CAPÍTULO 4
DISEÑO DE LA INTERFAZ ELECTRÓNICA

En este capítulo se describe el diseño final de la interfaz, el cual se dividió en cuatro partes: el hardware electrónico, que abarca el sistema de control seleccionado y cómo se realizaron todas sus conexiones; el hardware mecánico, que hace referencia al montaje físico de la interfaz; el firmware del sistema de control seleccionado; por último, la comunicación utilizada.

4.1 HARDWARE ELECTRÓNICO

El circuito electrónico está constituido por su sistema de control, con los circuitos necesarios para la conexión de los motores y los sensores; este hardware, cuyo diagrama de bloques se muestra en la Figura 11, se comunica con la PC a través del puerto USB con el programa de aplicación.

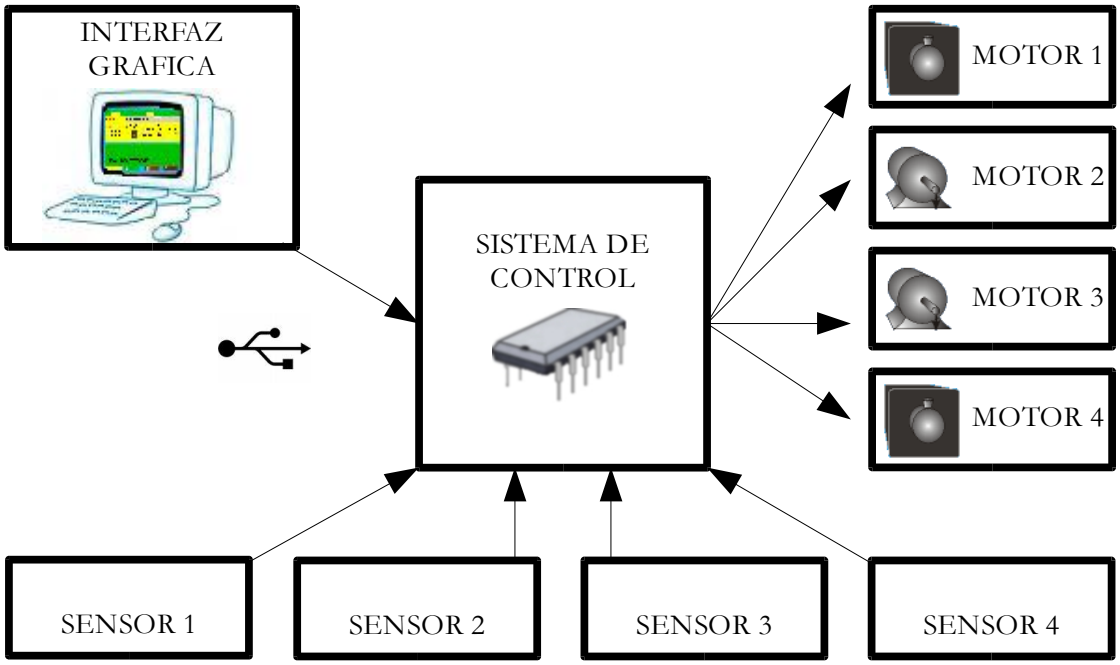

Figura 11 *Diagrama de bloques de la interfaz electrónica.*

De las opciones para el sistema de control consideradas, se decidió trabajar con los microcontroladores, debido a que se disminuye el número de componentes con respecto a las otras opciones analizadas en el capítulo anterior, y con ello la probabilidad de errores en su funcionamiento, además de contar con funcionalidades integradas como son los convertidores analógico digital, y la modulación PWM, que se requieren para el diseño de esta interfaz.

Dentro de la gama de microcontroladores existentes, se eligió uno de la compañía Microchip, debido a que tienen la opción de programarlos con lenguaje ensamblador o con lenguaje de programación C; el tipo de encapsulado es tipo DIP, por la siglas de *Dual In line Package*, por lo tanto se puede trabajar fácilmente con él, y además, porque se cuenta con la experiencia de haber trabajado previamente con estos dispositivos.

El microcontrolador seleccionado fue el PIC18F4550, debido a que tiene la posibilidad de comunicación por puerto USB, que los periféricos con los que cuenta serán de gran utilidad, y además porque su capacidad de memoria de programa es amplia. Este circuito integrado tendrá programadas las rutinas necesarias para el control de cada uno de los motores así como el acceso a la lectura de los sensores; dicho programa constituye el denominado *Firmware* del sistema.

