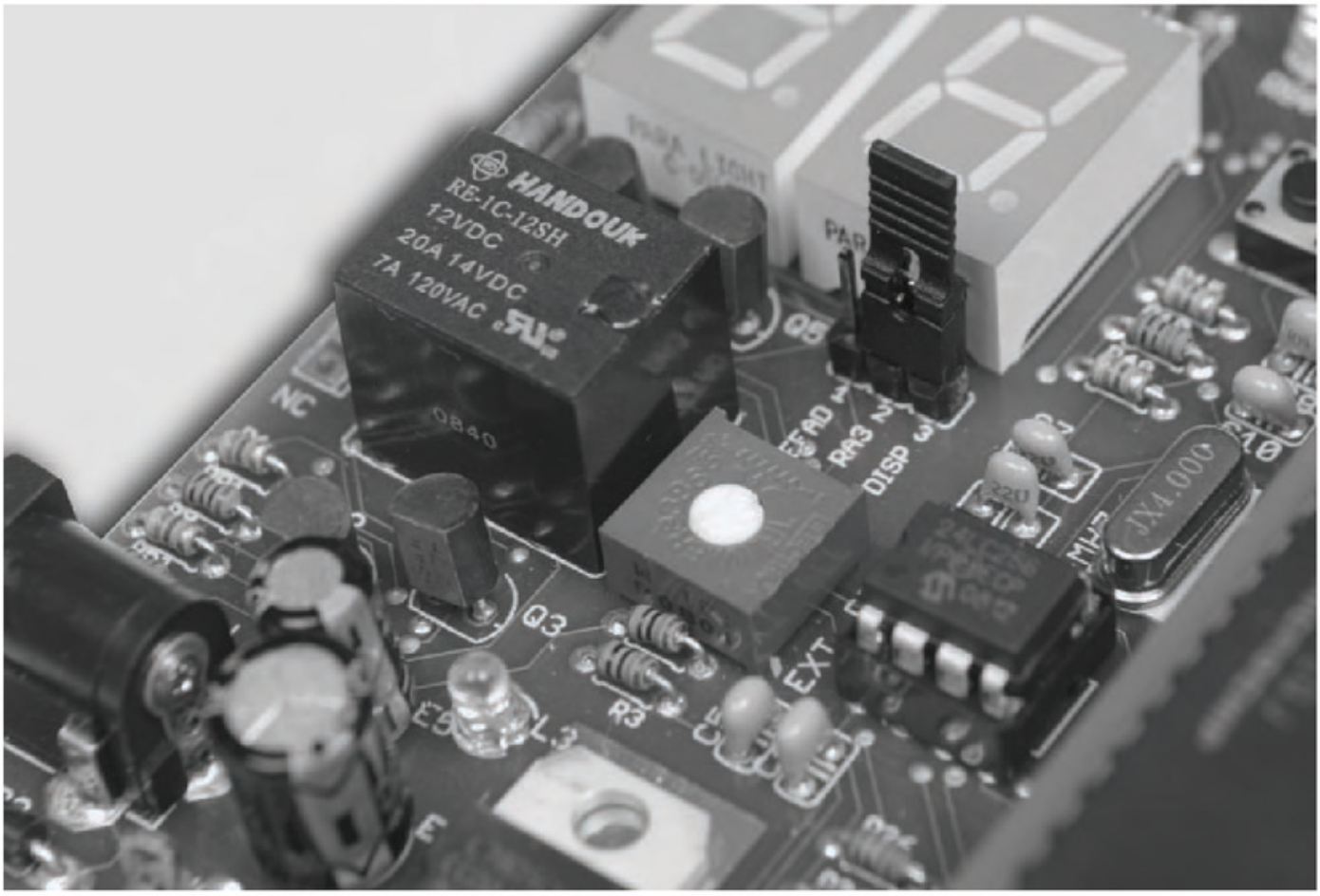


FIGURA 22. **Con el** potenciómetro de referencia**, podemos aplicar una referencia externa al conversor A/D del microcontrolador PIC.**

Finalmente, tenemos que aplicar alimentación a la placa y verificar su funcionamiento. Si queremos simular su accionar de un modo más rápido, podemos cambiar el sensor de temperatura por un potenciómetro de **10 KΩ** en el pin **RA5**, y con un destornillador, modificar el valor del pote para verificar el correcto funcionamiento de la práctica.

FIGURA 23. **Debemos utilizar un** tester digital **en la posición de tensión continua para medir la referencia externa que se aplica en el pin** RA3**; el valor debe ser de** 1,75 V **para nuestra práctica.**

Multiple **choice**

1 ¿Cuál de las siguientes funciones inicializa el LCD?

a- putsXLCD.
b- SetDDRamAddr.
c- OpenXLCD.
d- WriteDataXLCD.

2 ¿Cuál de las siguientes funciones establece la dirección de la visualización de caracteres?

a- putsXLCD.
b- SetDDRamAddr.
c- OpenXLCD.
d- WriteDataXLCD.

3 ¿Cuál de las siguientes funciones escribe un byte en el LCD?

a- putsXLCD.
b- SetDDRamAddr.
c- OpenXLCD.
d- WriteDataXLCD.

4 ¿Cuál de las siguientes funciones escribe un string de la memoria de datos en el LCD?

a- putsXLCD.
b- SetDDRamAddr.
c- OpenXLCD.
d- WriteDataXLCD.

5 ¿Qué significa la terminal E dentro de los pines del control del LCD?

a- Lectura/escritura.
b- Selector de registro.
c- Habilitado.
d- Ninguna de las opciones anteriores.

6 ¿Qué significa la terminal RS dentro de los pines del control del LCD?

a- Lectura/escritura.
b- Selector de registro.
c- Habilitado.
d- Ninguna de las opciones anteriores.

Respuestas: 1 c, 2 b, 3 d, 4 a, 5 c, 6 b.

Capítulo 3

Conectividad no inalámbrica

Veremos las diferentes opciones disponibles para conectar un sistema embebido a nuestra computadora.

RedUSERS.com

Sistemas embebidos

Un sistema embebido es un dispositivo controlado por un procesador, dedicado a realizar una única tarea o una serie de ellas. Un módem, por ejemplo, es un sistema embebido que maneja tareas de comunicación a través de la línea telefónica. Algunos sistemas embebidos son únicos en su tipo o se construyen para proyectos específicos.

Los microcontroladores se encuentran presentes en numerosos sistemas embebidos. Para realizar sus funciones, necesitan el apoyo de otros dispositivos digitales, como puede ser una computadora personal, una memoria o un display LCD, por nombrar sólo algunos. Con este objetivo, precisa comunicarse con ellos, es decir, transmitir y recibir información relevante al desempeño de las tareas para las cuales fue diseñado. Debido a esta necesidad de comunicación, se desarrollaron diversos protocolos, y en esta clase veremos, específicamente, los de conectividad no inalámbrica.

Protocolos **de conectividad**

SPI, **microwire**, **I2C** y **SMBus** son protocolos de interconexión utilizados en comunicaciones de microcontroladores con periféricos, memorias **EEPROM**, **displays LCD** u otro tipo de **circuitos integrados** que requieran una intercomunicación digital. Estas interfaces tienen un formato de transmisión serie y se encuentran, principalmente, en sistemas embebidos.

La computadora personal o PC, omnipresente en nuestra vida, es actualmente la herramienta tecnológica más popular junto con el teléfono móvil. Nos permite realizar todo tipo de tareas y nos ofrece una capacidad de procesamiento de datos cada vez más poderosa. Además, el avance de Internet la ha transformado en un puente de conexión con el mundo. Por todo esto, sería deseable poder establecer una comunicación entre un dispositivo periférico (diseñado por nosotros) y la PC. La **Tabla 1** muestra algunas de las opciones disponibles al respecto. Además de las mencionadas, existen otras interfaces populares. El **protocolo MIDI**, por ejemplo, se emplea en la conexión de instrumentos musicales electrónicos con la PC.

COMUNICACIÓN SERIE ASÍNCRONA Y RS-232

Analizaremos la comunicación serie asíncrona, su aplicación en la conectividad de un sistema embebido con una PC y el estándar **RS-232**. Este tipo de comunicación define una manera de transmitir la información de a un bit por vez, en un formato determinado, a una velocidad acordada de antemano por las partes. En general, existen muchos protocolos de comunicación serie que, además, son asíncronos (**Figura 1**). En particular, aquí estudiaremos

Conectar un dispositivo a la PC nos permite aprovechar todas las ventajas que ésta ofrece

INTERFAZ	FORMATO	NÚMERO MÁXIMO DE DISPOSITIVOS	DISTANCIA MÁXIMA	VELOCIDAD MÁXIMA (BITS/SEGUNDO)	USO TÍPICO
RS-232	Serie asíncrono	2	15 m	19200 a 15 m	Módem, mouse instrumentación
RS-485	Serie asíncrono (half-duplex)	32	1200 m a 100 Kbps	10 M a 12 m	Adquisición de datos y sistemas de control
USB	Serie asíncrono	127	5 m	1,5 M; 12 M; 480 M; 4,8 G (USB 3.0)	Mouse, teclado, memoria, audio, impresora, periférico personalizado
Ethernet	Serie		100 m par trenzado	10/100/1000 M	Comunicaciones de redes en general
Puerto paralelo	Paralelo	2	3-9 m	8 M	Impresora, escáner
IEEE-1394 (Firewire)	Serie	64	90 m	3,2 G	Video, audio, dispositivos de almacenamiento
IEEE-488 (GPIB)	Paralelo	15	18 m	8 M	Instrumentación

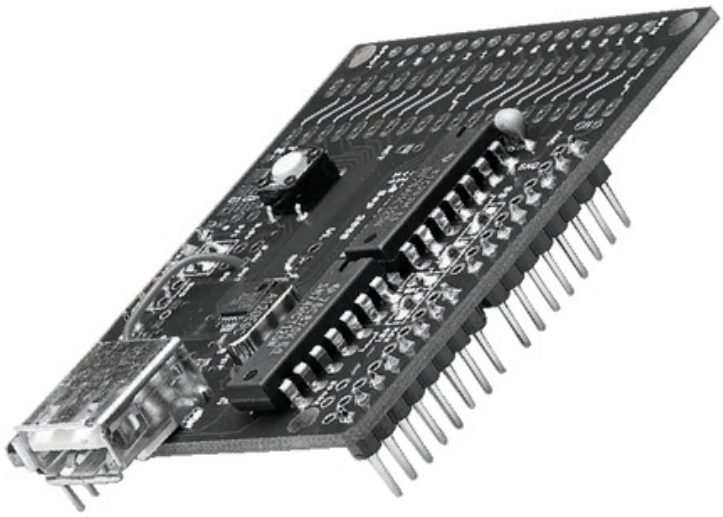
TABLA 1. **Existen múltiples opciones de conectividad no inalámbrica con la PC; en la tabla se presentan las más importantes.**

FIGURA 1. **La** conectividad USB **nos abre un nuevo mundo de posibilidades y nos permite ampliar las características y funciones de nuestros periféricos.**

lo relacionado con la comunicación elemental sobre la base de un puerto serie en una **PC** y una **UART** (*Universal Asynchronous Receiver/Transmitter*) en un microcontrolador.

LA COMUNICACIÓN

El concepto de **comunicación** es muy amplio; sin embargo, en este caso nos referiremos a la acción de conectar dos dispositivos electrónicos por un medio físico. En este sentido, la comunicación es **bidireccional** y, vista desde el microcontrolador, se realiza de la siguiente manera:



- En **ausencia de información**, la señal se mantiene en el estado lógico alto.
- Un carácter de información se inicia con un bit de **start**, cuya misión es indicar al receptor que la información sigue a continuación.
- El dato de información se envía con el bit menos significativo primero, y puede contener desde 5 hasta 8 bits. En algunos casos, se manda un noveno bit para aplicaciones particulares.
- De manera opcional, pero acordado de antemano, se envía un bit de paridad. Éste se calcula emparejando la cantidad de unos en el carácter, para que sea par o impar.
- El carácter se termina con 1 o 2 bits de **stop**, cuya finalidad es volver la línea de transmisión al estado de reposo durante un tiempo suficiente como para que el receptor pueda identificar el próximo bit de **start**. La **Figura 2** muestra el envío del carácter 'a' (0x61) en 8 bits.

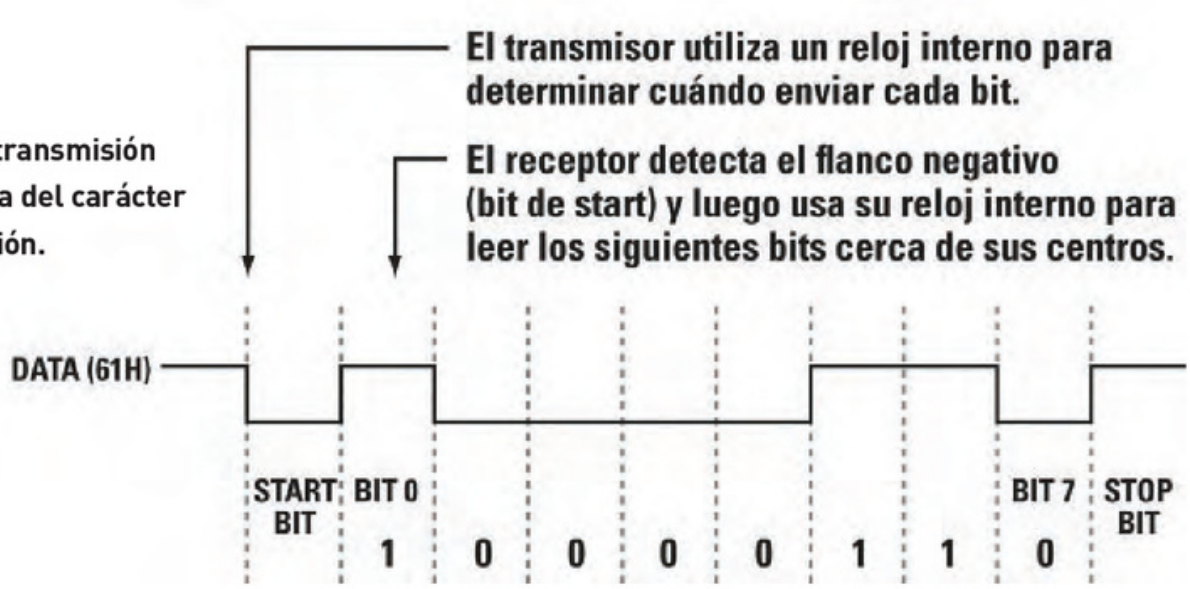


FIGURA 2. La transmisión serie asíncrona del carácter 'a' y su recepción.

FIGURA 3. La mayoría de las nuevas PCs no poseen puerto serie en su hardware. Sin embargo, es posible agregarlo mediante un convertidor de USB a serie.

Es importante aclarar que **asíncrono** significa que no utiliza una señal de reloj para la transmisión y recepción de datos. Los **relojes internos** de **transmisor** y **receptor** deben estar lo suficientemente próximos y mantener una estabilidad tal, que permita la comunicación de un carácter. Por lo general, el receptor sincroniza su reloj con el flanco descendente del bit de start y observa los datos en el centro de cada bit de información.

PUERTO SERIE

El puerto serie es la interfaz en una computadora u otro dispositivo que transmite datos de a un bit por vez. En el uso convencional, el término **puerto serie** se refiere a aquél que utiliza este protocolo **asíncrono** que acabamos de ver. Es bidireccional, ya que puede enviar y recibir información. En muchas ocasiones, podemos pensar que transmitir un bit a la vez es ineficiente para las comunicaciones actuales; sin embargo, este sistema tiene sus ventajas, como la posibilidad de utilizar cables de bajo costo y conectores pequeños (**Figura 3**). En una PC, las aplicaciones de software acceden al puerto serie mediante el llamado **COM port**.

EL ESTÁNDAR RS-232

El estándar V.24 de la **ITU-T** (*International Telecommunications Union, Telecommunications Standardization Sector*) define 25 líneas de conexión entre un **DTE** (*Data Terminal Equipment*, una computadora) y un **DCE** (*Data Communications Equipment*, un módem o periférico).

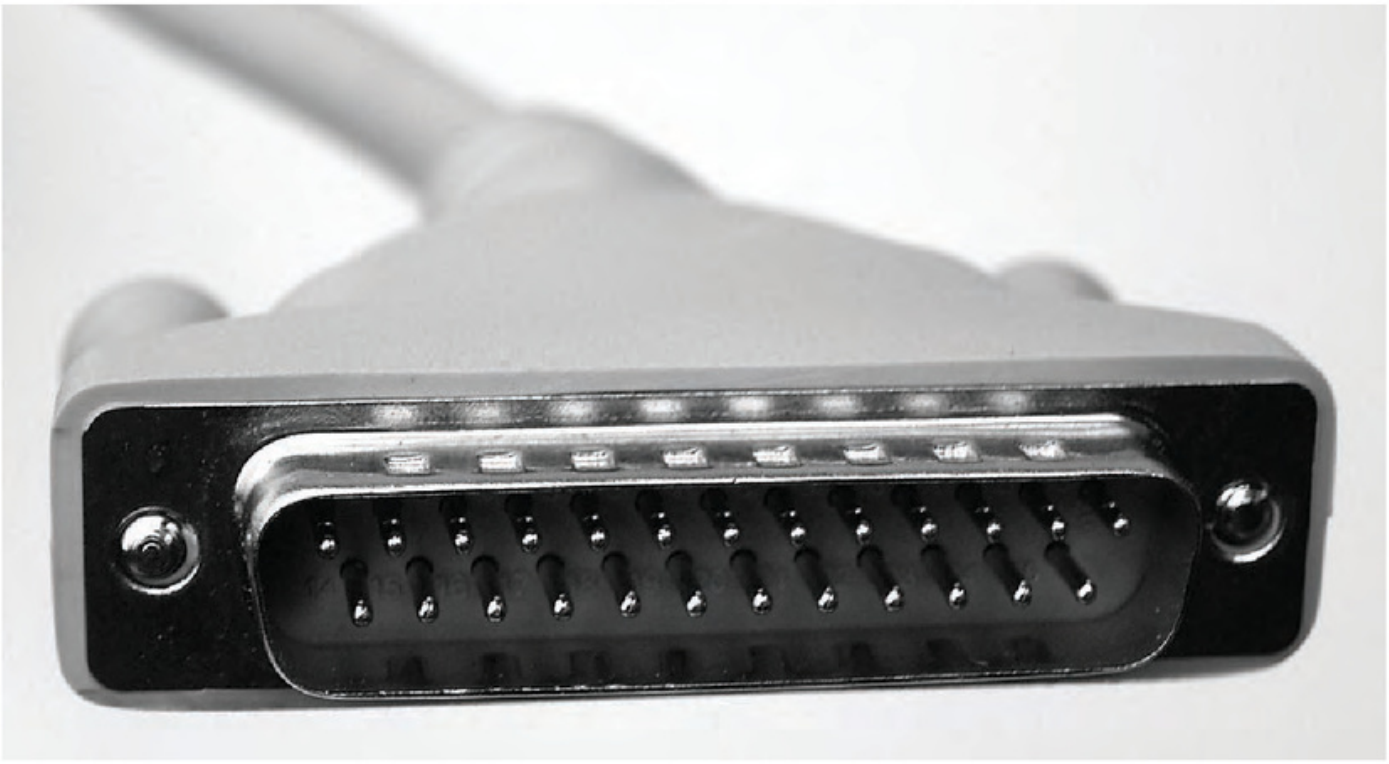


FIGURA 4. El estándar RS-232 utiliza un conector tipo DB-25 (de 25 pines) o DB-9 (de 9 pines).

El estándar **RS-232** de la **EIA/TIA** (*Electronics/Telecommunications Industries Association*) determina los niveles de tensión para realizar la comunicación (**Figura 4**). Existen variantes que fijan el nivel máximo, pero la mayoría coincide en aceptar lo siguiente:

- Un **0** lógico se representa con un nivel de tensión mayor a **3 V**.
- Un **1** lógico se representa con un nivel de tensión menor a **-3 V**.

La distancia máxima es de **15 m,** y la velocidad máxima a esa distancia es de **19200 bps**; velocidades de hasta **115200 bps** o más son posibles a distancias menores. La comunicación es por cable convencional, con referencia de masa (no-balanceado). Las señales más comunes de **RS 232** pueden apreciarse en la **Tabla 2**. Existe un modo de conexión muy utilizado, denominado **de 3 cables** (*3-wire*), que emplea sólo **TD**, **RD** y **GND**.

USO DE LA UART DEL PIC18F4620

Este microcontrolador posee un módulo denominado **EUSART**, que es una UART con capacidad de funcionar, además, en modo sincrónico. El módulo se habilita mediante el bit **SPEN** en el registro **RCSTA**. La trans-

El puerto serie es la interfaz en una PC u otro dispositivo que transmite datos de a un bit por vez

NOMBRE	PIN (25)	PIN (9)	SENTIDO	FUNCIÓN
TD (Transmit Data)	2	3	DTE→DCE	Datos por transmitir
RD (Received Data)	3	2	DTE→DCE	Datos recibidos
RTS (Request To Send)	4	7	DTE→DCE	El terminal desea transmitir
CTS (Clear To Send)	5	8	DTE→DCE	El terminal puede transmitir
DTR (Data Terminal Ready)	20	4	DTE→DCE	El terminal está operacional
DSR (Data Set Ready)	6	6	DTE→DCE	El módem está operacional
DCD (Data Carrier Detect)	8	1	DTE→DCE	El módem recibe portadora del módem remoto
RI (Ring Indicator)	22	9	DTE→DCE	Se recibe un llamado por la línea telefónica
GND (Ground)	7	5	COMÚN	

TABLA 2. **Algunas de las señales más comunes en una comunicación asíncrona.**

misión se efectúa a través del registro **TXREG**, y la recepción, mediante el **RCREG**. Existen dos registros de estado, **TXSTA** y **RCSTA**, que nos permiten controlar y conocer la actividad de la UART. La velocidad y forma de operación se controlan a través del registro **BAUDCON** y de un *Baud Rate Generator* (**BRG**) interno.

En **MPLAB C18**, disponemos de una función de incialización y de cuatro elementales para enviar y recibir información por la UART:

- **OpenUSART():** permite la configuración de la UART del microcontrolador, incluyendo velocidad e interrupciones. Configura un modo de trabajo sin interrupciones, en 8 bits por carácter, recepción continua a FOSC / (64 * (**spbrg** + 1)) bps. Por ejemplo:

```
OpenUSART( USART_TX_INT_OFF &
USART_RX_INT_OFF &
USART_ASYNCH_MODE & USART_EIGHT_BIT &
USART_CONT_RX & USART_BRGH_LOW, spbrg );
```

¿POR QUÉ UTILIZAR PUERTO SERIE?

Cuando el USB llegó al mercado, se pronosticó la desaparición del puerto serie. Muchos sistemas embebidos continúan usándolo porque su implementación es más económica y la programación, menos compleja. Además, permite el uso de cables de mayor longitud.

- **`WriteUSART()`**: escribe un byte en el buffer de transmisión de la UART.
- **`BusyUSART()`**: permite conocer si la UART está transmitiendo
- **`DataRdyUSART`**: permite conocer si se ha recibido un carácter.
- **`ReadUSART()`**: lee un byte del buffer de recepción de la UART.

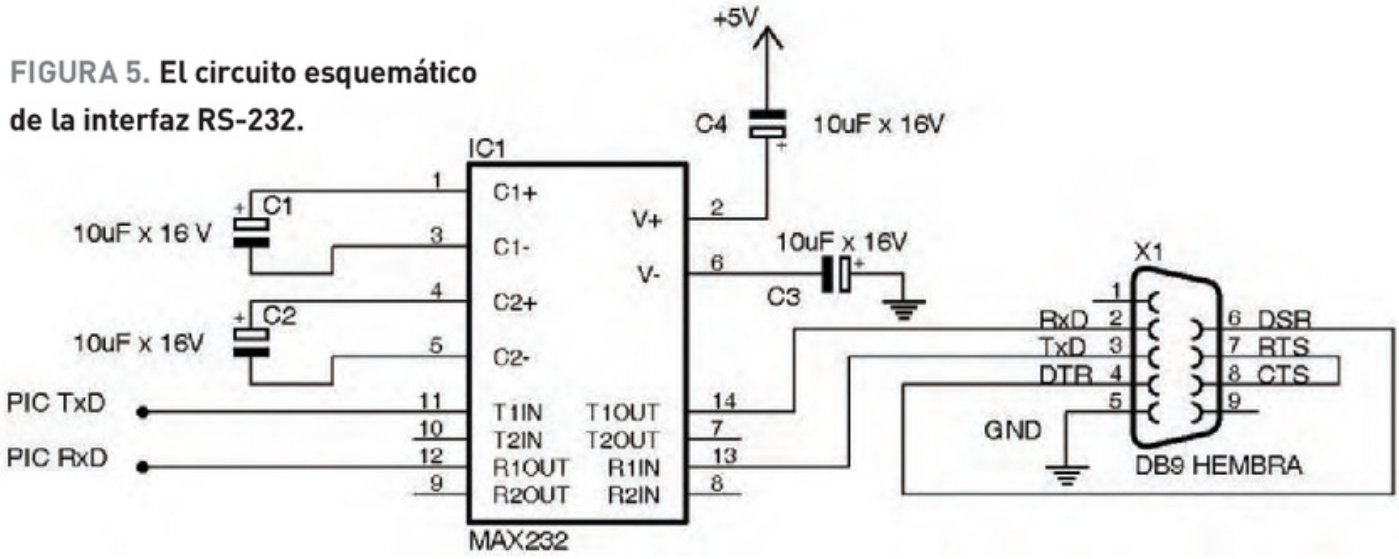
La operatoria en transmisión es verificar si la UART ha terminado de transmitir y, luego, enviar un carácter. En recepción, verificamos si hay uno disponible y, a continuación, llamamos a la función que nos lo entrega.

INTERFAZ RS-232

En la **Figura 5** vemos el circuito esquemático de una interfaz RS-232 genérica para nuestros proyectos. Se basa en el **MAX232**, que incluye dos excitadores para convertir las entradas de niveles TTL a RS-232 y dos receptores que se encargan de transformar entradas RS-232 en salidas a niveles TTL (**Figura 6**).

FIGURA 6. En el sitio web de Microchip, www.microchip.com, podemos encontrar una versión de evaluación del MPLAB C18.

FIGURA 5. El circuito esquemático de la interfaz RS-232.



Docklight

Docklight es un software para verificación, análisis y simulación de protocolos de comunicación serie (**RS-232**, **RS-485** y otros).

Veamos cómo utilizar este programa permite realizar el monitoreo de la comunicación serie entre dos dispositivos o simular el comportamiento de uno de ellos para verificar el comportamiento del otro. Las funciones de este software son las siguientes:

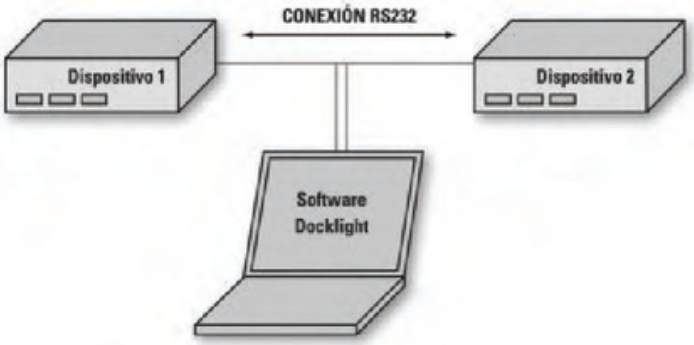


FIGURA 7. **Es posible insertar Docklight entre la comunicación de dos dispositivos serie para monitoreo. Debemos poseer dos puertos COM en la PC.**

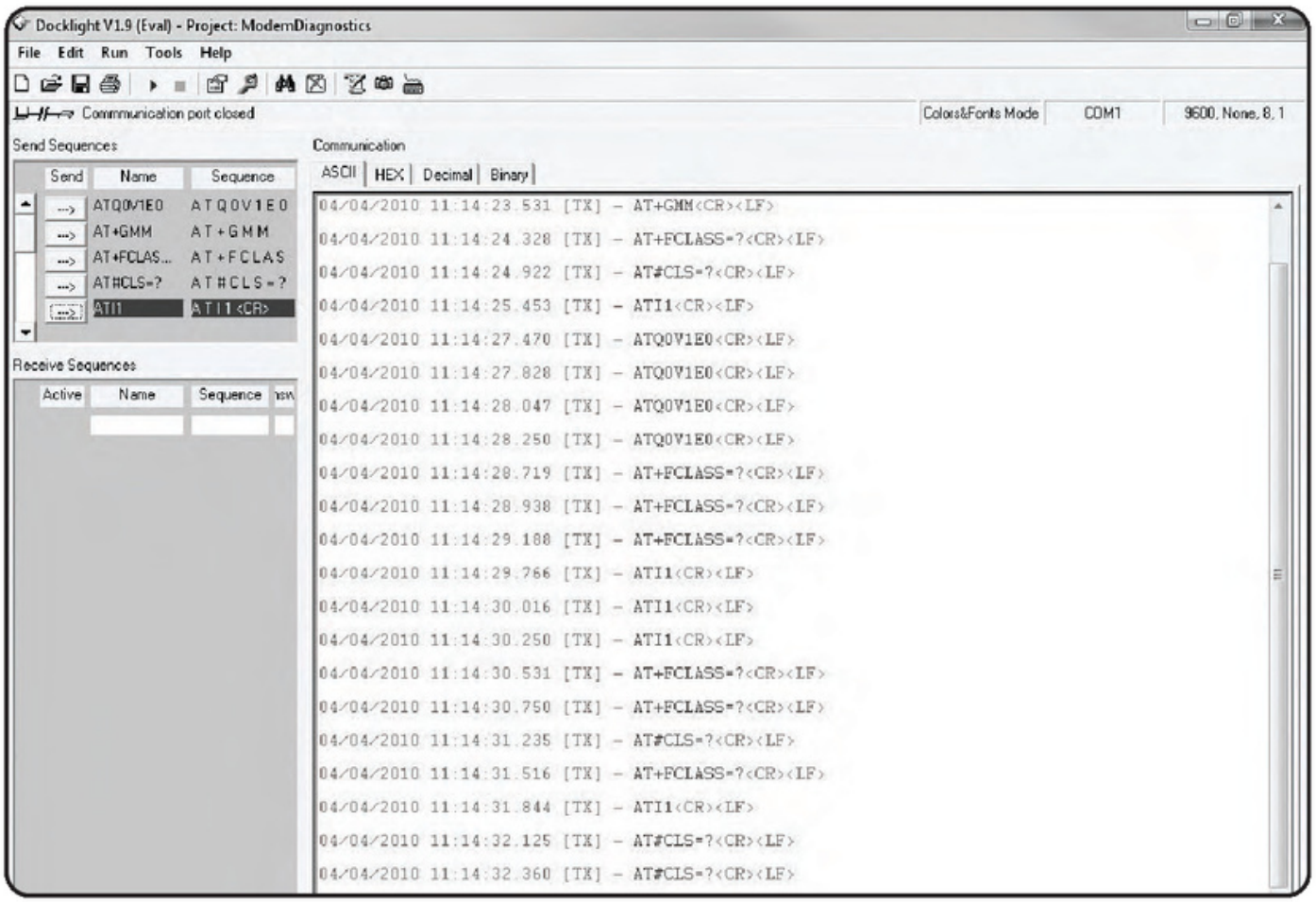


FIGURA 8. **Con Docklight podemos verificar la implementación del protocolo serie en un dispositivo. Es posible definir una secuencia de control reconocible por el dispositivo y analizar las respuestas que éste genera.**

• **Simular un protocolo serie**. Docklight puede enviar secuencias de datos definidas por el usuario según el protocolo utilizado; además, es capaz de actuar según las secuencias que reciba. Esto hace posible simular el comportamiento de un dispositivo serie, algo particularmente útil para generar condiciones de verificación que son difíciles de reproducir con el dispositivo original.

• **Registrar datos de RS-232**. Cualquier comunicación serie puede ser registrada utilizando dos formatos de archivos diferentes. Es posible emplear texto normal para registrar una gran cantidad de datos rápidamente, o utilizar un formato **HTML** con texto que permita distinguir fácilmente entre los datos entrantes y salientes.

• **Detectar secuencias de datos específicas**. En muchos casos, necesitamos verificar la ocurrencia de una secuencia específica dentro de la información en RS-232 que indica una condición de error. Docklight maneja una lista de estas secuencias, y es capaz de realizar acciones definidas por el usuario en caso de detectarlas.

• **Responder a datos entrantes**. Docklight da la posibilidad de especificar respuestas definidas por el usuario a diferentes secuencias recibidas. De esta manera, es posible construir un simulador básico para nuestro dispositivo serie, y rastrear errores enviando un comando de diagnóstico luego de recibir un mensaje de error.

Las celdas de una memoria PROM se construyen alrededor de fusibles, diodos y transistores

CÓMO UTILIZAR DOCKLIGHT

Este programa puede obtenerse desde su página web oficial: **www.docklight.de**. En el **Paso a paso 1** veremos la manera de configurar Docklight para enviar y reconocer secuencias, y operar manualmente. Esta última requiere de la activación del teclado, ya que el programa se orienta al envío de secuencias mediante un clic. Realizaremos una comunicación con un módulo **XBee** a través de un puerto serie virtual vía USB con un **chip FTDI**.

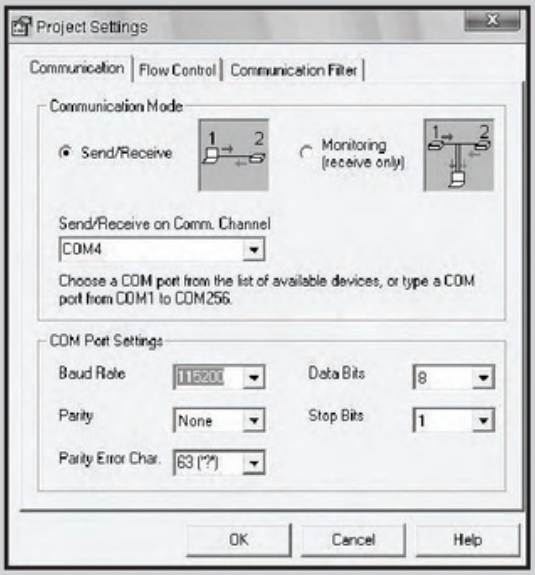
Docklight es un software de verificación, análisis y simulación de protocolos de comunicación serie

PASO A PASO /1
Utilizar Docklight

1

Al correr el programa por primera vez, aparece una ventana de registro. Como va a usarlo en modo evaluación, no debe registrarse.

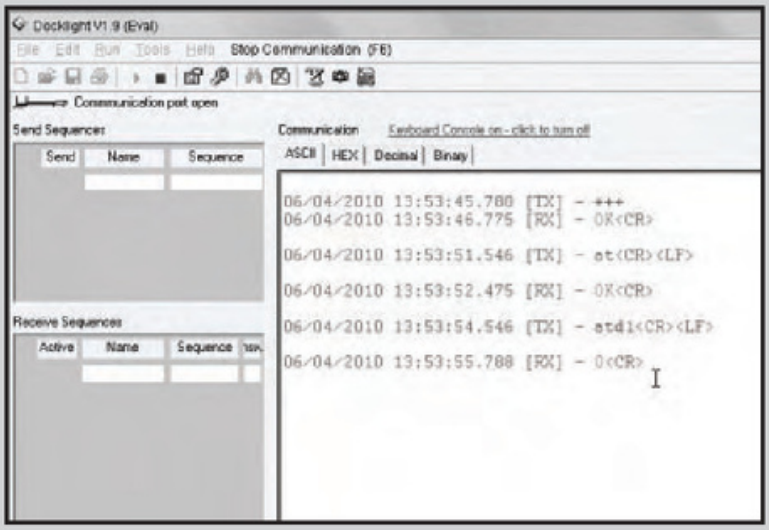
2



A continuación, se abre una ventana para elegir un proyecto. La versión de evaluación no permite guardar el trabajo, así que empezará con un proyecto nuevo. Lo primero que hará es configurar el puerto serie que va a usar. En este caso, se conectará a 115200 bps en el COM4. En el menú **`Tools`** seleccione **`Project Settings`** y, en la solapa **`Communications`**, configure **`Port`** y **`Speed`**.

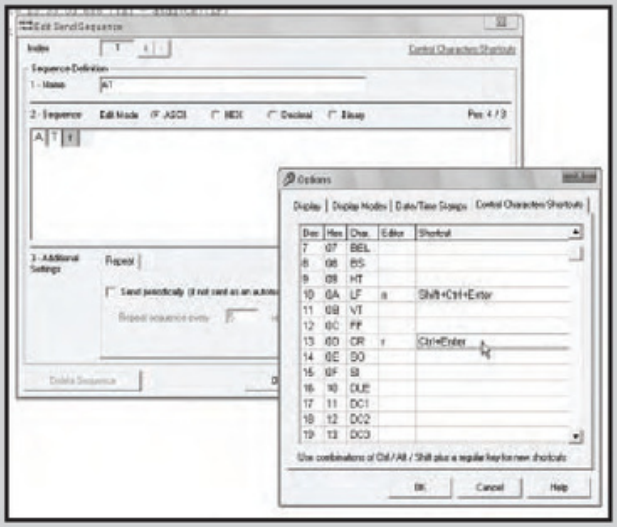
PASO A PASO /1 (cont.)

3



Para enviar caracteres manualmente, debe abrir el puerto y habilitar la operación de teclado. Con este fin, haga clic sobre el icono de reproducción y, luego, en el del teclado. Ingrese un texto, y vea que el programa muestra la fecha, la hora y el sentido de la comunicación.

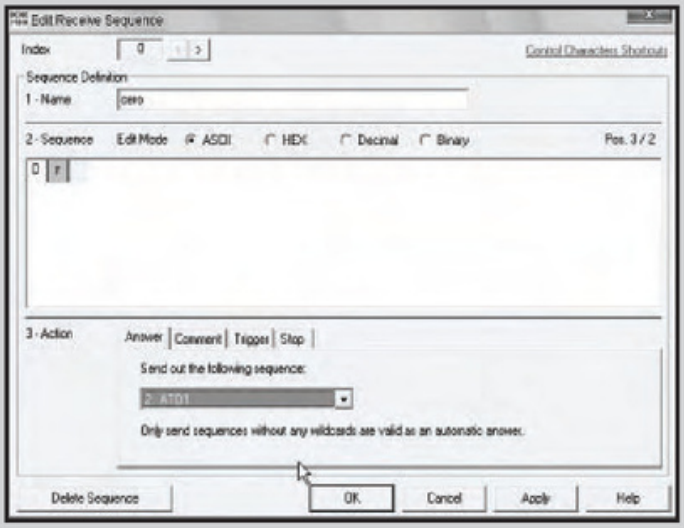
4



Puede definir una secuencia para enviar haciendo doble clic en la sección **Send sequences**. Ingrese los datos correspondientes y, si requiere caracteres de control (como **Enter**), puede pedir ayuda. Mande la secuencia haciendo clic sobre la flecha que está a la izquierda.

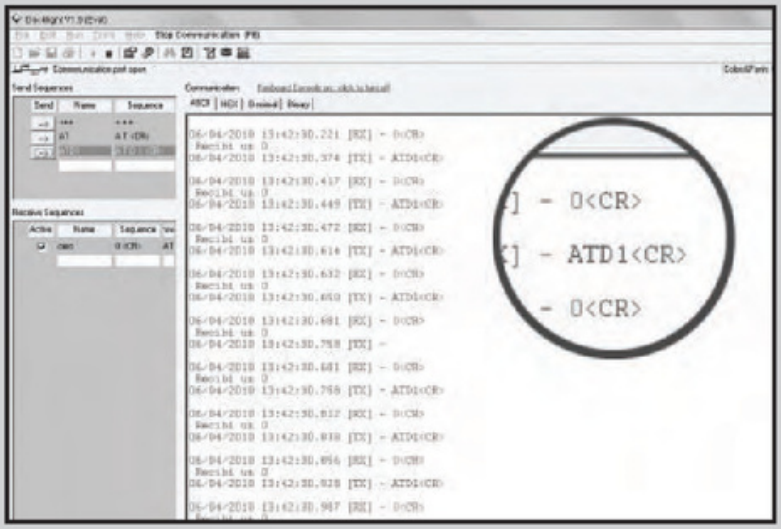
PASO A PASO /1 (cont.)

5



Puede definir una secuencia para recibir haciendo doble clic en la sección **`Receive sequences`**, para lo cual tiene que ingresar los datos correspondientes. Puede relacionarla con una respuesta por enviar o un comentario que se verá en pantalla junto a la información de fecha y hora.

6



Configure enviar **`ATD1`** cuando la respuesta es **`0<CR>`.** En la imagen se ve que el sistema funciona por sí solo ante un primer envío, que origina la primera respuesta e indica la recepción de la secuencia mediante el comentario introducido en el paso anterior: el texto "recibí un 0".

FIGURA 9. **Vemos aquí los conectores típicos del tipo A y B, utilizados corrientemente en dispositivos de interfaz USB.**

Universal **Serial BUS**

El puerto **USB** (*Universal Serial BUS*) es una interfaz de relativa complejidad. Una transferencia USB implica una comunicación serial, bidireccional y de índole "diferencial" sobre un cable compuesto por cuatro conductores: GND (común), VCC (alimentación) y un par diferencial balanceado para transporte de datos (**Figura 9**). Existen cuatro tipos de transferencias definidas para una comunicación USB:

Cada vez que un dispositivo USB se conecta a un host, se inicia un proceso de enumeración

- **Isocrónicas**: garantizan una velocidad de transferencia determinada, pero con posibles pérdidas de datos.
- **Del tipo interrupción**: esta opción de transferencias se utiliza cuando un dispositivo requiere garantizar transferencias rápidas.
- **Bulk**: este tipo realiza transferencias largas y esporádicas, que utilizan todo el ancho de banda remanente disponible, pero que no precisan de una entrega en un tiempo determinado. El **USB Host** le asigna la prioridad más baja a este tipo de transferencias.
- **De control**: usada para el envío de comandos cortos o información de estado al dispositivo.

Cada vez que un dispositivo USB se conecta a un host, se inicia un proceso de enumeración, que comienza mediante el envío de una señal de reset al dispositivo. Durante esta etapa, se determina la tasa de transferencia del periférico USB. El USB Host lee la información y le asigna una dirección única de 7 bits. Una comunicación USB está basada en el establecimiento de *pipes* o canales lógicos de conexión. Se denomina **endpoint** al bloque de memoria de datos o registro del chip controlador sobre el cual se establece la comunicación física. Es posible trabajar a distintas tasas de transferencia:

- **Show Speed**: 10 - 100 Kbps
- **Full Speed**: 500 Kbps - 10 Mbps
- **High Speed** (**USB 2.0**): 25 – 480 Mbps

Las diferentes tasas de transferencia involucran distintos niveles de tensión utilizados para la comunicación en el par diferencial. Cada intercambio involucra **tres paquetes**:

• Paquete tipo **token**: contiene la dirección del dispositivo (número de endpoint) para el direccionamiento.
• Paquete de **datos**: contiene los datos propiamente dichos de la transacción.
• Paquete de **terminación**: indica el fin de la transacción, de modo que pueda procesarse.

USB FÁCIL: FT2232D CHIP

El núcleo del **FT2232D**, fabricado por FTDI, está conformado por dos controladores que manejan el proceso de enumeración y otras comunicaciones sobre el bus USB (**Figura 10).** Cada uno ofrece distintos modos de configuración, que implementan la operatividad de otros dos chips de la familia: el **FT232B** y el **FT245B**:

• Una **interfaz full-speed USB-UART** (**asíncrona serial**) a través del módulo hardware **FT232BM** embebido.
• Una **interfaz full-speed paralela USB-FIFO** a través del módulo hardware **FT245BM** embebido.
• Una **interfaz serial sincrónica** (**USB-JTAG**, **USB-SPI** o **USB-I2C**) a través del módulo hardware configurable: **MPSSE** (*Multi-Protocol Serial Engine Interface*).
• Una **interfaz paralela compatible** con un **bus microcontrolador 8051** (**MCU** *Bus Host Emulation*).
• **Modos bit-bang sincrónicos** y **asíncronos**, de operación básica, donde no se necesita la conexión a una CPU externa. Se maneja directamente un bus E/S de 8 bits con el fin de controlar relays, LEDs o similares.

DRIVERS

Estos chips emplean drivers provistos por el fabricante, en dos opciones:

• **Virtual COM Port driver**: el dispositivo se ve como conectado a un puerto COM (RS-232). Trabaja bajo entorno Windows, y es utilizable por cualquier aplicación que emplee comunicaciones con un puerto COM.
• **D2XX Direct Driver:** para aplicaciones paralelo o en las que prevalezca la necesidad de velocidad, con funciones específicas para las aplicaciones que se comuniquen con el chip.

Algunas aplicaciones típicas son aquellas que necesitan manejar, a lo sumo, un puerto del tipo **bulk** o **isocrónico** para cada lado. El chip provee pines de comunicación con una memoria **microwire**, en la cual es posible grabar información como **Vendor ID** (identificador del fabricante), **Product ID** (identificador del producto), si es un circuito que se alimenta del bus o tiene alimentación externa (*self-powered*), etc.

HID

Human Interface Device es una clase de dispositivo USB especialmente diseñada para interactuar de manera directa con personas. Esta tecnología permitió innovar en dispositivos de entrada a computadoras y simplificar el procedimiento de instalación de éstos.

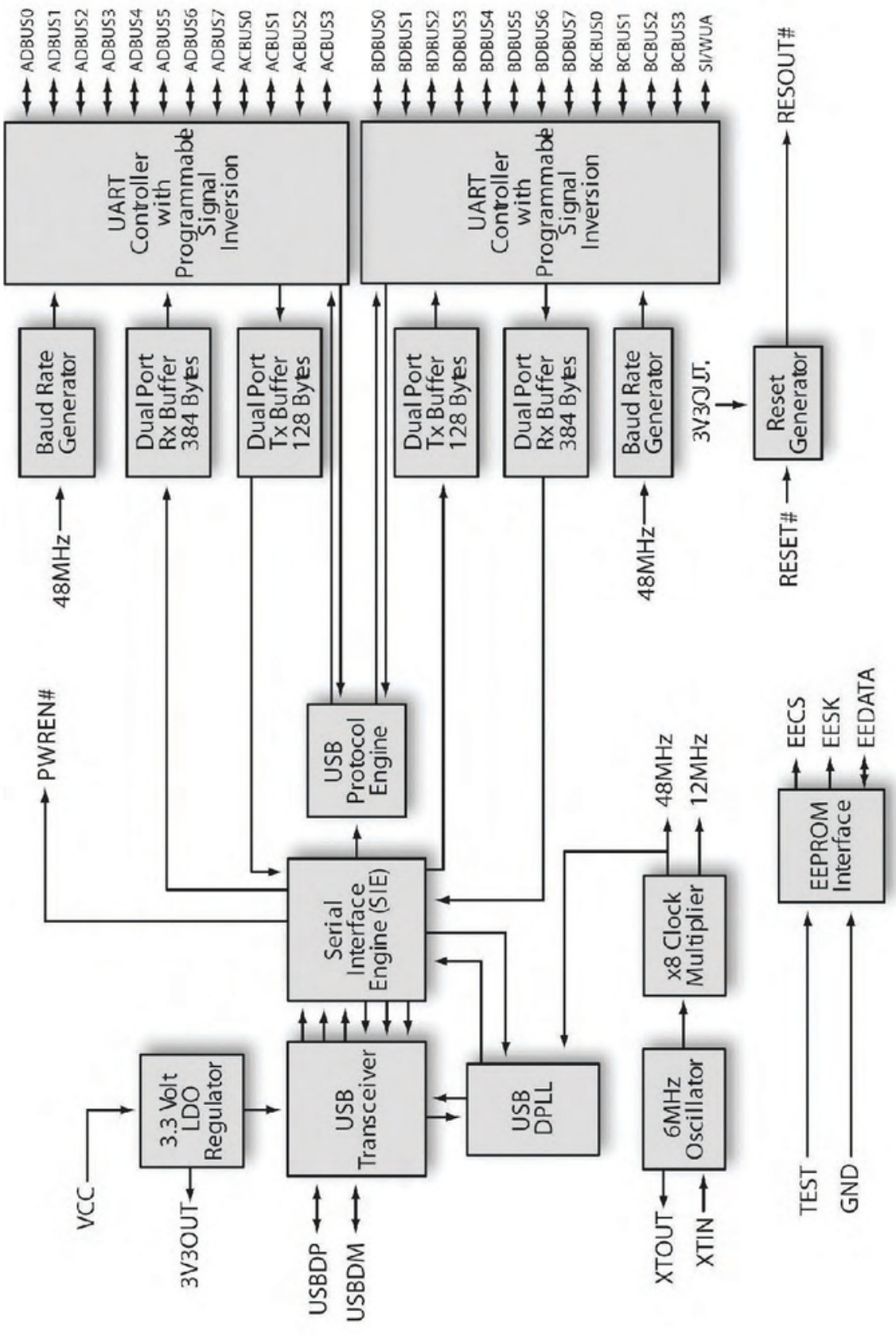


FIGURA 10. Diagrama en bloques del chip FT2232. Observamos la interfaz con el bus USB y con la memoria EEPROM de configuración, y los dos bloques principales que implementan sus puertos virtuales.