Los PIC18 tienen una arquitectura Harvard con un bus de programa de 16 bits y el bus de datos de 8 bits, con los que se puede acceder a la memoria de programa y a la de datos independientemente, tal como se ilustra en la Figura 12.

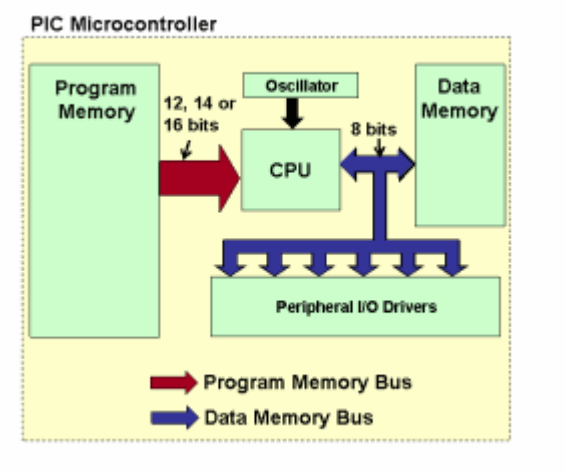

Figura 12 Arquitectura del PIC18F4550.

El PIC18F4550 cuenta con:

- Memoria de programa flash de 32768 bytes
- Memoria RAM de datos 2048 bytes
- Memoria EEPROM de datos 256 bytes
- Pila de 31 palabras de 21 bits
- 20 fuentes de interrupción

- Cuatro temporizadores
- Un módulo CCP (Comparación/Captura/PWM)
- Un módulo ECCP (Comparación/Captura/PWM mejorado)
- Comunicación serial MSSP, o Puerto Serial Síncrono Maestro por las siglas de *Master Synchronous Serial Port*, EUSART, o Transmisor Receptor Síncrono Asíncrono Universal Mejorado por las siglas en inglés de *Enhanced Universal Synchronous Asynchronous Receiver Transmitter.*
- Canal USB
- Puerto paralelo de transmisión de datos
- 13 canales de conversión analógico–digital de 10 bits
- Dos comparadores analógicos.

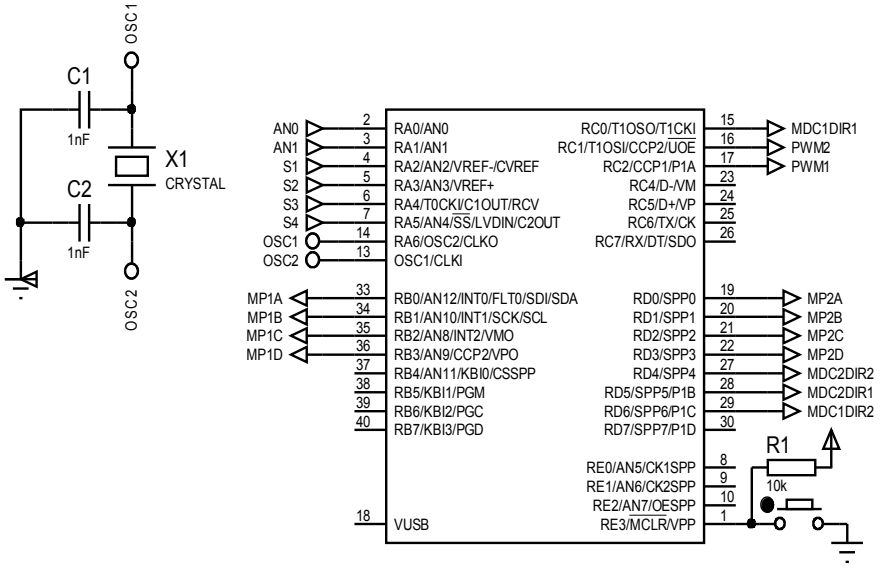

Figura 13 Configuración mínima para la operación del PIC 18F4550.

El PIC18F4550 requiere de un voltaje de polarización de 5 V, se le puede conectar un oscilador externo de 20 Mhz, con el que puede realizar instrucciones simples en 0.1 µs. Requiere la conexión de un interruptor normalmente abierto para reiniciar su operación, y de una conexión para la comunicación vía USB con la PC. En la Figura 13 se pueden observar su configuración mínima.