MODO FT245BM

El buffer de transmisión de esta interfaz es de **128 bytes**, en tanto que el de recepción es de 384 bytes de capacidad. Esta interfaz paralela, configurada en modo FT245BM, está conformada por **ocho líneas** de datos y **cuatro de control**:

- **RXF**: indica a la CPU conectada que tiene datos del host para leer.
- **RD**: es activada por la CPU para leer los datos.
- **TXE**: indica a la CPU que puede enviar datos al host si se lo requiere.
- **WR**: es activada por la CPU para escribir los datos.

MODO FT232BM

Este módulo convierte una transferencia USB en una del tipo serial asíncrona. Los tamaños de los buffers son exactamente los mismos que para el caso anterior. La velocidad de transferencia máxima que se puede alcanzar es de **300 Kbps** con un bit de **start** y uno de **stop**.

En la **Figura 11** se muestra la distribución y la función de los pines para una interfaz configurada en este modo. Las líneas implementadas son las correspondientes a una terminal DTE (*Data Terminal Equipment*), definidas en el estándar EIA/TIA-232, en tensiones de uso en lógica.

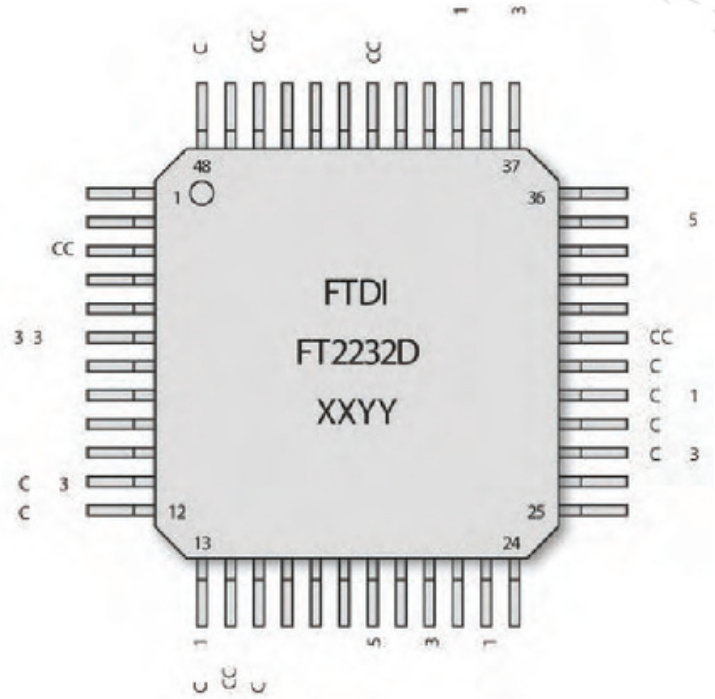


FIGURA 11. **Encapsulado LQFP de 48 pines del FTDI2232H.**

El FT232 provee un puerto serie vía USB. Lo conectado a este chip se ve como conectado a una UART

VENTAJAS DRAM

Esta tecnología permite, en un mismo espacio físico, lograr matrices de memoria con una capacidad total muy elevada por sobre una RAM convencional o estática. Otra de las ventajas es que el consumo de potencia es muy inferior.

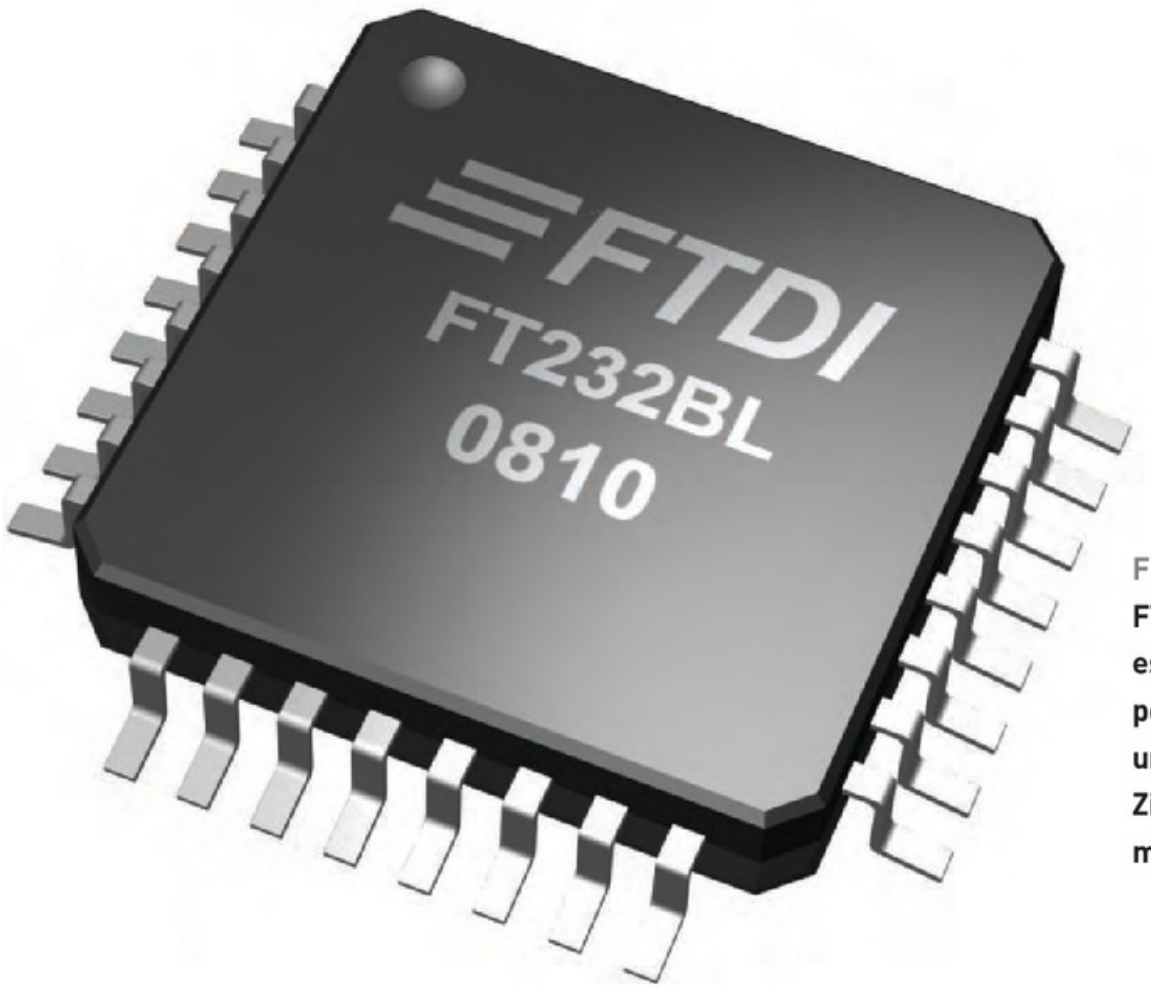


FIGURA 12. El FT232B cuando está en circuito, permite conectar un módulo ZigBee a una PC mediante USB.

Serial Peripheral **Interface**

SPI fue introducido en el mercado por la empresa **Motorola**. Sin embargo, este protocolo ofrece distintas variantes. **SPI** es un protocolo **serial**, **sincrónico** y **full-duplex** muy utilizado en la comunicación entre microcontroladores y periféricos (memorias Flash EEPROM, conversores A/D y D/A, RTCs, etc.), y de microcontroladores entre sí. Sus características principales son:

- Utiliza muy pocas líneas de comunicación, y esto lo convierte en una alternativa excelente frente a los buses paralelos.
- Tiene velocidades del orden de 5 a 10 MHz, e incluso mayores.
- La comunicación es del tipo maestro/esclavo. Existe un **maestro** (**host**) encargado de establecer una comunicación sincrónica, y un **esclavo** (generalmente, un **periférico**) que responde a los requerimientos (comandos) de aquél.
- Es posible la convivencia de múltiples **esclavos** sobre el mismo bus.

Es importante recordar que el modo multi-master permite tener más de un maestro SPI

EL BUS SPI

Está compuesto por dos líneas de **datos**, un **reloj** y una línea de **selección** por cada esclavo. Por ejemplo:

- **MOSI** (*Master Output - Slave Input*): datos de **maestro** a **esclavo**.
- **MISO** (*Master Input - Slave Output*): datos de **esclavo** a **maestro**.
- **CLK** o **SCLK**: reloj de sincronización que entrega el **maestro** a todos los esclavos del circuito.
- **/CS** o **/SS** (*Chip Select* o *Slave Select*): líneas de habilitación de los **esclavos**, generalmente activas en nivel bajo. Si el esclavo no es seleccionado (**/CS** se encuentra en nivel alto), su línea de datos de salida (SO, *Slave Output*) queda en estado de alta impedancia, de manera de no interferir con el periférico habilitado en ese momento.

Queda claro, entonces, que cada **esclavo** recibe de su **maestro** tanto el reloj (**CLK** o **SCLK**) como la línea de selección/habilitación (**/CS** o **/SS**). El **maestro** tiene el control de la comunicación: establece el sincronismo llevando el reloj a todos los periféricos, y selecciona con quién se quiere comunicar en cada momento mediante **/SS** o **/CS** (**Figura 13**).

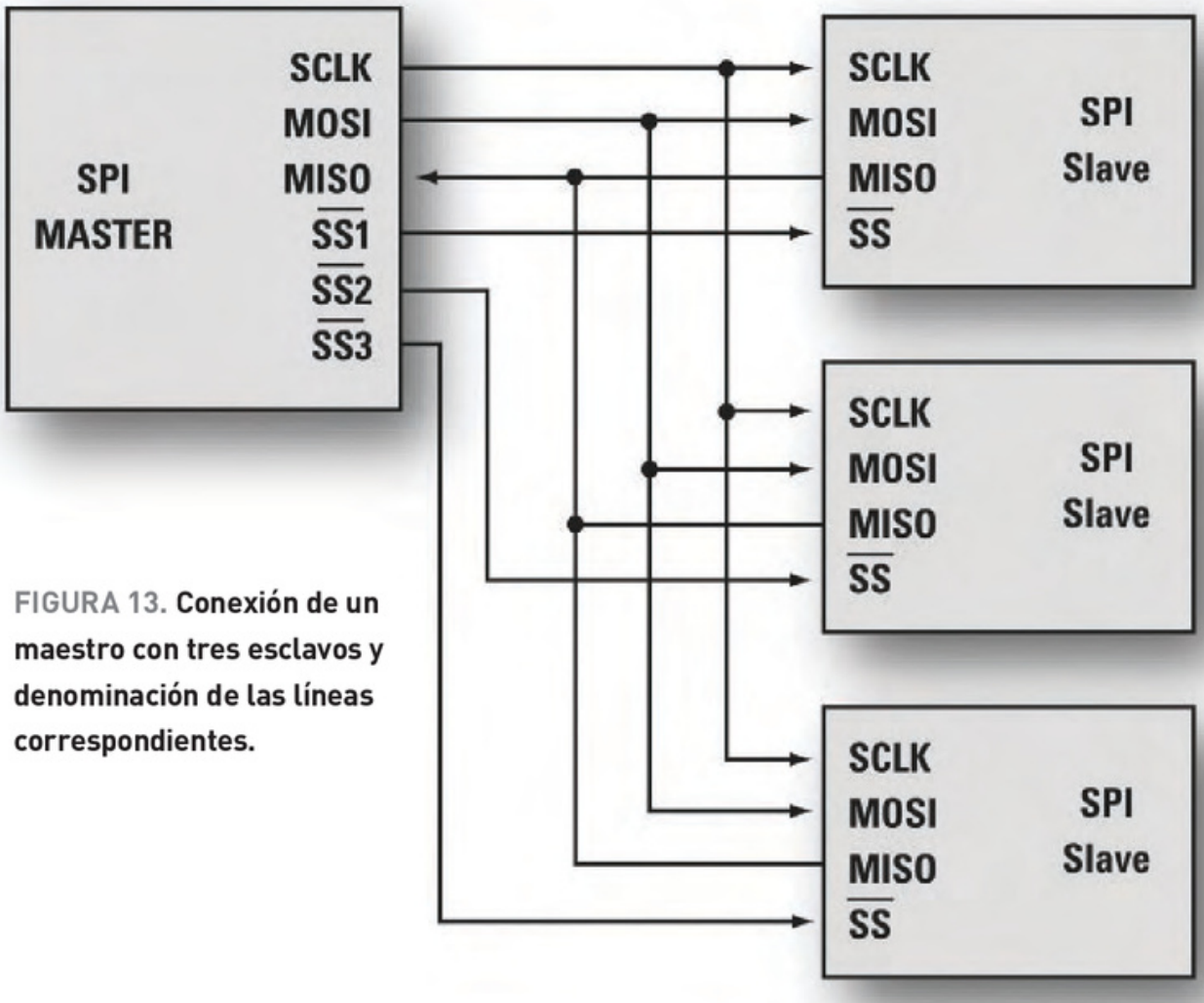
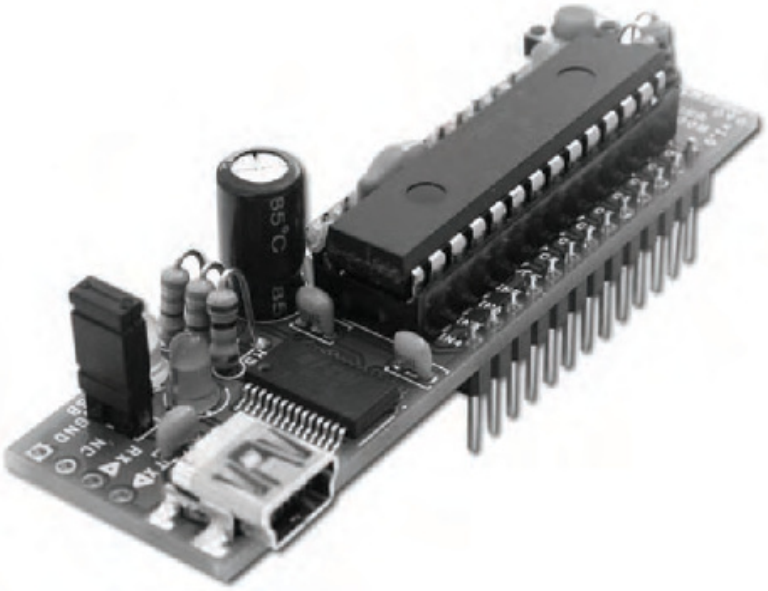


FIGURA 13. **Conexión de un maestro con tres esclavos y denominación de las líneas correspondientes.**



LA COMUNICACIÓN SPI

Esta comunicación es, básicamente, el diálogo entre dos registros de desplazamientos de 1 byte (aunque el protocolo no está limitado a sólo 8 bits de datos): uno, ubicado en el **maestro**; y el otro, perteneciente al **esclavo**. Es un anillo circular, donde la información, comandos de operación y/o datos se desplazan sincrónicamente a medida que los flancos del reloj del **maestro** disparan el traspaso bit a bit (**Figura 14**). La información obtenida por ambos queda luego disponible internamente en forma paralela.

La comunicación es full-duplex, al mismo tiempo que el **maestro** transmite información al **esclavo** por la línea **MOSI** (*Master Output - Slave Input*); éste le responde mediante **MISO** (*Master Input - Slave Output*). La información viaja al mismo tiempo en ambos sentidos.

Debido a la existencia de la línea de reloj para sincronismo, es posible encontrar variaciones sobre ella sin que esto se traduzca en pérdida o distorsión de los datos. Es así que SPI se presenta como un protocolo apropiado para usar con microcontroladores regidos por relojes de relativa precisión, como osciladores **RC**.

MODOS SPI

Antes del inicio de una comunicación, son necesarias dos tareas:

- Establecer y configurar una frecuencia de **CLK** apropiada para la comunicación que esté dentro del rango de operación del o los **esclavos**.
- Establecer las reglas de interpretación de la señal de sincronismo. Esto significa definir tanto la **polaridad CPOL** (*Clock POLarity*) como la **fase**

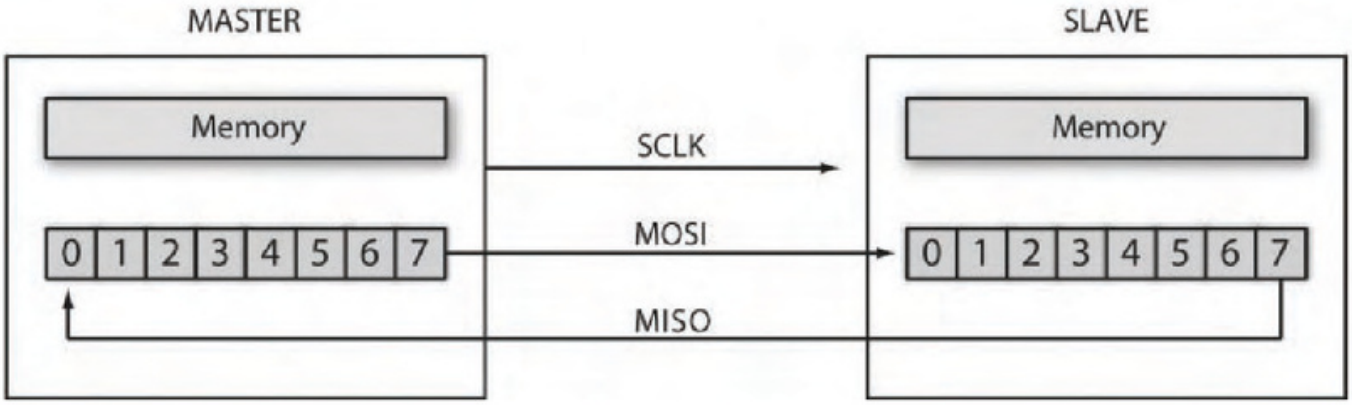


FIGURA 14. Configuración del hardware de una interfaz SPI, donde se establece un anillo circular entre los dos registros de desplazamiento.

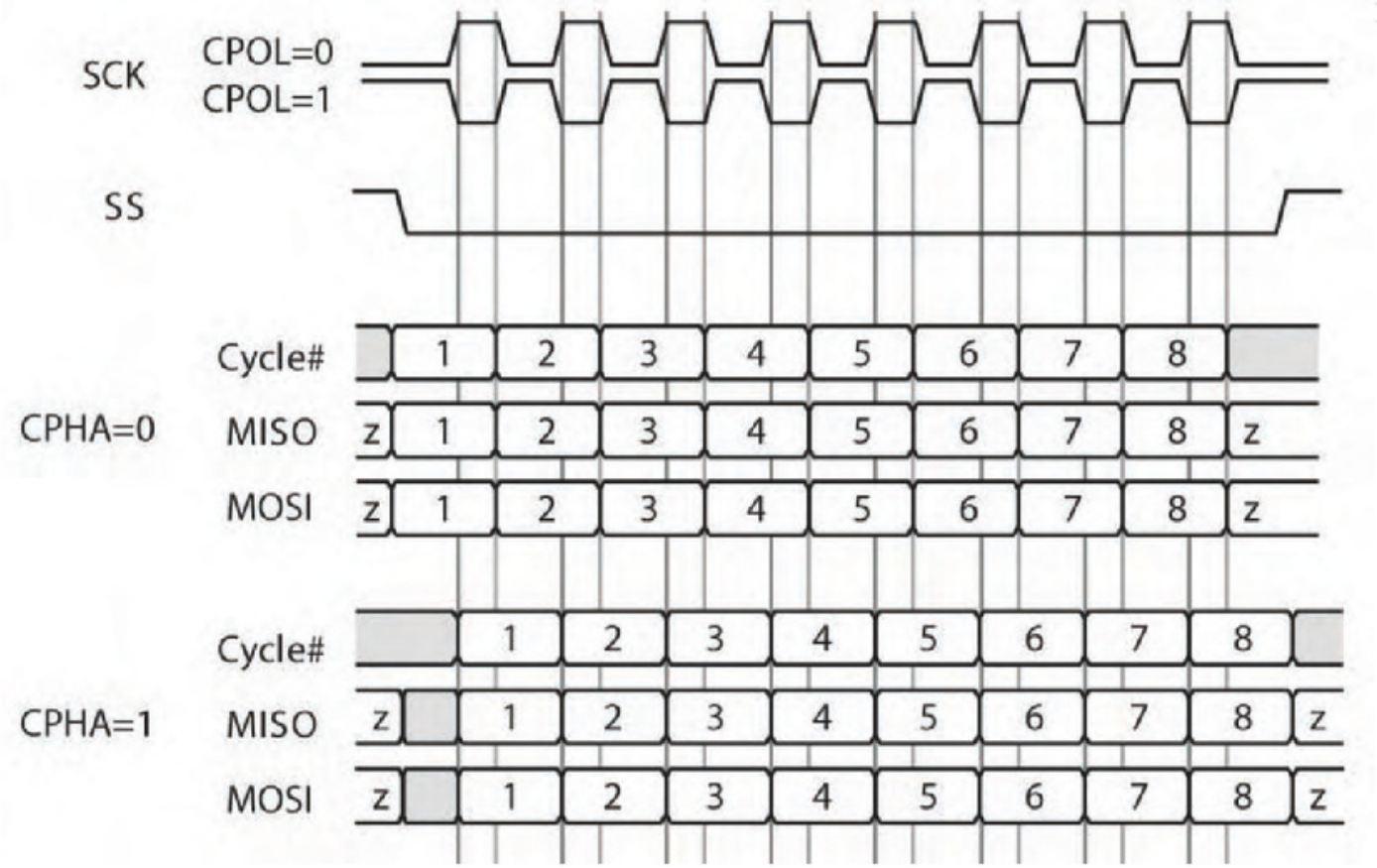


FIGURA 15. Los cuatro modos SPI y la interpretación de la información de reloj con respecto a la transferencia de datos.

del reloj **CPHA** (*Clock PHAse*) en cuestión, con respecto a la transferencia de datos.

A estas reglas de interpretación se las conoce como **modos**. De acuerdo con la convención tomada de Motorola, existen **cuatro modos de operación**, cada uno de ellos determinado por un valor del vector (**CPOL**; **CPHA**), (**Tabla 3**).

A partir del diagrama de tiempos de la **Figura 15**, podemos extraer las siguientes conclusiones:

A) Con **CPOL = 0**, el valor de **reposo** del CLK es **0** (**nivel bajo**).

A1) Si **CPHA = 0**, los datos se leerán en el primer flanco de CLK; para este caso: el flanco de subida. A su vez, los datos cambiarán en el segundo flanco de CLK, es decir, en el de bajada.

MODO	CPOL	CPHA
0	0	0
1	0	1
2	1	0
3	1	1

TABLA 3. CPOL y CPHA marcan los cuatro modos SPI.

A2) Si **CPHA = 1**, los datos se leerán en el segundo flanco de CLK; para este caso: el de bajada. A su vez, los datos cambiarán en el primer flanco de CLK, es decir, en el de subida.

B) Con **CPOL = 1**, ahora el valor de **reposo** del CLK es **1** (**nivel alto**).

B1) Si **CPHA = 0**, los datos se leerán en el primer flanco de CLK; para este caso: el flanco de bajada. A su vez, los datos cambiarán en el segundo flanco de CLK, es decir, en el de subida.

B2) Si **CPHA = 1**, los datos se leerán en el segundo flanco de CLK; para este caso: el flanco de subida. A su vez, los datos cambiarán en el primer flanco de CLK, es decir, en el de bajada.

Algunas conclusiones que se desprenden de la **Figura 15** son que, con **CPHA = 0**, el dato en la línea **MOSI** debe estar disponible al menos medio ciclo de reloj antes del primer flanco. También se requiere que la polaridad de **CLK** esté bien establecida antes de habilitar con **/SS** al dispositivo **esclavo**.

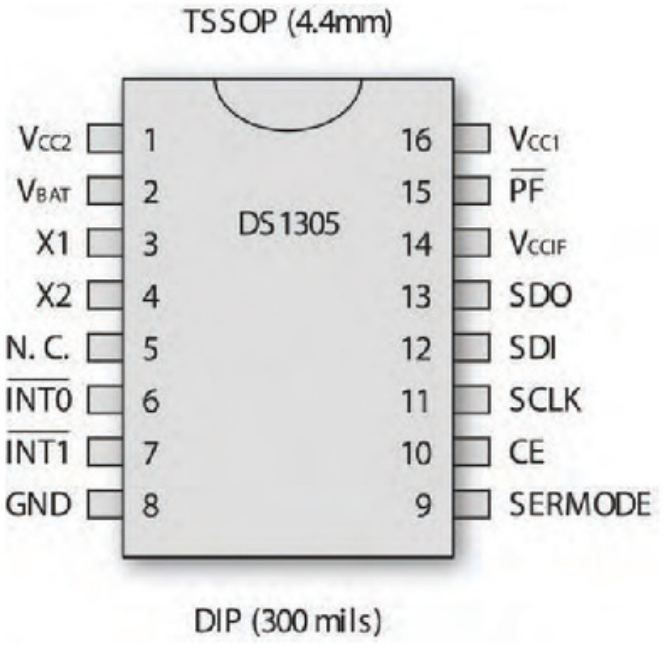


FIGURA 16. Chip DS1305 de Maxim, distribución de pines y encapsulado.

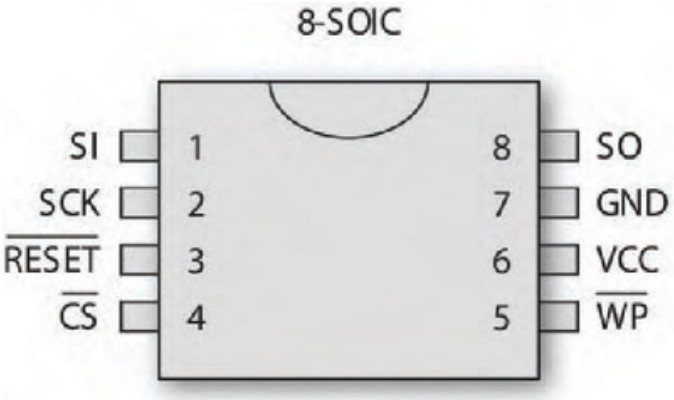


FIGURA 16. Chip AT45DB041b de Atmel, distribución de pines y encapsulado.

TIPOS DE PERIFÉRICOS

Existe una gran variedad de periféricos que incluyen el protocolo SPI como interfaz para intercambio de datos y configuración. Entre los más conocidos y habituales podemos destacar los siguientes (**Figuras 16** y **17**):

- **Memorias Flash**, **EEPROM**, **FRAM**: por ejemplo, **AT45DB041B** de Atmel (flash), **25LC256** (EEPROM) y **FM27L256B** (FRAM).
- **RTC (Real Time Clocks)** y **accesorios**: por ejemplo, el **DS1305** de Maxim y los *Processor Companions*, como el **FM3316** de Ramtron.
- **Conversores A/D** y **D/A**: por ejemplo, el **MAX5222** de Maxim, doble D/A de 8 bits en encapsulado **SOT23-8**.

USO DE SPI EN EL PIC18F4620

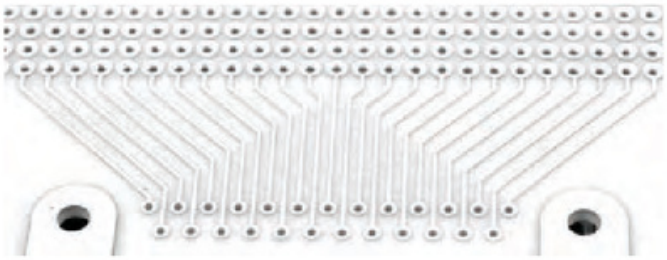
El módulo **MSSP** (*Master Serial Synchronous Port*) de este micro puede ser configurado para operar en modo SPI, estableciendo los registros de control

SSPCON1 y **SSPSTAT**. Observemos las siguientes características para su uso como **maestro**:

- Los pines que son salidas (**SDO** y **SCK**) deberán configurarse de tal modo.
- La entrada (**SDI**) es controlada por el módulo **MSSP**.
- En el registro **SSPCON1**, el bit **CKP** controla la polaridad del reloj.
- En el registro **SSPSTAT**, los bits **SMP** y **CKE** controlan en qué flanco se leen y escriben los datos, respectivamente.
- Los bits **SSPM0** a **SSPM3** eligen el modo **maestro** o **esclavo** y la frecuencia de clock.
- La escritura se inicia escribiendo el registro **SSPBUF**.
- La lectura se realiza al final del ciclo de escritura, lo cual es indicado por el micro seteando el flag **BF** en el registro **SSPSTAT**. Si no se requiere escribir nada, se ingresa cualquier valor, como **0xFF**. Por ejemplo, en C, para configurar el **MODO 3**:

```
SSPEN=1; // Habilita las líneas MOSI, MISO
 y SCK para la interfaz SPI.

 CKP=1;    // "Clock Polarity": indica el
 estado inactivo ó de reposo del SCK.
 STAT_CKE=0; // "Clock Edge Select": para
 Mode 3 los datos son transmitidos en
 // el flanco de subida, por lo que CKE=0.
 STAT_SMP=1;          // "Sample Bit": Los datos
 de entrada deben ser muestreados al
 // final del tiempo de los datos de salida.
```

El módulo MSSP del PIC18F4620 puede ser configurado para operar en modo SPI



```
SSPM3=0;            // Aquí se configura la
frecuencia del CLK a Tosc*4
SSPM2=0;
SSPM1=0;
SSPM0=0;
```

Luego, para realizar una escritura seguida de una lectura, efectuamos lo siguiente:

```
SSPBUF=txdata;
// Carga del registro
SSPBUF con el dato a transmitir
while(!STAT_BF);    // Espera a que se setee
el flag "Buffer full".
rxdata= SSPBUF;
// Se efectúa la lectura del Esclavo
```

CONEXIÓN DE UNA MEMORIA SPI

En el **Paso a paso 2** veremos cómo agregar un dispositivo SPI a una placa de desarrollo. La utilizaremos en conjunto con una memoria **EEPROM 25LC256** para realizar la práctica (**Figura 18**).

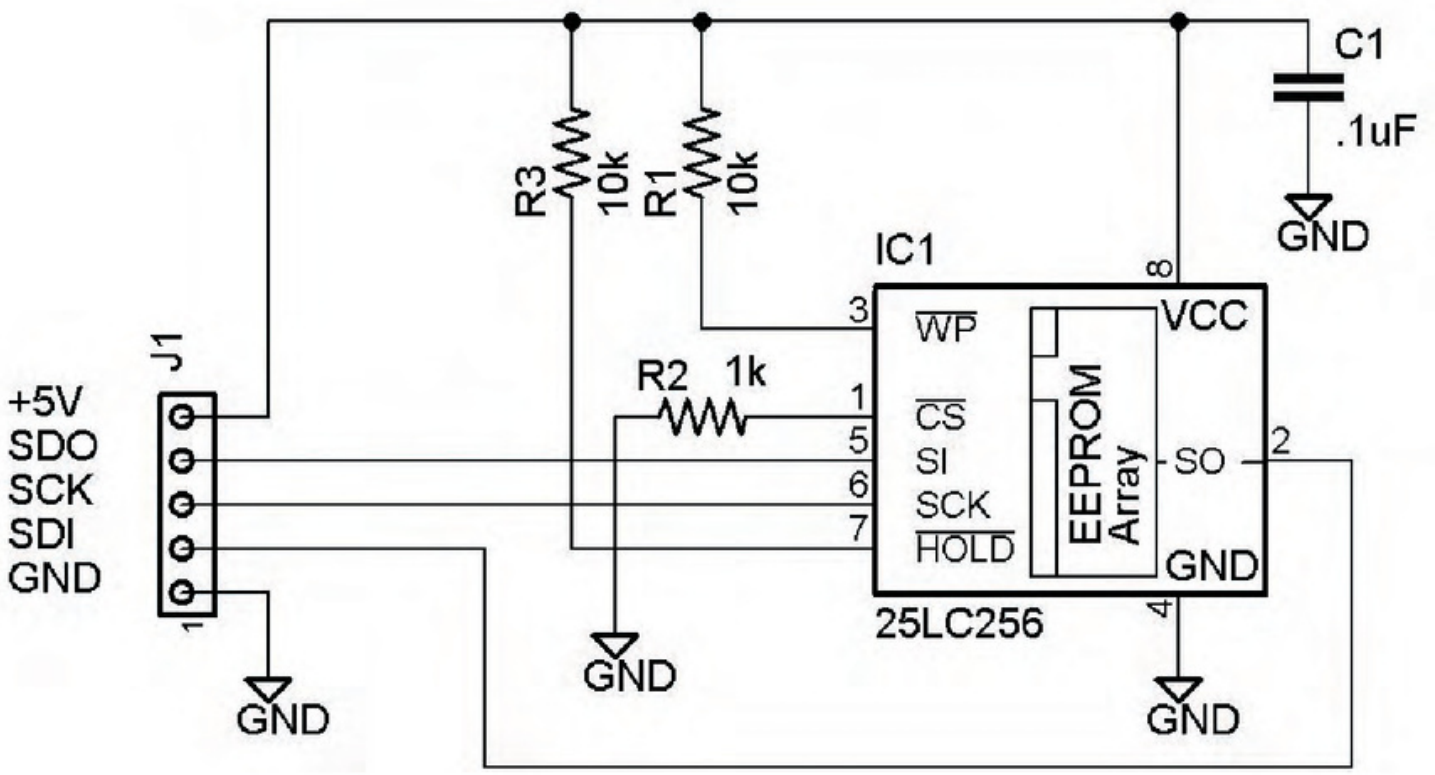
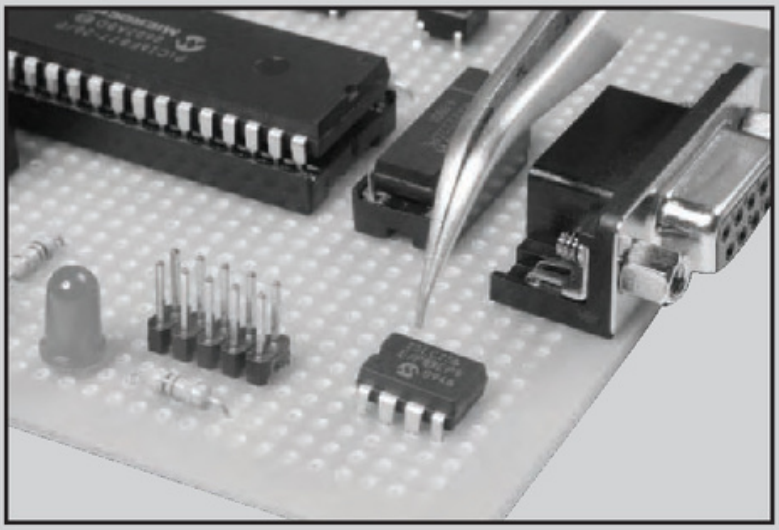


FIGURA 18. Circuito esquemático de una interfaz genérica para una memoria SPI, en este caso, una 25LC256.

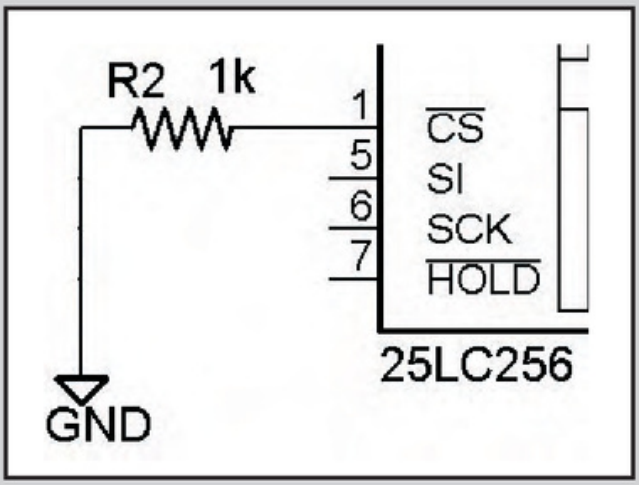
PASO A PASO /2
Conexión de una memoria SPI

1



Monte la memoria EEPROM sobre la placa con cuidado de no doblar ni quebrar ninguno de sus pines. Utilice cables de color para realizar las conexiones al bus de alimentación del protoboard: rojo para el positivo (pin número 8) y negro para la masa (pin número 4). Suelde también el capacitor de desacople.

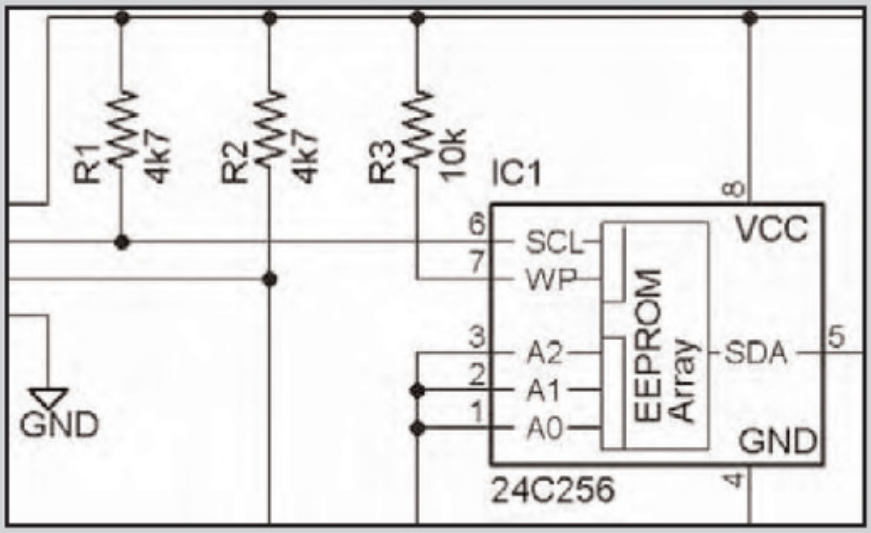
2



Es el turno del pin número 1: Chip Select (CS), que permite habilitar o no el dispositivo. Se lo utiliza con el fin de direccionar diferentes dispositivos que se encuentran conectados al mismo bus SPI. Como en esta práctica sólo dispone de una memoria, debe habilitarlo conectándolo a través de una resistencia de 1 kΩ a masa.

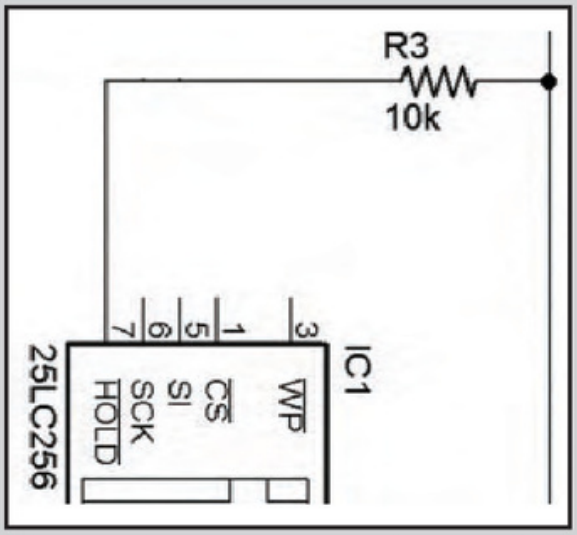
PASO A PASO /2 (cont.)

3



Siguiendo con la memoria, debe conectar, mediante una resistencia de 10 k?, el pin número 3: Write-Protect (WP) a positivo. Este pin se utiliza para proteger contra escritura el contenido de la memoria y se activa mediante un nivel lógico bajo. Como no le daremos utilidad, lo dejará desactivado para trabajar sin problemas.

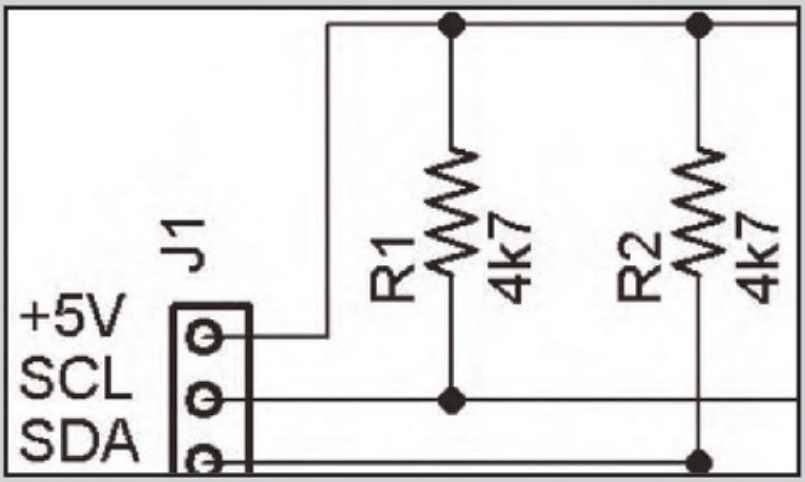
4



El próximo pin por conectar es el 7: HOLD. Éste funciona a modo de pausa, es decir que coloca la comunicación en un estado de suspensión ignorando todas las peticiones que lleguen del dispositivo maestro. Tampoco lo utilizará por el momento, de modo que lo conecta al positivo mediante una resistencia de 10 k?.

PASO A PASO /2 (cont.)

5



Conecte los pines de comunicación: el 5 (SI), el 2 (SO) y el 6 (SCK), mediante cables a los correspondientes pines del microcontrolador: 24 (SDO), 23 (SDI) y 18 (SCK). Observe que salida y entrada se interconectan una con la otra.

Bus de **comunicación I²C**

El bus **I²C** (*Inter Integrated Circuits*), diseñado por **Philips Semiconductors** en la década del '80, nació con el fin de realizar conexiones entre circuitos integrados. Es un bus **half-duplex** de comunicación de datos, de tipo serial sincrónico. Se desarrolló inicialmente para comunicar microcontroladores con distintos dispositivos electrónicos a través de dos hilos de forma bidireccional. En la actualidad, se lo usa en aparatos electrónicos de consumo, como televisores, equipos de audio, reproductores de DVD y monitores de computadora.

Permite conectar gran cantidad de circuitos integrados utilizando la configuración maestro/esclavo y con sólo dos hilos: datos (SDA) y reloj (SCL). Dichas líneas son bidireccionales y están polarizadas a positivo mediante sendas resistencia de **pull-up**, por lo que en reposo se encuentran en nivel alto (**Figura 19**).

Los dispositivos **maestros** se encargan de generar la señal de reloj, de iniciar el protocolo de comunicación y de recibir las peticiones de los **esclavos**. La transmisión es en formato serie, y es posible alcanzar tasas del orden de **100 kbits** por segundo en la modalidad estándar, de **400 kbits** por segundo en el modo rápido y de hasta **3,4 Mbits** por segundo en el modo de alta velocidad.

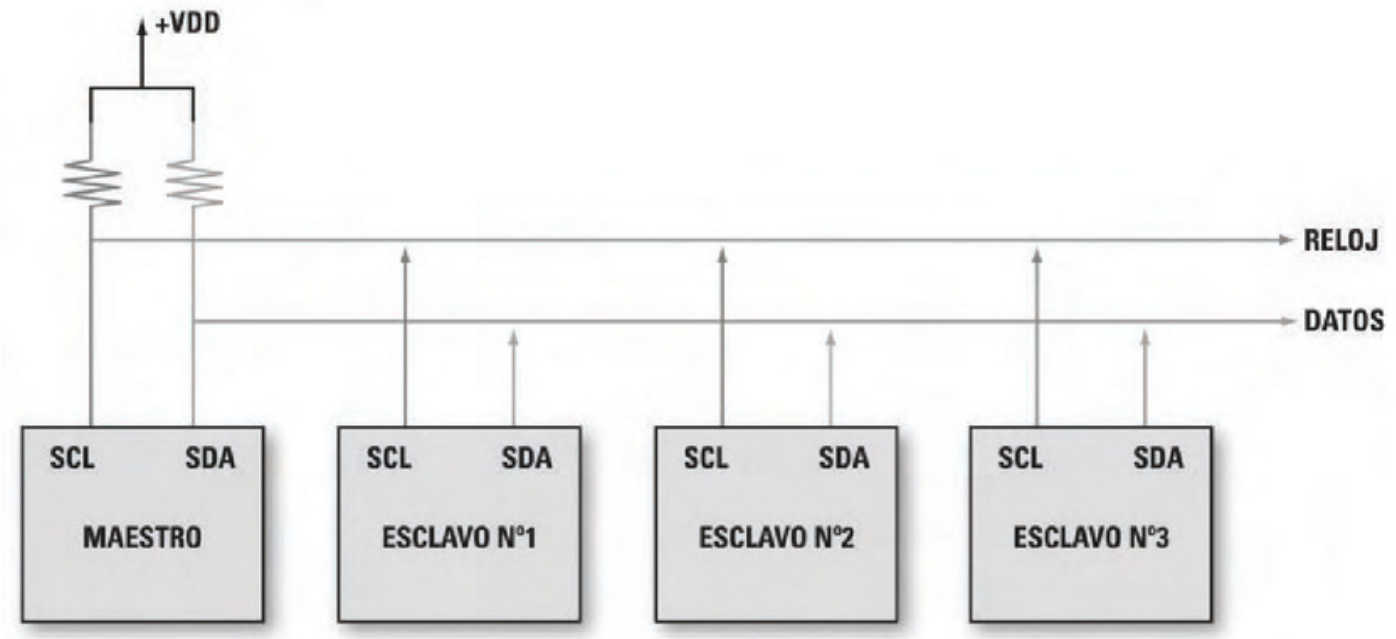


FIGURA 19. **Conexión de múltiples dispositivos mediante el bus. Es necesario que ambas líneas se encuentren conectadas a positivo mediante** resistencias pull-up.

Cada chip conectado al bus posee una dirección única para su identificación, seleccionable por software. Es posible que en la misma red convivan varios dispositivos maestros, debido a que el protocolo cuenta con un detector de colisiones. El **I²C** trabaja con palabras de datos de 8 bits, tanto para las direcciones como para la información que se transmite.

PROTOCOLO DE COMUNICACIÓN

Ya sabemos que para realizar una comunicación entre dos o más dispositivos mediante el bus necesitamos sólo dos líneas: datos y reloj. Cada dispositivo puede actuar como maestro o esclavo indistintamente, y posee una dirección que lo identifica. Es por eso que es necesario cumplir una serie de condiciones para que la comunicación sea exitosa; en otras palabras, debemos seguir el protocolo de comunicación.

El I²C trabaja con palabras de datos de 8 bits

CONDICIONES DE START Y STOP

El primer paso antes de enviar palabras de datos entre los distintos dispositivos conectados es avisar que se iniciará una transmisión, es decir, ingresar en la condición de **START**. El encargado de hacerlo es el **maestro.** Para esto, debe enviar la línea de datos **SDA** a nivel bajo, mientras mantiene la de **reloj SCL** en nivel alto. A continuación, los dispositivos podrán enviar y recibir datos. Una vez que concluye el proceso de transmisión, el maestro debe informarlo; es decir, ingresaremos en la condición de **STOP**. Para esto se debe llevar la línea SDA a nivel alto, mientras que la SCL permanece en nivel alto (**Figura 20**).

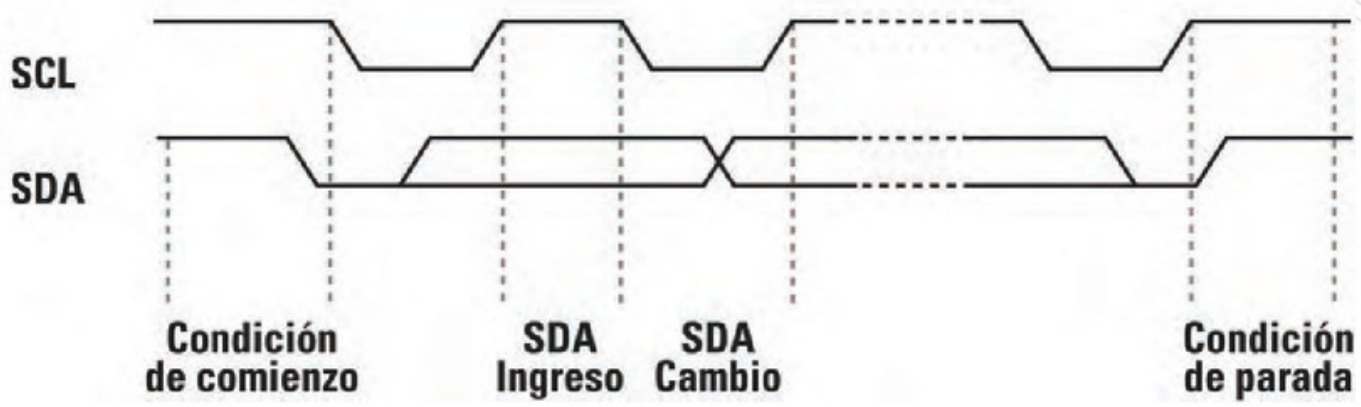


FIGURA 20. **Diagrama de tiempos en el que se muestran las condiciones de START y STOP, y los niveles que deberán adoptar las líneas SDA y SCL.**

TRANSFERENCIA DE DATOS

Una vez que el dispositivo maestro generó la condición de inicio, comienza la transferencia. Como ya sabemos, las palabras deben tener una longitud de 8 bits, y la primera en enviarse representa la dirección del dispositivo esclavo seleccionado para establecer la comunicación. Dicha dirección está compuesta por 7 bits, lo que da un máximo de 128 dispositivos posibles (0 a 127). El bit restante es el **R/W**, que permite seleccionar entre el modo de escritura y el de lectura.

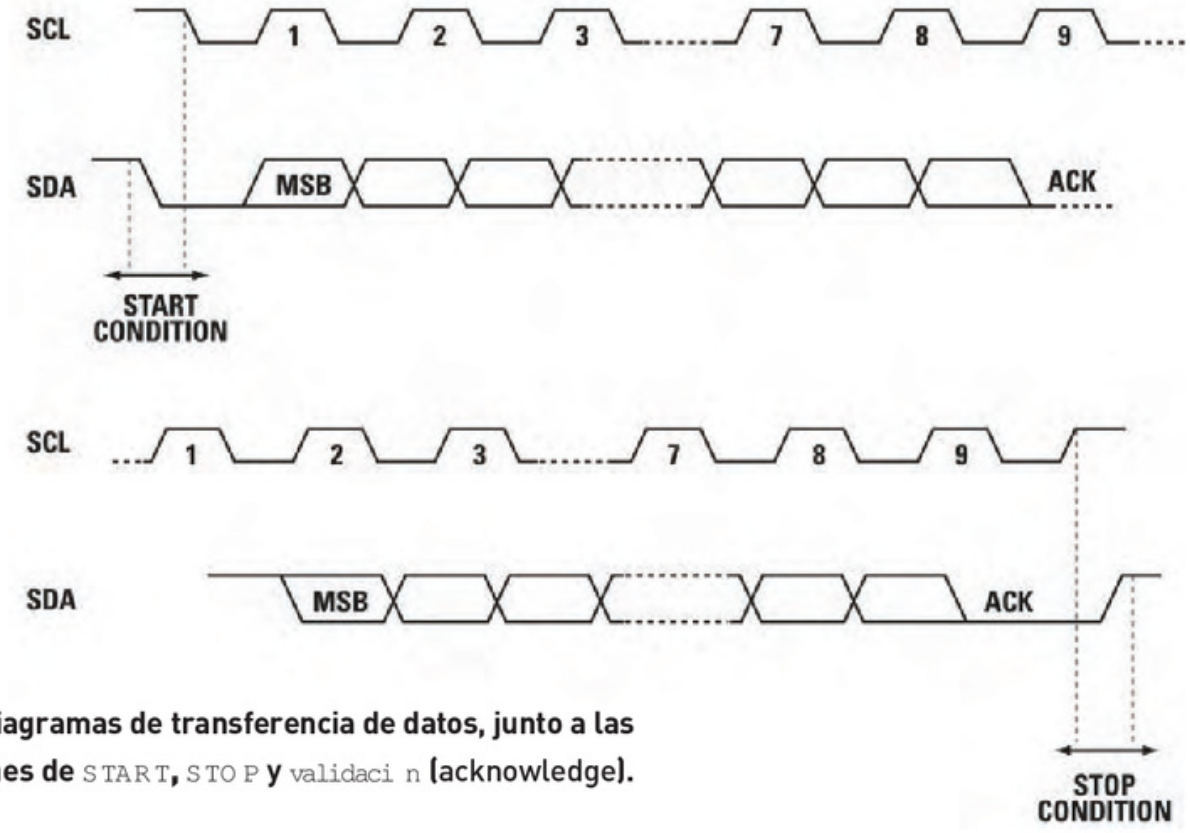


FIGURA 21. **Diagramas de transferencia de datos, junto a las condiciones de START, STOP y validación (acknowledge).**

El próximo paso que debe seguir el maestro es leer el estado de la línea SDA, impuesto por el dispositivo **esclavo**. En caso de que sea un nivel bajo, la transferencia continúa normalmente; en cambio, si la línea adopta un nivel alto, quiere decir que el esclavo no reconoce la transferencia o no está listo para la comunicación. En el paso siguiente se genera desde el maestro la condición de **STOP** para dejar el bus libre. Este proceso de validación se denomina **ACK**, *acknowledge* (**Figura 21**). Siempre que finaliza la transmisión, el maestro debe generar la condición de STOP para liberar al bus y dejar el sistema a la espera de nuevas comunicaciones.

COMPARACIÓN ENTRE I²C Y SPI

Si bien los buses de comunicación **I²C** y **SPI** transmiten la información en formato serie y permiten conectar dispositivos similares, presentan grandes diferencias en su funcionamiento. El bus SPI tiene las siguientes características:

- Permite realizar una comunicación **full-duplex,** a diferencia de la **half-duplex** que maneja el bus **I²C**.
- Logra mayores velocidades de transmisión.
- No limita la palabra de datos que se va a transmitir a 8 bits.
- Utiliza más líneas por dispositivo.
- Direcciona los dispositivos mediante líneas específicas, mientras que **I²C** utiliza una palabra de 7 bits para realizarlo.
- No posee bit de reconocimiento (ACK). Esto puede derivar en un gran problema, debido a que es posible enviar datos desde el maestro sin que ningún esclavo responda.
- No permite utilizar, por lo menos fácilmente, varios dispositivos maestros conectados en el bus.

DISPOSITIVOS I²C

En la actualidad, una enorme cantidad de dispositivos electrónicos integran el protocolo de comunicación **I²C**, lo que facilita enormemente la tarea de realizar buses de comunicación en sistemas electrónicos complejos. El bus ha tomado tanta popularidad en el diseño electrónico, que es posible encontrarlo integrado desde en equipos domésticos, hasta en sistemas a nivel industrial. Por ejemplo, es común disponer de series de diversos circuitos integrados con el bus incluido:

- Memorias seriales de tipo EEPROM
- Microcontroladores de diferentes gamas
- Displays
- Convertidores analógico-digitales y digital-analógicos
- Procesadores de sonido
- Sensores de distintos tipos
- Relojes de tiempo real (RTC)
- Llaves digitales

USO DE I²C EN EL PIC18F4620

El módulo **MSSP** (*Master Serial Synchronous Port*) de este micro puede ser configurado para operar en modo I²C, operando sobre los registros de control **SSPCON1**, **SSPCON2** y **SSPSTAT**. Observemos las siguientes características para el uso como **maestro**:

- Los pines (**SDA** y **SCL**) deben configurarse como entradas en el registro **TRISC.**
- Los bits **SSPM0** a **SSPM3** de SSPCON1 se configuran al valor **1000b** para modo **maestro.**
- La velocidad de operación se controla mediante el registro **SPADD.**
- La interfaz se configura para transmitir o recibir mediante el bit **RCEN** en el registro **SSPCON2.**

• El módulo facilita la operación por interrupciones; luego de cada operación elemental se configura el flag **SSPIF** en el registro **PIR1**, que debe reseteare por software.

Una secuencia típica de direccionamiento de un dispositivo es la siguiente:

• El modo se configura en transmisión.
• La condición de START se inicia seteando el bit **SEN** en el registro **SSPCON2.**
• La transmisión de la dirección se inicia escribiendo el registro **SSPBUF**, el modo es el bit **0.**
• El **ACK** del dispositivo se observa en el bit **ACKSTAT** en el registro **SSPCON2.**

Según se trate de lectura o escritura, se cambia o no el modo de la interfaz (bit **RCEN**):

• Para una transmisión, los bits se ingresan en el registro **SSPBUF**.
• Para una recepción, el envío (o no) de **ACK** al dispositivo se configura en el bit **ACKDT** en el registro **SSPCON2**. La lectura de cada byte se efectúa en el registro **SSPBUF.**
• La operación se termina con una condición de **STOP**, que se inicia seteando el bit **PEN** en el registro **SSPCON2**.

CONEXIÓN Y MANEJO DE UN PERIFÉRICO I²C

En el **Paso a paso 3** veremos cómo agregar un dispositivo I²C a una placa de desarrollo con una memoria **EEPROM 24LC256**, ver **Figura 22**. El **PIC18F4620** incluye un módulo **I²C**, lo que nos simplifica la tarea de escribir el código.

Montamos la memoria sobre la placa tomando los recaudos necesarios para no doblar o quebrar ninguno de los pines de manera accidental. Utilizando cable de color rojo (positivo) y de color negro (negativo), realizamos las conexiones correspondientes de la alimentación de la memoria y el capacitor de desacople.

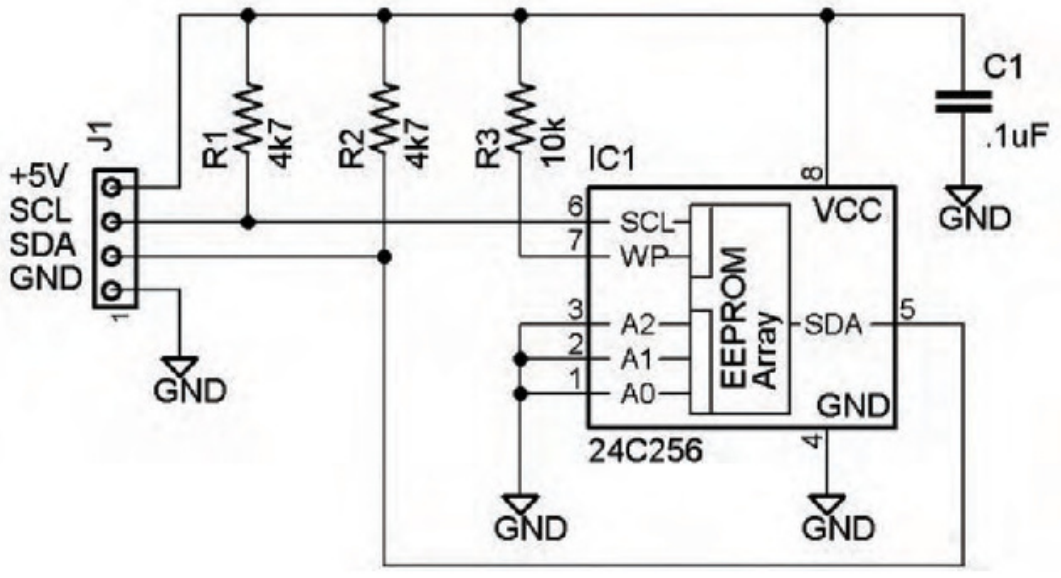


FIGURA 22.
Circuito esquemático de una interfaz genérica para una memoria I²C, en este caso, una 24LC256.

Multiple **choice**

1 ¿Qué es Docklight?

a- La interfaz en una computadora u otro dispositivo que transmite datos de a un bit por vez.
b- Un software para verificación, análisis y simulación de protocolos de comunicación serie.
c- Una comunicación entre dos registros de desplazamientos de 1 byte.
d- Una comunicación serial, bidireccional y de índole "diferencial".

2 ¿Qué es un puerto USB?

a- La interfaz en una computadora u otro dispositivo que transmite datos de a un bit por vez.
b- Un software para verificación, análisis y simulación de protocolos de comunicación serie.
c- Una comunicación entre dos registros de desplazamientos de 1 byte.
d- Una comunicación serial, bidireccional y de índole "diferencial".

3 ¿Qué es un puerto serie?

a- La interfaz en una computadora u otro dispositivo que transmite datos de a un bit por vez.
b- Un software para verificación, análisis y simulación de protocolos de comunicación serie.
c- Una comunicación entre dos registros de desplazamientos de 1 byte.
d- Una comunicación serial, bidireccional y de índole "diferencial".

4 ¿Qué es la comunicación SPI?

a- La interfaz en una computadora u otro dispositivo que transmite datos de a un bit por vez.
b- Un software para verificación, análisis y simulación de protocolos de comunicación serie.
c- Una comunicación entre dos registros de desplazamientos de 1 byte.
d- Una comunicación serial, bidireccional y de índole "diferencial".

5 ¿Quién es el encargado de avisar que se iniciará una conexión?

a- El botón START.
b- El dispositivo maestro.
c- La línea de datos SDA.
d- El reloj SCL.

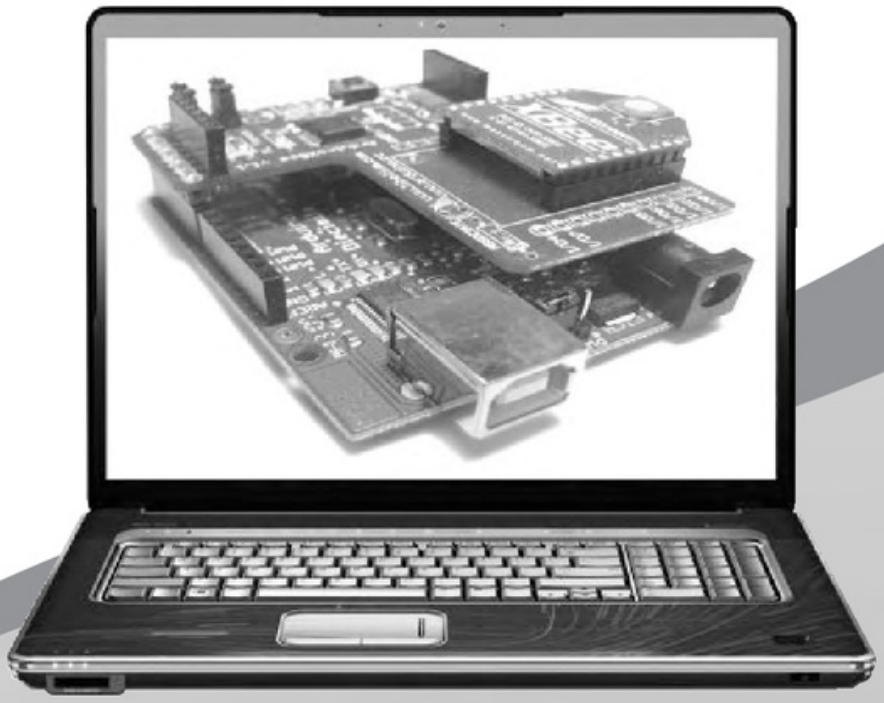
6 ¿A qué nivel se debe llevar la línea SDA una vez que concluye el proceso de transmisión?

a- Alto.
b- Medio.
c- Bajo.
d- Mantener como estaba.

Respuestas: 1 b, 2 d, 3 c, 4 a, 5 b, 6 a.

Capítulo 4

Conectividad inalámbrica



En este capítulo, estudiaremos cómo establecer una conexión sin cables entre dos o más dispositivos.

Conectividad **inalámbrica**

Hace muchos años, era impensable imaginar un sistema de transmisión de datos que no se realizara mediante un alambre de cobre. Sin embargo, la demanda del mercado y la tecnología permitieron que esta idea se hiciera realidad. Tanto es así, que en la actualidad a nadie le asombra que un conjunto de bits puedan ser transmitidos entre dispositivos empleando como medio el espacio.

Los medios de transmisión de datos inalámbricos nacen de la limitación propia de los cables. Hasta el momento, hemos visto las opciones de conectividad para agregar memorias, conversores y, por qué no, para reportar nuestras acciones a una PC, siempre a través de cables. En este capítulo veremos otras alternativas, esta vez, sin cables.

A partir del año **1901**, **Guillermo Marconi** llevó a la práctica las teorías de **Maxwell** y **Hertz**, realizando la primera emisión transatlántica por medio de las ondas de radio. Desde entonces, los avances han sido impresionantes. A continuación, veremos la evolución de esta tecnología aplicada en diferentes aspectos.

OPCIONES DE COMUNICACIÓN CON LA PC

Al tomar como unidad de medida básica una computadora personal, podemos decir que las opciones de comunicación con sensores remotos, para controlar a distancia y tomar mediciones, son inagotables. Una computadora dispone de varios conectores de comunicación con el mundo exterior. Cada uno de ellos tiene sus propias características y reglas que deben cumplirse para poder utilizarlos.

El puerto **paralelo** ha quedado obsoleto y ya no existe en las nuevas máquinas desde hace varios años. El **serie USART** es el puerto ideal para los experimentos de conectividad de un aficionado, pero fue desplazado casi por completo por el estándar **USB** (puerto serie universal, **Figura 1**).

Una computadora es más un concepto que un sustantivo. Un microcontrolador o microprocesador es toda una computadora. Por lo tanto, todo lo que sea aplicado a una PC podrá aplicarse también a todos nuestros proyectos electrónicos con microcontroladores, para hacer que éstos se comuniquen entre sí y con periféricos externos.

Entre las posibilidades inalámbricas de comunicación más comunes, podemos destacar **Bluetooth,**

OPCIONES DE CONECTIVIDAD

Mediante **Bluetooth**, ya sea integrado o por medio de un accesorio USB, es posible conectar la PC con diversos proyectos. Conectando un módulo **ZigBee** u **802.15.4** a un puerto serie, integrado o USB, podemos, además, recibir datos de redes de sensores.

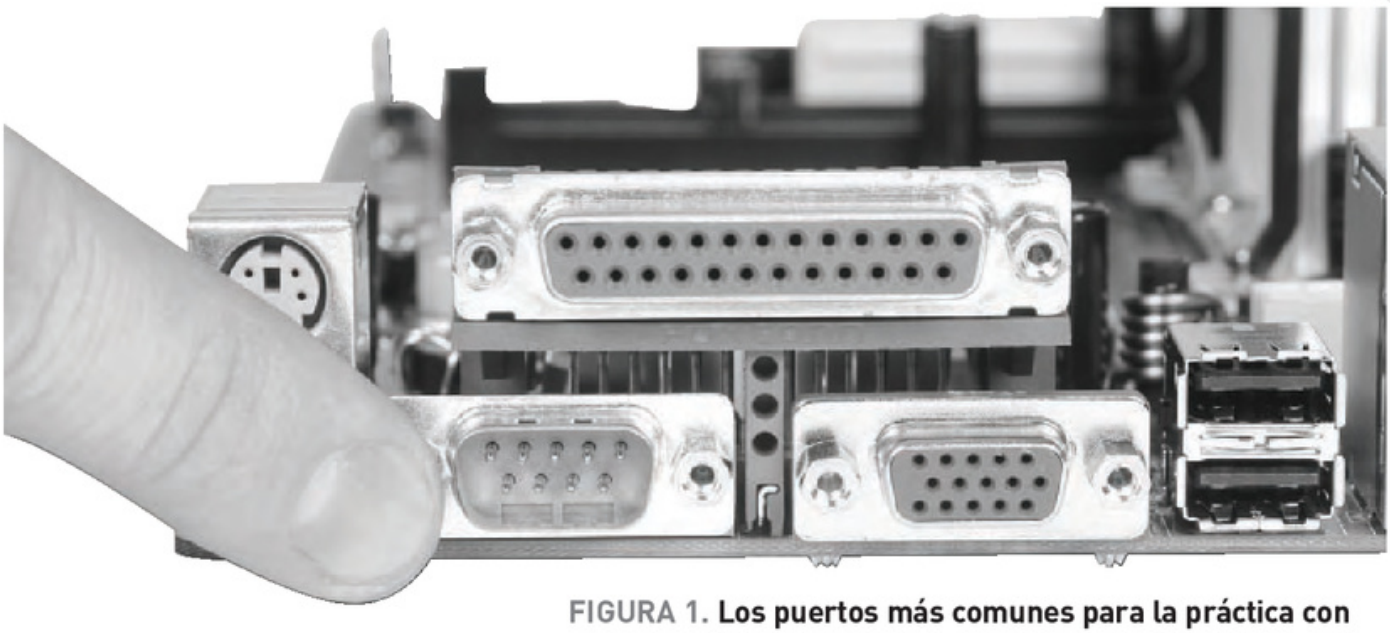


FIGURA 1. **Los puertos más comunes para la práctica con proyectos electrónicos son el** RS-232 **y el** USB**.**

802.15.4 y ZigBee. Por supuesto que existen otras tecnologías, pero utilizan mayores recursos y están indicadas para casos especiales, como la transmisión a larga distancia; por ejemplo, es posible emplear un módem **GPRS** para controlar a escala mundial un dispositivo por medio de la red celular **GSM**.

TECNOLOGÍA BLUETOOTH

Bluetooth es el nombre de una especificación técnica que permite conectar dispositivos entre sí, para evitar el uso de cables. Es de corto alcance –entre 1 y 100 metros–, dependiendo de la potencia o especificación.

En un principio, esta tecnología tenía como meta eliminar los cables en los accesorios de teléfonos móviles. Su frecuencia de operación es de **2,4 GHz** y, además, permite la comunicación de datos y voz.

El perfil SPP de Bluetooth permite emular un cable serie entre dos dispositivos de manera inalámbrica

IRDA VS. BLUETOOTH

Otro de los medios de transmisión de datos a corta distancia es el infrarrojo (IrDA). Sin embargo, Bluetooth tiene la ventaja de que el infrarrojo requiere de una comunicación lineal entre transmisor y receptor, para su efectiva transmisión.

Sus principales aplicaciones son los manos-libres para telefonía móvil, auriculares y micrófonos inalámbricos, mouses, impresoras, cámaras fotográficas digitales y mandos a distancia en consolas de juegos, entre otras posibilidades.

La velocidad de transmisión de datos es, como máximo, de **1 Mbps** o megabit por segundo (recordemos que la "b" minúscula corresponde a bit, en tanto que la "B" mayúscula equivale a byte). Si bien este valor es muy bajo en comparación con otras especificaciones –como Wi-Fi que, normalmente, es de 54 Mbps o 108 Mbps–, es más que suficiente para interconectar casi el 100% de los dispositivos o periféricos que actualmente existen en el mercado.

A pesar de la especificación original, Bluetooth sigue creciendo en velocidad, bajo consumo y versatilidad. Como podemos ver, muchas especificaciones que tenían un objetivo inicial y único fueron modificándose a lo largo de los años, para así llegar a satisfacer mayores demandas de velocidad e interacción con distintos medios.

Por lo tanto, hoy no existe una clara línea divisoria entre todas estas tecnologías, tal como pudo haber en la época de las especificaciones alámbricas con los puertos paralelo y serie de una computadora. De alguna manera, ahora todo puede conectarse a casi todo, dependiendo de las características destacables que más graviten en nuestros proyectos, las cuales determinarán la adopción de una u otra alternativa.

Módulos prearmados (Kcwirefree)

La adopción de un módulo prearmado y probado hará que nuestros proyectos sean un verdadero placer, y nos evitará tener que armarlos y depurarlos, lo cual elevaría su costo final. Para ejemplificar el uso de estos módulos, podemos tomar los **Bluetooth de Kcwirefree**.

Recordemos que Bluetooth permite la comunicación de datos, pero también la de voz y audio. Dado que el enlace de audio no reviste mayores inconvenientes al momento de utilizarlo –porque los módulos pueden comandarse directamente por medio de **dip-switches** (o un microcontrolador, por supuesto)–, veremos los de comunicación de datos, que nos permitirán enlazar nuestros circuitos entre sí, con una PC, un celular o un PDA.

PERFILES DUN Y SPP

Las características de operación en Bluetooth se agrupan en perfiles. El perfil DUN permite utilizar un dispositivo como módem. En cambio, el SPP simula una conexión serie, de manera que vemos el equipo remoto como si estuviéramos conectados a él por un cable.

Para utilizar y controlar este módulo (al igual que cualquier otro, aunque hay excepciones, claro), se emplean unos comandos inventados por la compañía **Hayes**, llamados **comandos AT**. Esta firma desarrolló este lenguaje como un método simple de comunicación entre módems, que, por aquella época, eran de acople telefónico. En esos dispositivos, el auricular de un aparato telefónico se colocaba encima del módem mismo para captar los sonidos digitales y, así, convertirlos en datos.

En la actualidad, los comandos AT son un estándar de la industria para controlar todo tipo de dispositivos inalámbricos o alámbricos, teléfonos móviles, y otros; además de seguir manejando los obsoletos módems telefónicos dial-up. Las letras **AT** provienen de ATention (atención), y le indican al módem o módulo que inmediatamente se enviará el comando propiamente dicho. Todo comando es antecedido por el prefijo AT.

En una Palm con Bluetooth es posible usar el programa TriConnect para ver un proyecto

La conexión a los módulos es de tipo serie. Para el caso particular del módulo Bluetooth, utilizaremos una conexión serie a **115.200 baudios**, **8N1**. El módulo viene configurado por defecto a ese valor, pero permite alcanzar una velocidad de control de **3 Mbps**. Podemos utilizar sólo tres cables (TX, RX y masa) o cinco, para efectuar un control de flujo por medio de las señales **CTS** y **RTS**.

Un dato importante es que, además de la conversión de las señales **RS-232** de la computadora, es necesario adecuar el nivel de los valores lógicos a **3,3 V**.

Es posible utilizar el programa **Docklight** para acceder al módulo, o gestionarlo directamente desde un microcontrolador. Pero como en este caso estamos estudiándolo, lo más conveniente es emplear la PC. No bien alimentemos el módulo, éste responderá algo similar a lo siguiente:

```
AT-ZV -CommandMode-
AT-ZV BDAddress 00043e3a3111
```

En la hoja de datos del fabricante se encuentran todos los comandos disponibles y su explicación en detalle. Incluso, se provee de un software especial para ensayar el módulo, sin necesidad de escribir todos los comandos.

CONECTARSE

Desde el módulo y mediante el comando AT+ZV Discovery, podemos obtener un listado de los dispositivos Bluetooth próximos. Tomamos la dirección del que nos interesa y, enviando AT+ZV SPPConnect dirección, nos conectamos a él.

Una última consideración para tener en cuenta es el modo de operación del módulo. Al momento de alimentarlo, éste se encuentra en el modo **COMANDO**, quiere decir que podemos configurarlo e interactuar con todos los comandos disponibles. Luego de realizar una conexión con otro terminal, el módulo pasa al modo **BYPASS**, siendo éste transparente para los datos emitidos y recibidos. Para regresar al anterior, debemos enviarle una secuencia denominada "de escape", que consiste en los caracteres: **^#^$^***. La secuencia que fuerza al modo BYPASS –estando en COMANDO– es: **AT + ZV Bypass**.

Existen muchas opciones de conectividad, y todas apuntan a satisfacer distintas necesidades

¿Por qué tantas **opciones?**

Hemos mencionado algunas opciones al comienzo de este capítulo y, seguramente, ya conocemos otras. Las razones de esta proliferación están en la búsqueda de la satisfacción de distintas necesidades por caminos diferentes. Empecemos por diferenciar entre simple **conectividad** y **networking**.

Por **conectividad** entendemos la posibilidad de **establecer una conexión** o comunicación entre dos dispositivos, de modo que puedan **dialogar entre sí**. Con **networking**, nos referimos a la interconexión de **varios dispositivos en red** soportando un stack de protocolos robusto que permita comunicar varias y diversas aplicaciones. Hagamos, entonces, el análisis desde este ángulo.

TECNOLOGÍA WI-FI

La necesidad que da origen a **Wi-Fi** es mantener el esquema y la velocidad de una red, pero sin tener que recurrir a cables. Como tal, se la puede considerar una especie de reemplazo de los sistemas **Ethernet**. Los dispositivos ya implementan un stack de protocolo, generalmente, **TCP/IP**; lo que hace Wi-Fi es reemplazar la parte física, es decir, la de conexión al medio de comunicaciones (**Figura 2**).

En este entorno, no nos sorprende que las velocidades de transferencia sean altas (del orden de los 50 MBps), el consumo resulte elevado, y el stack, complejo. En particular, lo que se refiere a encriptación requiere un hardware con una potencia de procesamiento elevada.

BLUETOOTH

Este stack de protocolo nace para mantener la conectividad que requiere un equipo respecto a sus periféricos, sin depender de los cables. Como tal, podemos pensar en él como un reemplazo de **USB**. En particular, el equipo central suele ser un celular o una **PDA** (*Personal Digital Assistant*, o asistente personal digital), de modo que el requerimiento es mantener la libertad de operación del individuo brindándole servicios de control, conexión y audio.

El stack resultante tiene una complejidad menor a la de **Wi-Fi**, pero suficiente como para requerir un procesador importante. No se pretende que haya conexión de periféricos entre sí, pero sí autodetección y una velocidad como para que varios periféricos puedan hacer su trabajo. El consumo es menor, pues los dispositivos involucrados funcionan con baterías, que el usuario debe recargar periódicamente.

FIGURA 2. **Podemos observar un robot que incorpora la tecnología** Wi-Fi 802.11b/g. **También se pueden ver estos dispositivos con** Bluetooth **o** GSM.

ZIGBEE Y 802.15.4

En este caso, el escenario es diferente. El origen de **802.15.4** puede asociarse a la necesidad de conectar sensores remotos a un sitio central que recolecta la información. Como tal, podemos considerarlo un reemplazo de **RS-485**, utilizado en conexiones multipunto cableadas. El factor que tiene más peso en las decisiones, en este caso, es el consumo, pues la idea es que los sensores operen con baterías y no es factible recolectarlas para recargarlas.

De modo similar, se trata de entes que envían un mensaje de forma esporádica, por lo que no se pretende que haya gran velocidad de comunicación. Sí se espera que haya muchos de estos entes en la red, por lo que se prioriza el hecho de evitar las colisiones en el acceso al medio de comunicaciones. Los sensores suelen ser dispositivos simples, de modo que el stack de protocolo también tiene que serlo.

ZigBee RF4CE es una implementación particular del stack para formar redes de control remoto

ALTERNATIVAS A ZIGBEE

Utilizando 802.15.4 como base, o no, existen muchas alternativas a ZigBee. Microchip ofrece Mi-Wi, Texas Instruments tiene SimpliciTI, Digi brinda DigiMesh, y en especial, existe un protocolo llamado ANT, diseñado para mantener el mínimo consumo posible.

ZigBee y 802.15.4

Sobre esta base se apoya **ZigBee**, proveyendo, además, la extensión de la conectividad mediante la repetición de los mensajes por dispositivos llamados **routers** (routing), y un **conjunto de especificaciones** para soportar **aplicaciones distribuidas** con interoperabilidad entre distintos fabricantes. El objetivo es proveer a estas aplicaciones de un **marco de operación**. Por ejemplo, ZigBee brinda los códigos necesarios para implementar dispositivos dedicados que realicen tareas de automatización del hogar. Estos marcos de operación se denominan **perfiles**.

Describiremos a continuación los estándares de comunicaciones inalámbricas personales **802.15.4** y **ZigBee**. Gracias al dictado de estas normas, la industria pudo fabricar dispositivos inalámbricos capaces de dialogar entre sí.

802.15.4

Este estándar está destinado a regular el acceso al medio de comunicaciones para redes personales inalámbricas. Como veremos en el apartado corres-

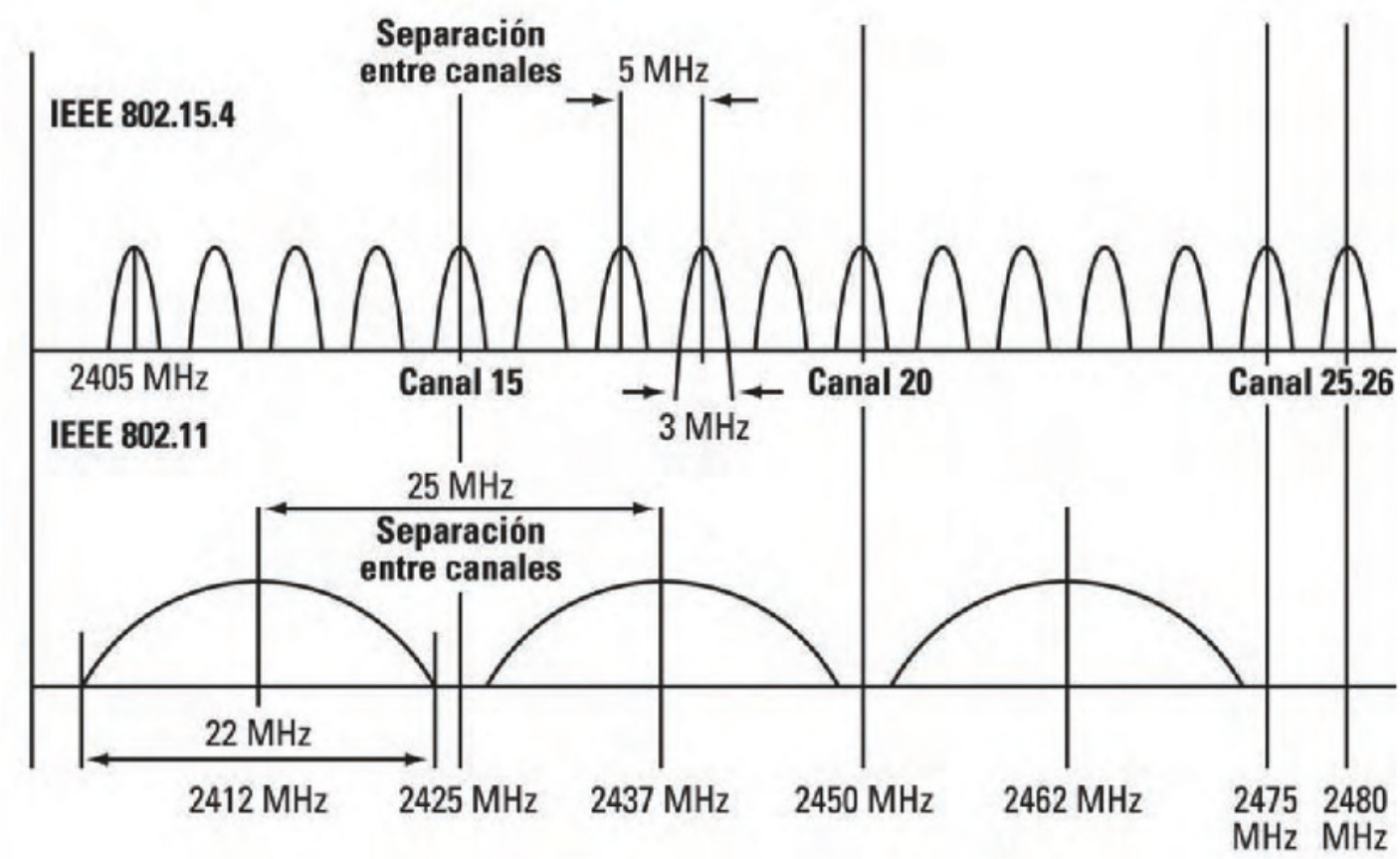


FIGURA 3. **Distribución de los canales de** 802.15.4 **y** ZigBee**. Notemos cómo es compartida la misma banda con el estándar** 802.11g **(Wi-Fi).**

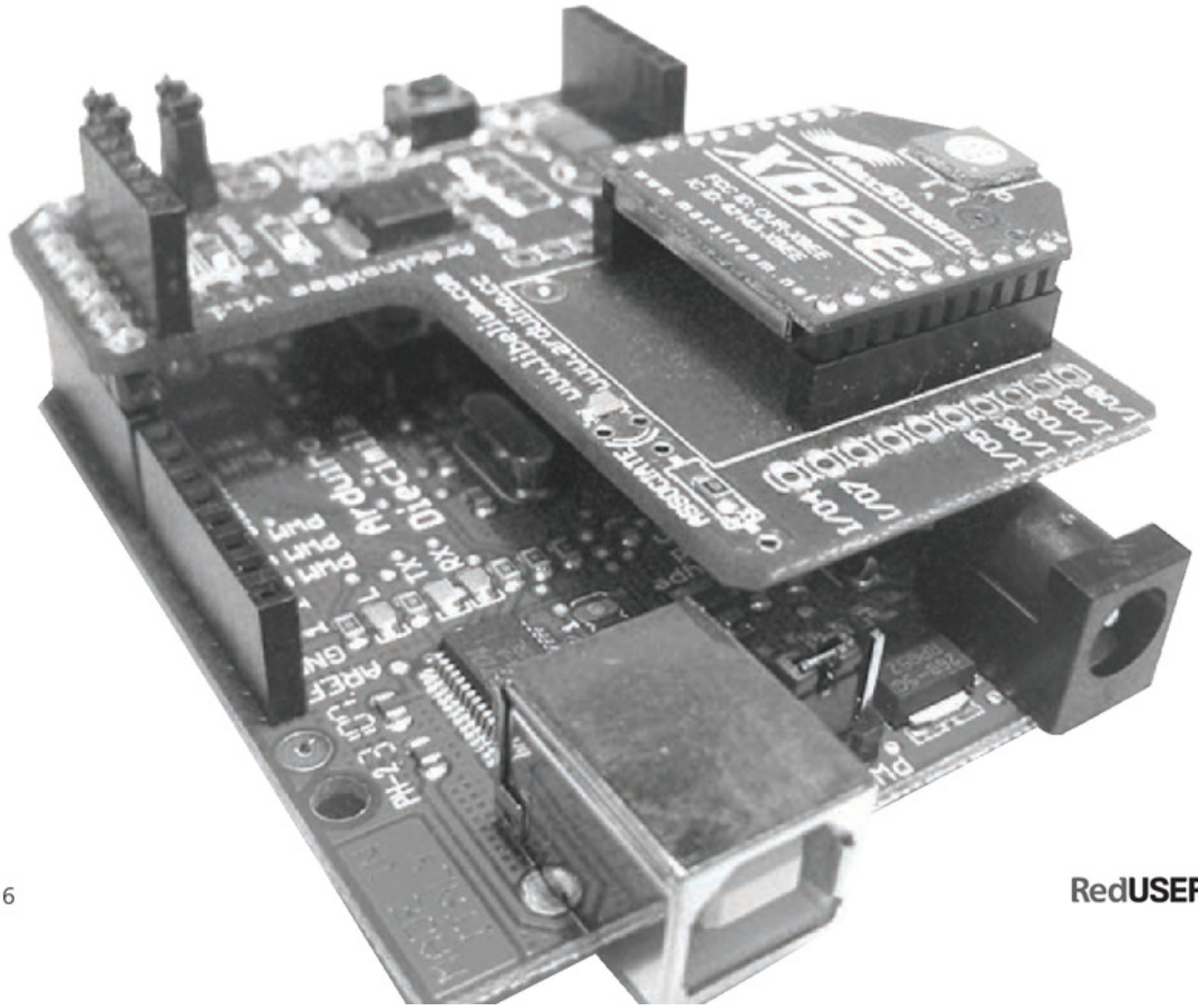
pondiente, **ZigBee** se basa en él y le agrega una mayor versatilidad para la mayoría de las aplicaciones (**Figura 3**).

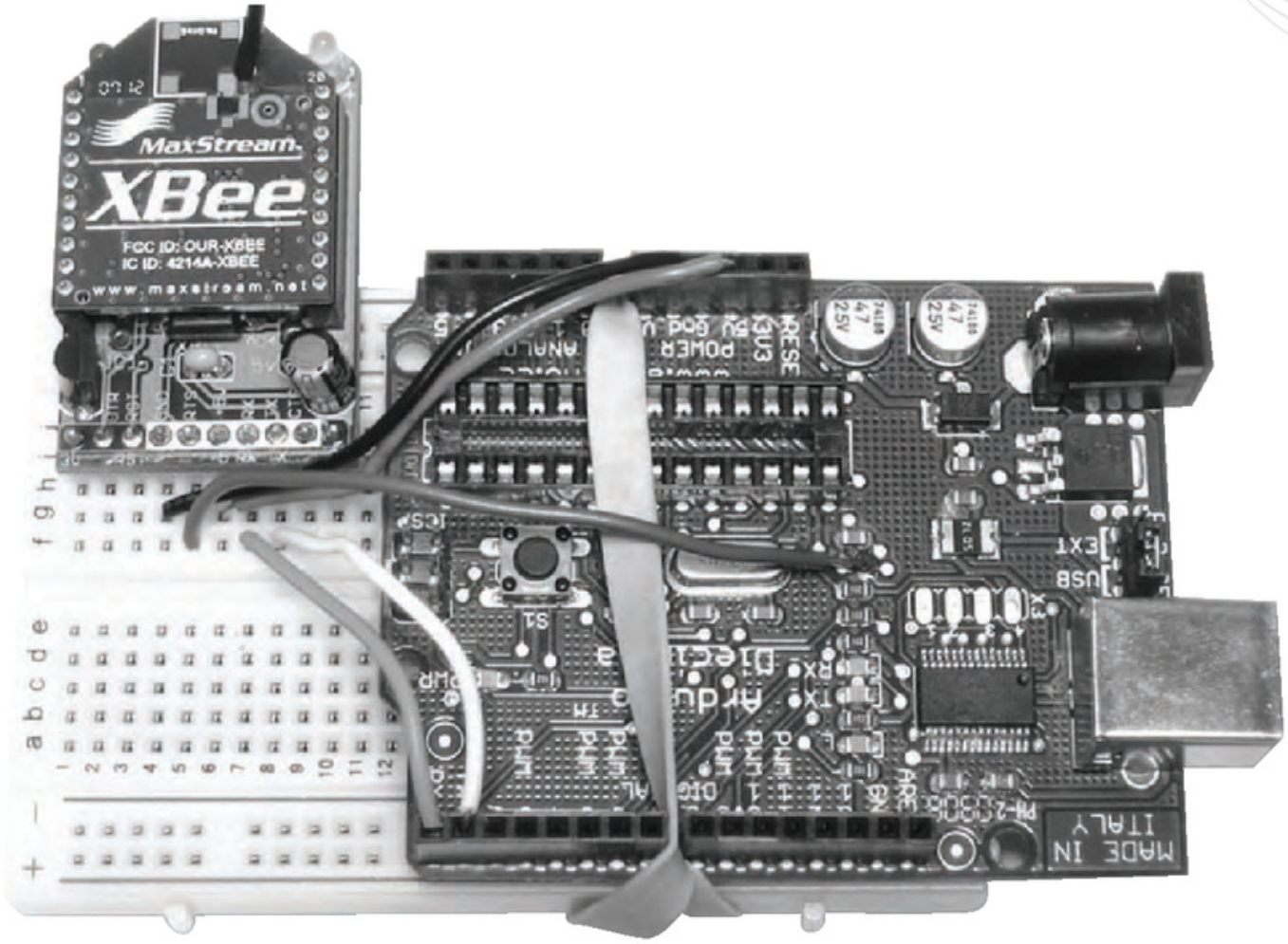
El estándar **802.15.4** define una banda de frecuencias sobre la cual operar: **868 MHz**, **915 MHz** y **2,4 GHz**. Tomando como ejemplo la de 2,4 GHz, el estándar especifica **16 canales** (denominados 11 a 26), que están separados cada **5 MHz,** desde 2405 hasta 2480 MHz (frecuencia central). Por ejemplo, el canal 11 estará ubicado en 2405 MHz, y el 12, en 2410 MHz.

Contiene todos los pormenores en relación a la sincronización entre dispositivos: control de errores y selección automática del canal menos ruidoso (con menos interferencias). También es posible prefijar el canal deseado. Cada dispositivo tiene una dirección única en la red, por lo que pueden direccionarse con un número, tal como lo hacemos cuando efectuamos una llamada telefónica.

Dentro del estándar, se prevén todos los mecanismos necesarios para conocer si el receptor de un determinado mensaje lo ha recibido, ya que el mismo responde con un código **ACK,** que es un acuse de recibo.

Un punto muy importante para considerar es que la comunicación es **half-duplex,** es decir que sólo es posible emitir y recibir, pero no al mismo tiempo, como ocurre con las comunicaciones **full-duplex**. La tasa de transferencia es de **100 Kbps** en los dispo-





sitivos que operen en la banda de **868/915 MHz**, y de hasta **250 Kbps** en los que lo hacen en **2,4 GHz**.

ZIGBEE

Mencionamos al principio de este apartado que el estándar **ZigBee** se alza por sobre el estándar **802.15.4**. Para comprender las diferencias entre ellos, es preciso conocer las particularidades de ambos.

El **throughput** y la latencia miden la performance del sistema inalámbrico. El primero, también llamado velocidad efectiva, es la cantidad de información que podemos enviar por unidad de tiempo a través del canal; mientras que el segundo, o demora, mide el tiempo en que esos datos llegan y son procesados por el dispositivo remoto, para producir una respuesta. Como podemos observar, a menor latencia, mejor es el throughput.