El microcontrolador proporciona una intensidad de corriente máxima de 25 mA por terminal, y por consiguiente, tal como se mencionó en el capítulo anterior, es necesario proporcionar la intensidad de corriente suficiente para el funcionamiento de los motores. Se eligió el circuito integrado L293D, que proporciona hasta 600 mA por terminal, y cuyo diagrama se muestra en la Figura 14. En la interfaz son necesarios tres integrados; uno para cada motor de pulsos y uno más para los dos motores de CD.

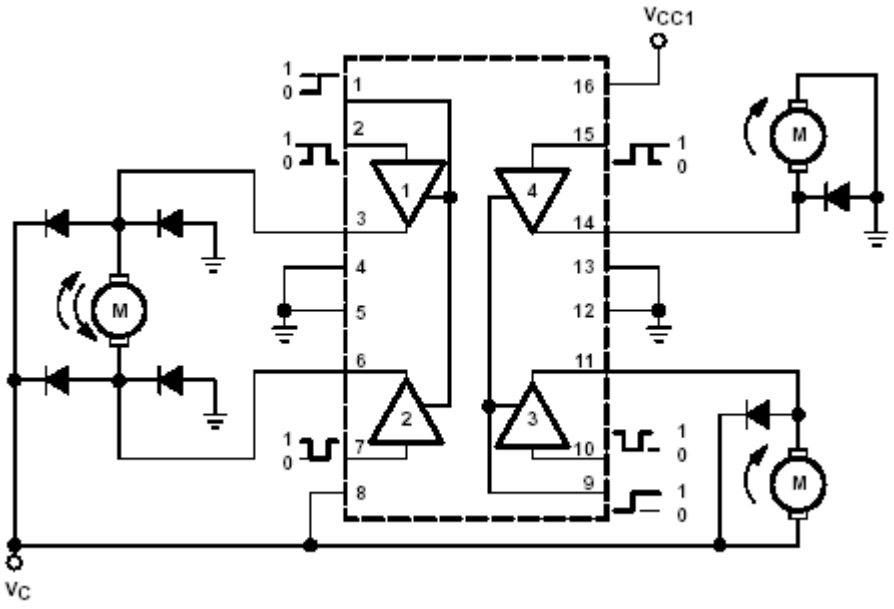

Figura 14 Diagrama del circuito integrado L293D.

En algunos sensores, como los fotorreflectivos, su funcionamiento se modifica dependiendo de la luz ambiental, y por tanto es necesario calibrar su transductor

para asegurar su funcionamiento correcto. Dicho transductor se implementa con un circuito comparador, el cual tiene dos entradas, una de referencia y la otra la señal a comparar, que se conectan de manera que su salida sea de 0 V si la entrada a comparar es menor que la referencia, o 5 V en caso contrario. Dado que el valor del voltaje de referencia debe variar, dependiendo de la cantidad de luz ambiental, esta entrada se conectó a un trimpot de 50 kΩ polarizado con 5 V.

La tensión que suministra la fuente de poder a la interfaz puede ser hasta de 15 V, y por medio del regulador de voltaje fijo LM2940CT–5.0 es posible proporcionar los 5 V necesarios para la polarización del microcontrolador, del comparador y del puente H. La energización de los motores es directamente de dicha fuente, sin necesidad de regulación.

Con respecto a la conexión de los sensores, se instalaron terminales a los que pudieran conectarse cualquiera de los sensores considerados en este trabajo. Por medio de dichas terminales se proporciona el voltaje de alimentación que requieren los sensores para su funcionamiento, y se obtiene su señal, la cual es conectada a una entrada del PIC. Los elementos necesarios para el correcto funcionamiento de los sensores deben de ser conectados externamente a cada uno de ellos.

Para garantizar la recepción de las señales tanto de los sensores como de los actuadores, la longitud de los cables de conexión entre la interfaz y dichos componentes debe de ser tal que no presente pérdidas significativas. Por otra parte, es necesario que también sea lo suficientemente holgada para permitir adaptar los componentes a distintos diseños de robots pedagógicos. La longitud de los cables que finalmente se decidió para cada uno de los diversos componentes fueron de 15 cm para los motores de CD, 15 cm para los de pulsos, y para todos los sensores de 5 cm.