Las redes basadas en **802.15.4** permiten transmitir de forma constante a **115200 bps** durante cortos períodos de tiempo. Dado que ZigBee es una arquitectura que extiende el estándar 802.15.4, el throughput resultante será menor, ya que la latencia introducida será más alta debido al procesamiento adicional que se hace sobre los datos, como veremos seguidamente.

En las redes **802.15.4** sólo es posible la comunicación directa entre los dispositivos. Es decir, no se puede dialogar entre dos dispositivos que no sean visibles entre sí, aunque exista uno intermedio que vea a ambos.

En resumidas cuentas, una red 802.15.4 no permite hacer routing (encaminamiento) entre dispositivos que no sean visibles. ZigBee nace como consecuencia de esta limitación, como de otras también, y da la posibilidad de comunicar dispositivos que no se ven entre sí, ni están al alcance uno del otro, empleando otros intermedios que cumplirán el papel de routers (**Figura 4**).

Debido al routing, una red ZigBee permite tener una red múltiplemente conexa. Es decir, pueden comunicarse dispositivos directos y distantes sin que haya comunicación radial entre sí, efectuando saltos intermedios en otros componentes ZigBee hasta llegar a destino.

Una red ZigBee permite transportar mensajes entre dispositivos que no tienen comunicación directa

En un dispositivo ZigBee, el stack o conjunto de aplicaciones de protocolos que corren dentro del procesador del dispositivo para gestionar y controlar la comunicación es más complejo que el de un dispositivo 802.15.4. Por lo tanto, uno ZigBee requiere mucha más memoria, y microprocesadores o microcontroladores más caros.

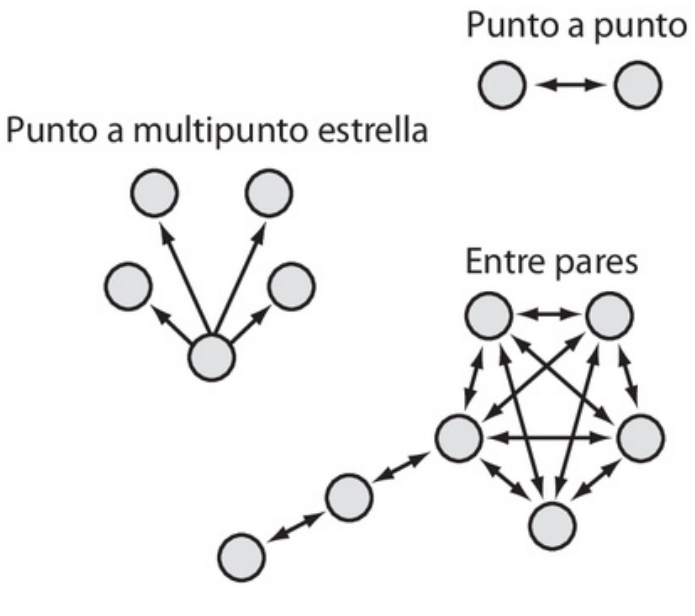
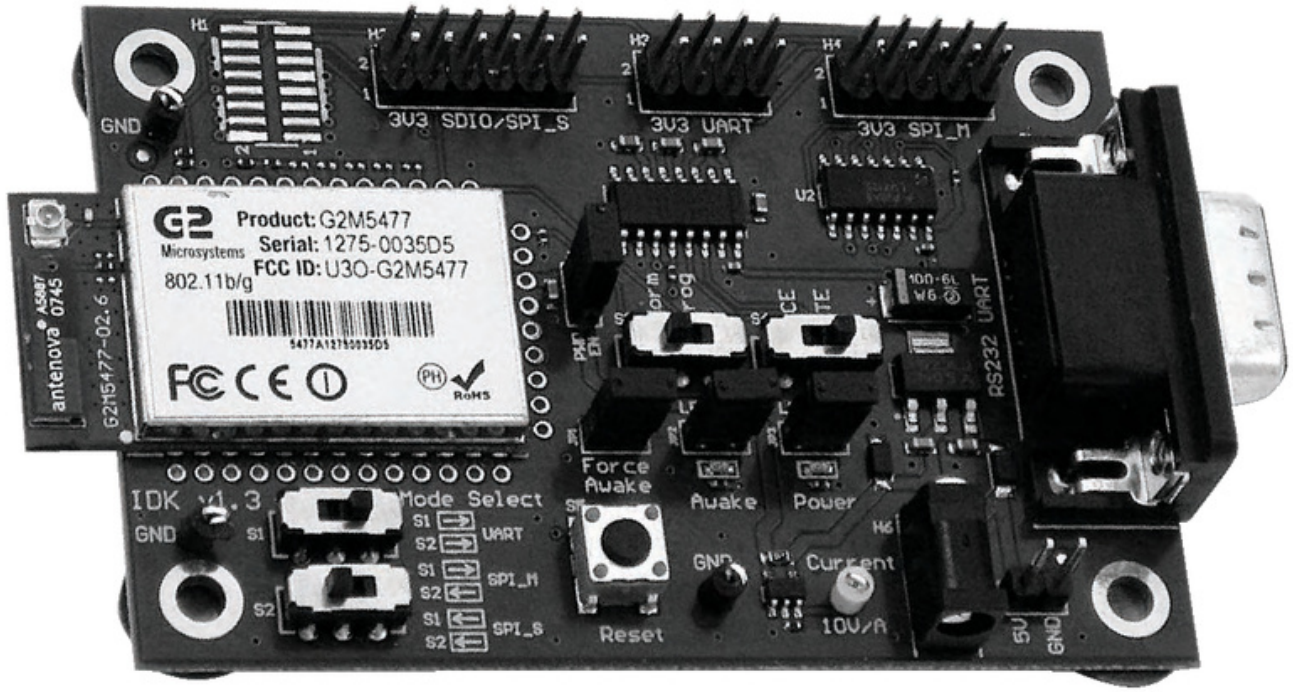


FIGURA 4. **La topología de la red en 802.15.4 sólo puede ser** peer-to-peer **y en** estrella **o** punto a punto**, que es la versión mínima de una estrella.**

ROUTING EN ZIGBEE

ZigBee incorpora el concepto de ruteo: los mensajes viajan de nodo en nodo hasta llegar a destino. Cada nodo conoce a sus vecinos, y si recibe un mensaje que no es para él, les pregunta a ellos cómo llegar al destinatario.



ZIGBEE EN CONCRETO

Una red ZigBee se basa en el estándar **802.15.4-2003** para la comunicación entre dispositivos comunicados directamente entre sí. Además, define un nivel de routing mediante el cual aquellos dispositivos que actúan como router pueden transportar mensajes para otro destinatario, lo que extiende el alcance de la red.

Existen tres tipos de nodos: coordinador, router y end-devices. El primero es el encargado de dar inicio a una red ZigBee, escogiendo el canal más silencioso y emitiendo una trama de datos llamada **Beacon Request,** para armar una lista de los dispositivos ZigBee encontrados.

El coordinador es el nodo que inicia y controla una red ZigBee

Los routers tienen la capacidad de encaminar los mensajes entre distintos dispositivos de la red y, además, de almacenar temporalmente aquellos destinados a los **end-devices**, que están durmiendo en bajo consumo hasta que despierten.

COMUNICACIÓN DE APLICACIONES

Basándose en el esquema de routing, las aplicaciones ZigBee se mandan mensajes entre sí, formando una gran aplicación distribuida que se encarga de resolver tareas específicas, de acuerdo con el perfil de cada una; por ejemplo, la automatización del hogar.

Los end-devices son dispositivos con funcionalidad reducida, ya que son los encargados de recibir o enviar los mensajes propiamente dichos a los sensores, actuadores o microcontroladores que deseamos manejar en forma inalámbrica (**Figura 5**).

Módulos prearmados (XBee y XBee ZB)

Si bien podemos diseñar y construir nuestros propios módulos 802.15.4 o ZigBee ajustándonos a las especificaciones del estándar, lo cierto es que éste es un trabajo muy complejo, ya que involucra diversos elementos y alta especialización, en particular, en RF (radio frecuencia). Por este motivo, existen módulos ya armados y listos para funcionar (**Figura 6**).

XBEE 802.15.4

Por medio de este módulo, podemos crear una red 802.15.4. El XBee 802.15.4 tiene una potencia de transmisión de **1 mW** y una sensibilidad del receptor de **-92 dBm**, por lo que podremos esperar un alcance de hasta 100 metros en espacios abiertos y de 30 metros en zonas urbanas. Existe otra versión de mayor potencia denominada **XBee –PRO 802.15.4,** de **60 mW** y una sensibilidad del receptor de **-100 dBm**, que alcanzan una distancia de hasta 1000 metros y 300 metros, respectivamente.

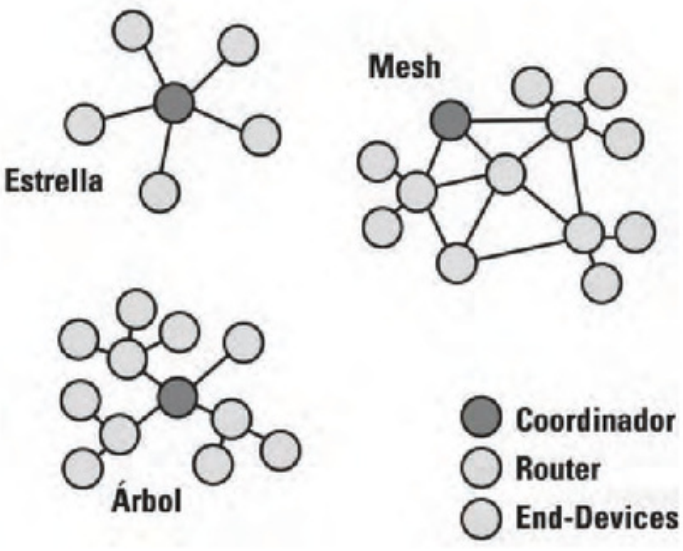


FIGURA 5. **Dada la presencia de routers, es posible establecer una topología en árbol o mesh, donde un end-device no es directamente visible para otro que quiera enviarle un mensaje.**

FIGURA 6. **Módulo XBee-PRO con antena externa. Es más potente y cuenta con un receptor más sensible, con lo que logra distancias superiores.**

CONEXIÓN PUNTO A PUNTO

Realizar una conexión punto a punto es una tarea muy simple, porque los módulos ya vienen con la configuración predeterminada. Lo único que tenemos que hacer es conectar ambos, uno en el extremo remoto y otro al control central, que debe ser capaz de conectarse al módulo por medio de una **interfaz UART**. De esta manera, los módulos operan en forma transparente; es decir, los datos que se reciben por el puerto serie de un módulo son transmitidos al módulo remoto a la salida de su puerto serie y, recíprocamente, el remoto, al central. Es conveniente especificar las direcciones concretas de los módulos para evitar conflictos con otros que agreguemos. Para esto, configuramos los parámetros **MY=<dirección propia>** y **DL=<dirección remoto>** antes de dar inicio a la red (**Figura 7**). Por ejemplo:

Módulo 1

MY=1234
DL=5678

Módulo 2

MY=5678
DL=1234

PUNTO A MULTIPUNTO CON COORDINADOR

Un **coordinador** es un dispositivo que opera como central de toda la red y que tiene la capacidad de almacenar hasta dos mensajes en espera de que los módulos correspondientes despierten para poder

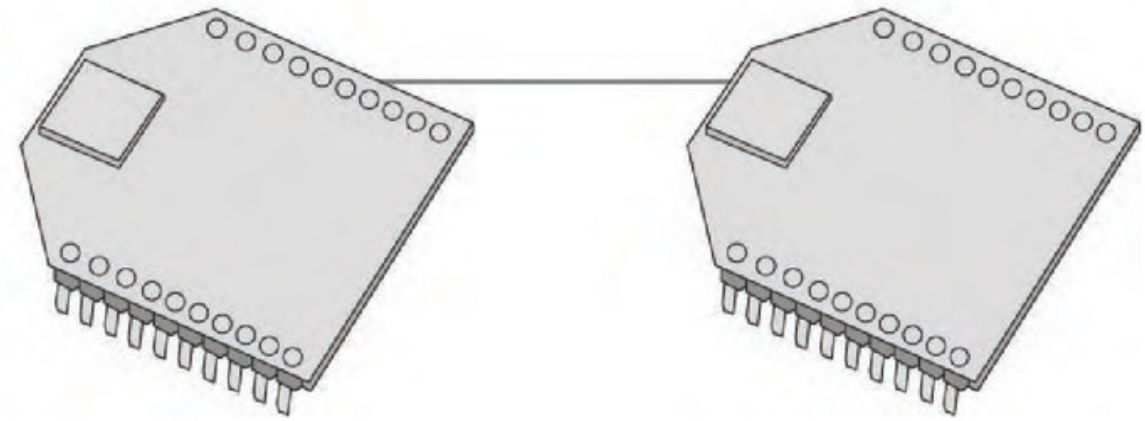


FIGURA 7. **Ésta es la manera de interconectar dos módulos punto a punto. Con los valores de fábrica, es posible alimentarlos y ponerlos a funcionar de inmediato.**

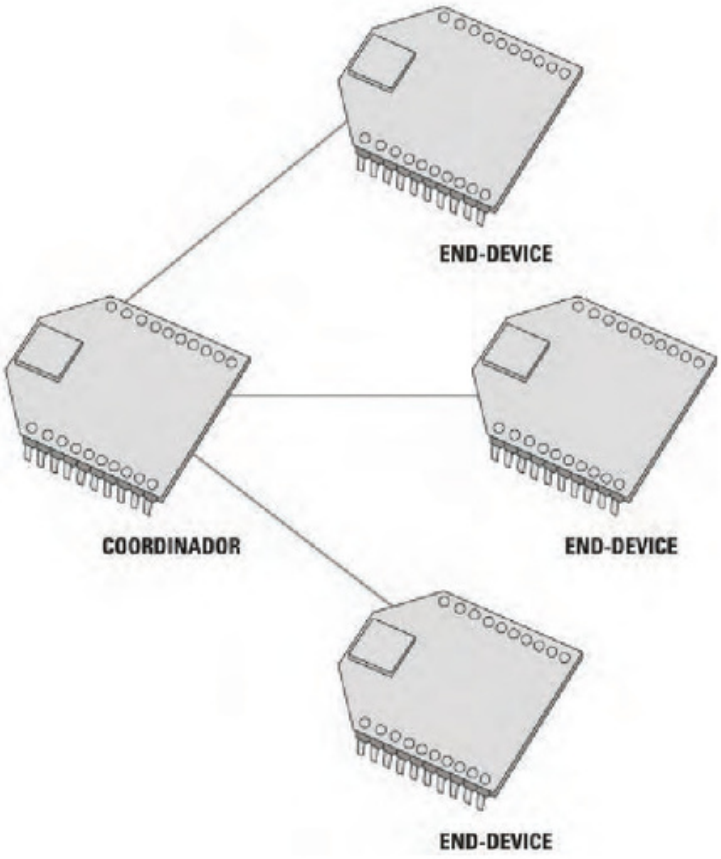


FIGURA 8. **Conexión punto a** multipunto **con coordinador. En esta configuración, todos los** end-devices **se dirigen al coordinador, el que conocerá las direcciones de todos los módulos y almacenará hasta dos mensajes en espera de que el módulo direccionado despierte.**

Un coordinador es un dispositivo que opera como central de toda la red, con capacidad de almacenar hasta dos mensajes en espera

entregárselos. Esto soluciona el problema de consumo en las redes punto a punto, en el que ningún módulo puede entrar en el modo de bajo consumo. El número máximo de dispositivos dependerá del tráfico de la red.

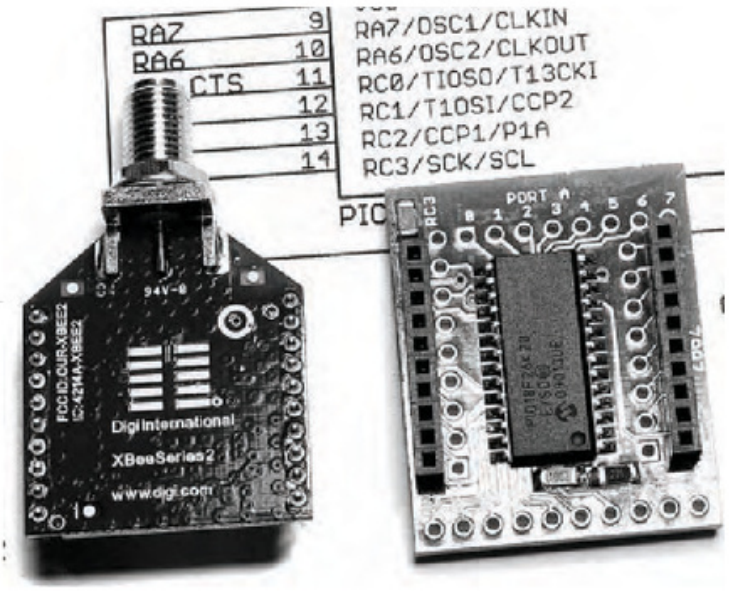
Para crear una red multipunto con coordinador, debemos configurar el módulo que será coordinador con el parámetro **CE = 1,** mientras que en todos los remotos deberá estar configurado **CE = 0**.
Al iniciar la red, cada remoto manda mensajes para descubrir a un coordinador y asociarse a él. El coordinador le asigna una dirección al remoto. Además, en los remotos debe configurarse el **bit 2** en el parámetro **A1**. Para que el coordinador asocie remotos que se lo solicitan, hay que configurar el **bit 2** del parámetro **A2** en el coordinador.

La unidad central o computadora o microcontrolador que esté conectada al coordinador necesitará saber de qué remoto provienen los mensajes. Para esto, hay que agregar al mensaje de los remotos algún dato que los identifique, o utilizar un modo especial y documentado en las especificaciones del módulo, llamado modo **API**.

Para enviar un mensaje desde el coordinador hasta un remoto, tenemos que cambiar el parámetro **DH-DL** del coordinador por la dirección del remoto en cuestión. Los remotos siempre transmiten al coordinador (**Figura 8**).

MÓDULOS XBEE ZB

Los módulos **XBee ZB** permiten crear una red ZigBee. Están disponibles en distintas potencias, y cuentan con entradas y salidas digitales, y con entradas analógicas,



para efectuar tomas de mediciones reales en la entrada del conversor A/D de que disponen. También tienen una salida **PWM** que permite leer la intensidad de la señal de recepción de acuerdo con su ciclo de trabajo (al igual que el XBee 802.15.4), entre otras utilidades, según la configuración del módulo (**Figura 9**).

El módulo puede configurarse para un determinado modo de funcionamiento, que dependerá del programa o firmware que le grabemos, por medio del software gratuito **X-CTU**. A través de él, también vamos a ajustar los parámetros que sean necesarios, de una forma sencilla. Los modos de funcionamiento son dos:

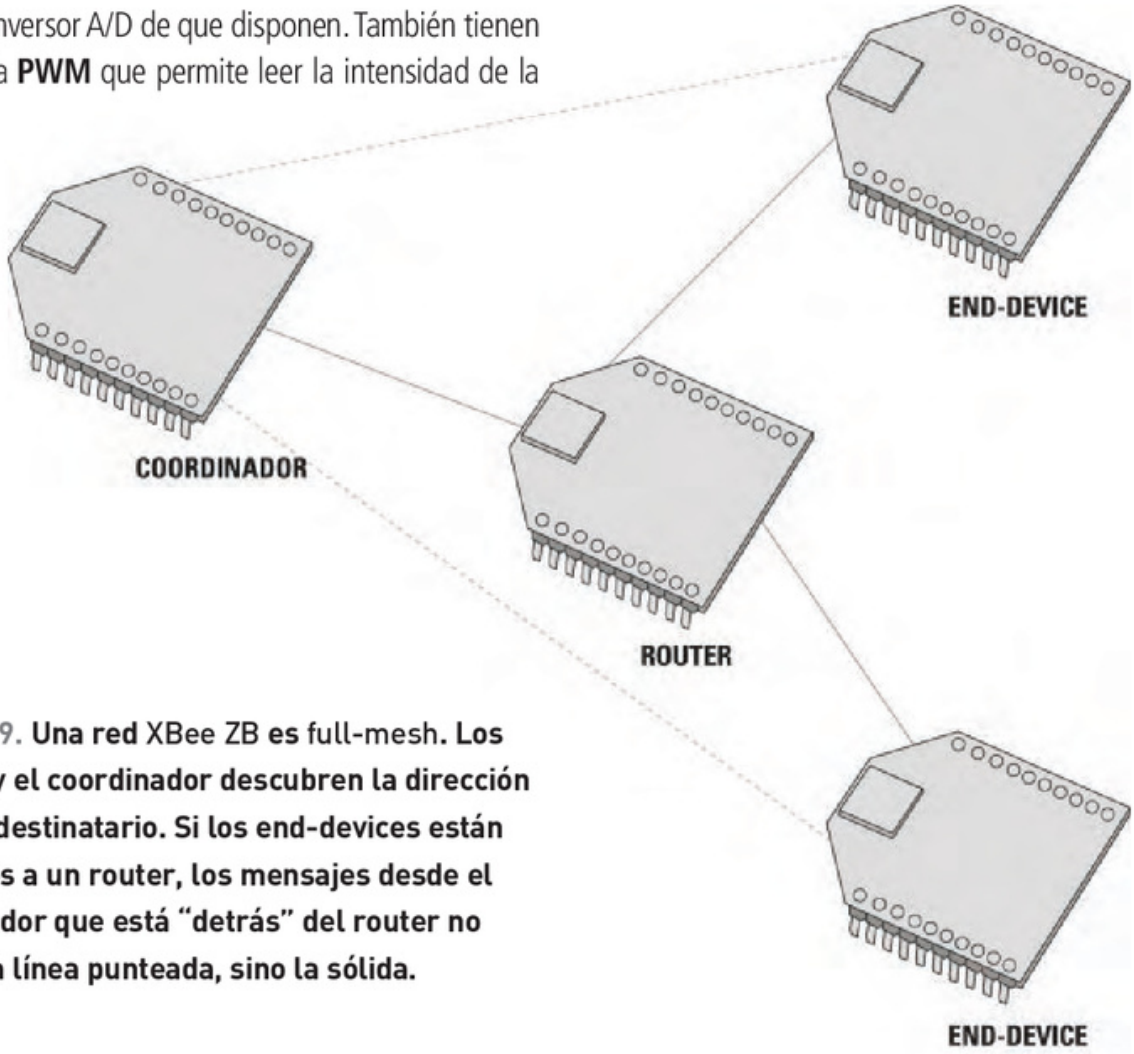
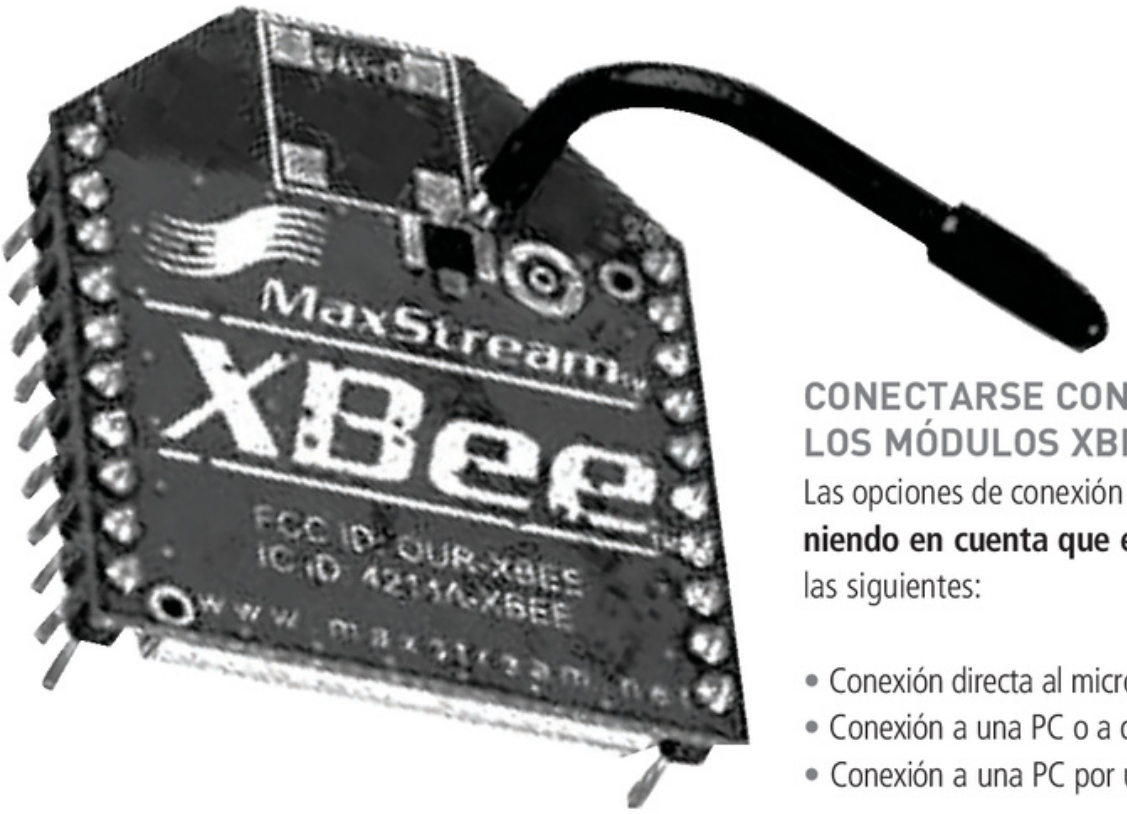


FIGURA 9. **Una red** XBee ZB **es** full-mesh. **Los routers y el coordinador descubren la dirección hacia el destinatario. Si los end-devices están asociados a un router, los mensajes desde el coordinador que está "detrás" del router no siguen la línea punteada, sino la sólida.**



• **Transparente + AT:** el módulo envía y recibe los datos de los remotos por medio de su interfaz serie, y es controlado por los comandos AT "escapados".
• **API**: por medio de un mismo mensaje, permite codificar los datos y comandos de una forma sencilla, en una única interfaz serie.

COMUNICACIÓN A UN SITIO CENTRAL

Éste es el modo predeterminado al cargar el firmware AT y que opera de forma transparente. Es decir, une de manera inalámbrica los puertos serie de los módulos remotos con el de un sitio central, y obtiene los datos como si se tratase de un cable directo que los vinculara. El sitio central es, por defecto, el coordinador de la red.

CONECTARSE CON LOS MÓDULOS XBEE

Las opciones de conexión con los módulos XBee, **teniendo en cuenta que el módulo es de 3 V**, son las siguientes:

- Conexión directa al micro en los pines de una UART.
- Conexión a una PC o a otro dispositivo por RS-232.
- Conexión a una PC por un puerto serie USB.

En todos los casos, nos basamos en lo visto en el **Capítulo 3**.

CONEXIÓN CON LA PC Y CONFIGURACIÓN

La interfaz de conexión con la PC nos permitirá configurar los módulos XBee, actualizar el firmware y utilizarlos para recibir la información enviada por los módulos remotos. En la **Figura 10** vemos el esquemático para conectar los módulos XBee a la PC mediante un puerto serie USB; sigamos las instrucciones en el **Paso a paso 1.**

La interfaz de conexión con la PC permite configurar los módulos XBee

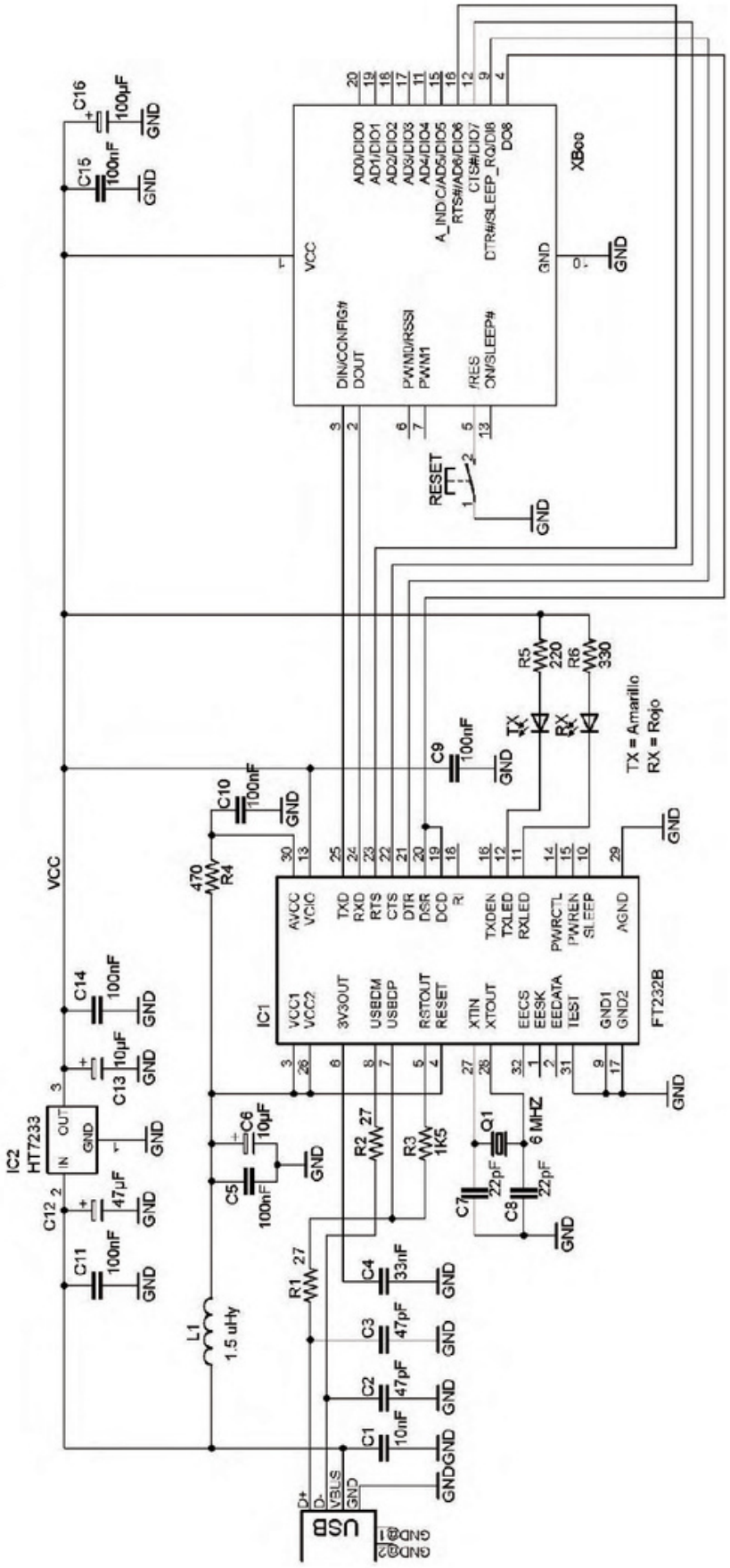
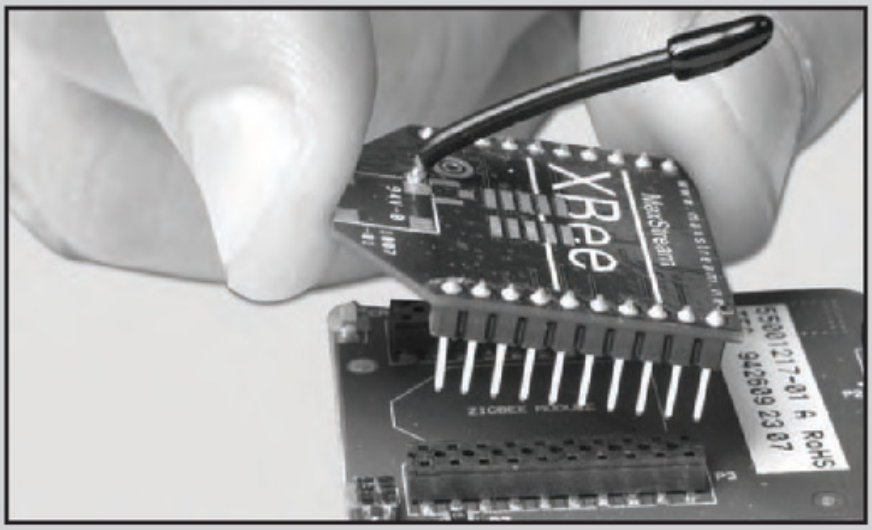


FIGURA 10. El FT232 viene en SMD, y los módulos tienen pines de paso de 2 mm, que requieren un zócalo.

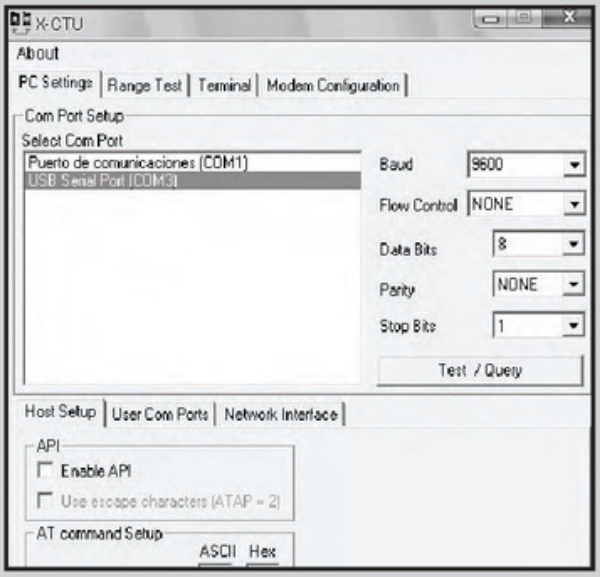
PASO A PASO /1
Conexión con la PC y configuración

1



La hembra de pines de 2 mm viene en formato doble, ya sea SMD o de inserción. Como siempre, debe ubicar los capacitores de desacople lo más cerca posible del chip correspondiente: regulador, FT232, XBee. Inserte el módulo XBee en el zócalo, teniendo cuidado de hacer coincidir los pines correctamente.

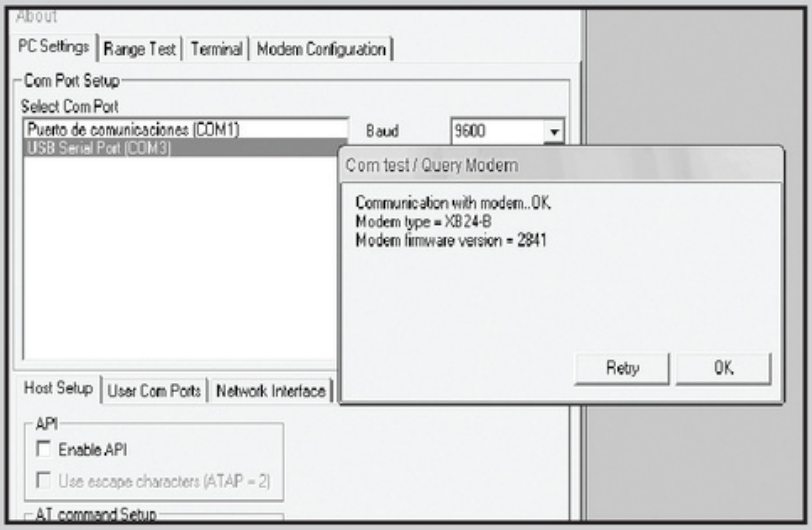
2



Instale el programa **X-CTU** desde la página web del fabricante: **www.digi.com**. Conecte el puerto USB al hardware de interfaz y corra el programa. Seleccione el puerto serie USB.

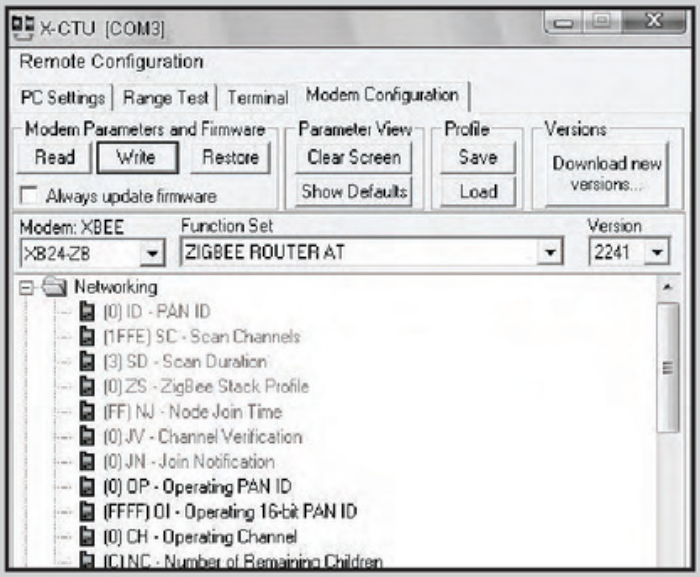
PASO A PASO /1 (cont.)

3



Presione el botón **`Test/Query`**; se abrirá una ventana que indicará la revisión de firmware del módulo. De no ser así, debe revisar las conexiones con el módulo.

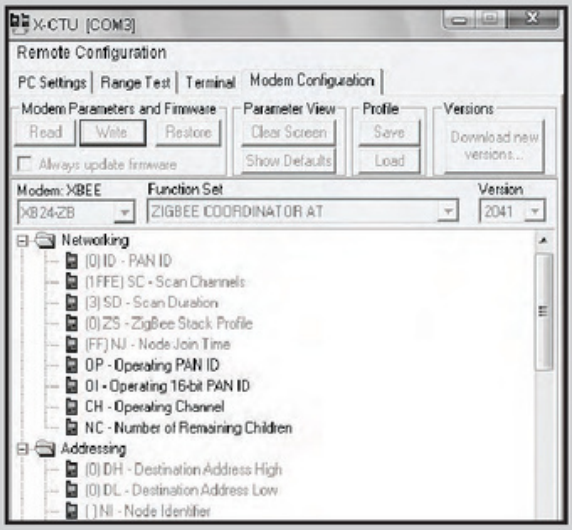
4



En la solapa **`Modem Configuration`**, puede observar la configuración del módulo. Para obtenerla, presione el botón **`Read`**. Realice cambios operando sobre la ventana, y lo que cambie se verá en azul. Modifique el valor de **`JV`** (**`Channel verification`**) a **`1`** (**`Enable`**). Guarde lo hecho con el botón **`Write`**.

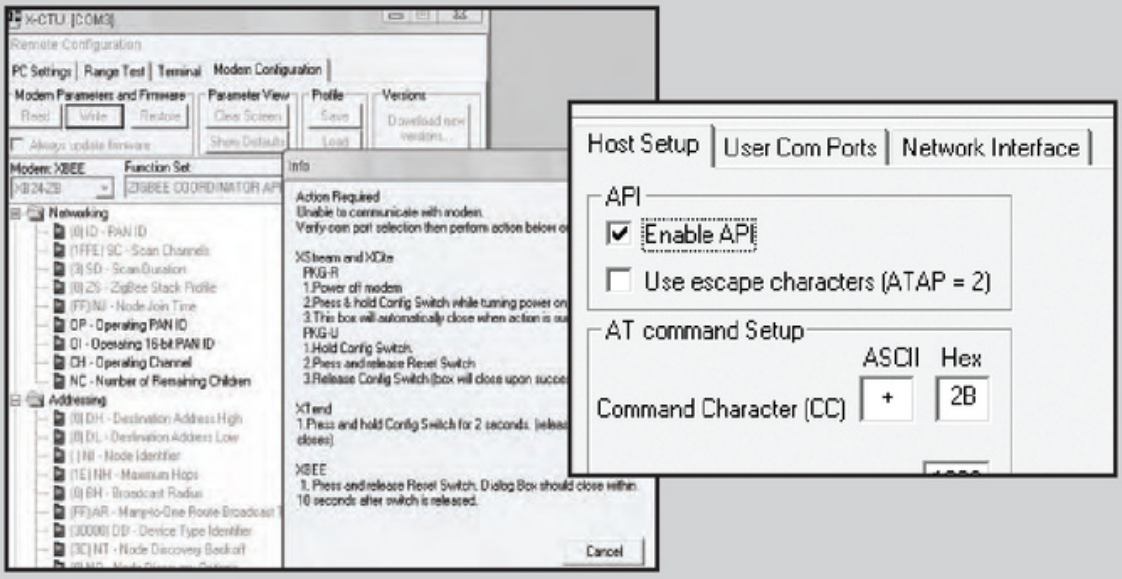
PASO A PASO /1 (cont.)

5



Uno de los módulos deberá ser coordinador. Desconecte la interfaz, cambie el módulo y vuelva a leerlo. En **`Function Set`**, seleccione la opción **`ZIGBEE COORDINATOR API`** y, en **`Version`**, **`2141`**. Grabe presionando el botón **`Write`**.

6



La comunicación con el firmware API (en vez de AT) requiere que se habilite este modo de operación en X-CTU. Como no está así, una vez terminada la actualización, aparecerá un anuncio que debe cancelar. Luego, proceda a habilitar el modo API en X-CTU, marcando el casillero **`Enable API`**, dentro de la solapa **`Modem Configuration`**.

Red de sensores ZigBee

Una **red ZigBee** debe tener un coordinador, que se encarga de iniciarla y darle un identificador. Éste, además de un router, puede derivar mensajes a otros dispositivos y ser padre de **end-devices**, dispositivos que pueden estar en bajo consumo, con el receptor apagado. Puede haber otros **routers** y **end-devices**; estos últimos los pondremos en sitios alejados, para tomar mediciones. En esta red utilizamos, en lo posible, la configuración por defecto.

EL COORDINADOR

El coordinador de la red será, en nuestro caso, el sitio central donde recibiremos los reportes. Tendrá **firmware API**, para reportar los mensajes que reciba, indicando la dirección de quien los manda y los detalles de la medición. Su dirección es por defecto **0**, que coincide con el destino prefijado para transmitir en los otros dispositivos. Si no utilizamos **routers**, no debemos configurar nada más. En caso contrario, aplicamos los cambios que hagamos en ellos, también al coordinador. Recordemos que cuando configuramos el coordinador, tenemos que habilitar el modo API marcando el casillero **`Enable API`**, en la solapa **`Modem Configuration`**, y revertir esta acción cuando pasamos a configurar otro dispositivo (**Figura 11**).

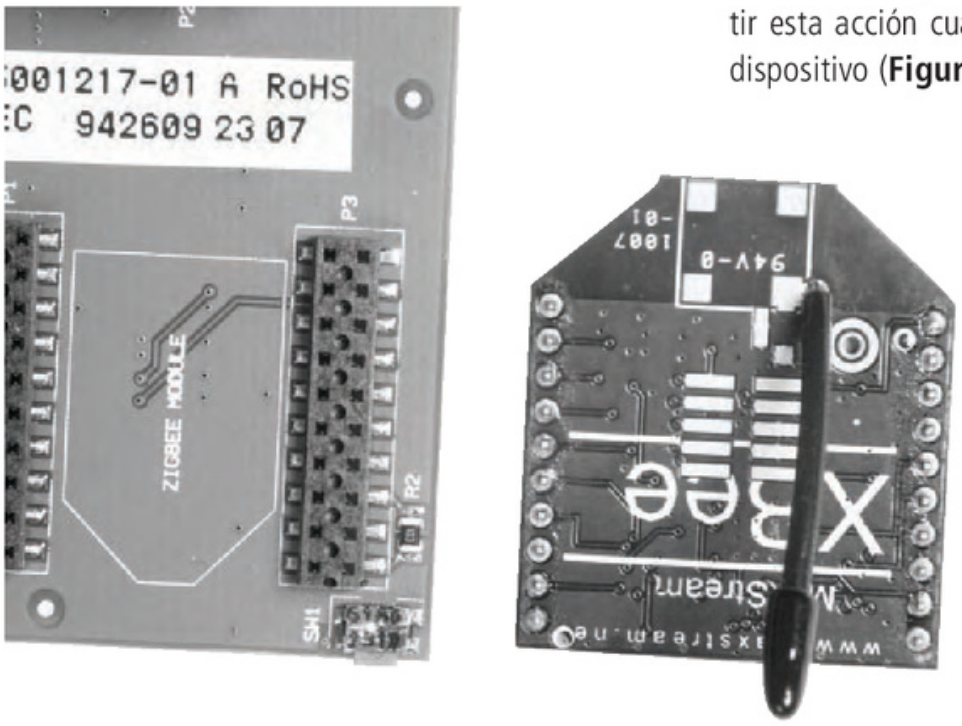


FIGURA 11. **El kit Rabbit RCM4500W incorpora un módulo XBee con interfaz USB para oficiar de coordinador de una red ZigBee.**

MODO API

El fabricante denomina modo API a una forma de comunicación con el módulo. En ella, en vez de comandos AT y datos transparentes, se utiliza una trama de información con encabezado, checksum e identificadores. Esto facilita la operación de envío de mensajes.

LOS ROUTERS

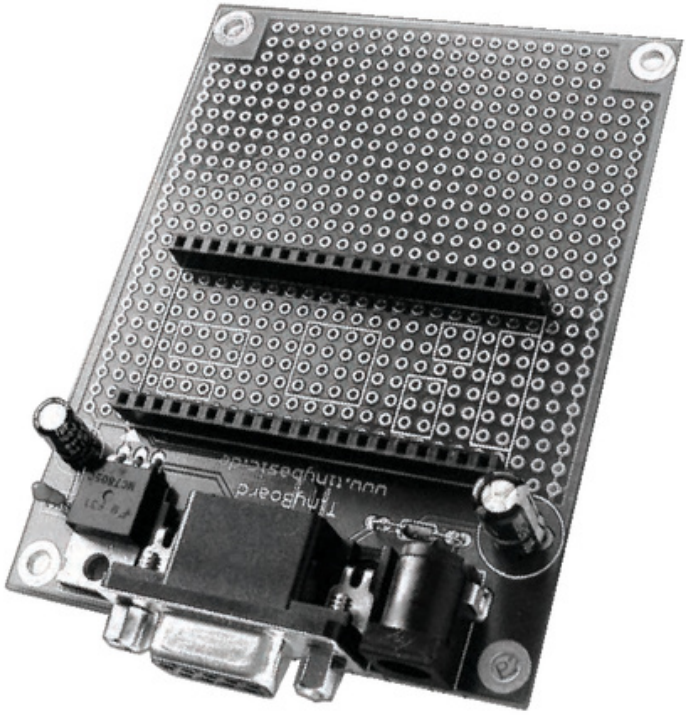
Los routers están siempre atentos y atienden los mensajes para los end-devices, ya que éstos pueden permanecer en bajo consumo durante largos períodos de tiempo, con su receptor apagado. También recibirán sus reportes al despertar y los retransmitirán a destino. En los routers configuramos el parámetro **JV = 1** para que busquen un coordinador al arrancar. Dejamos el parámetro **ID = 0**, para que se asocien a cualquier red (de todos modos, tenemos sólo una). También configuramos los parámetros **SP** y **ST** igual que lo hacemos con los **end-devices**.

LOS END-DEVICES

Para armar end-devices, grabamos el firmware **ZIGBEE END DEVICE AT 2841** en un módulo. Configuramos el parámetro **SP** para el tiempo en que queremos que el módulo duerma (tiempo entre reportes).

Aquí podemos configurar los pines para que sean asignados al **conversor A/D** interno, entradas o salidas digitales, mediante los parámetros **D1** a **D4**, según lo que deseemos conectar. El parámetro **IR**, que se encuentra en la carpeta **[I/O Sampling]**, nos permite configurar el tiempo entre reportes cuando el módulo está despierto. Es decir, transcurrido **SP**, reportará cada **IR** hasta que transcurra **ST** y, luego, volverá a dormir.

En modo API, el coordinador entrega la dirección de quien envía el mensaje



Si queremos conectar un micro al puerto serie, éste recibirá los mensajes que se envíen al módulo, y deberá contestar mientras se encuentre despierto.

LECTURA DE LOS REPORTES

El formato **API** tiene sus vericuetos, ya que está pensado para ser parte de un sistema de comunicaciones. Sin embargo, podemos leer los mensajes de manera muy simple, como si fuera una tabla. Por ejemplo:

- El byte en la posición 3 es el identificador de mensaje. El **valor 0x90** corresponde a datos por puerto serie, en tanto que el **valor 0x92** es una muestra de **I/O.**
- Los 8 bytes a continuación son la dirección del remitente.
- El mensaje se encuentra a partir de la posición 15.

En un mensaje de I/O, las entradas y salidas digitales (si hay alguna habilitada) se leen en los bytes que están en las posiciones **19** y **20**. Los canales analógicos habilitados están a continuación.

La información completa del formato del mensaje puede obtenerse de la hoja de datos del módulo, en la página web del fabricante. Vemos un ejemplo en la **Figura 12**.

Control por RF

Si bien la radiofrecuencia (**RF**) requiere de un particular cuidado en el diseño y la construcción, existe una serie de módulos prearmados que facilita notablemente su aplicación. De esta manera, conectándonos directamente a un pin del microcontrolador, podemos dotar rápida y fácilmente a nuestro proyecto de la capacidad de controlar o ser controlado de forma remota. En la **Figura 13** vemos un esquema simplificado.

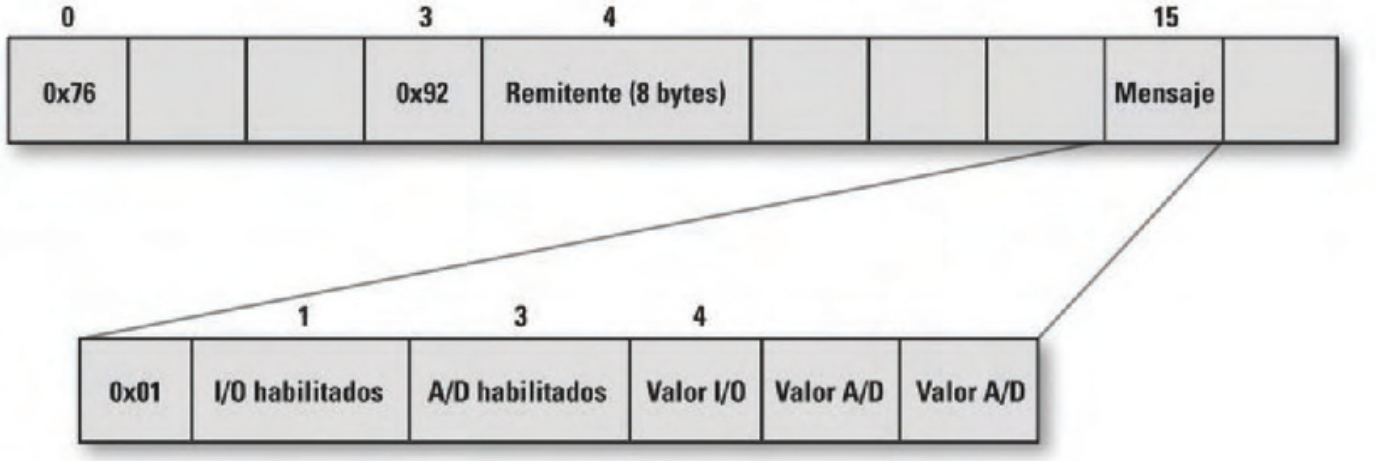


FIGURA 12. **Ejemplo de un mensaje de I/O en formato API. Podemos acceder a la información en posiciones fijas.**

CONOCER MÁS SOBRE API

Implementar el modo API no es difícil, pero requiere algo de experiencia en comunicaciones. En la hoja de datos del módulo hay simples ejemplos, pero podemos encontrar más información sobre 802.15.4, ZigBee y los módulos XBee en www.ldir.com.ar/libros/equisbi.

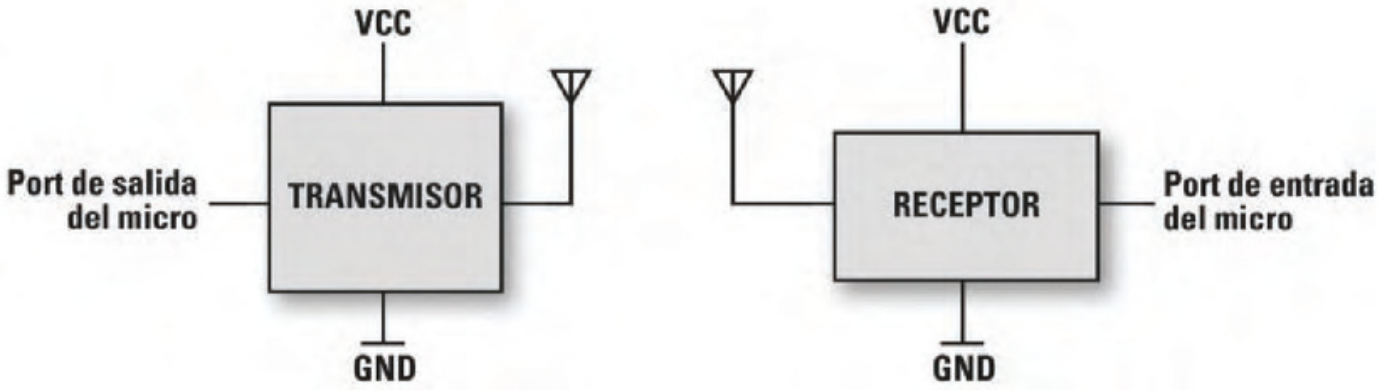


FIGURA 13. **Uso de los módulos transmisores y receptores de RF.**

Muchos codificadores para uso en módulos de RF trabajan con un ancho de pulso diferente para el 1 y el 0

EL TRANSMISOR

Los módulos transmisores de **RF** (**Figura 14**) en su mayoría utilizan un esquema de modulación denominado **ASK** (*Amplitude Shift Keying*). En él, las señales de datos ocasionan que la amplitud de la portadora varíe entre dos niveles. En general, se utiliza **OOSK** u **OOK** (*On-Off Shift Keying* u *On-Off Keying*), que consiste en transmitir portadora ante un estado lógico y anularla en el otro.

El esquema circuital de un transmisor es muy simple; por lo general, es sólo un transmisor y, a lo sumo, un elemento estabilizador, como un resonador **SAW** (*Surface Acoustic Wave*, onda acústica de propagación superficial). Posee un pin para recibir los datos, y dos pines de alimentación. En algunos casos, existe un tercer pin para habilitar la transmisión.

EL RECEPTOR

Los módulos receptores suelen ser circuitos **regenerativos** o **súper-regenerativos**, con un detector de

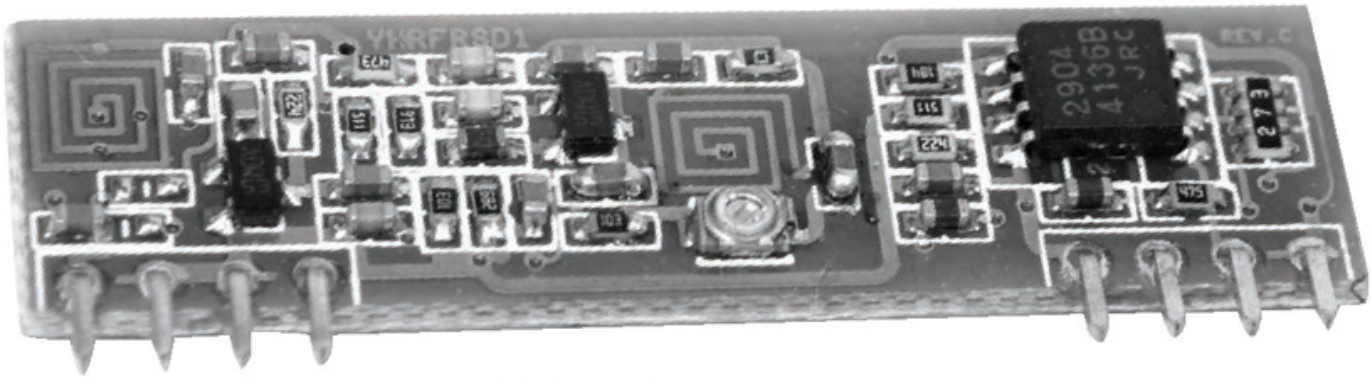


FIGURA 14. **Un módulo receptor de RF. El circuito integrado que se observa es el comparador.**

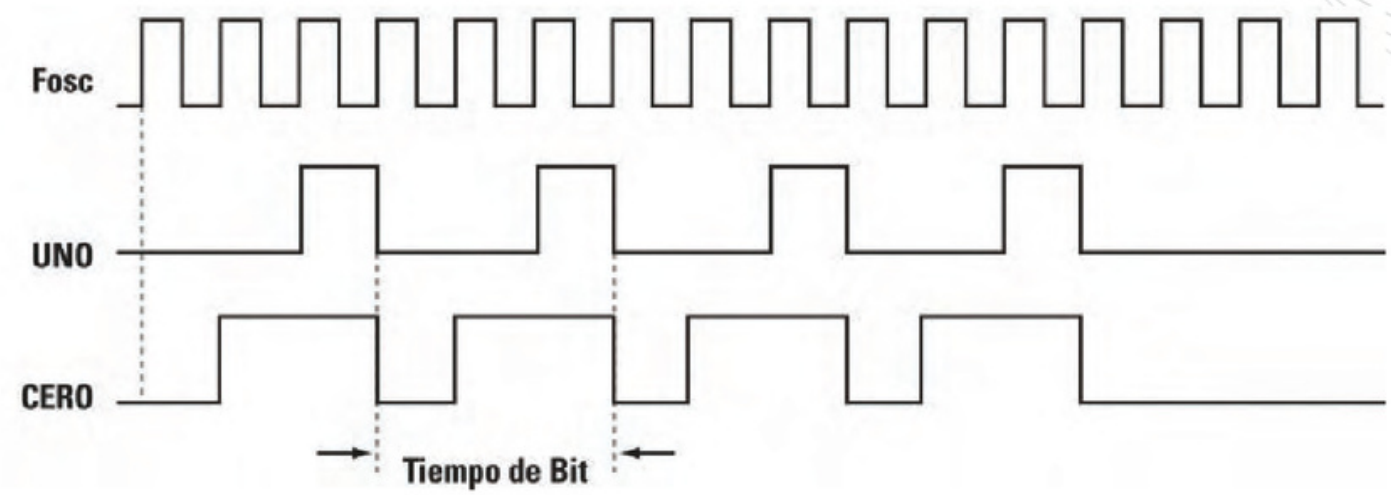


FIGURA 15. **La codificación de unos y ceros en un HT12E, muy utilizado como transmisor de telecontrol.**

umbral basado en un comparador a la salida. Estos receptores son circuitos con prestaciones muy por debajo de las que ofrece el más simple de los receptores **súper-heterodinos** (la clásica radio AM portátil). Se trata de receptores sencillos, económicos y, por lo general, sin estabilización de frecuencia. Su diseño los hace sensibles y poco selectivos, es decir, capaces de detectar señales débiles sin ser muy exigentes con la frecuencia. Estos módulos disponen de un pin que entrega la salida de datos (la salida del comparador) y de dos pines de alimentación.

Un receptor de este tipo, en ausencia de señal, presenta ruido a la salida. En presencia de una portadora constante, la señal de salida indica el estado correspondiente (en general, un 1 lógico). En muchos casos, retorna otra vez al estado inactivo, debido a que estos módulos suelen estar diseñados para trabajar con señales que varían constantemente y no, para transmitir estados lógicos permanentes. Por estos motivos, no se los suele conectar directamente a una **UART,** sino que se codifican los datos, y se agrega un preámbulo al inicio de la transmisión para detectar la presencia de una señal válida frente al ruido.

La codificación usada suele mantener un tiempo de pulso constante, y emplea un ciclo de trabajo para un nivel lógico y otro diferente para el otro. Como vemos en la **Figura 15**, es común aplicar la relación **1/3** para un estado y **2/3** para el otro. De este modo, se puede recuperar la información aun frente a modificaciones del ancho de pulso, lo cual

HISTORIAS DE RECEPTORES

El receptor regenerativo se basa en la aplicación de realimentación positiva, formando un amplificador a punto de oscilar. Fue inventado en 1914 por Edwin Armstrong, quien además desarrolló el receptor súper-regenerativo en 1922, y el súper-heterodino en 1918.

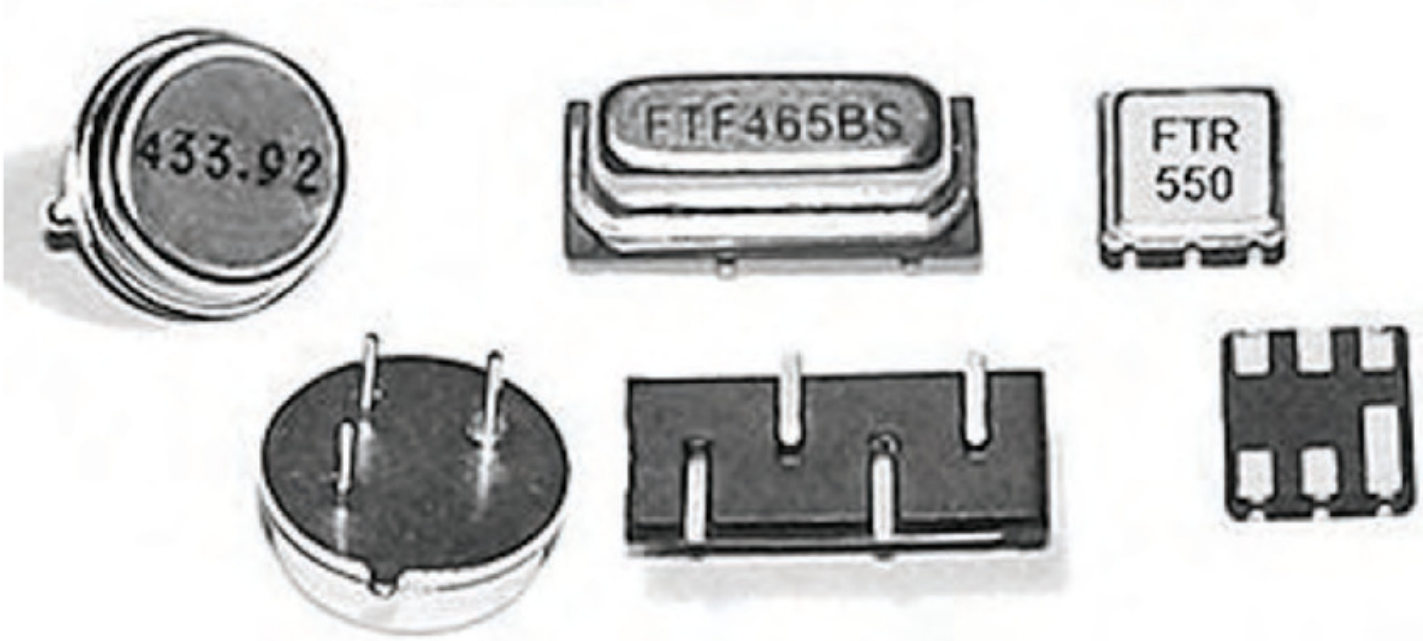
es frecuente en este tipo de receptores. Para recuperar la información codificada de esta manera, podemos observar la duración del pulso y compararla con un tiempo preestablecido, u observar el estado en la mitad del tiempo de bit.

Algunos módulos entregan también la salida analógica del receptor propiamente dicho, de forma que el usuario pueda discernir entre señal y ruido, o proveer un mejor detector. El rango de frecuencias utilizado suele ser la banda de **433 MHz**, destinada a este tipo de usos sin requerir licencia; aunque hay módulos para otras frecuencias.

Control por **infrarrojos**

El control remoto por infrarrojos tiene algunos estándares de facto, como **RC5** y **RECS80**; se trata de maneras de codificar la información que han sido adoptadas por un grupo de fabricantes. El esquema general es similar al visto para los módulos de **RF**. Las diferencias se producen en la forma de codificación, los tiempos asignados al preámbulo y los bits, así como en la generación de un **anti-código** para realizar detección de errores en el receptor.

Un transmisor de control remoto envía la información así codificada pulsando la emisión de un LED infrarrojo a una alta frecuencia; las más utilizadas están entre **30** y **38 KHz**. De este modo, durante la porción activa del ciclo, se transmiten pequeños pulsos de luz, y no se transmite nada durante la porción inactiva.



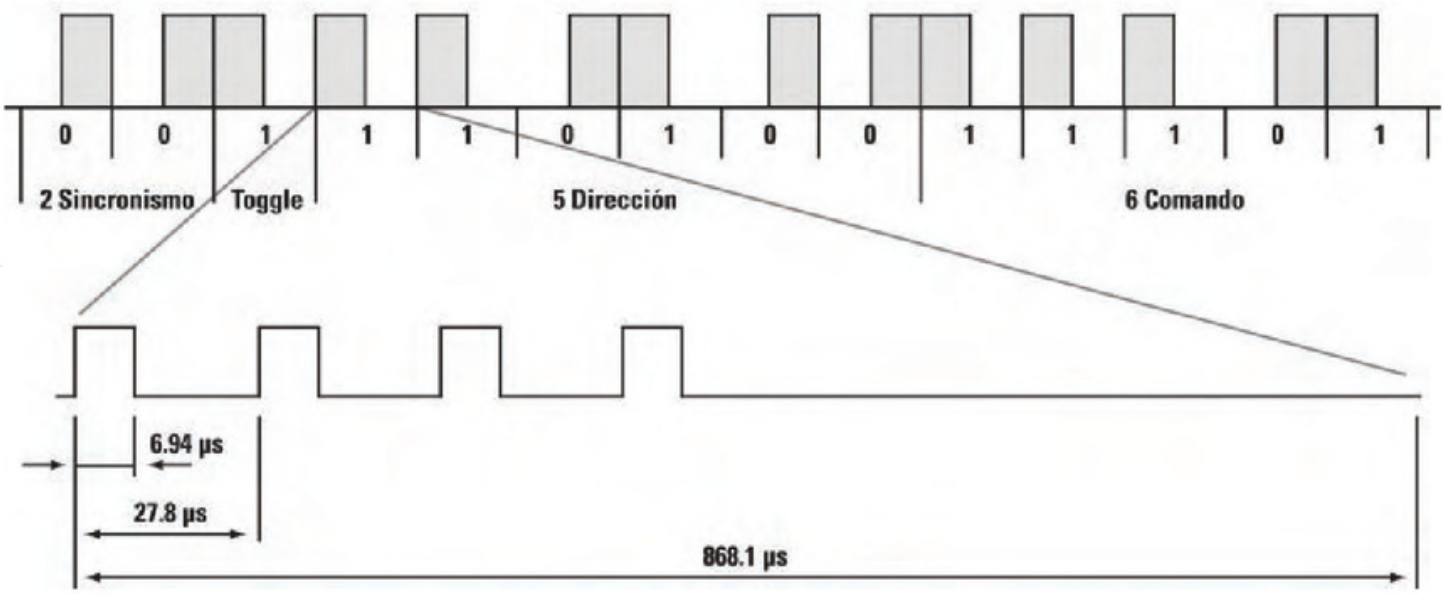


FIGURA 16. **El código** RC5**. Vemos el envío del comando** 011101 **a la dirección** 11010**.**

ESTÁNDAR RECS80

El código **RECS80** utiliza un esquema de ancho de pulso variable. Sus características principales son:

- Un pulso inicial para sincronización que dura **10 T** (10 veces un cierto tiempo T), seguido de un silencio de 4 T.
- Los 1 son codificados con un pulso de **1 T** y silencio de **3 T.**
- Los 0 se codifican como pulso de **1 T** y silencio de **1 T**.

El código RC5 es el más utilizado entre los fabricantes de controles remotos

Un valor común de **T** es **400** microsegundos.

Se envían **32 bits**, de los cuales los últimos 8 corresponden al **anti-código,** y los 8 anteriores, al código transmitido.

ESTÁNDAR RC5

RC5 (**Figura 16**) emplea una codificación similar a **Manchester** (utilizada por Ethernet), en cuanto a que siempre presenta transiciones en la mitad del tiempo de bit, y el sentido de esta transición identifica al bit transmitido. Por ende, podemos observar el estado durante la primera parte del bit, antes de la transición, para recuperar el valor. La trama transmitida contiene:

- 2 bits de sincronismo (00).
- 1 bit de control (toggle, cambia cada vez que se presiona una tecla).

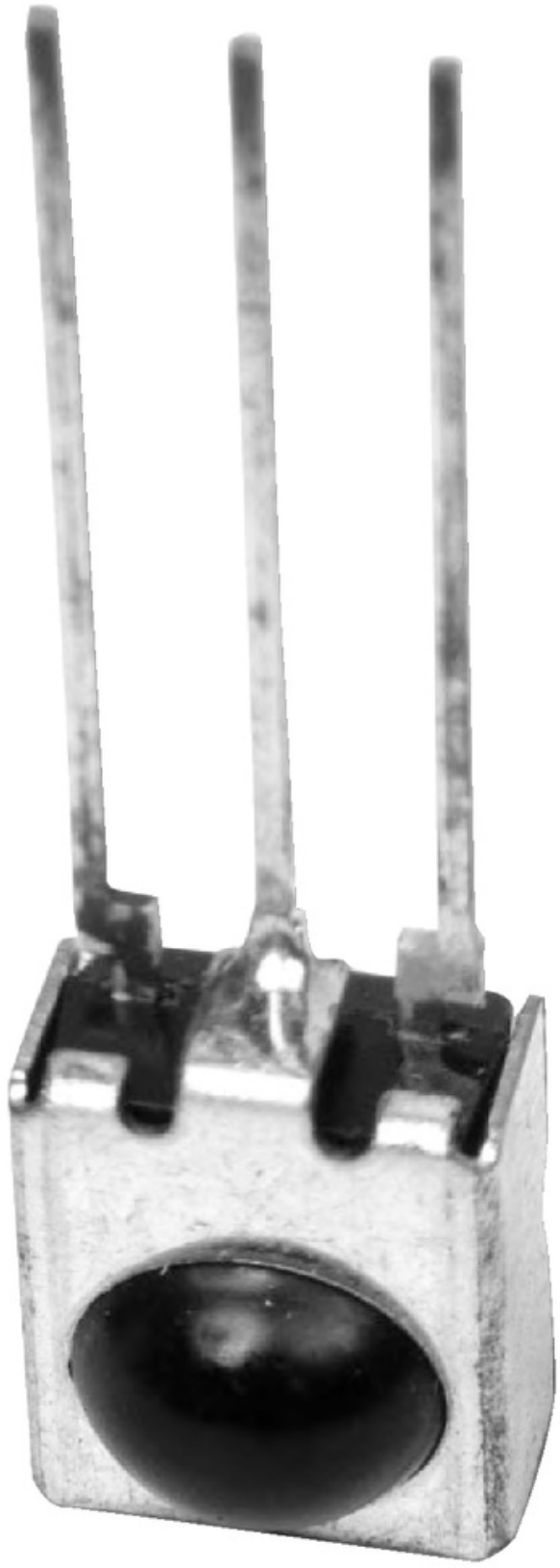


FIGURA 17. **Un receptor infrarrojo integrado. Observemos el filtro que aparece en su frente.**

- 5 bits para indicar el dispositivo.
- 6 bits para el comando.

El tiempo de bit consta de **32 ciclos** de la portadora de **36 KHz**. Durante cada ciclo de la portadora, si se transmite, el ciclo de actividad del transmisor es de 1/5; es decir, un pulso de 6,94 microsegundos seguido de un silencio hasta totalizar 27,8 microsegundos.

Una manera muy simple de generar estos códigos es empleando un puerto de salida un microcontrolador. También podemos generar la portadora con un oscilador externo, el cual controlamos mediante una compuerta AND comandada por el pin del microcontrolador.

EL RECEPTOR

Un receptor de control remoto posee un elemento sensible a los infrarrojos, como un fotodiodo o un fototransistor. Estos elementos generan una pequeña tensión al recibir luz, por efecto fotoeléctrico. Aun los sensibles al espectro infrarrojo tienen una cierta sensibilidad a la luz del día y, particularmente, al efecto de encendido y apagado de los tubos fluorescentes en la iluminación hogareña. Por este motivo, suelen incorporar un filtro óptico (**Figura 17**). Las señales recuperadas son muy pequeñas, por lo que

Un receptor de control remoto posee un elemento sensible a los infrarrojos

los módulos incorporan un preamplificador. En muchos casos, el conjunto se blinda y se entrega de ese modo (**Figura 18**).

La transmisión de la información pulsada que realiza el transmisor permite que el receptor separe fácilmente la información de la interferencia causada por otras fuentes de luz.

La salida de un módulo receptor es una señal discreta que puede leerse directamente por un puerto de entrada de un microcontrolador. La señal entregada es la envolvente, es decir, los bits de información. Leyendo esta señal en determinados instantes, comandados generalmente por un timer del microcontrolador, podemos extraer la información enviada y usarla para nuestros propósitos.

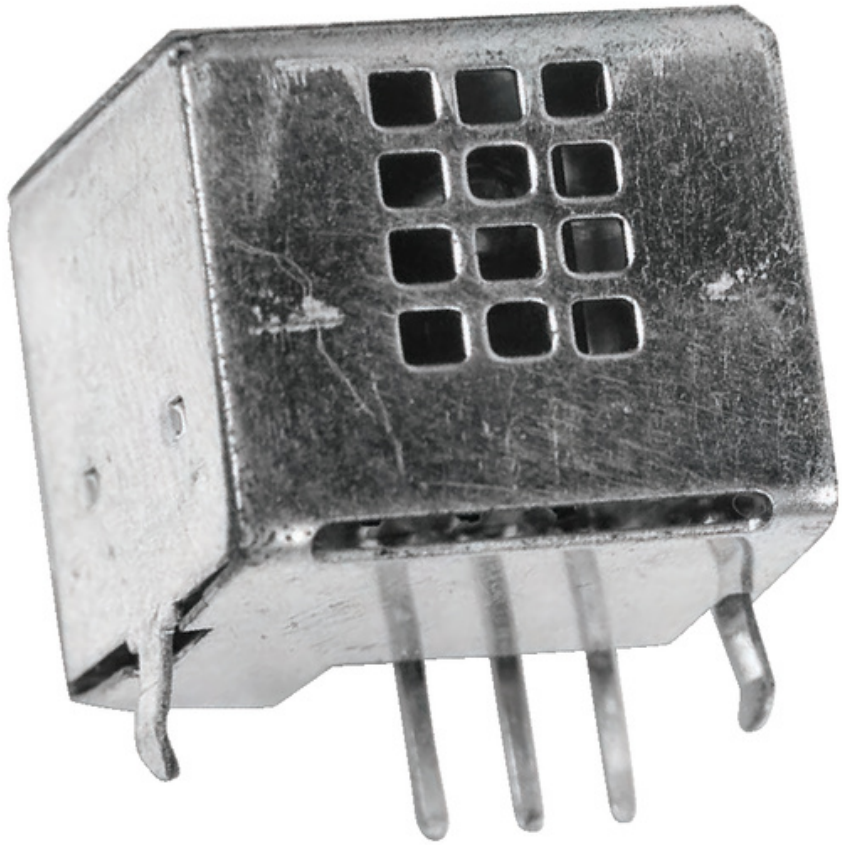


FIGURA 18. **Un receptor infrarrojo blindado. Todo el circuito receptor y preamplificador se encuentra dentro del blindaje.**

Multiple **choice**

1 **¿En qué año se realizó la primera emisión transatlántica por medio de las ondas de radio?**

a- 1851.
b- 1901.
c- 1951.
d- 1971.

2 **¿Cuál de los siguientes puertos ha quedado obsoleto?**

a- El puerto USB.
b- El puerto serie USART.
c- El puerto paralelo.
d- Ninguno de los anteriores ha quedado obsoleto.

3 **¿Qué nos permite la tecnología Bluetooth?**

a- Conectar dispositivos entre sí, evitando el uso de cables.
b- Conectarse con los módulos XBee.
c- Acceder al módulo, o gestionarlo directamente desde un microcontrolador.
d- Configurar los pines para que sean asignados al conversor A/D interno.

4 **¿Qué nos permite Docklight?**

a- Conectar dispositivos entre sí, evitando el uso de cables.
b- Conectarse con los módulos XBee.
c- Acceder al módulo, o gestionarlo directamente desde un microcontrolador.
d- Configurar los pines para que sean asignados al conversor A/D interno.

5 **¿Qué esquema de modulación utilizan los módulos transmisores de RF?**

a- OOSK.
b- OOK.
c- ASK.
d- RC5.

6 **¿Cuál de los siguientes es un estándar de facto del control remoto por infrarrojo?**

a- OOSK.
b- OOK.
c- ASK.
d- RC5.

Respuestas: 1 b, 2 c, 3 a, 4 c, 5 c, 6 d.

Capítulo 5

Módulos Rabbit y displays LCD gráficos

Conoceremos los módulos Rabbit con todos sus elementos a bordo y estudiaremos los displays LCD.

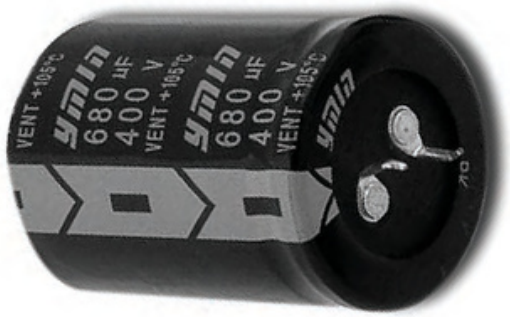
RedUSERS.com

Historias de **microprocesadores**

Mucho tiempo ha pasado desde el desarrollo del primer microprocesador y aún muchos de los conceptos originales siguen vivos en los microcontroladores más modernos.

DE 8080 A RABBIT 5000

El primer microprocesador exitoso fue el 8080 de Intel, sobre el que se basó el **Z80** de la empresa **Zilog**. Con ellos, nació uno de los primeros sistemas operativos de amplia distribución: el **CP/M**.



La característica distintiva de estos micros fue su gran cantidad de registros y un variado **set de instrucciones** orientado al procesamiento de datos. El Z80, en particular, introdujo la innovación del set alternativo de registros, para acelerar el cambio de contexto en interrupciones, tema que desarrollaremos más adelante. Otra novedad fue un área especial para el acceso a puertos de **entrada/salida** (**I/O**) fuera del mapa de memoria de datos. Sobre esta plataforma se desarrolló **Rabbit**.

DEL 6800 AL HCS08

El primer microprocesador exitoso de **Motorola**, hoy **Freescale**, fue el **6800**, contemporáneo y competidor del 8080 e, incluso, del Z80. Las líneas de 8 bits de mayor éxito de Freescale en la actualidad son las **HC08** y **HCS08** (**S08**), basadas en la **CPU08**, que, a su vez, es descendiente de la original **6800**. La característica distintiva de estos micros es un modelo de programación simple y compacto, con gran cantidad de instrucciones cortas y eficientes, orientadas a la resolución de tareas simples (**Figura 1**).

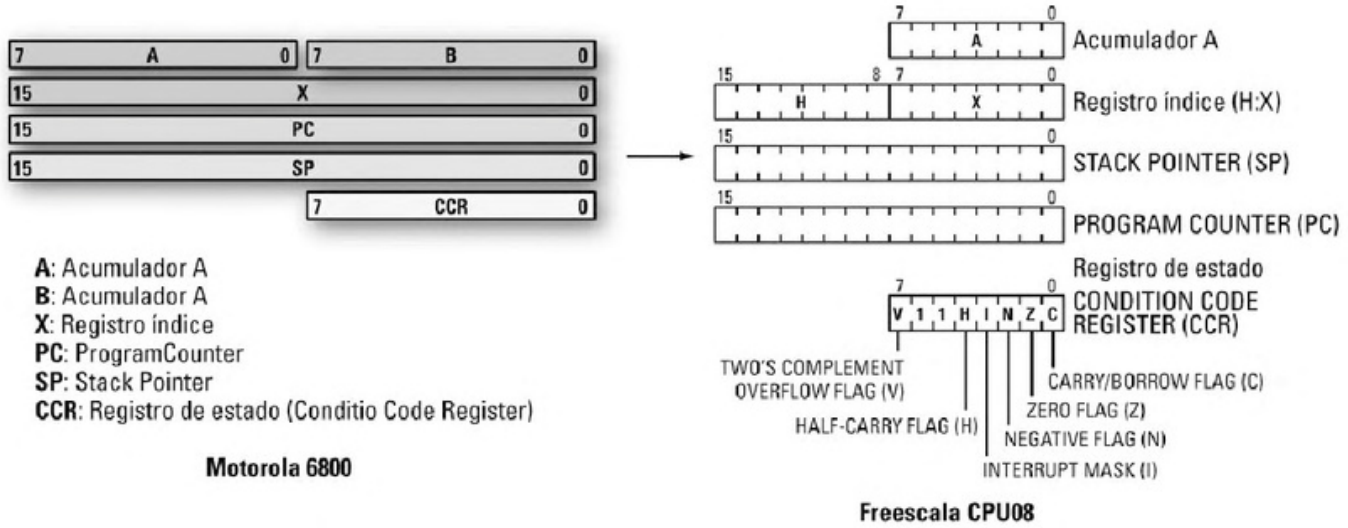


FIGURA 1. **Esta figura nos muestra la evolución del modelo 6800 a la CPU08.**

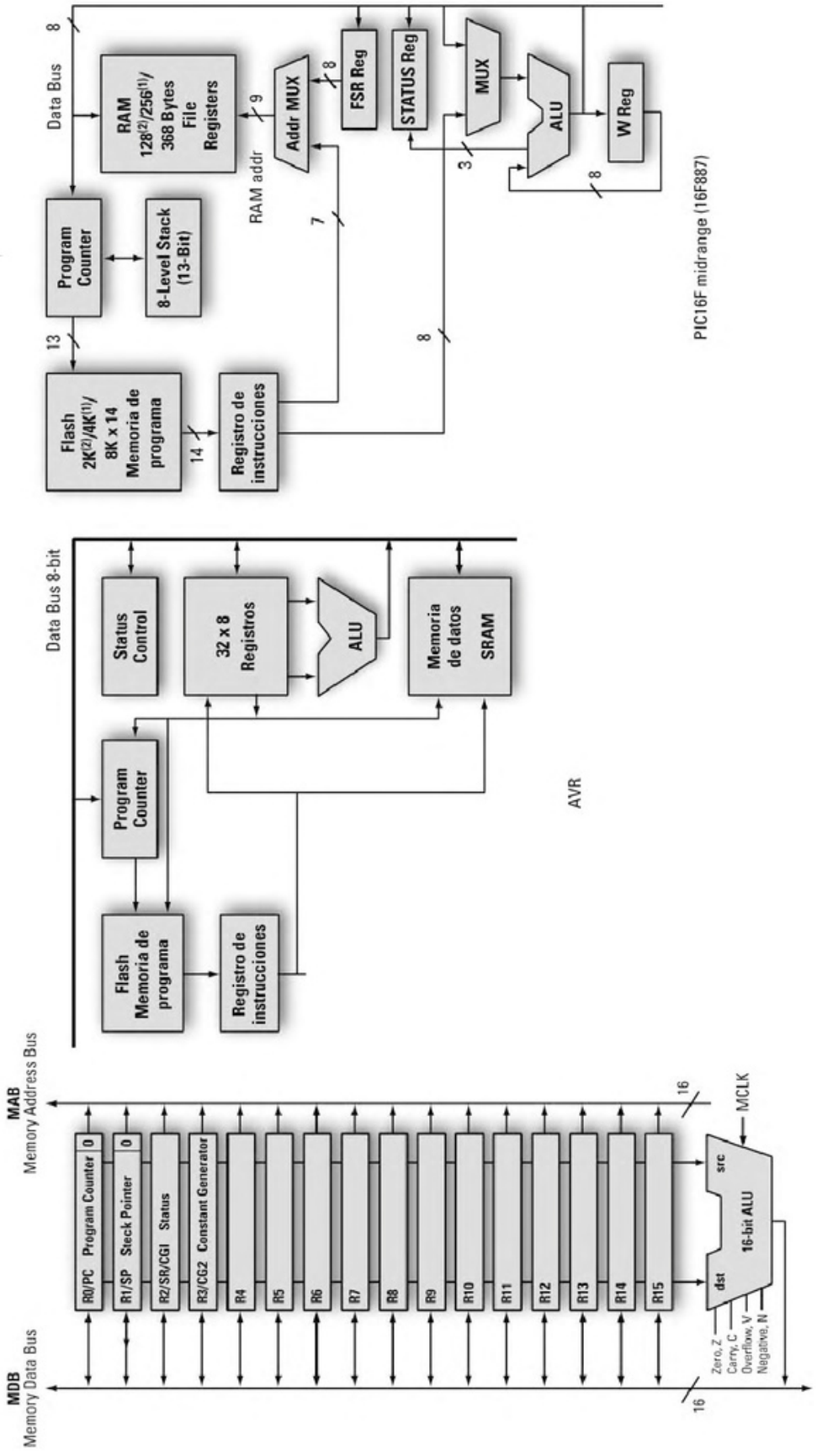


FIGURA 2. Detalles de la estructura interna simplificada de los núcleos: MSP430, AVR ATmega y PIC16F.

EL 8051

El **MCS51**, perteneciente al 8051, es un microcontrolador diseñado por Intel que ha pasado a ser el estándar de facto en el mundo de los microcontroladores, y que es modificado y producido por decenas de fabricantes. Su particularidad distintiva es que puede procesar bits individualmente, dentro de un área de **128** bits (32 bytes).

MSP430 Y AVR

Dos **cores** de relativamente reciente aparición son, por un lado, el **MSP430**, de 8/16 bits, fabricado por Texas Instruments; y, por el otro, el **AVR**, de **Atmel**. Ambos se caracterizan por tener una gran cantidad de registros del procesador, fuera del mapa de memoria de datos, además de un poderoso set de instrucciones.

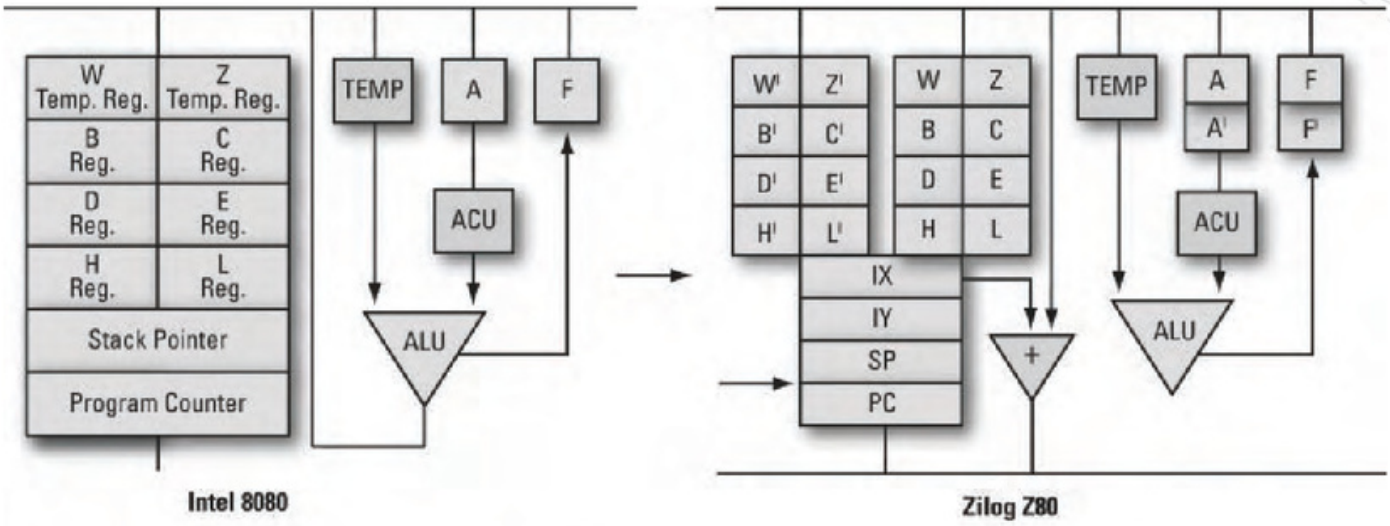
El **MSP430** permite operar en 8 o 16 bits sobre los registros o la memoria. AVR incluye, además, un área de I/O. Ambos núcleos permiten que cualquier registro o posición de memoria sea fuente y/o destino de casi cualquier operación (**Figura 2**).

MICROCHIP PIC

Otro core de gran repercusión es el **PIC**, de **Microchip**, elegido por la mayoría de los aficionados en el mundo. Se distingue por la simpleza y eficiencia del core, además de tener un muy compacto set de instrucciones de rápida ejecución, con acceso a toda la memoria de datos como uno de los operandos y posible destino de muchas de las operaciones. Éstos son sólo algunos de los núcleos de 8 bits de algunos de los fabricantes de mayor repercusión.

Módulos Rabbit

Un módulo Rabbit es una placa de circuito impreso con conectores que le permiten vincularse al resto del circuito que controla. Cumple la función de un microcontrolador, pero algunos modelos agregan, además, ciertas características adicionales, como zócalo para tarjetas **mini-SD**, conector **RJ-45** para Ethernet e interfaz de antena para **Wi-Fi**.



W y Z son registros temporarios inaccesibles para el programador F (Flags) es el registro de Estado.

FIGURA 3. **La evolución de la estructura interna del 8080 al Z80, en el que se basa el Rabbit 2000.**

LA ARQUITECTURA

La familia **Rabbit** se basa en el **Z80**, en cierto modo, sucesor del famoso **8080** (**Figura 3**). La arquitectura empleada es **Von Neumann**, que, si bien tiene numerosas desventajas, también presenta varios puntos a favor:

- Permite mayor sencillez al requerir sólo un bus de datos y direcciones, por consiguiente, un área de memoria.
- No requiere que los programas en **lenguajes de alto nivel**, como **C**, deban preocuparse de a qué área apuntan los punteros.

En particular, nos ocuparemos del **Rabbit 5000**, que, si bien es mucho más complejo internamente que el **2000** –el primero, nacido en ese año–, resulta más simple de manejar, sin necesidad de conocer todos sus vericuetos. Este core es capaz de operar a **100 MHz**, empleando desde **dos** ciclos para resolver la operación más simple (**50 MIPS** pico), con un consumo de **0,57 mA/MHz**.

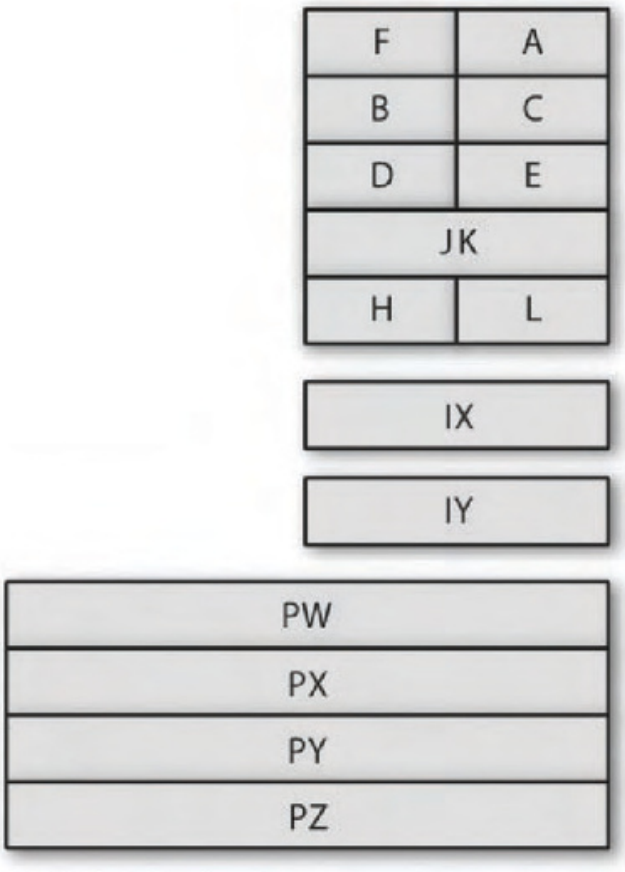


FIGURA 4. **Aquí podemos ver el mapa de registros de la CPU Rabbit 5000, donde apreciamos registros desde 8 hasta 32 bits.**

Un módulo Rabbit es una microcomputadora en un módulo con pines de interfaz

El modelo de CPU es **CISC** (*Complex Instruction-Set Computer*, o computador con set de instrucciones complejas), donde se busca generar código más compacto proveyendo instrucciones que resuelven la mayoría de los problemas. Sus más de **400 instrucciones** permiten resolver desde sumas entre registros, hasta movimiento de bloques de datos, incluyendo, por supuesto, multiplicaciones (en 32 bits).

EL MICRO POR DENTRO

Internamente, el micro tiene un **bus de 16 bits** e instrucciones de 8 y 16 bits, con algunas de 32 bits. En su set de registros (**Figura 4**) vemos el acumulador **A**, de **8 bits**, y varios registros de 8 bits que pueden agruparse de a pares y utilizarse en **16 bits**.

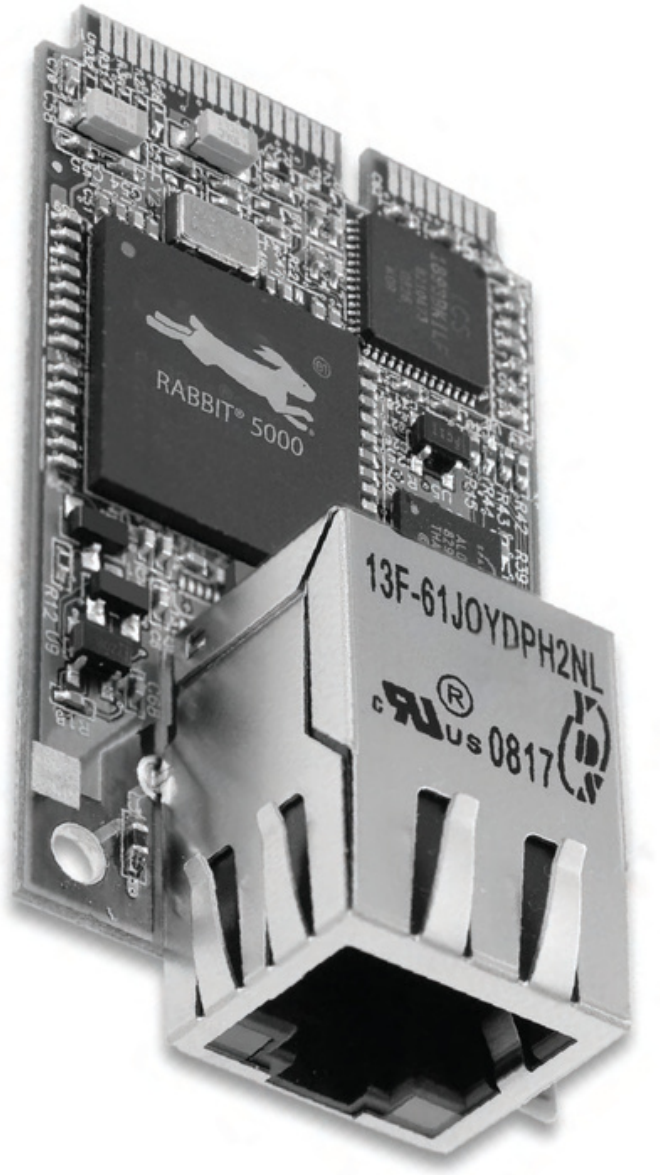
Por ejemplo, el par **HL**, que es la concatenación de los registros **H** y **L**, funciona como un acumulador de 16 bits. A su vez, el par **JK** puede agruparse con **HL**, y el **BC**, con **DE**, formando **JKHL** y **BCDE**, dos registros de **32 bits** que permiten algunas operaciones lógicas y aritméticas. También disponemos de un **set alternativo**, como en el **Z80**, pero en este caso, dichos registros pueden emplearse más libremente, casi duplicando la cantidad de registros.

PUERTOS DE ENTRADA/SALIDA

Al igual que el **8080** y los **80x86** (que, en cierto modo, también derivan del 8080), Rabbit posee un área especial para **I/O**. Estos puertos no comparten el mapa de memoria; por este motivo, tal como en las

PRIORIDAD DE EJECUCIÓN

La CPU puede operar en cuatro niveles distintos de prioridad. Las interrupciones se configuran para un nivel de operación. En todo momento, sólo son aceptadas aquellas cuyo nivel de prioridad supera al de ejecución en dicho instante.



PCs, se requiere de una instrucción especial en **Assembler** y una función especial en C para escribir o leer un puerto de I/O.

El **Rabbit 5000** incorpora seis puertos de I/O: **A**, **B**, **C**, **D**, **E** y **H**. Los pines se comparten con los demás periféricos. El puerto **H** normalmente se dedica a proveer los 8 bits altos para acceder a la memoria en 16 bits.

La arquitectura Rabbit diferencia, además, entre **espacio de I/O interno** y **externo**. Los periféricos del micro se hallan en el espacio interno, mientras que en el externo es posible conectar cualquier periférico y direccionarlo en el **mapa de I/O externo**. Algunos de los puertos de I/O permiten configurar sus pines como decodificadores de espacio de direcciones y seleccionar periféricos externos, eliminando la decodificación externa (*glue-logic*).

MANEJO DE MEMORIA

El Rabbit 5000 puede manejar hasta **16 MB** de memoria, mediante sus cuatro registros de 32 bits (**PW**, **PX**, **PY**, **PZ**) e instrucciones de acceso a memoria en **24 bits**. Para mayor economía, también incluye una unidad de manejo de memoria (**MMU**, *Memory Management Unit*), que permite emplear punteros y direccionamiento de **16 bits**.

PERIFÉRICOS

Dentro del chip **Rabbit 5000** encontramos los siguientes periféricos:

- Seis puertos serie (**A**, **B**, **C**, **D**, **E** y **F**) asincrónicos con capacidad IrDA (comunicación por infrarrojo).
- Un conjunto de timers de **8 bits** (**A**) para relojes de los demás periféricos.
- Un timer de **10 bits** (**B**) con dos módulos de comparación.
- Un contador de módulo variable hasta **16 bits** (**C**) con cuatro juegos de registros de comparación set/reset. Este esquema es muy útil para controlar motores paso a paso y generar señales **PWM** (modulación de ancho de pulso) multifase, para regular la velocidad de motores trifásicos.
- Un contador asincrónico (*ripple counter*) de **48 bits** con alimentación independiente para operar como **RTC** (*Real Time Clock*, o reloj de tiempo real).

- **IOSTROBEs** (chip selects para periféricos externos).
- Ocho canales de **DMA** (*Direct Memory Access*, o acceso directo a memoria).
- **Dos detectores de cuadratura** de 8 o 10 bits, que permiten leer encoders rotativos y determinar, por ejemplo, el sentido de giro de motores.
- Cuatro canales **PWM** con posibilidad de interrupción y supresión de pulsos, más DMA.
- **Dos unidades de captura y cuenta de eventos**, para medir pulsos de muy corta duración, del orden de los nanosegundos.
- **Ethernet 10/100** integrada y con DMA.
- **Wi-Fi 802.11b/g** integrada y con DMA.

En la página del fabricante, **www.rabbit.com**, podemos obtener el manual del usuario, donde consultar una descripción detallada del micro y todos sus periféricos (**Figura 5**).

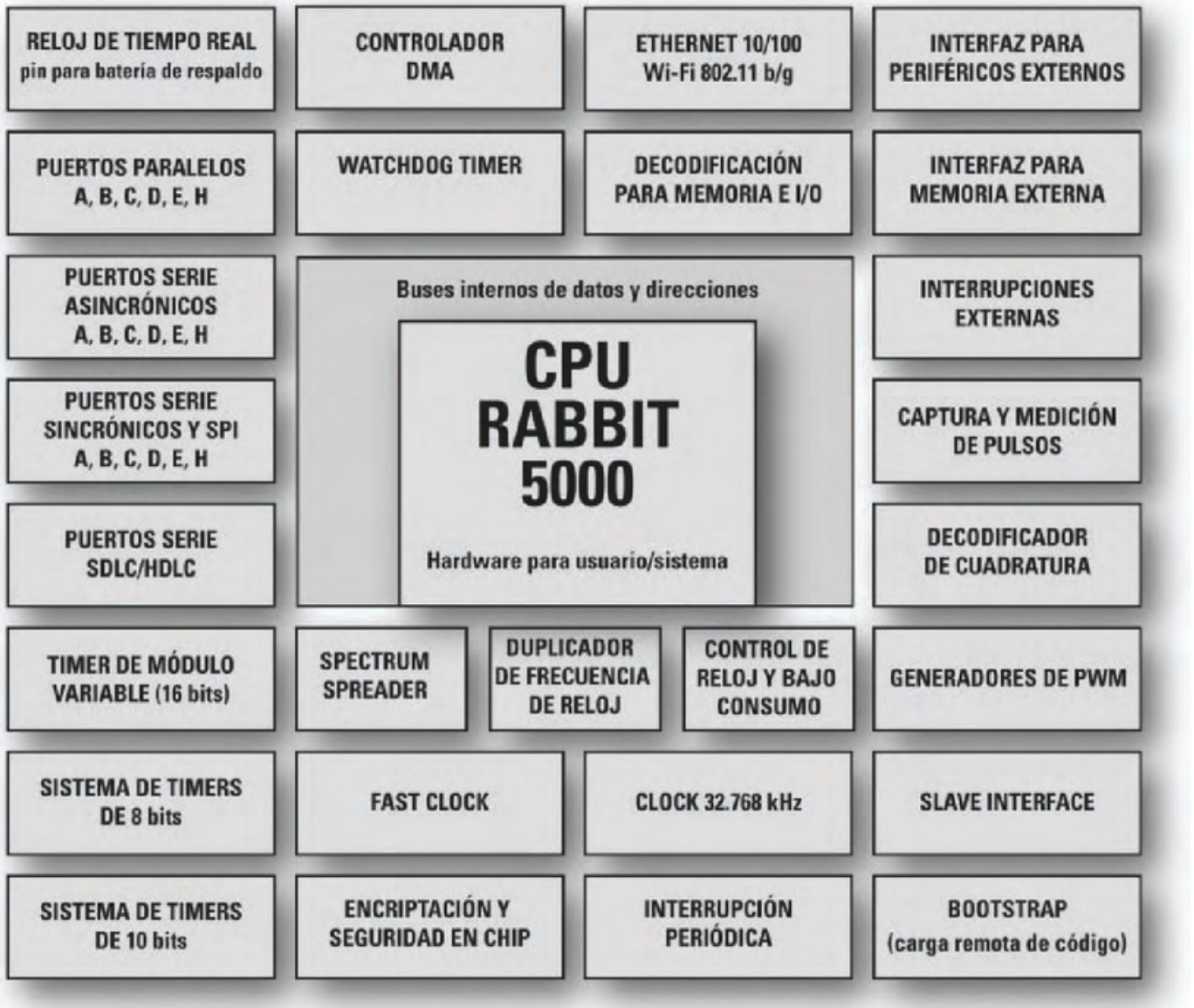


FIGURA 5. **Diagrama en bloques del Rabbit 5000, junto con sus periféricos y bloques constitutivos.**

Dynamic C

La historia de **Dynamic C** es más larga que la de las **CPU Rabbit**. Comenzó como un compilador propietario, y fue lentamente transformándose en un entorno de desarrollo completo. En un principio, no era compatible con el estándar **ANSI** (*American National Standards Institute*), pero poco a poco fue incorporando las fortalezas de éste, sin perder las propias. Podemos decir que Dynamic C es una implementación de C con funciones específicas para Rabbit (**Figura 6**).

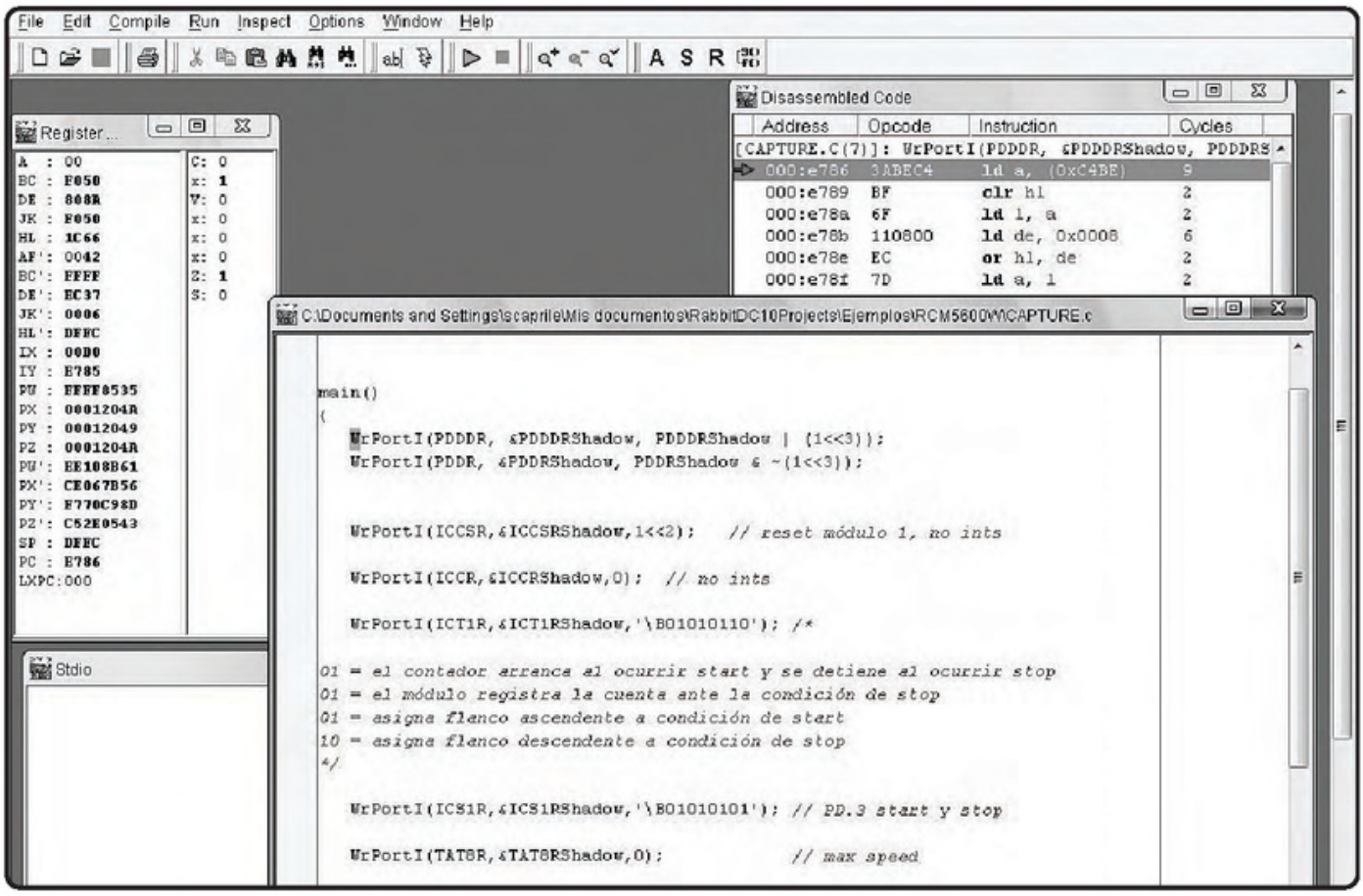


FIGURA 6. **Aquí observamos el entorno de** Dynamic C**. El cursor verde indica la sentencia actual para depuración en circuito.**

UNIDAD DE MANEJO DE MEMORIA

La MMU permite operar sobre una gran cantidad de memoria, sin perder la eficiencia de un direccionamiento de 16 bits en un micro de 8 bits, y sin emplear bancos. Controla la ventana en el espacio de 64 KB que permite ver una porción del espacio total de 16 MB.

La unidad de captura de eventos permite medir pulsos del orden de los nanosegundos

CARACTERÍSTICAS

El compilador se aleja del estándar **ANSI,** que el fabricante considera preferible para los sistemas dedicados, por ejemplo, Dynamic C:

- No pone a cero las variables estáticas que no fueron inicializadas.
- No utiliza **include files** por defecto, sino que la directiva **#use** reemplaza a **#include**.
- Compila a la memoria del sistema objeto.
- Incluye soporte para **multitarea cooperativo**
- Agrega tipos para identificar variables compartidas por contextos diferentes o almacenados en memoria no volátil.

En Dynamic C, los enteros son de **16 bits**, y los enteros largos, de **32 bits**. Los números en coma flotante son sólo de simple precisión (**32 bits**). Para indicar una función o un fragmento de código Assembler, sólo tenemos que escribirlo entre las directivas **#asm** y **#endasm**.

VARIABLES PROTEGIDAS

Si nuestro sistema se resetea en medio de la actualización de una variable **multibyte**, como un **array**, éste quedará dañado. Si declaramos la variable de tipo **protected**, el compilador generará código que creará una **copia de seguridad** de la variable antes de modificarla.

Si el sistema llegara a resetearse al momento de modificar dicha variable, el valor original podría recuperarse al reiniciar el sistema. La declaración es, por ejemplo: **protected long enterolargo**.

VARIABLES COMPARTIDAS

Si compartimos variables entre el programa principal y una rutina de interrupciones, una interrupción en medio de un acceso del programa principal a una de estas variables puede ser catastrófica. La rutina de interrupciones modifica el contenido de la variable al momento en que el programa principal había accedido a parte de ella. Entonces, la lectura que hará el programa principal será, cuando menos, incorrecta.

Si declaramos la variable de tipo **shared**, todos los **cambios** a ella son **atómicos**, es decir que se inhiben las interrupciones mientras se la modifica. Por ejemplo:

```
shared long varcomp;
```

C Y VON NEUMANN

Muchos de los lenguajes de alto nivel contemporáneos fueron desarrollados sobre máquinas con arquitectura Von Neumann, y mantienen el concepto de ver toda la memoria como una. En Dynamic C, no se requieren directivas para identificar donde apuntan los punteros.

Algunas variables globales compartidas que incorpora el sistema son:

SEC_TIMER: contiene el tiempo del sistema en segundos.
MS_TIMER: lo contiene en milisegundos.

UTILIDAD DE CONFIGURACIÓN DE I/O

El Rabbit 5000 incluye varios niveles de asignación de periféricos a los pines del encapsulado. Para facilitar la tarea al programador, existe un programa que permite realizar este proceso de forma gráfica, una tendencia actual en muchos fabricantes.

RABBIT BIOS

El BIOS (*Basic Input/Output System*, o sistema básico de entrada/salida) es, en este caso, la **interfaz entre el hardware y Dynamic C**. Provee los servicios de bajo nivel y la inicialización del procesador, y es el encargado de cargar el código y la **depuración** (**in-circuit debugging**).

Además, da la posibilidad de identificar el módulo mediante un área especial de la memoria flash, denominada **IDblock**, en la que se guardan estos datos. También provee de soporte para que el programador guarde sus datos en un sector llamado **User Block** (**Figura 7**).

> Dynamic C es una implementación de C con funciones específicas para Rabbit

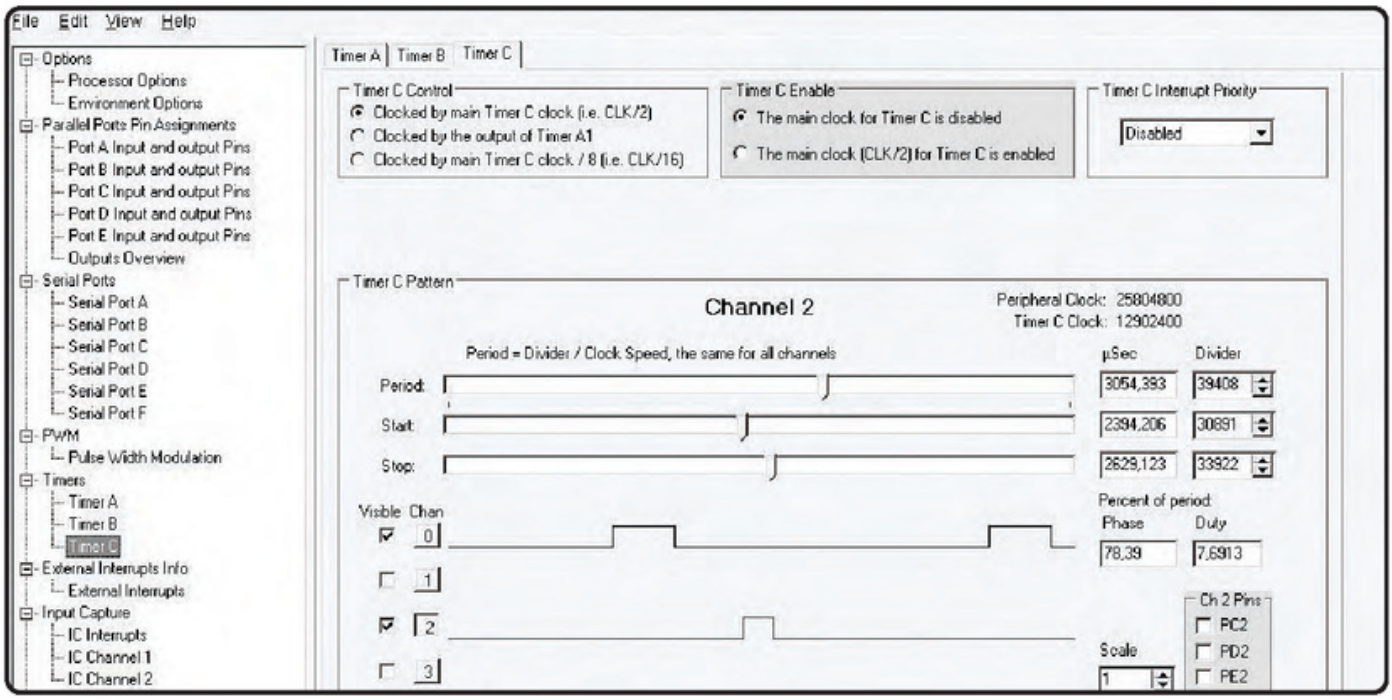


FIGURA 7. La utilidad de configuración operando sobre el timer C. El recuadro amarillo indica que prestemos atención.

BIBLIOTECAS DE FUNCIONES

Junto con el compilador, el fabricante distribuye una serie de bibliotecas de funciones que resuelven tareas comunes. De este modo, el programador puede enfocarse en su aplicación. Excepto en algún caso en particular, como el de **Wi-Fi**, donde existe propiedad intelectual del fabricante, las bibliotecas de funciones se encuentran en código fuente (C y Assembler), que el usuario puede observar y modificar a gusto (**Figura 8**).

Para utilizar funciones de una **library**, debemos incluirla mediante la directiva **#use**. Esto es, en cierto modo, equivalente a **#include**, sólo que **Dynamic C** maneja un sistema de ayuda incluido en las **libraries**. Las siguientes son las más comunes, excluyendo lo relativo a **networking**.

PUERTOS DE I/O

Cuando el micro presenta un espacio de I/O independiente del de memoria, como en las PCs, necesitamos funciones para acceder a los puertos. Como Rabbit, además del espacio de I/O de un microprocesador, incorpora periféricos internos, tenemos funciones para ambos espacios, cuyos nombres terminan en las letras **E** (espacio **E**xterno) e **I** (espacio **I**nterno).

```
BitRdPortI(port,bit);
// Devuelve el estado de
un bit en un port interno.
BitWrPortI(port,shadow,estado,bit);
// Modifica un bit en un port interno.
```

FIGURA 8. El RCM5400W provee de conectividad Wi-Fi con el controlador incluido en el chip Rabbit 5000.

ATOMICIDAD Y CAMBIOS ATÓMICOS

Cuando determinada área se accede mediante contextos diferentes, es preciso que los cambios dentro de cada ente de dicha área se realicen de forma atómica. Es decir, toda la operación debe desarrollarse sin interrupciones, como una única acción indivisible.

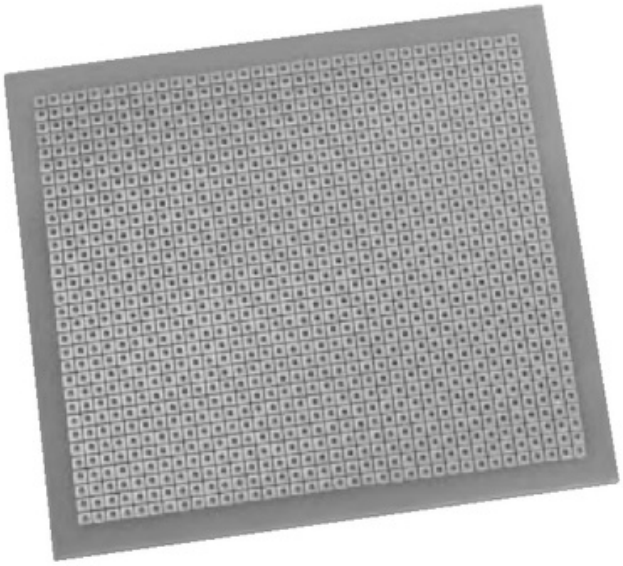
```
RdPortI(port);
// Lee un port interno.
WrPortI(port,shadow,data);
// Escribe en un port interno.
```

GRABAR DATOS EN FLASH

Esto se realiza, como dijimos, en el User Block. Para leer y escribir en esta aplicación, disponemos de dos funciones:

- Una función lee **num** bytes del User Block, desde **offset** hacia **dest_addr**:

```
readUserBlock(dest_addr,offset,num);
```



- Otra función graba **num** bytes en el User Block, desde **src_addr** hacia **offset**:

```
writeUserBlock(offset,src_addr,num);
```

LIBRERÍA RS-232

Para operar sobre los puertos serie, disponemos de la library **rs232.lib**, que provee de una interfaz (API) similar a la que encontramos en otros sistemas:

```
serBopen(9600);
// elige 9600 bps
serBdatabits(PARAM_8BIT);
// elige 8 bits
serBparity(PARAM_NOPARITY);
// elige sin paridad
serBputs(string);
// envía el texto apuntado por string
serBread(data, num, tmout);
// recibe hasta num bytes, los coloca
// en el array data, mientras no
// transcurra un tiempo tmout sin
// que haya datos
serBwrite(data,num);
// escribe num bytes del array data
```

INSTRUCCIONES DE I/O

Las CPUs como Rabbit poseen un espacio de I/O separado del mapa de memoria. La operación de I/O se realiza mediante funciones especiales, de modo que el compilador pueda insertar las instrucciones Assembler correspondientes para operar sobre este espacio.

Para operaciones complejas existe **packet.lib**, que permite operar sobre mensajes delimitados por un carácter especial, un tiempo de silencio o una ruptura de formato.

EL GPS

La mayoría de los módulos **GPS** (*Global Positioning System*, o sistema de posicionamiento global) entrega la información de posición y fecha/hora empleando un protocolo llamado **NMEA-0183**. La información es texto, y el formato es simple y relativamente fácil de procesar, más aún si llamamos a esta función de **gps.lib**:

```
gps_get_position(newpos, sentence);
```

El parámetro **newpos** es un puntero a una estructura **GPSPosition**, con latitud y longitud separados en los elementos de ésta; y **sentence** es un puntero a un array donde está la sentencia NMEA que obtuvimos del GPS por el puerto serie (**Figura 9**).

LA LIBRERÍA PWM

Esta library provee de la interfaz con los generadores de PWM:

```
pwm_init(frec);
// configura la frecuencia de operación
pwm_set(ch, pw, NULL);
// setea el canal ch con un ciclo de
// trabajo pw de 0 a 1024
```

Recordemos que el **valor medio** de una señal PWM es el valor de tensión del pulso por el **ciclo de trabajo**. En este caso: **V = Vpulso x pw / 1024**

RELOJ DE TIEMPO REAL

Operamos sobre el **RTC** mediante funciones de **rtclock.lib**. Podemos ver un ejemplo en el código que se encuentra al principio de la siguiente página.

La library rs232.lib permite operar sobre un puerto serie de forma similar a como se hace en una PC

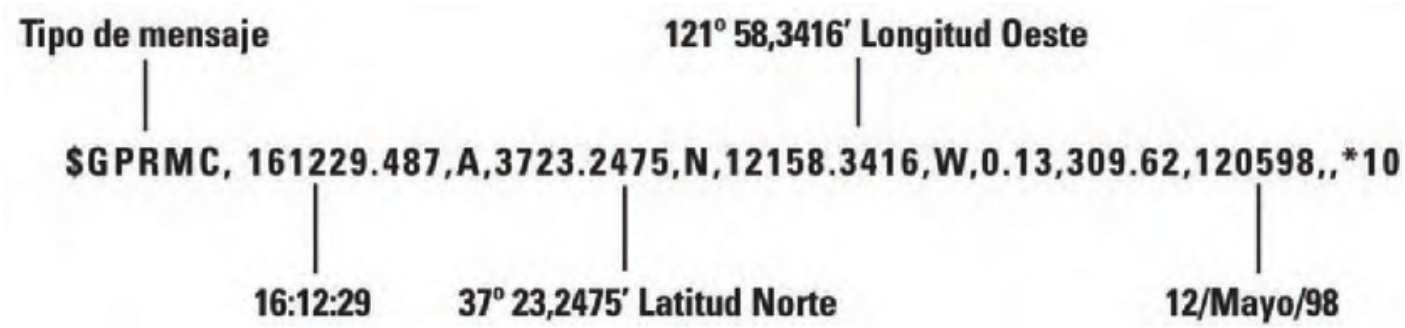


FIGURA 9. Detalles de un mensaje de un módulo GPS en protocolo NMEA-0183. Para extraer la información, sólo debemos seguir las comas.

```
unsigned long secs ;
struct tm thetm;

secs = read_rtc();
mktm(&thetm, SEC_TIMER);
```

Así obtuvimos en la estructura **thetm** la información de fecha y hora con cada dato en un elemento de ella.

DETECTORES DE CUADRATURA

La library **qd.lib** permite trabajar con los detectores de cuadratura. Leyendo el valor actual y comparando con el anterior, podemos detectar la velocidad y el sentido de giro. Esto permite leer, por ejemplo, un encoder rotativo para una interfaz de usuario.

```
qd_init(int);
// inicializa y configura prioridad
de interrupciones
qd_zero(mod);
// resetea el contador del módulo mod
qd_read(mod);
// devuelve el valor del
contador del módulo mod
```

DISPLAYS CON PANTALLA SENSIBLE

Existen dos libraries con funciones de soporte para displays monocromáticos de 320 x 240 píxeles y touchscreens. Ambas manejan todo lo relativo a posicionamiento de gráficos, texto, dibujo, lectura de botones con texto y gráficos, para realizar interfaces de control al tacto.

I²C Y SPI

Las libraries **i2c.lib** y **spi.lib** ofrecen funciones para controlar dispositivos mediante estos protocolos que estudiamos en el **Capítulo 3**. Por ejemplo, las funciones **i2c_init()** y **SPIinit()** inicializan el hardware.

POSICIONAMIENTO GLOBAL

La constelación de satélites mantiene una base de tiempo común y coordinado. El GPS recibe la posición y el tiempo de cada satélite.stimar la distancia de cada uno y determinar su posición en el espacio (x, y, z) y tiempo local (t).

```
I2CWrite(slave, index, src, num);
I2CRead(slave, index, dest, num);
SPIRead(dest, num);
SPIWrite(src, num);
SPIWrRd(src, dest, num);
// lee y escribe a la vez
```

Aquí, **slave** es la dirección del dispositivo (I2C), e **index**, la dirección en memoria de éste; **num** es la cantidad de bytes (I2C) o bits (SPI), **src** es la dirección donde están los datos en memoria, y **dest**, donde se guardan.

CODIFICADOR EN CUADRATURA

Permite codificar la velocidad y el sentido de giro mediante dos señales desfasadas **90º**. Por ejemplo, una rueda dentada en la que un transmisor y dos receptores infrarrojos están mecánicamente separados la mitad del ángulo ocupado por los dientes (**Figura 10**). Estos dispositivos son muy utilizados en robótica.

Mundo Rabbit

Existen al momento cuatro CPU Rabbit: 2000, 3000, 4000 y 5000. Las dos primeras CPUs contienen una unidad de manejo de memoria que les permite direccionar hasta **1 MB** de código o datos (recordemos que utiliza la arquitectura Von Neumann). Esa memoria es vista a través de un espacio lógico de **64 KB**.

Para cantidades de datos que no quepan en ese espacio, es necesario utilizar técnicas de programación no muy simples para el principiante.

A partir del Rabbit 4000, los **punteros de 32 bits** y las nuevas instrucciones permiten que el compilador utilice el concepto de **far pointers** (punteros lejanos) y direccione toda la memoria (16 MB) de manera más eficiente, sin que el programador deba ocuparse de hacerlo.

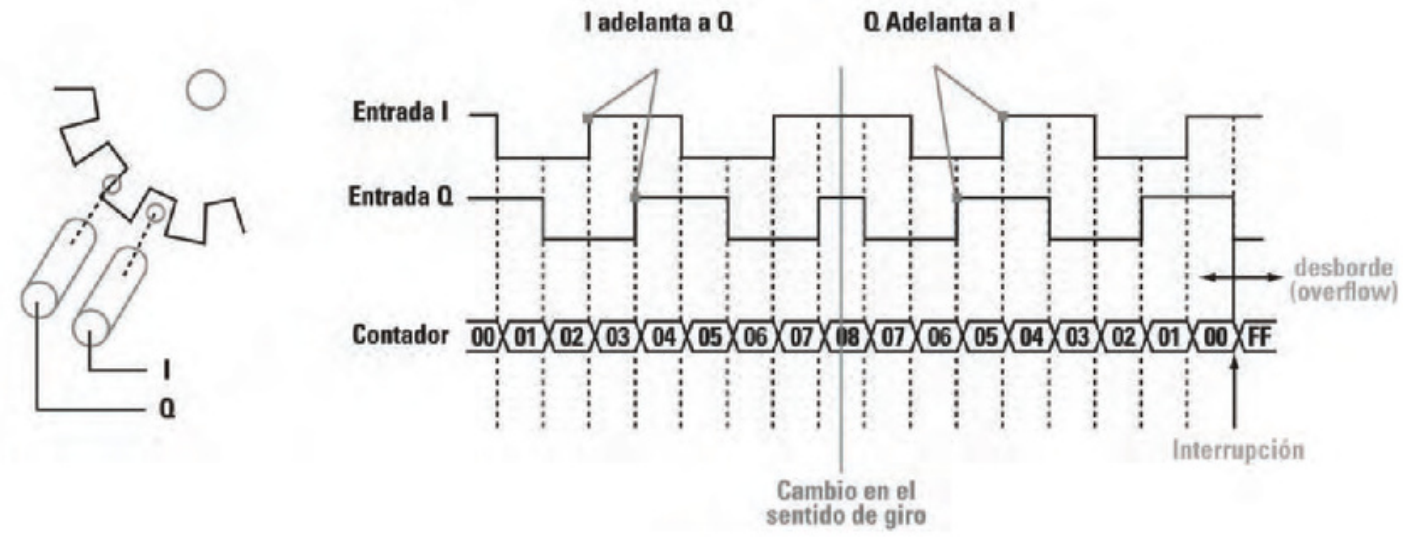


FIGURA 10. Un codificador en cuadratura y la operación del detector de Rabbit. El principio se basa en detectar los flancos y recordar el estado anterior.

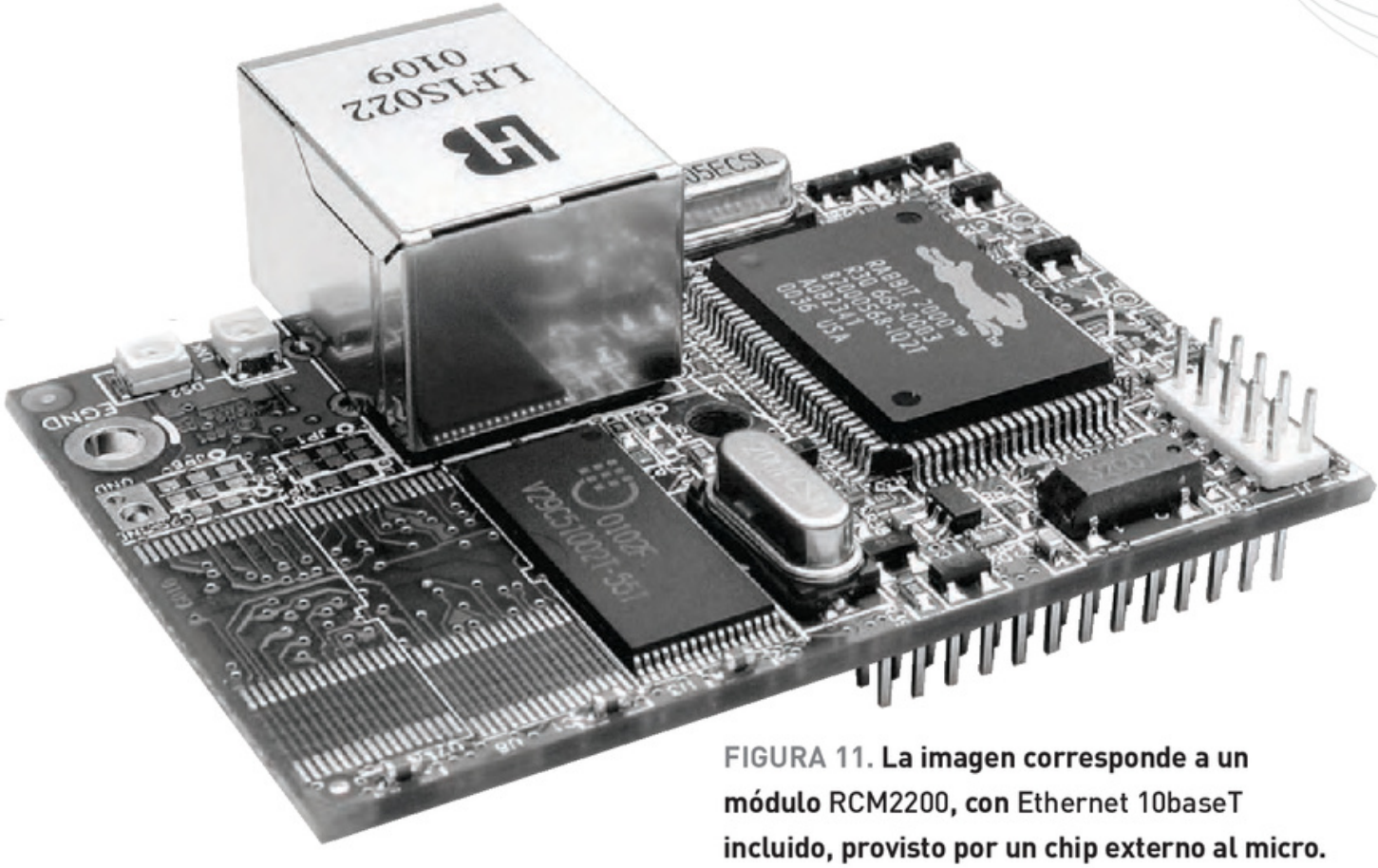


FIGURA 11. **La imagen corresponde a un módulo** RCM2200, **con** Ethernet 10baseT **incluido, provisto por un chip externo al micro.**

MÓDULOS DE 5 VOLTS

La familia de módulos **RCM2xxx** está basada en la CPU Rabbit 2000, la primera de todas, bastante menos poderosa (**Figura 11**). Son módulos de **5 V**. El más versátil y utilizado es el **RCM2200**, que incorpora **128 KB de RAM** y **256 KB de flash**.

MÓDULOS DE 3,3 VOLTS

La CPU **Rabbit 3000** funciona a **3,3 V**. La familia de módulos **RCM3xxx** está basada en esta CPU, y son módulos con puertos de **I/O** de **3,3 V**, pero tolerantes a **5 V**. Esto significa que, si bien el micro entrega tensiones de 3,3 V, permite recibir datos de un periférico de 5 V. Si los niveles lógicos del periférico pueden reconocer el valor de tensión entregado por el Rabbit 3000 como un **1 lógico**, la conexión es posible. El más simple y utilizado de estos módulos es el **RCM3720**, que se alimenta de 5 V e incluye un regulador interno (**Figura 12**). Contiene **256 KB de RAM**, **512 KB de flash**, y una memoria **flash serie** de **1 MB** que puede utilizarse mediante el sistema de archivos **FAT**.

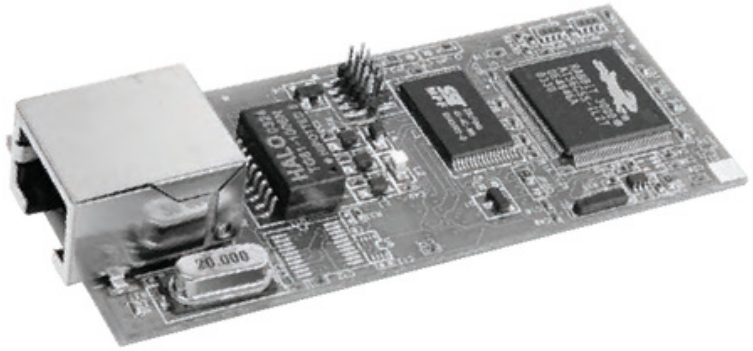


FIGURA 12. **El módulo** RCM3720 **tiene pines de paso .1" y posee conexión** Ethernet 10baseT **mediante un chip externo al micro.**

MÓDULOS DE 3,3 VOLTS

El chip anterior al **Rabbit 5000** es el Rabbit 4000, que tiene un controlador **Ethernet** en el chip, no posee controlador **Wi-Fi**, y su bus interno de datos es de 8 bits, como los anteriores. Es el primero en incorporar **punteros de 32 bits** para facilitar el manejo de memoria cuando la cantidad de datos no cabe en el espacio de **64 KB** que sus antecesores manejan directamente.

Se trata de módulos de **3,3 V** que no toleran **5 V** en sus entradas. En este caso, ya disponemos de más opciones, como tarjetas **mini-SD** y **módulos ZigBee** (**Figura 13** y **14**). Las tarjetas pueden ser leídas en cualquier otro sistema, pues emplean archivos **FAT** (*File Allocation Table*, o tabla de asignación de archivos).

También existen módulos con **Wi-Fi 802.11 b/g** basados en el Rabbit 5000, como el **RCM5400W**, que incorpora **512 KB de RAM** y **512 KB de flash**. Dada la velocidad de la CPU, este micro incluye **otros 512 KB RAM** para copiar el programa de la memoria flash y ejecutarlo en **RAM**, y una **flash serie** de **1 MB**.

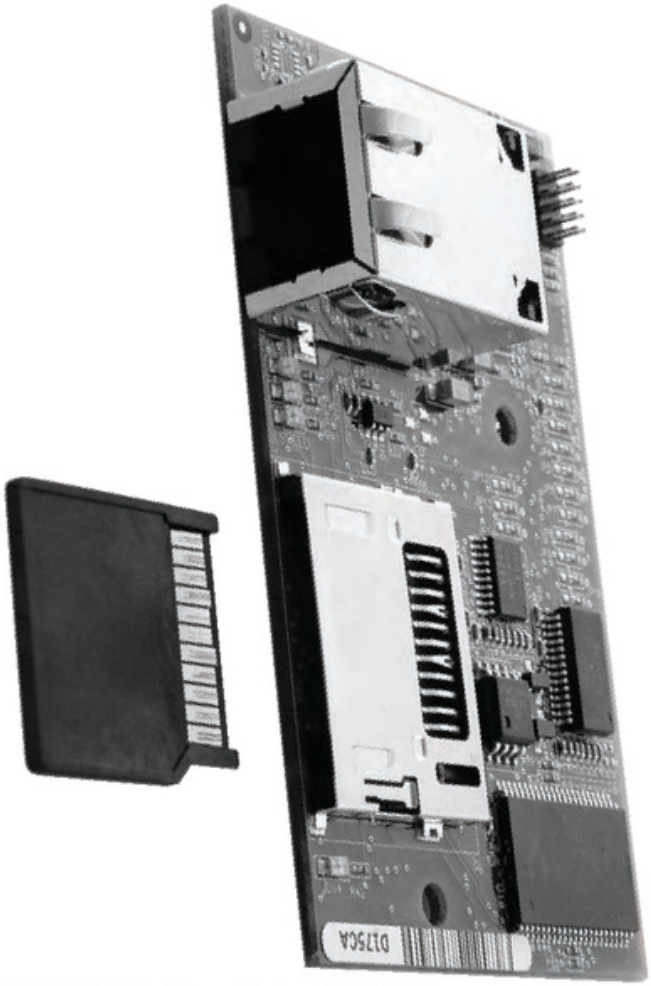


FIGURA 13. El módulo RCM4300 provee de un zócalo para tarjeta mini-SD, conversor AD de 12 bits y conexión Ethernet 10baseT con el controlador interno.

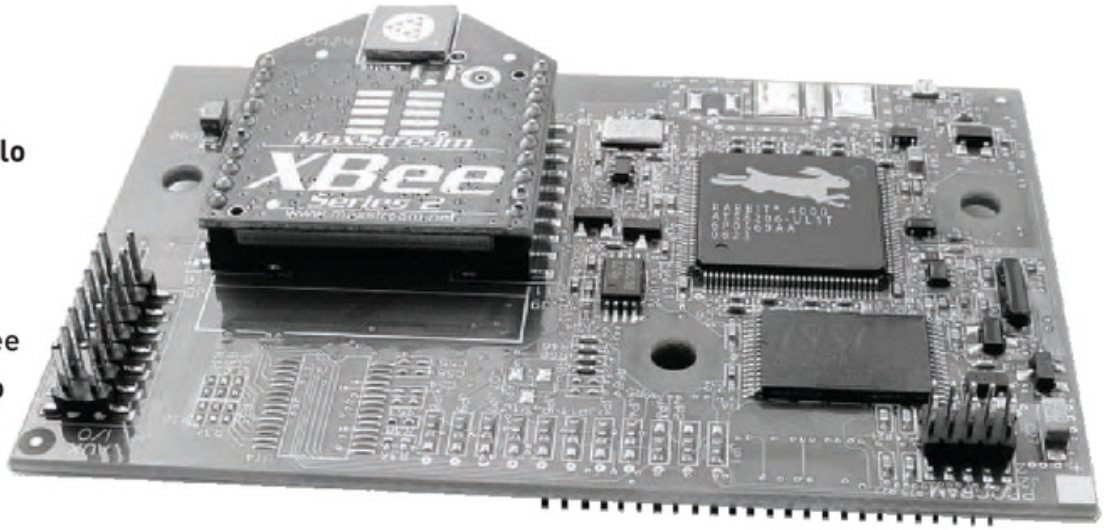
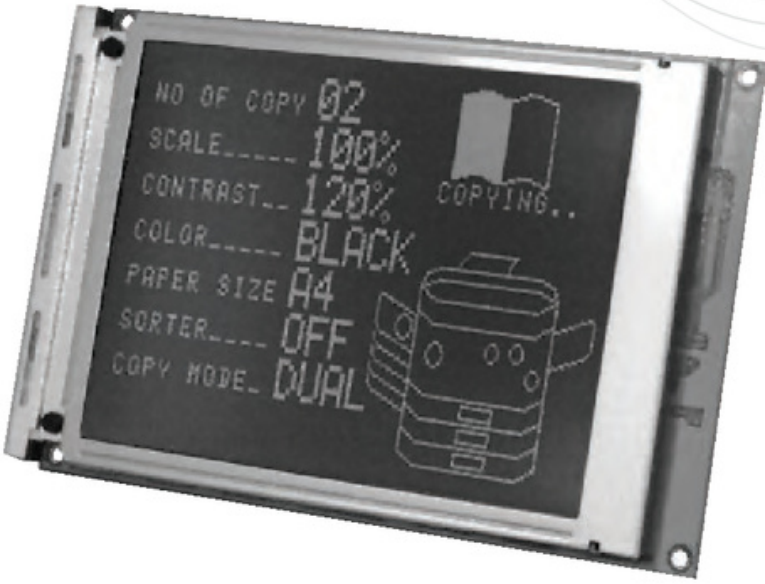


FIGURA 14. El módulo RCM4510W ofrece 512 KB de RAM, 512 KB de flash y conectividad ZigBee mediante un módulo XBee ZB.

MINICORES (3,3 V)

Basados en el Rabbit 5000, los minicores son la opción más económica del fabricante y son los que emplearemos en este capítulo. El **RCM5600W** no tiene memoria flash paralelo, sino que emplea una **flash serie de 1 MB**, cuyo contenido es copiado por el micro a la **RAM de 1 MB** al inicializar, y se ejecuta desde allí. La CPU opera a **74 MHz**, y el tiempo de acceso requerido (2 ciclos) es de aproximadamente 25 nanosegundos (**~25 ns**), imposible con una memoria flash en la actualidad.

El **RCM5700** incorpora una **flash paralela de 1 MB**, pero su CPU opera a menor frecuencia (**50 MHz**) y se configura para insertar tiempos de espera (wait-states) al acceder a la flash. Como la **RAM**, utiliza los **128 KB internos** del Rabbit 5000.



Displays gráficos LCD

Existen varias clases de displays gráficos LCD. Esto se debe a que distintos tamaños de display llevan a diferentes resoluciones (puntos) y, a su vez, existen distintos tipos de controladores, orientados más a uno o a otro tipo de aplicación. Nos concentraremos aquí en un tipo de displays cuyo controlador es similar al utilizado en los alfanuméricos; de hecho, la interfaz eléctrica es prácticamente la misma. El controlador original es el **HD61202**, desarrollado por Hitachi; en la actualidad muchos displays emplean el **KS0108** de Samsung, que es virtualmente idéntico en operación, aunque presenta algunas diferencias en los niveles lógicos. El **KS0108** acepta señales de **3,3 V**, mientras que el **HD61202** no lo hace. Estos displays suelen presentarse en dos formatos: **64 x 64 píxeles** y **128 x 64 píxeles**. En este último caso, el display dispone de dos controladores, uno para el lado izquierdo y el otro para el lado derecho, pues el chip no puede controlar más de **64 columnas** y **8 filas**.

EL DISPLAY

La **Figura 15** muestra la estructura de memoria de estos módulos gráficos. El primer punto de pantalla –arriba a la izquierda– corresponde al bit menos significativo del primer byte de memoria, del primer controlador. El punto en pantalla –abajo a la derecha– corresponde al bit más significativo del último byte de memoria, del segundo controlador. Seleccionamos el byte por leer o escribir en memoria por medio de comandos, como en los displays alfanuméricos, pero ahora, tenemos dos direcciones:

- horizontal (número de byte, de 0 a 63)
- vertical (número de página, de 0 a 7)

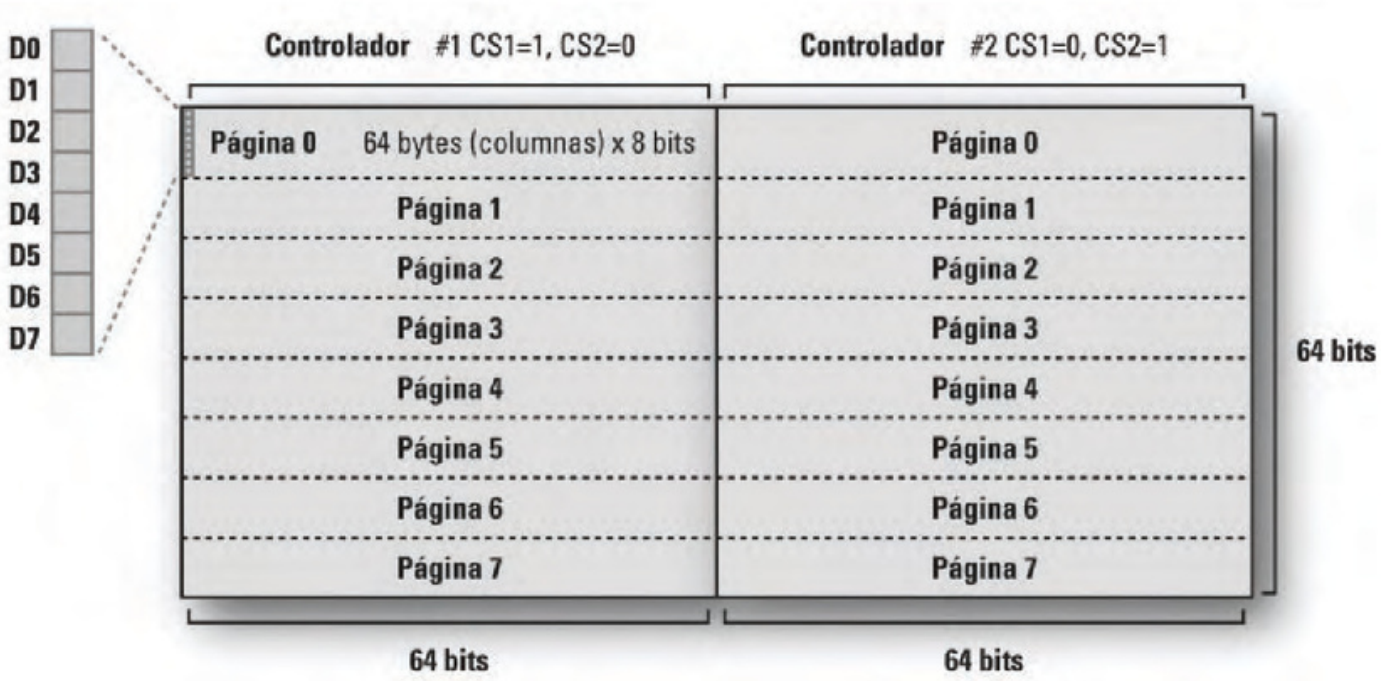


FIGURA 15. **Estructura de la memoria (mapa de la imagen) en el display utilizado. Cada punto encendido es un bit en una posición de memoria.**

Un contador autoincrementado, que apunta a la dirección siguiente, luego de una lectura o escritura, controla la dirección horizontal.

CONTROL DE CONTRASTE

Estos displays requieren una tensión negativa para el control de contraste. Afortunadamente, gran cantidad de módulos la generan de manera interna y la proveen en un pin.

La interfaz es la ya familiar originada en el bus del **procesador 6800**. Para controlar el display con un microcontrolador, imitamos la operación de ese bus, asignando las señales de control a puertos de I/O del micro; esto se conoce como **bit-banging**.

En el caso de displays de **128 x 64 píxeles**, tenemos dos pines de control adicionales, cada uno de los cuales habilita a un controlador.

Para un micro de **5 V**, conectamos directamente a los puertos de **I/O**, en tanto que para micros de **3,3 V**,

DISPLAYS Y CONTROLADORES

Hay displays gráficos que incluyen al controlador, y otros que no lo hacen. Existen varios tipos de controladores, cada uno con interfaz y comandos diferentes. El HD61202 y su clon, el KS0108, son sólo una alternativa, pero resultan la más simple y económica.

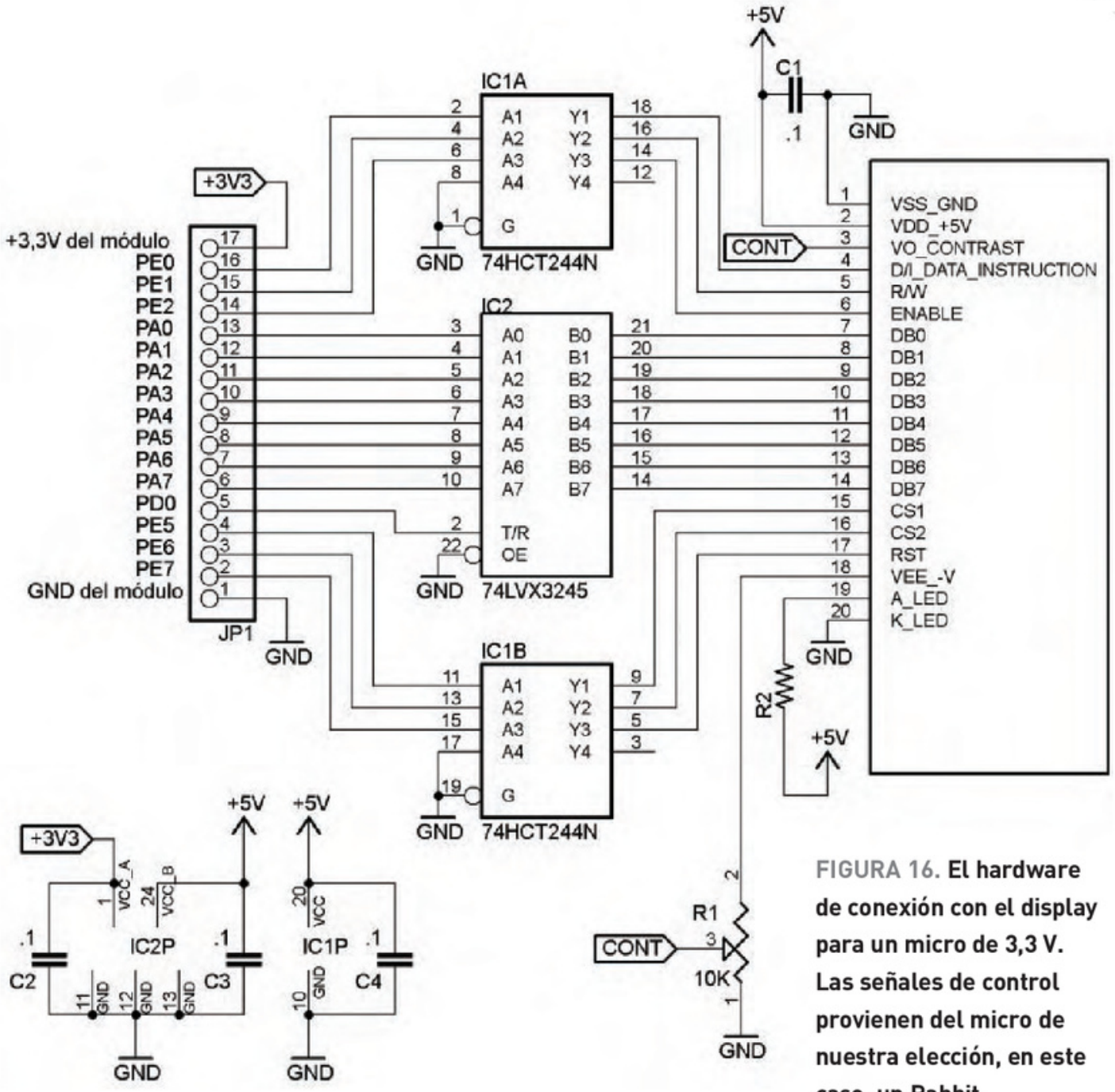


FIGURA 16. El hardware de conexión con el display para un micro de 3,3 V. Las señales de control provienen del micro de nuestra elección, en este caso, un Rabbit.

debemos agregar un **conversor de nivel**. La opción genérica es emplear un **bus transceiver**, como el **74LVX3245**, que se controla desde el lado del micro a 3,3 V, y admite del otro lado un periférico a 5 V, convirtiendo niveles en uno u otro sentido. Para las señales de control, utilizamos un **buffer** de tecnología **HCT** (*High-speed CMOS with TTL levels*). Sus niveles lógicos de entrada le permiten reconocer 3,3 V como un **1** lógico, alimentado a 5 V.

En el circuito de la **Figura 16**, tomamos los 3,3 V del módulo Rabbit y los 5 V de una fuente.

CONTROL DEL DISPLAY

El manejo de los pines de control del display requiere tener en cuenta el control del **74LVX3245** y el orden de activación de las salidas (**Figura 17**). El pin **I/D** selecciona el envío de comandos (**I**) o el operar sobre la memoria (**D**). La **Figura 18** muestra la operatoria de manejo de control en

El control de contraste requiere una tensión negativa que los displays generan en el módulo

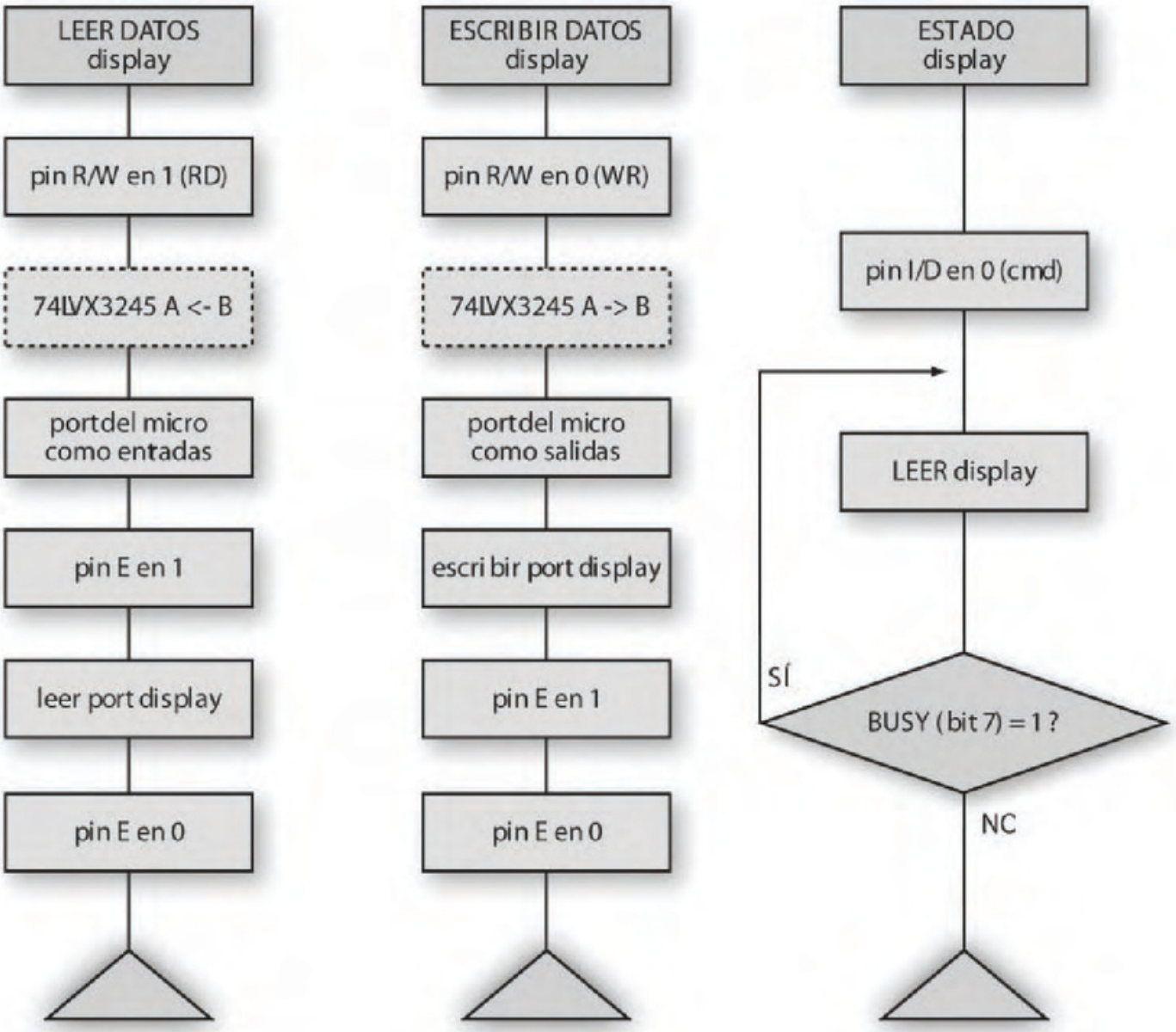


FIGURA 17. **Operaciones elementales de acceso al display. La operatoria es similar a lo visto para el display alfanumérico.**

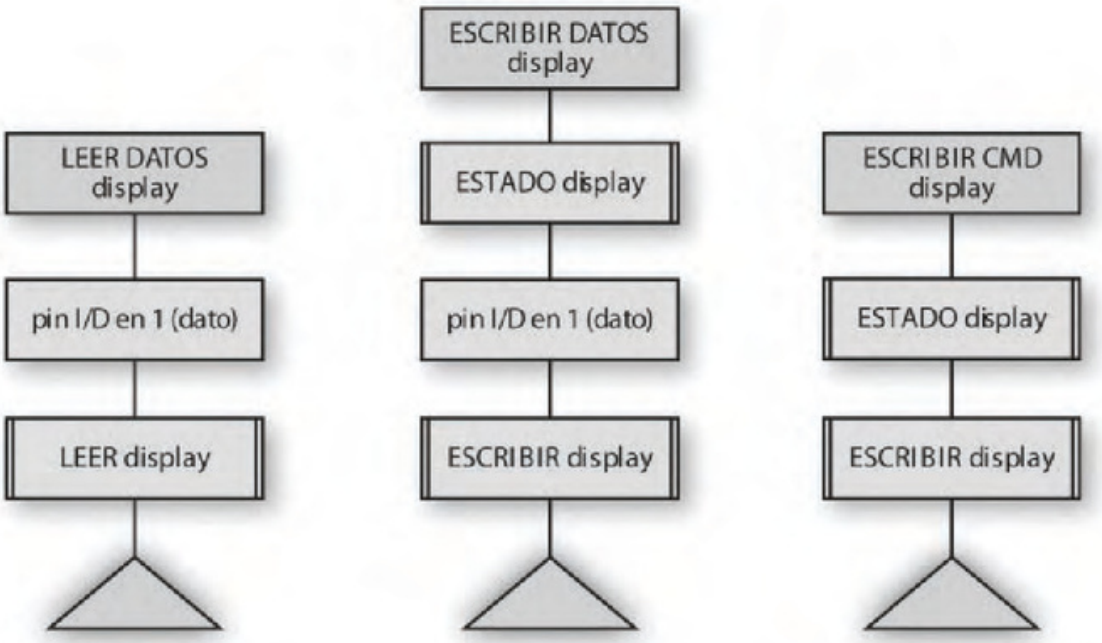


FIGURA 18. Operaciones de bajo nivel del display. Obsérvese la similitud con el controlador del display alfanumérico.

bajo nivel del display, cuyas actividades usuales son: leer un dato (la memoria) del display, escribir un dato en el display y enviar un comando.

Si el micro es muy rápido, deberemos insertar demoras para cumplir con los **tiempos de acceso** del display, especificados en su hoja de datos.

ALGORITMOS

Ya estamos en condiciones de desarrollar los algoritmos que nos permitirán mostrar textos, gráficos (iconos) y funciones en el display. Comenzamos definiendo que el punto con coordenadas (0;0) es el del extremo superior izquierdo de la pantalla. Hecho esto, podemos enunciar algunos algoritmos:

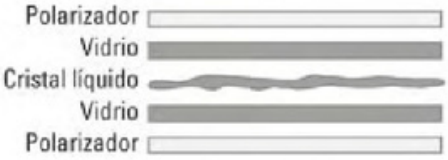
- Dado un punto con coordenadas **(x ; y)**, el controlador correspondiente es el izquierdo si **x < 64**. En caso contrario, es el derecho.
- La dirección dentro del controlador debe modificarse a **x = x - 64** si **x > 64**. En caso contrario, es el derecho.
- El número de página es la **parte entera** de dividir **y** por **8.**
- El bit es el **resto** de la división anterior; es decir, **y - 8 * página**

COMANDOS

Con los comandos, nos posicionamos sobre el byte (columna) dentro de la página (fila) que contiene el bit correspondiente al punto que deseamos encender o apagar y operamos sobre él. Podemos obtener más información de estos controladores en sus hojas de datos.

INFOGRAFÍA 2: ARQUITECTURA DE UN DISPLAY DIGITAL

En esta infografía veremos la arquitectura interna de un display digital de tecnología LCD (Liquid Crystal Display) y las características técnicas de cuatro de los dispositivos más famosos del mercado.



Displays tipo TN (*Twisted Nematic*):

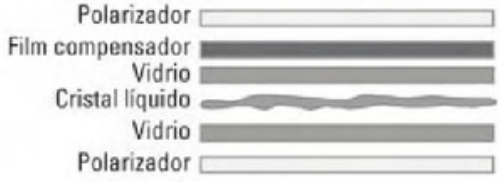
En estado de reposo, estos displays tienen 90° de rotación en la alineación de las moléculas de cristal líquido con respecto al área de transición de las dos superficies de vidrio. Los polarizadores utilizados en las superficies exteriores están orientados de manera de permitir el paso de la luz. Como sabemos, una vez aplicado un campo eléctrico, las moléculas se alinean con el mismo impidiendola rotación de los haces de luz y generando contraste entre zonas de "encendido" y "apagado".

Displays tipo HTN (*High Performance Twisted Nematic*):

Ofrecen mayor rotación, mejor ángulo de visión y menor costo que los materialesl TN.

Displays tipo STN (*Super Twisted Nematic*):

Ofrecen mayor ángulo de rotación (240° contra 90° del TN), con mayor contraste y ángulo de vista. Posee una desventaja, que es el efecto de birrefringencia o "doble refracción". Estos materiales desdoblan un rayo de luz incidente en dos linealmente polarizados en forma perpendicular entre sí. El efecto causado es que cambia el color de fondo a amarillo - verde y el de los caracteres a azul.



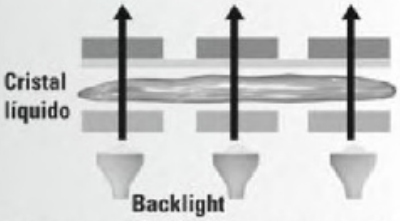
Displays tipo FSTN (Film Super Twisted Nematic):

Este material agrega un film de retardo que compensa el efecto de birrefringencia. Éste permite la existencia de displays blanco-negro, con excelente visualización cuando se utiliza backlight.

Los displays LCD no emiten luz ni ningún tipo de radiación, es por eso que se los agrupa dentro de los dispositivos "pasivos" de visualización. Utilizan la luz ambiente del entorno, por lo que consumen muy poca energía.

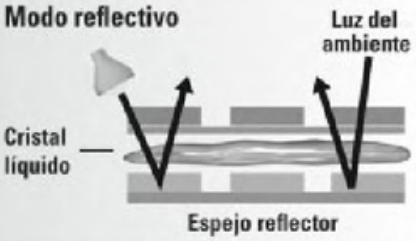
MÉTODOS DE ILUMINACIÓN

Modo transmisivo



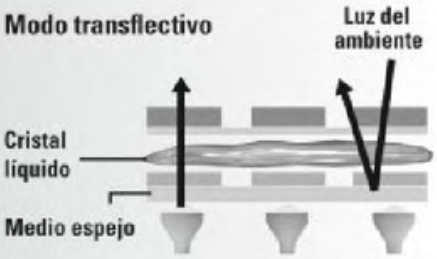
Ofrecen un aspecto distinto del resto debido a la luminosidad que ofrece el backlight siempre presente. Ideales para ser usados en ambientes oscuros.

Modo reflectivo



Poseen sobre el fondo un espejo que brinda reflexión completa. No usan backlights: son dispositivos de bajo consumo, ya que utilizan la luz del ambiente.

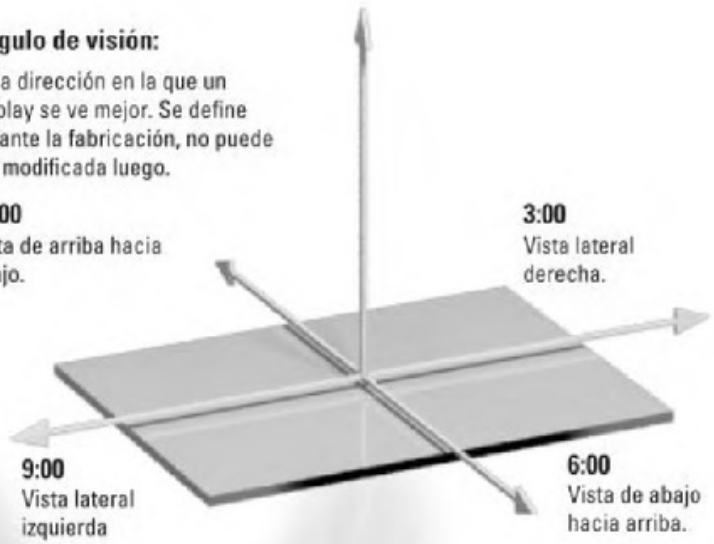
Modo transflectivo



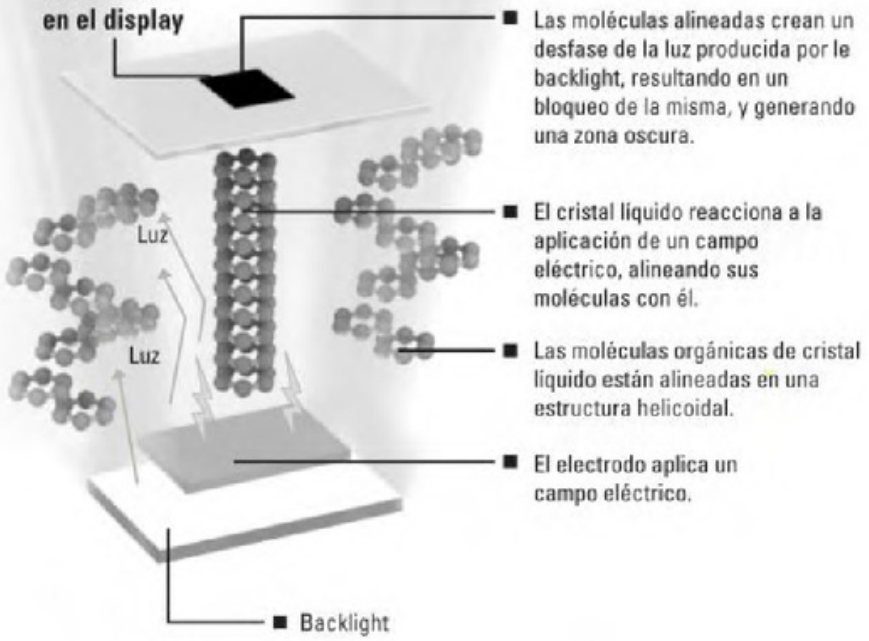
Combinan los efectos reflectivos y transmisivos. Pueden ser utilizados en cualquier situación de iluminación, contemplando, en malas condiciones, el encendido del backlight.

Ángulo de visión:

Es la dirección en la que un display se ve mejor. Se define durante la fabricación, no puede ser modificada luego.



Zona oscura en el display



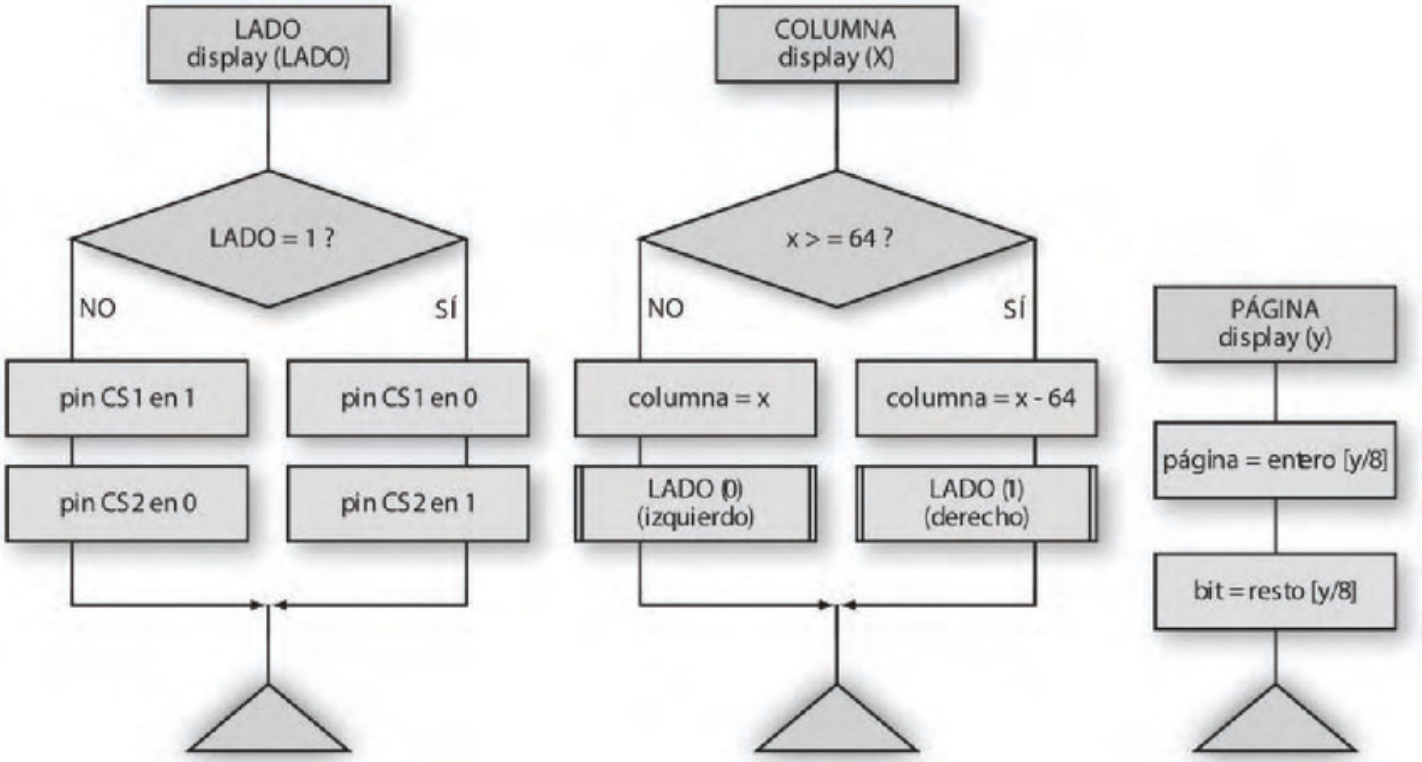


FIGURA 19. Algoritmos de posicionamiento en el display. Ésta es la clave para graficar imágenes, textos y funciones.

PLOT, ENCENDIENDO PÍXELES

Para encender un punto determinado, leemos del display el contenido del byte que lo contiene, configuramos el bit que le corresponde, y lo volvemos a escribir.

Al indicar la dirección, debemos hacer una lectura ficticia para darle tiempo al controlador. Antes de escribir, como el contador se autoincrementa, tenemos que volver a indicar la dirección (**Figura 19**).

Para enviar gráficos al display, conviene que éstos sean de un tamaño vertical

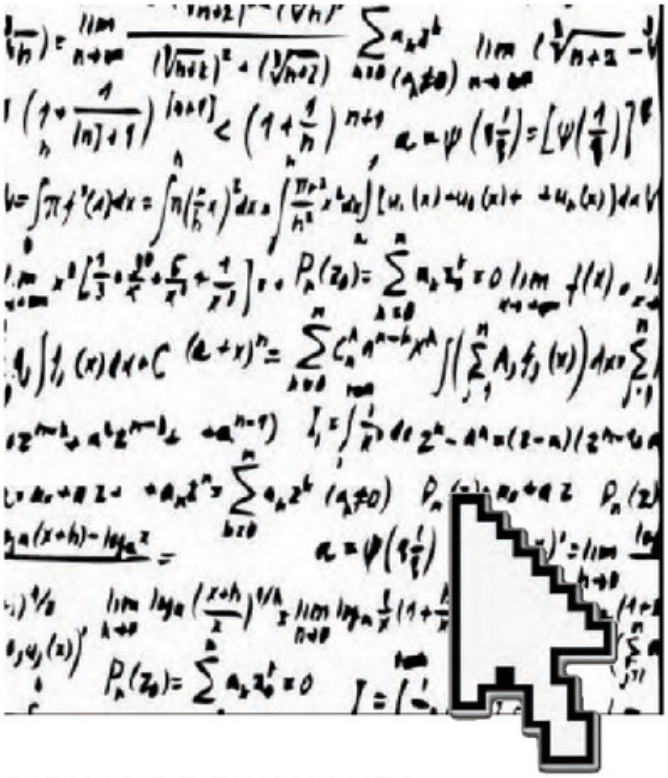
CÓMO DIBUJAR GRÁFICOS Y FUNCIONES

Para enviar gráficos al display, conviene que éstos sean de un tamaño vertical y que quepan en un número entero de páginas, es decir, 8, 16, 24 puntos, etc. De esta forma, el dibujo puede imaginarse dividido en páginas y columnas. La tarea requerida es la siguiente:

- Seleccionar la dirección inicial para la página.
- Enviar todos los bytes de dicha página, cuidando que no se produzca un cambio de controlador (**x≥64**), en cuyo caso deberemos repetir la operación de seteo de página para el controlador derecho.
- Repetir por el número de páginas.

Por ejemplo, para mostrar un gráfico alojado en **imgdata**, de un cierto **alto** y **ancho**, en la página **pag**, posición **x** (**Figura 20**).

```
while(alto–) {
// para cada fila
        bytes = ancho;
// bytes a enviar por fila
        fila=x;
// posicion de inicio en la fila
        LADOdisplay(x>63);
// 0a63 => lado 0; 63a127 => lado 1
        x=(x>63)?(x-64):x;
// corrige x si x > 63
        ESCRIBIRCMDdisplay (0x40+x);
// horizontal = x
        ESCRIBIRCMDdisplay(0xB8+pag);
// página while(bytes–) {
        ESCRIBIRDATdisplay(*(imgdata++));
// datos
        if(++fila == 64) {
// cambio de controlador ?
        fila=0;
// corrige columna
        LADOdisplay(1);
// controlador derecho
        ESCRIBIRCMDdisplay (0x40);
// horizontal = 0
        SCRIBIRCMDdisplay(0xB8+page);
// página
                }
        }
        pag++;
}
```



DIBUJO DE FUNCIONES

Para dibujar funciones, tomamos una muestra de ésta para cada punto horizontal y calculamos la coordenada vertical de cada punto de la función. Por ejemplo, un coseno:

```
for(i = -64 ; i < 63 ; i++)
PUNTOdisplay( 64 + i, 32 - (int)
(32 * (cos (i * 3.14159 / 64))));
```

Cómo dibujar letras

Asociando un "**renglón**" a cada página del display, y manteniendo todos los puntos de un carácter dentro del espacio de un mismo controlador, podemos simplificar en gran medida el proceso de dibujar caracteres.

Utilizando una tipografía de **5 x 7**, tenemos 7 puntos de alto, que cabe dentro de los 8 bits de una página, dejando separación entre líneas.

Si dejamos un punto de separación entre caracteres en sentido horizontal, tenemos ocupados 6 puntos por carácter. Dentro del espacio de un controlador (64 puntos) caben, entonces, 10 caracteres (60 puntos), con 4 puntos sin usar a ambos lados.

De este modo, tenemos la posibilidad de imprimir en una hoja de 8 renglones con 20 filas cada uno (10 filas por controlador). Por ejemplo, para imprimir un texto apuntado por **string** en el renglón **fila**, columna **col**, hacemos lo siguiente:

```
while (chr=*(string++)){
x = (col>9)? 6 * (col-10) : 6*col;
// corrige si col > 9
if(col > 9)
// 0 a 9 => lado 0; 10 a 20 => lado 1
LADOdisplay(1);

else {
LADOdisplay(0);
        x += 4;
// dejo un borde de 4 pixels a los lados
        }
        ESCRIBIRCMDdisplay (0x40+x);
// horizontal = x
        ESCRIBIRCMDdisplay(0xB8+fila);
// página
        base = 5 * (chr - 0x20);
        for(i=0; i < 5; i++)
ESCRIBIRDATdisplay( tabla5x7[ base + i] );
        ESCRIBIRDATdisplay(0);
// separador
        col++;                          }
```

DIBUJAR CARACTERES

Los caracteres se guardan en una tabla de **5 bytes** por carácter; cada byte contiene una columna del carácter. Copiamos esos 5 bytes a las 5 columnas de la página deseada del display. Recordemos que una columna del display contiene **8 bits** en sentido vertical, con el menos significativo arriba. El carácter se forma en pantalla como vemos en la **Figura 21**.

DIBUJAR GRÁFICOS Y TEXTOS PASO A PASO

Analizaremos algunos detalles prácticos del diseño y armado de la placa de conexión con el display. En el

DIBUJO DE LÍNEAS

Para dibujar líneas, se utiliza el algoritmo de Bresenham, que permite obtener, paso a paso, los puntos de la recta que pasa por dos puntos dados, utilizando sólo sumas y restas. Esto puede usarse para interpolar los puntos intermedios, entre otras cosas.

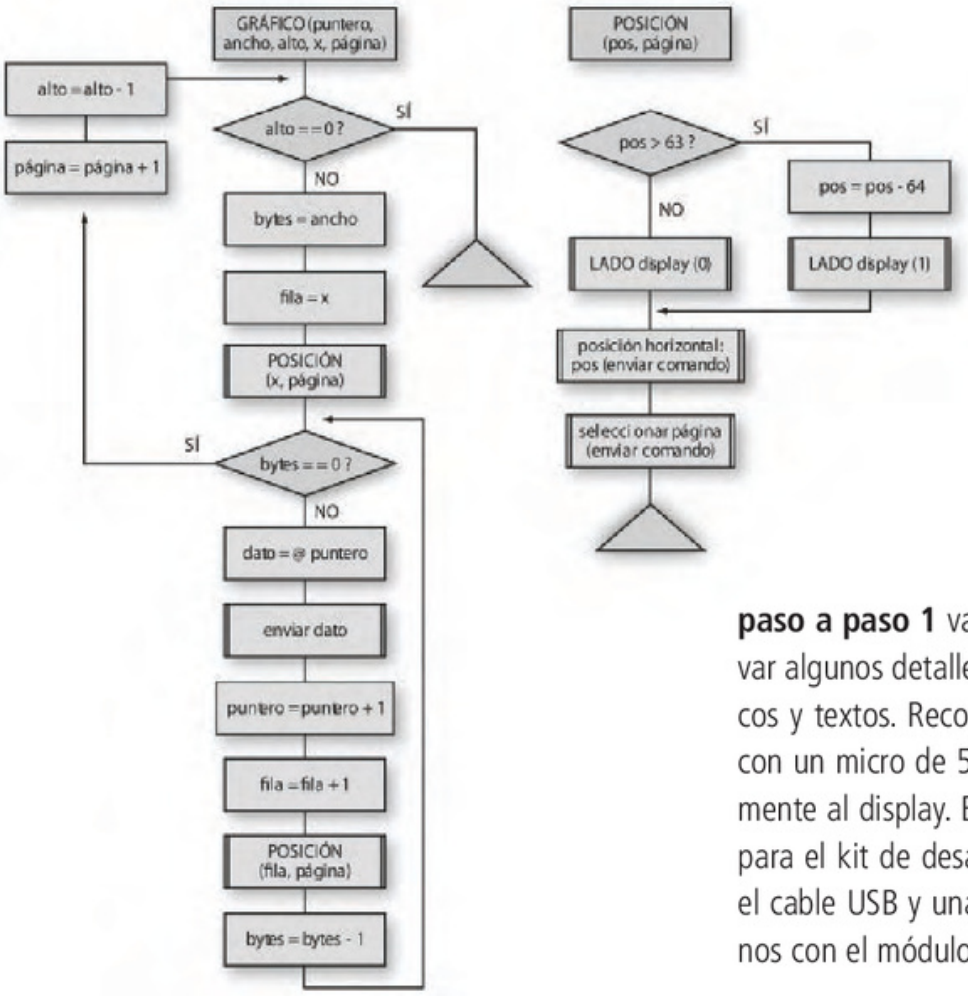


FIGURA 20. Algoritmos de envío de iconos al display. Simplificamos el proceso aprovechando las páginas de éste.

paso a paso 1 vamos a compilar, depurar y observar algunos detalles de operación para dibujar gráficos y textos. Recordemos que si deseamos trabajar con un micro de 5 V, podemos conectarnos directamente al display. En este caso, las instrucciones son para el kit de desarrollo de **RCM5700**, que incluye el cable USB y una placa de soporte para conectarnos con el módulo.

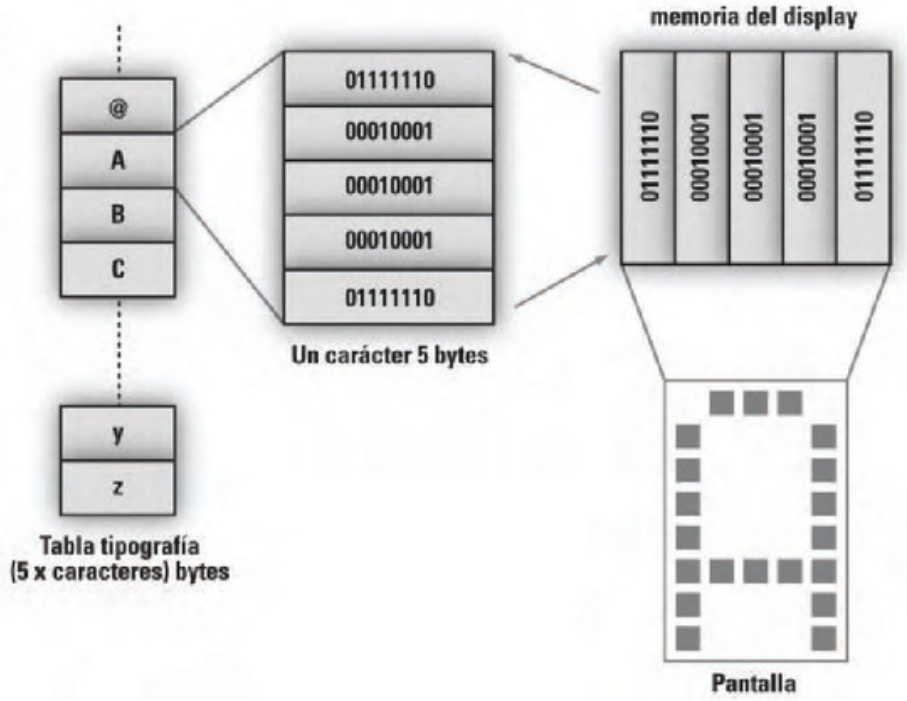


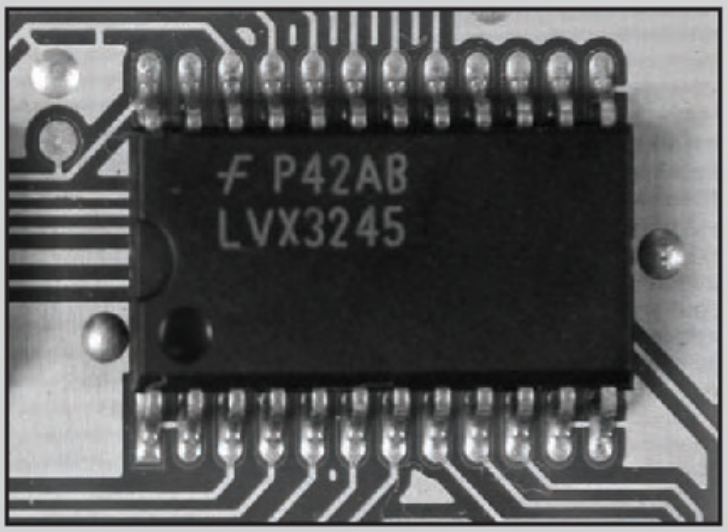
FIGURA 21. El proceso de dibujar una letra en pantalla. Las tipografías se arman en una tabla.

PASO A PASO /1
Cómo dibujar gráficos y textos

1

Los displays vienen en varios formatos; los más comunes son pines en una línea y en dos líneas; en este caso, usamos el primero. Es ideal soldar una tira de pines a **90º** del display y usar una hembra para pines a la que se le suelda un cable plano.

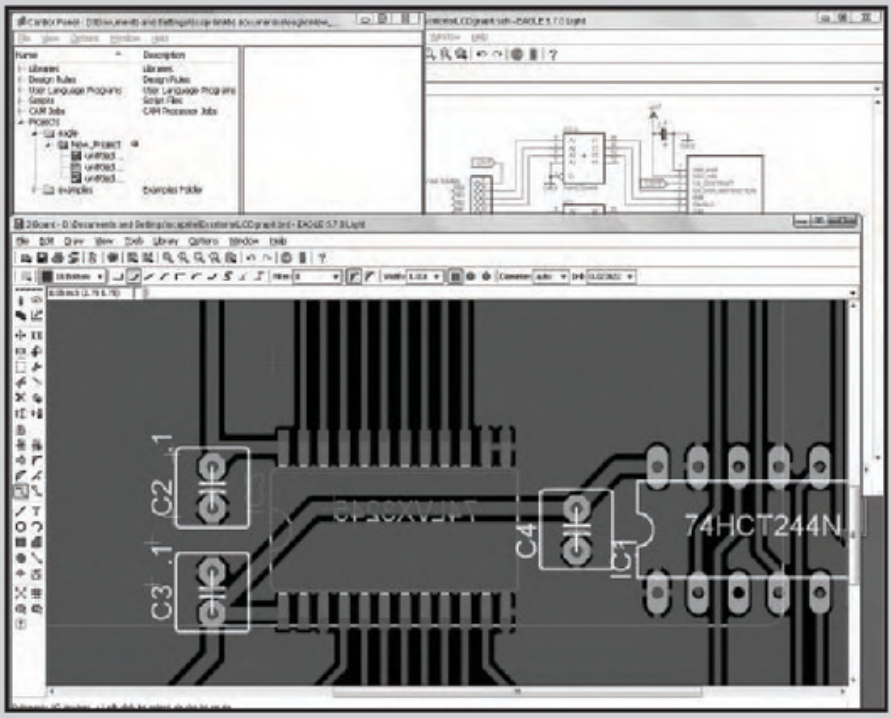
2



El **74HCT244** se consigue en encapsulado **DIP**, pero el **74LVX3245** es **SMD**. Por esta razón, se recomienda realizar una placa de circuito impreso en **Pertinax**. De igual modo, puede hacer el diseño en **Eagle**.

PASO A PASO /1 (cont.)

3



En el diseño del **PCB**, debe prestar atención a los capacitores de **1 µF**, tienen que estar cerca de los chips correspondientes. La conexión a la fuente de alimentación (**5 V**) tiene que hacerse cerca del display. Para la conexión al display y micro, emplee tiras de pines.

4

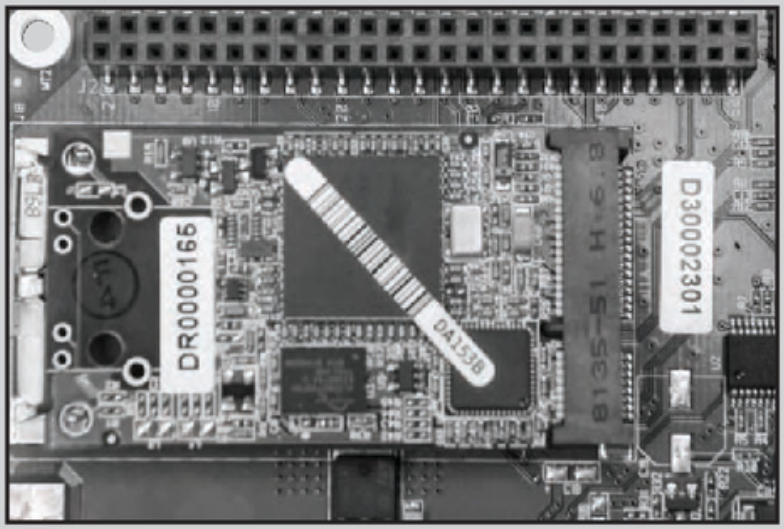
ELECTRONICAL CHARACTERISTICS

ITEM	SYMBOL	CONDITION	STANDARD			UNIT
			MIN	TYP	MAX	
Input voltage	Vdd	+ 5V	4.7	5.0	5.5	V
Supply current	Idd	Vdd=5V	---	8	---	mA
Recommended LCD driving voltage for normal temp version module	Vdd-V0	0 °C	---	9.8	---	V
		25 °C	---	9.5	---	
		50 °C	---	9.3	---	
LED forward voltage	VF	25 °C	---	4.2	4.5	V
LED forward current	IF	25 °C	---	360	---	mA

Verifique la conexión del **backlight** en la hoja de datos del display y calcule el valor de **R2 = (5 V - VLED) / ILED**. El fabricante da VLED e ILED máximas. Debe elegir una corriente menor de la mitad.

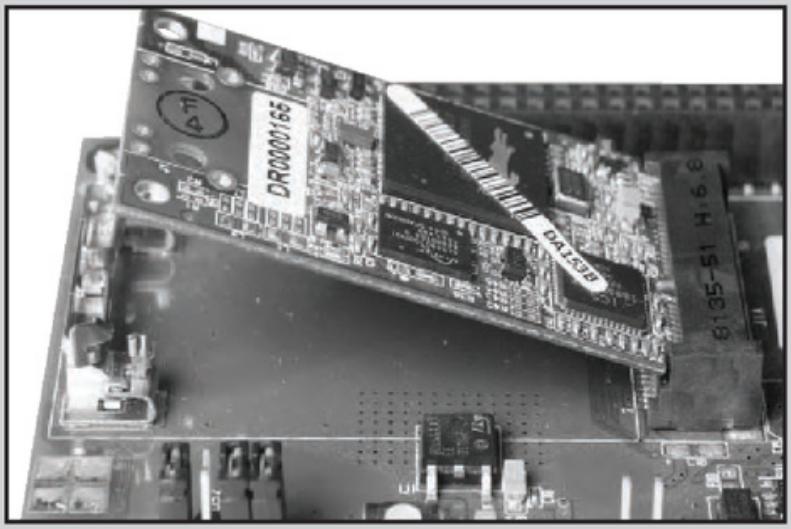
PASO A PASO /1 (cont.)

5



Para la conexión con el micro, puede usar un esquema similar al del display con pines, conector hembra y cable plano. Del lado que conecta a la placa de desarrollo, use la conexión correspondiente. En el caso del **RCM5700**, dispone de una hembra de pines doble; puede usar una tira de pines y soldarles el cable.

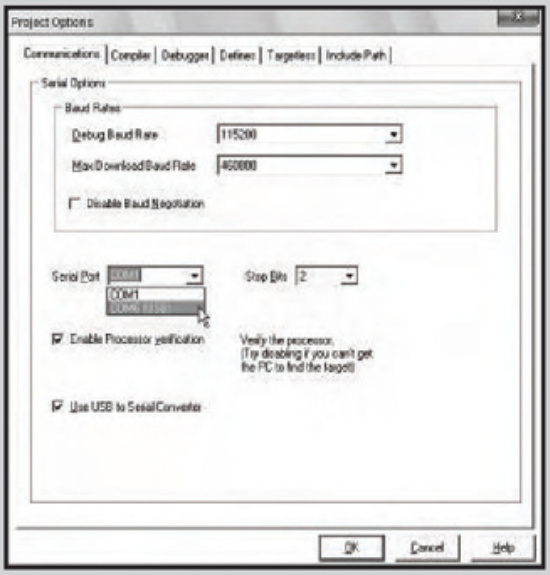
6



El código para **Dynamic C** se encuentra en la página web de la editorial. El compilador se provee con el kit o puede bajarlo del sitio del fabricante. Luego, inserte el módulo **RCM5700** en la placa de desarrollo. A continuación, debe conectar la placa de desarrollo por el conector **mini-USB** a un puerto USB de la PC.

PASO A PASO /1 (cont.)

7



Una vez que la PC reconoce el puerto que provee el chip FTDI de la placa de desarrollo, configurelo en Dynamic C, al elegir en el menú **Options** la alternativa **Project Options**, y luego, la solapa **Communications**.

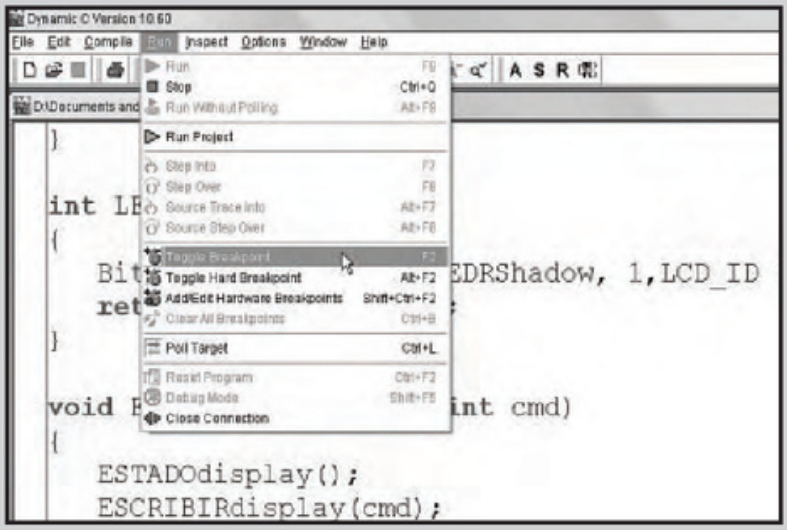
8

```
/* PROGRAMA PRINCIPAL */

main()
{

      INITdisplay();

      BORRARdisplay();

      GRAFICOdisplay(users,0,2,128,4);

      TEXTOENdisplay(0,3,"Coleccionables");
      TEXTOENdisplay(7,0,"Electronica Digital");

      while(1);

}
```

Abra el archivo **demoLCD.c**, compílelo y ejecútelo presionando el icono de reproducción o la tecla **F5**. Si quiere ejecutar el programa paso a paso, puede detenerlo con el icono correspondiente y avanzar mediante las teclas **F7** y **F8**, como en la mayoría de los entornos de desarrollo.

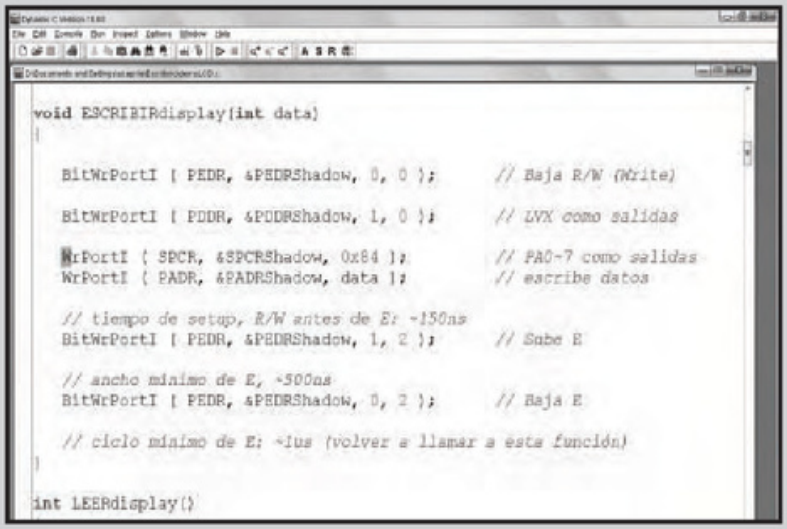
PASO A PASO /1 (cont.)

9



Posicione el cursor y ponga un breakpoint en la función **`ESCRIBIRDATdisplay()`**, mediante la opción **`Run:Toggle Breakpoint`** o al pulsar la tecla **F2**; verá un rectángulo **rojo**. Cuando la ejecución se detiene en dicha función, observará un rectángulo verde.

10



Ejecute paso a paso la función mediante la tecla **F7** y podrá observar, con un multímetro u osciloscopio, cómo cambia el estado de los pines y el orden de activación del conversor de nivel, el display y el puerto de I/O del micro.

PASO A PASO /1 (cont.)

11

Elimine el breakpoint y coloque otro en la función **`GRAFICOdisplay()`**. Ejecute el código y, cuando tenga el cursor de color verde, ejecute paso a paso sin entrar en las demás funciones, con la tecla **F8**. Observe cómo los datos van siendo enviados al display uno a uno.

12

Terminado el proceso, vea el logo completo. Luego, puede eliminar el breakpoint y poner uno en el bloque **`for()`**, que realiza el dibujo de un carácter en la función **`TEXTOENdisplay()`**. Ejecute paso a paso mediante la tecla **F7** y observe cómo se dibuja el carácter en el display.

Multiple **choice**

1 ¿A qué empresa perteneció el primer microprocesador exitoso?

a- Motorola.
b- AMD.
c- IBM.
d- Intel.

2 ¿Cuál de los siguientes microprocesadores introdujo la innovación del set alternativo de registros?

a- 6800.
b- Z80.
c- AVR.
d- MCS51.

3 ¿Cuál de los siguientes fue el primer microprocesador exitoso de Motorola?

a- 6800.
b- Z80.
c- AVR.
d- MCS51.

4 ¿Cuál de los siguientes se caracteriza por tener una gran cantidad de registros del procesador?

a- 6800.
b- Z80.
c- AVR.
d- MCS51.

5 ¿Cuál de los siguientes es un microcontrolador diseñado por Intel que ha pasado a ser el estándar de facto?

a- 6800.
b- Z80.
c- AVR.
d- MCS51.

6 ¿En cuál microprocesador se basa la familia Rabbit?

a- 6800.
b- Z80.
c- AVR.
d- MCS51.

Respuestas: 1 d, 2 b, 3 a, 4 c, 5 d, 6 b.

Servicios **al lector**

Encontraremos información adicional relacionada con el contenido, que servirá para complementar lo aprendido.

Índice **temático**

A

B

C

D

F

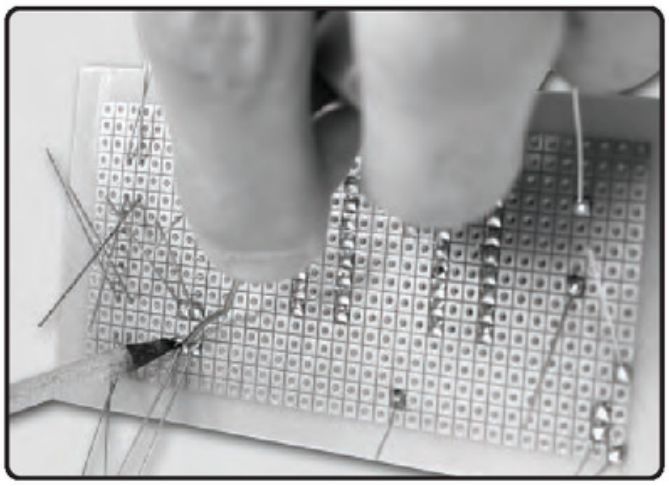
H

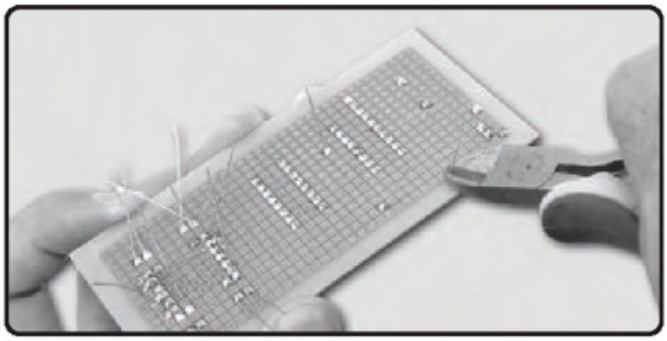
I

K

L

M





O

P

R

S

T

U

V

W

X

Z

Programas **relacionados**

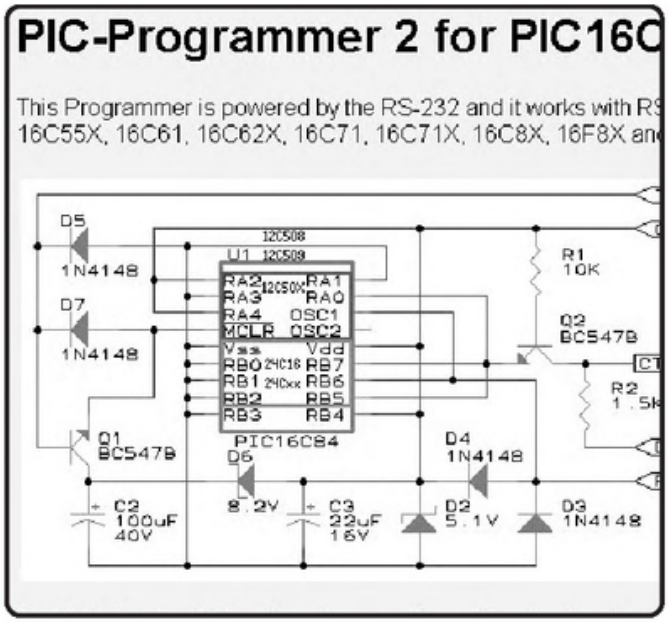
MPLAB IDE

www.microchip.com

Programa para la escritura y la simulación de programas en lenguaje ensamblador para PIC, desarrollado por el fabricante de los microcontroladores PIC. Podemos descargarlo y utilizarlo en forma gratuita.

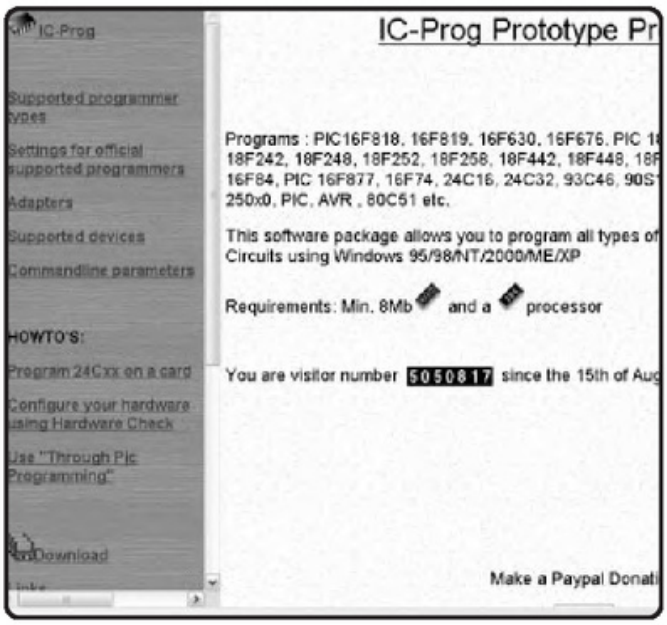
IC-PROG

www.ic-prog.com



Programa que nos permite la grabación de microcontroladores PIC y otros dispositivos a través de un grabador compatible conectado a la computadora.

PIC DELAYER



Este programa calcula automáticamente subrutinas de retardo en lenguaje ensamblador para microcontroladores PIC. Sólo debemos colocar la frecuencia del oscilador y el tiempo deseado.

PIC SIMULATOR IDE

www.oshonsoft.com/pic.html

Software de simulación de proyectos con microcontroladores PIC. Permite simular los programas escritos para una gran variedad de microcontroladores de la familia PIC de Microchip.

CLAVES PARA COMPRAR
UN LIBRO DE COMPUTACIÓN

1 SOBRE EL AUTOR Y LA EDITORIAL

Revise que haya un cuadro "sobre el autor", en el que se informe sobre su experiencia en el tema. En cuanto a la editorial, es conveniente que sea especializada en computación.

2 PRESTE ATENCIÓN AL DISEÑO

Compruebe que el libro tenga guías visuales, explicaciones paso a paso, recuadros con información adicional y gran cantidad de pantallas. Su lectura será más ágil y atractiva que la de un libro de puro texto.

3 COMPARE PRECIOS

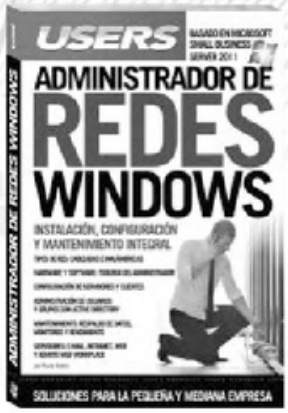
Suele haber grandes diferencias de precio entre libros del mismo tema; si no tiene el valor en tapa, pregunte y compare.

4 ¿TIENE VALORES AGREGADOS?

Desde un sitio exclusivo en la Red, un Servicio de Atención al Lector, la posibilidad de leer el sumario en la Web para evaluar con tranquilidad la compra, y hasta la presencia de adecuados índices temáticos, todo suma al valor de un buen libro.

5 VERIFIQUE EL IDIOMA

No solo el del texto; también revise que las pantallas incluidas en el libro estén en el mismo idioma del programa que usted utiliza.



Administrador de RedesWindows

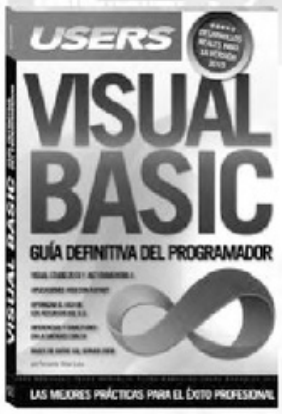
En esta obra presentamos los fundamentos para la instalación, configuración y puesta en marcha de una red corporativa. El libro está centrado en Windows Small Business Server 2011, sistema operativo pensado para pequeñas y medianas empresas que necesitan llevar adelante soluciones de networking a bajo costo.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / 978-987-1773-80-0

HTML 5

En esta obra presentamos un nuevo paradigma de Internet que cambia de manera sustancial todo lo que conocíamos sobre el diseño y desarrollo web. A lo largo de sus capítulos, aprenderemos sobre los estándares web que existen, y conoceremos las diferencias entre las versiones anteriores y la actual del lenguaje.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / 978-987-1773-79-4



Visual Basic

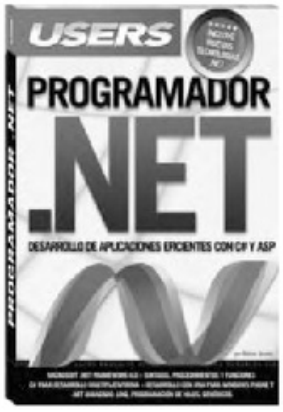
Este libro está escrito para aquellos usuarios que quieran aprender a programar en VB.NET. Desde el IDE de programación hasta el desarrollo de aplicaciones del mundo real en la versión 2010 de Visual Studio, todo está contemplado para conocer en profundidad VB.NET al finalizar la lectura.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / 978-987-1773-57-2

Microcontroladores

Este manual es ideal para aquellos que quieran iniciarse en la programación de microcontroladores. A través de esta obra, podrán conocer los fundamentos de los sistemas digitales, aprender sobre los microcontroladores PIC 16F y 18F, hasta llegar a conectar los dispositivos de forma inalámbrica, entre muchos otros proyectos.

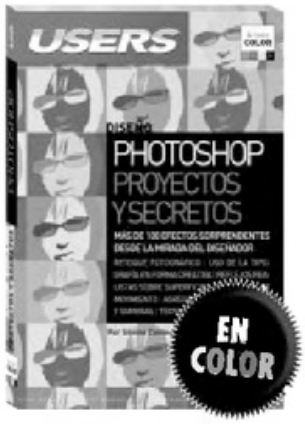
→ COLECCIÓN: MANUALES USERS
→ 320 páginas / 978-987-1773-56-5



Programador.NET

Este libro está dirigido a todos aquellos que quieran iniciarse en el desarrollo bajo lenguajes Microsoft. A través de los capítulos del manual, aprenderemos sobre POO y la programación con tecnologías .NET, su aplicación, cómo interactúan entre sí y de qué manera se desenvuelven con otras tecnologías existentes.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / 978-987-1773-26-8



Photoshop: proyectos y secretos

En esta obra aprenderemos a utilizar Photoshop, desde la original mirada de la autora. Con el foco puesto en la comunicación visual, a lo largo del libro adquiriremos conocimientos teóricos, al mismo tiempo que avanzaremos sobre la práctica, con todos los efectos y herramientas que ofrece el programa.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / 978-987-1773-25-1

¡Léalo antes Gratis!

En nuestro sitio, obtenga GRATIS un capítulo del libro de su elección antes de comprarlo.

WordPress

Este manual está dirigido a todos aquellos que quieran presentar sus contenidos o los de sus clientes a través de WordPress. En sus páginas el autor nos enseñará desde cómo llevar adelante la administración del blog hasta las posibilidades de interacción con las redes sociales.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / 978-987-1773-18-3

Administrador de servidores

Este libro es la puerta de acceso para ingresar en el apasionante mundo de los servidores. Aprenderemos desde los primeros pasos sobre la instalación, configuración, seguridad y virtualización; todo para cumplir el objetivo final de tener el control de los servidores en la palma de nuestras manos.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / ISBN 978-987-1773-19-0

Windows 7: Trucos y secretos

Este libro está dirigido a todos aquellos que quieran sacar el máximo provecho de Windows 7, las redes sociales y los dispositivos ultraportátiles del momento. A lo largo de sus páginas, el lector podrá adentrarse en estas tecnologías mediante trucos inéditos y consejos asombrosos.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / ISBN 978-987-1773-17-6

Desarrollo PHP + MySQL

Este libro presenta la fusión de dos de las herramientas más populares para el desarrollo de aplicaciones web de la actualidad: PHP y MySQL. En sus páginas, el autor nos enseñará las funciones del lenguaje, de modo de tener un acercamiento progresivo, y aplicar lo aprendido en nuestros propios desarrollos.

→ COLECCIÓN: MANUALES USERS
→ 432 páginas / ISBN 978-987-1773-16-9

Excel 2010

Este manual resulta ideal para quienes se inician en el uso de Excel, así como también para los usuarios que quieran conocer las nuevas herramientas que ofrece la versión 2010. La autora nos enseñará desde cómo ingresar y proteger datos hasta la forma de imprimir ahorrando papel y tiempo.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / ISBN 978-987-1773-15-2

Técnico Hardware

Esta obra es fundamental para ganar autonomía al momento de reparar la PC. Aprenderemos a diagnósticar y solucionar las fallas, así como a prevenirlas a través del mantenimiento adecuado, todo explicado en un lenguaje práctico y sencillo.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / ISBN 978-987-1773-14-5

PHP Avanzado

Este libro brinda todas las herramientas necesarias para acercar al trabajo diario del desarrollador los avances más importantes incorporados en PHP 6. En sus páginas, repasaremos todas las técnicas actuales para potenciar el desarrollo de sitios web.

→ COLECCIÓN: MANUALES USERS
→ **400** páginas / ISBN 978-987-1773-07-7

AutoCAD

Este manual nos presenta un recorrido exhaustivo por el programa más difundido en dibujo asistido por computadora a nivel mundial, en su versión 2010. En sus páginas, aprenderemos desde cómo trabajar con dibujos predeterminados hasta la realización de objetos 3D.

→ COLECCIÓN: MANUALES USERS
→ **384** páginas / ISBN 978-987-1773-06-0

Windows Seven Avanzado

Esta obra nos presenta un recorrido exhaustivo que nos permitirá acceder a un nuevo nivel de complejidad en el uso de Windows 7. Todas las herramientas son desarrolladas con el objetivo de acompañar al lector en el camino para ser un usuario experto.

→ COLECCIÓN: MANUALES USERS
→ **352** páginas / ISBN 978-987-1773-08-4

Photoshop

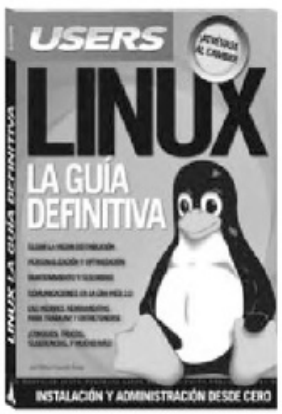
En este libro aprenderemos sobre las más novedosas técnicas de edición de imágenes en Photoshop. El autor nos presenta de manera clara y práctica todos los conceptos necesarios, desde la captura digital hasta las más avanzadas técnicas de retoque.

→ COLECCIÓN: MANUALES USERS
→ **320** páginas / ISBN 978-987-1773-05-3

Grabación y producción de música

En este libro repasaremos todos los aspectos del complejo mundo de la producción musical. Desde las cuestiones para tener en cuenta al momento de la composición, hasta la mezcla y el masterizado, así como la distribución final del producto.

→ COLECCIÓN: MANUALES USERS
→ **320** páginas / ISBN 978-987-1773-04-6



Linux

Este libro es una completa guía para migrar e iniciarse en el fascinante mundo del software libre. En su interior, el lector conocerá las características de Linux, desde su instalación hasta las opciones de entretenimiento, con todas las ventajas de seguridad que ofrece el sistema.

→ COLECCIÓN: MANUALES USERS
→ **320** páginas / ISBN 978-987-26013-8-6

¡Léalo antes Gratis!

En nuestro sitio, obtenga GRATIS un capítulo del libro de su elección antes de comprarlo.

Premiere + After Effects

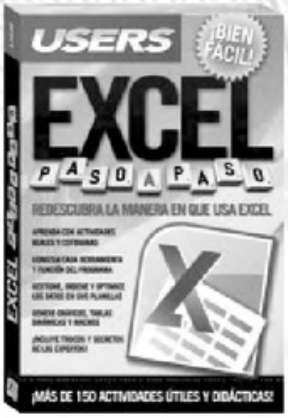
Esta obra nos presenta un recorrido detallado por las aplicaciones audiovisuales de Adobe: Premiere Pro, After Effects y Soundbooth. Todas las técnicas de los profesionales, desde la captura de video hasta la creación de efectos, explicadas de forma teórica y práctica.

→ COLECCIÓN: MANUALES USERS
→ **320** páginas / ISBN 978-987-26013-9-3

Office 2010

En este libro aprenderemos a utilizar todas las aplicaciones de la suite, en su versión 2010. Además, su autora nos mostrará las novedades más importantes, desde los minigráficos de Excel hasta Office Web Apps, todo presentado en un libro único.

→ COLECCIÓN: MANUALES USERS
→ **352** páginas / ISBN 978-987-26013-6-2



Excel Paso a Paso

En esta obra encontraremos una increíble selección de proyectos pensada para aprender mediante la práctica la forma de agilizar todas las tareas diarias. Todas las actividades son desarrolladas en procedimientos paso a paso de una manera didáctica y fácil de comprender.

→ COLECCIÓN: PASO A PASO
→ **320** páginas / ISBN 978-987-26013-4-8

C#

Este libro es un completo curso de programación con C# actualizado a la versión 4.0. Ideal tanto para quienes desean migrar a este potente lenguaje, como para quienes quieran aprender a programar desde cero en Visual Studio 2010.

→ COLECCIÓN: MANUALES USERS
→ **400** páginas / ISBN 978-987-26013-5-5

200 Respuestas Seguridad

Esta obra es una guía básica que responde, en forma visual y práctica, a todas las preguntas que necesitamos contestar para conseguir un equipo seguro. Definiciones, consejos, claves y secretos, explicados de manera clara, sencilla y didáctica.

→ COLECCIÓN: MANUALES USERS
→ **320** páginas / ISBN 978-987-26013-1-7

Funciones en Excel

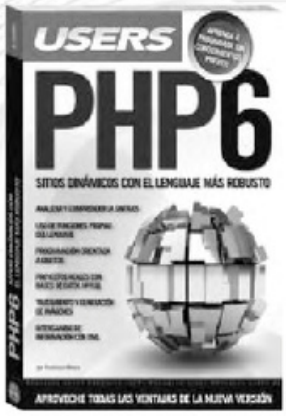
Este libro es una guía práctica de uso y aplicación de todas las funciones de la planilla de cálculo de Microsoft. Desde las funciones de siempre hasta las más complejas, todas presentadas a través de ejemplos prácticos y reales.

→ COLECCIÓN: 200 RESPUESTAS
→ **368** páginas / ISBN 978-987-26013-0-0

Proyectos con Windows 7

En esta obra aprenderemos cómo aprovechar al máximo todas las ventajas que ofrece la PC. Desde cómo participar en las redes sociales hasta las formas de montar una oficina virtual, todo presentado en 120 proyectos únicos.

→ COLECCIÓN: MANUALES USERS
→ **352** páginas / ISBN 978-987-663-036-8



PHP 6

Este libro es un completo curso de programación en PHP en su versión 6.0. Un lenguaje que se destaca tanto por su versatilidad como por el respaldo de una amplia comunidad de desarrolladores, que lo convierten en un punto de partida ideal para quienes comienzan a programar.

→ COLECCIÓN: MANUALES USERS
→ **368** páginas / ISBN 978-987-663-039-9

200 Respuestas: Blogs

Esta obra es una completa guía que responde a las preguntas más frecuentes de la gente sobre la forma de publicación más poderosa de la Web 2.0. Definiciones, consejos, claves y secretos, explicados de manera clara, sencilla y didáctica.

→ COLECCIÓN: 200 RESPUESTAS
→ **320** páginas / ISBN 978-987-663-037-5

Hardware paso a paso

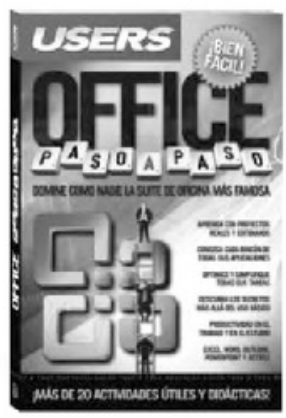
En este libro encontraremos una increíble selección de actividades que abarcan todos los aspectos del hardware. Desde la actualización de la PC hasta el overclocking de sus componentes, todo en una presentación nunca antes vista, realizada íntegramente con procedimientos paso a paso.

→ COLECCIÓN: PASO A PASO
→ **320** páginas / ISBN 978-987-663-034-4

200 Respuestas: Windows 7

Esta obra es una guía básica que responde, en forma visual y práctica, a todas las preguntas que necesitamos conocer para dominar la última versión del sistema operativo de Microsoft. Definiciones, consejos, claves y secretos, explicados de manera clara, sencilla y didáctica.

→ COLECCIÓN: 200 RESPUESTAS
→ **320** páginas / ISBN 978-987-663-035-1



Office paso a paso

Este libro presenta una increíble colección de proyectos basados en la suite de oficina más usada en el mundo. Todas las actividades son desarrolladas con procedimientos paso a paso de una manera didáctica y fácil de comprender.

→ COLECCIÓN: PASO A PASO
→ **320** páginas / ISBN 978-987-663-030-6

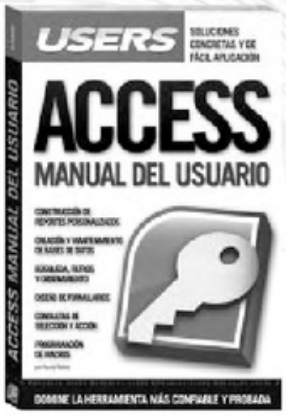
¡Léalo antes Gratis!

En nuestro sitio, obtenga GRATIS un capítulo del libro de su elección antes de comprarlo.

101 Secretos de Hardware

Esta obra es la mejor guía visual y práctica sobre hardware del momento. En su interior encontraremos los consejos de los expertos sobre las nuevas tecnologías, las soluciones a los problemas más frecuentes, cómo hacer overclocking, modding, y muchos más trucos y secretos.

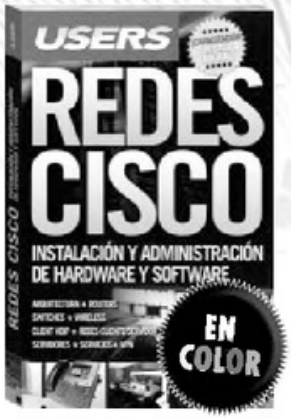
→ COLECCIÓN: MANUALES USERS
→ **352** páginas / ISBN 978-987-663-029-0



Access

Este manual nos introduce de lleno en el mundo de Access para aprender a crear y administrar bases de datos de forma profesional. Todos los secretos de una de las principales aplicaciones de Office, explicados de forma didáctica y sencilla.

→ COLECCIÓN: MANUALES USERS
→ **320** páginas / ISBN 978-987-663-025-2



Redes Cisco

Este libro permitirá al lector adquirir todos los conocimientos necesarios para planificar, instalar y administrar redes de computadoras. Todas las tecnologías y servicios Cisco, desarrollados de manera visual y práctica en una obra única.

→ COLECCIÓN: MANUALES USERS
→ **320** páginas / ISBN 978-987-663-024-5

Proyectos con Office

Esta obra nos enseña a usar las principales herramientas de Office a través de proyectos didácticos y útiles. En cada capítulo encontraremos la mejor manera de llevar adelante todas las actividades del hogar, la escuela y el trabajo.

→ COLECCIÓN: MANUALES USERS
→ **352** páginas / ISBN 978-987-663-023-8

Dreamweaver y Fireworks

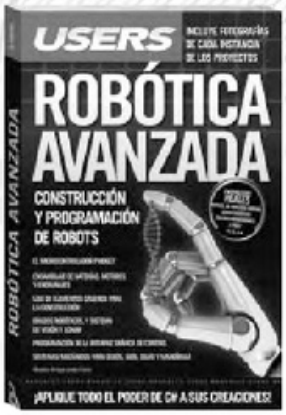
Esta obra nos presenta las dos herramientas más poderosas para la creación de sitios web profesionales de la actualidad. A través de procedimientos paso a paso, nos muestra cómo armar un sitio real con Dreamweaver y Fireworks sin necesidad de conocimientos previos.

→ COLECCIÓN: MANUALES USERS
→ **320** páginas / ISBN 978-987-663-022-1

Excel revelado

Este manual contiene una selección de más de 150 consultas de usuarios de Excel y todas las respuestas de Claudio Sánchez, un reconocido experto en la famosa planilla de cálculo. Todos los problemas encuentran su solución en esta obra imperdible.

→ COLECCIÓN: MANUALES USERS
→ **336** páginas / ISBN 978-987-663-021-4



Robótica avanzada

Esta obra nos permitirá ingresar al fascinante mundo de la robótica. Desde el ensamblaje de las partes hasta su puesta en marcha, todo el proceso está expuesto de forma didáctica y sencilla para así crear nuestros propios robots avanzados.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / ISBN 978-987-663-020-7

Windows 7

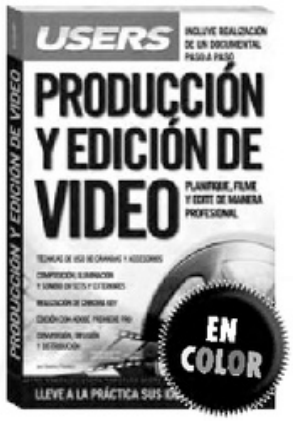
En este libro encontraremos las claves y los secretos destinados a optimizar el uso de nuestra PC tanto en el trabajo como en el hogar. Aprenderemos a llevar adelante una instalación exitosa y a utilizar todas las nuevas herramientas que incluye esta versión.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / ISBN 978-987-663-015-3

De Windows a Linux

Esta obra nos introduce en el apasionante mundo del software libre a través de una completa guía de migración, que parte desde el sistema operativo más conocido: Windows. Aprenderemos cómo realizar gratuitamente aquellas tareas que antes hacíamos con software pago.

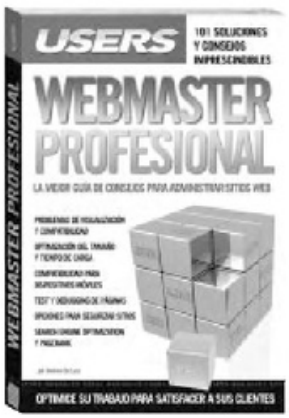
→ COLECCIÓN: MANUALES USERS
→ 336 páginas / ISBN 978-987-663-013-9



Producción y edición de video

Un libro ideal para quienes deseen realizar producciones audiovisuales con bajo presupuesto. Tanto estudiantes como profesionales encontrarán cómo adquirir las habilidades necesarias para obtener una salida laboral con una creciente demanda en el mercado.

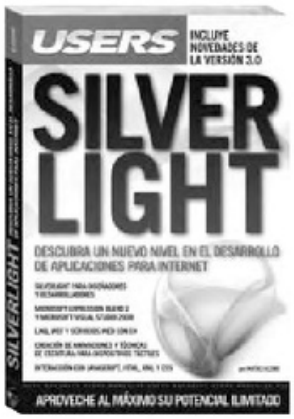
→ COLECCIÓN: MANUALES USERS
→ 336 páginas / ISBN 978-987-663-012-2



Webmaster Profesional

Esta obra explica cómo superar los problemas más frecuentes y complejos que enfrenta todo administrador de sitios web. Ideal para quienes necesiten conocer las tendencias actuales y las tecnologías en desarrollo que son materia obligada para dominar la Web 2.0.

→ COLECCIÓN: MANUALES USERS
→ 336 páginas / ISBN 978-987-663-011-5



Silverlight

Este manual nos introduce en un nuevo nivel en el desarrollo de aplicaciones interactivas a través de Silverlight, la opción multiplataforma de Microsoft. Quien consiga dominarlo creará aplicaciones visualmente impresionantes, acordes a los tiempos de la incipiente Web 3.0.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / ISBN 978-987-663-010-8

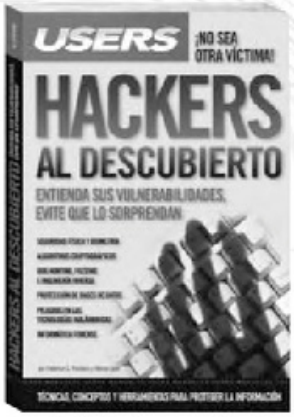
¡Léalo antes Gratis!

En nuestro sitio, obtenga GRATIS un capítulo del libro de su elección antes de comprarlo.

Flash Extremo

Este libro nos permitirá aprender a fondo Flash CS4 y ActionScript 3.0 para crear aplicaciones web y de escritorio. Una obra imperdible sobre uno de los recursos más empleados en la industria multimedia que nos permitirá estar a la vanguardia del desarrollo.

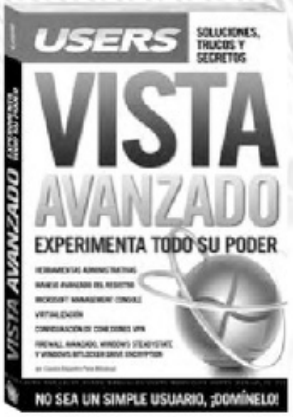
→ COLECCIÓN: MANUALES USERS
→ 320 páginas / ISBN 978-987-663-009-2



Hackers al descubierto

Esta obra presenta un panorama de las principales técnicas y herramientas utilizadas por los hackers, y de los conceptos necesarios para entender su manera de pensar, prevenir sus ataques y estar preparados ante las amenazas más frecuentes.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / ISBN 978-987-663-008-5



Vista avanzado

Este manual es una pieza imprescindible para convertirnos en administradores expertos de este popular sistema operativo. En sus páginas haremos un recorrido por las herramientas fundamentales para tener máximo control sobre todo lo que sucede en nuestra PC.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / ISBN 978-987-663-007-8

101 Secretos de Excel

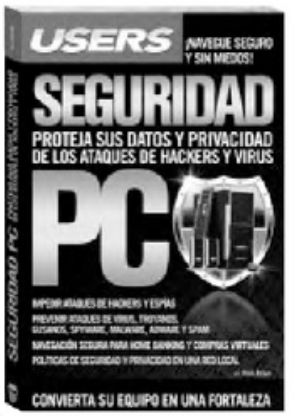
Una obra absolutamente increíble, con los mejores 101 secretos para dominar el programa más importante de Office. En sus páginas encontraremos un material sin desperdicios que nos permitirá realizar las tareas más complejas de manera sencilla.

→ COLECCIÓN: MANUALES USERS
→ 336 páginas / ISBN 978-987-663-005-4

Electrónica & microcontroladores PIC

Una obra ideal para quienes desean aprovechar al máximo las aplicaciones prácticas de los microcontroladores PIC y entender su funcionamiento. Un material con procedimientos paso a paso y guías visuales, para crear proyectos sin límites.

→ COLECCIÓN: MANUALES USERS
→ 368 páginas / ISBN 978-987-663-002-3



Seguridad PC

Este libro contiene un material imprescindible para proteger nuestra información y privacidad. Aprenderemos cómo reconocer los síntomas de infección, las medidas de prevención a tomar, y finalmente, la manera de solucionar los problemas.

→ COLECCIÓN: MANUALES USERS
→ 336 páginas / ISBN 978-987-663-004-7

Hardware desde cero

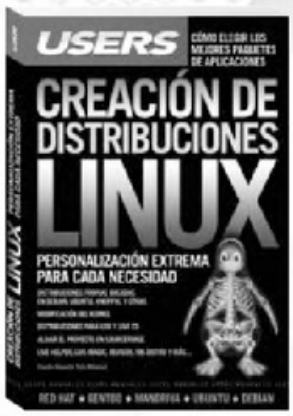
Este libro brinda las herramientas necesarias para entender de manera amena, simple y ordenada cómo funcionan el hardware y el software de la PC. Está destinado a usuarios que quieran independizarse de los especialistas necesarios para armar y actualizar un equipo.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / ISBN 978-987-663-001-6

200 Respuestas: Photoshop

Esta obra es una guía que responde, en forma visual y práctica, a todas las preguntas que necesitamos contestar para conocer y dominar Photoshop CS3. Definiciones, consejos, claves y secretos, explicados de manera clara, sencilla y didáctica.

→ COLECCIÓN: 200 RESPUESTAS
→ 320 páginas / ISBN 978-987-1347-98-8



Creación de distribuciones Linux

En este libro recorreremos todas las alternativas para crear distribuciones personalizadas: desde las más sencillas y menos customizables, hasta las más avanzadas, que nos permitirán modificar el corazón mismo del sistema, el kernel.

→ COLECCIÓN: MANUALES USERS
→ 336 páginas / ISBN 978-987-1347-99-5

Métodos ágiles

Este libro presenta una alternativa competitiva a las formas tradicionales de desarrollo y los últimos avances en cuanto a la producción de software. Ideal para quienes sientan que las técnicas actuales les resultan insuficientes para alcanzar metas de tiempo y calidad.

→ COLECCIÓN: DESARROLLADORES
→ 336 páginas / ISBN 978-987-1347-97-1

SuperBlogger

Esta obra es una guía para sumarse a la revolución de los contenidos digitales. En sus páginas, aprenderemos a crear un blog, y profundizaremos en su diseño, administración, promoción y en las diversas maneras de obtener dinero gracias a Internet.

→ COLECCIÓN: MANUALES USERS
→ 352 páginas / ISBN 978-987-1347-96-4

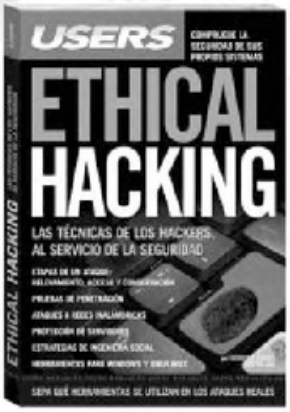
UML

Este libro es la guía adecuada para iniciarse en el mundo del modelado. Conoceremos todos los constructores y elementos necesarios para comprender la construcción de modelos y razonarlos de manera que reflejen los comportamientos de los sistemas.

→ COLECCIÓN: DESARROLLADORES
→ 320 páginas / ISBN 978-987-1347-95-7

¡Léalo antes Gratis!

En nuestro sitio, obtenga GRATIS un capítulo del libro de su elección antes de comprarlo.



Ethical Hacking

Esta obra expone una visión global de las técnicas que los hackers maliciosos utilizan en la actualidad para conseguir sus objetivos. Es una guía fundamental para obtener sistemas seguros y dominar las herramientas que permiten lograrlo.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / ISBN 978-987-1347-93-3

UNIX

Esta obra contiene un material imperdible, que nos permitirá dominar el sistema operativo más sólido, estable, confiable y seguro de la actualidad. En sus páginas encontraremos las claves para convertirnos en expertos administradores de FreeBSD.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / ISBN 978-987-1347-94-0

200 Respuestas: Excel

Esta obra es una guía básica que responde, en forma visual y práctica, a todas las preguntas que necesitamos conocer para dominar la versión 2007 de Microsoft Excel. Definiciones, consejos, claves y secretos, explicados de manera clara, sencilla y didáctica.

→ COLECCIÓN: 200 RESPUESTAS
→ 320 páginas / ISBN 978-987-1347-91-9

Hardware Extremo

En esta obra aprenderemos a llevar nuestra PC al límite, aplicar técnicas de modding, solucionar fallas y problemas avanzados, fabricar dispositivos inalámbricos caseros de alto alcance, y a sacarle el máximo provecho a nuestra notebook.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / ISBN 978-987-1347-90-2

Servicio Técnico de PC

Ésta es una obra que brinda las herramientas para convertirnos en expertos en el soporte y la reparación de los componentes internos de la PC. Está orientada a quienes quieran aprender o profundizar sus conocimientos en el área.

→ COLECCIÓN: MANUALES USERS
→ 320 páginas / ISBN 978-987-1347-89-6

Solución de Problemas PC

En esta obra encontraremos un material sin desperdicios que nos permitirá entender los síntomas que presentan los problemas graves, solucionarlos en caso de que algún imprevisto nos sorprenda y, finalmente, evitar que se repitan.

→ COLECCIÓN: MANUALES USERS
→ 336 páginas / ISBN 978-987-1347-88-9

Diseño Gráfico

Esta obra es una herramienta imprescindible para dominar las principales aplicaciones del paquete más famoso de Adobe y conocer los secretos utilizados por los expertos para diseñar de manera profesional.

→ COLECCIÓN: DISEÑO
→ 320 páginas / ISBN 978-987-1347-87-2

200 Respuestas: Redes

Esta obra es una guía básica que responde, en forma visual y práctica, a todas las preguntas que necesitamos plantearnos para conocer y dominar el mundo de las redes hogareñas, tanto cableadas como Wi-Fi.

→ COLECCIÓN: 200 RESPUESTAS
→ 320 páginas / ISBN 978-987-1347-86-5

200 Respuestas: Office

Una guía básica que responde, en forma visual y práctica, a todas las preguntas que necesitamos conocer para dominar la versión 2007 de la popular suite de Microsoft. Definiciones, consejos, claves y secretos, explicados de manera clara y didáctica.

→ COLECCIÓN: 200 RESPUESTAS
→ 320 páginas / ISBN 978-987-1347-85-8

Finanzas con Microsoft Excel

Este libro es una obra con un claro enfoque en lo práctico, plasmado en ejemplos no sólo útiles sino también reales; orientada a los profesionales que tienen la necesidad de aportar a sus empresas soluciones confiables, a muy bajo costo.

→ COLECCIÓN: PROFESSIONAL TOOLS
→ 256 páginas / ISBN 978-987-1347-84-1

Marketing en Internet

Este libro brinda las herramientas de análisis y los conocimientos necesarios para lograr un sitio con presencia sólida y alta tasa de efectividad. Una obra imprescindible para entender la manera en que los negocios se llevan a cabo en la actualidad.

→ COLECCIÓN: PROFESSIONAL TOOLS
→ 288 páginas / ISBN 978-987-1347-82-7

200 Respuestas: Hardware

Esta obra es una guía básica que responde, en forma visual y práctica, a todas las preguntas que necesitamos hacernos para dominar el hardware de la PC. Definiciones, consejos, claves y secretos de los profesionales, explicados de manera clara, sencilla y didáctica.

→ COLECCIÓN: 200 RESPUESTAS
→ 320 páginas / ISBN 978-987-1347-83-4

PROYECTOS CON MICROCONTROLADORES

En esta obra continuamos con los proyectos con microcontroladores que comenzamos a desarrollar en el libro anterior. En esta oportunidad, iniciaremos la construcción de una placa experimental PIC18, utilizaremos los periféricos internos del PIC y estudiaremos las posibilidades de conectividad inalámbrica.

DENTRO DEL LIBRO ENCONTRARÁ

■ Placa experimental para PIC18LF4620 ■ Consideraciones de armado ■ Periféricos externos ■ Contador y PWM ■ Conversores analógico-digitales ■ Tecnologías de displays LCD ■ Proyecto de alarma térmica ■ Conectividad no inalámbrica e inalámbrica ■ Docklight ■ USB ■ Módulos prearmados ■ ZigBee y 802.15.4 ■ Conexión con la PC y configuración

ADEMÁS

ELECTRÓNICA PRÁCTICA
Aprenda a analizar, simular y construir circuitos

MICROCONTROLADORES
Funcionamiento, programación y aplicaciones prácticas

NETWORKING CON MICROCONTROLADORES
Descubra cómo acceder remotamente a sus equipos

SOBRE LA COLECCIÓN: ELECTRÓNICA

■ **Aprendizaje guiado mediante explicaciones claras y concisas** ■ **Proyectos prácticos basados en necesidades reales** ■ **Consejos de los profesionales** ■ **Infografías y procedimientos paso a paso** ■ **Producciones fotográficas profesionales**

MICROCONTROLLERS PROJECTS

In this book, we will continue the study of mic**rocontrolllers that** we started in *Microcontrollers*. This time, we are going to build a PIC18 experimental board, we will learn about PIC internal peripherics, and apply wireless conectivity in several projects.

NIVEL DE USUARIO			
BÁSICO	INTERMEDIO	AVANZADO	EXPERTO
	■	■	

RedUSERS.com

Nuestro sitio reúne a la mayor comunidad de tecnología en América Latina. Aquí podrá comunicarse con lectores, editores y autores, y acceder a noticias, foros y blogs constantemente actualizados. Además, podrá descargar material adicional de los libros y capítulos gratuitos, o conocer nuestras otras publicaciones y acceder a comprarlas desde cualquier parte del mundo.

Si desea más información sobre el libro: Servicio de atención al lector **usershop@redusers.com**

ISBN 978-987-1773-23-7

9 789871 773237 >