Para aquellos sensores cuya lectura es analógica, se destinaron las terminales A0 y A1 del microcontrolador, que corresponden a dos canales del convertidor analógico–digital.

Descripción de las terminales para los sensores:

- Dos entradas digitales con calibración, en las que se pueden conectar sensores sensibles a la luz y cuyo funcionamiento se ve afectado por la cantidad de luz ambiental, por lo que se requieren calibrar.
- Dos entradas digitales simples, diseñadas para conectar todo tipo de interruptores como pueden ser los de palanca o lámina, los interruptores normalmente abiertos denominados *push botton,* y los conocidos como de tipo cola de rata.
- Dos entradas analógicas, con voltaje de alimentación de 5 V, que es una condición que deben cumplir los sensores que se conecten a ellas.

4.2 HARDWARE MECÁNICO

Considerando que los usuarios de esta interfaz electrónica para el control de robots pedagógicos serán niños, se consideró deseable que ellos pudieran observar cómo es físicamente este dispositivo. Por consiguiente, se decidió proteger toda la circuitería electrónica con un estuche transparente, en el que se puedan visualizar todos los circuitos integrados y el alambrado de la interfaz. Por ser modular y para facilitar su manipulación, se optó por que todos los cables externos al estuche pudieran conectarse y desconectarse con facilidad.

Aunque parece ser de poca importancia, las terminales forman parte importante de este proyecto. Se debían de elegir aquéllas que fueran fáciles de manipular, ya que los niños son quienes realizarían las conexiones, además de que debían ser

robustas, seguras y que su tamaño no fuera muy grande, ya que repercutiría en el tamaño final de la interfaz.

Después de una larga búsqueda de terminales para tarjeta impresa que pudieran emplear los niños con facilidad, se encontraron las que se muestran en la Figura 15, y que consisten en arreglos de conectores con palanca. Para conectar a éste un cable, basta con levantar la palanca, introducir el cable en el orificio que se forma y soltar la palanca que retornará a su posición inicial, prensando al cable mencionado.

Con ese tipo de terminales se pueden conectar y desconectar los cables de los sensores y de los motores, de tal manera que a la hora de trabajar con la interfaz se puedan conectar únicamente los elementos a utilizar, mejorando de esta manera su funcionalidad.

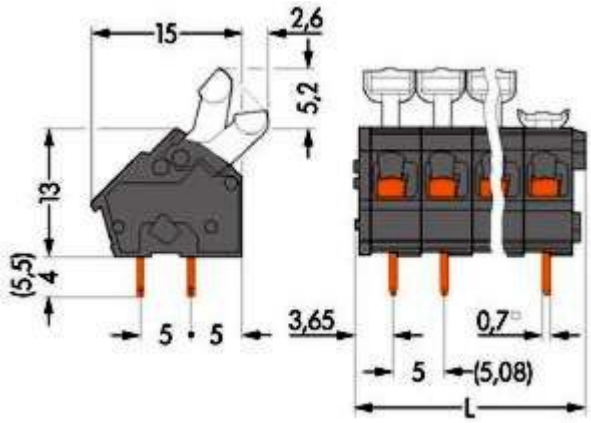

Figura 15 Terminales utilizadas para la conexión de los cables.

Hay otros dos tipos de conectores que se utilizaron: uno tipo USB para la comunicación con la PC, y otro tipo JT para la alimentación externa.

4.3 FIRMWARE DEL MICROCONTROLADOR

En el firmware del PIC, además del programa de control estarán los descriptores del dispositivo, que se requieren para establecer la comunicación con la PC, ya que sin ellos, aunque exista conexión física con la PC, ésta no los reconoce.

La programación del firmware se realizó con el software conocido como *CCS C Compiler*, que es el programa compilador de lenguaje C para los microcontroladores Microchip, que desarrolló la compañía *Custom Computer Services*, el cual es un lenguaje de programación con la estructura de C tradicional, pero con funciones especificas para los dispositivos de la compañía citada.

Puede utilizarse en el PIC un Sistema Operativo en Tiempo Real, o RTOS por las siglas en inglés de *Real Time Operating System*, que le permite trabajar en modo multitarea, sin necesidad de interrupciones [4]. El *CCS C Compiler* cuenta con un planificador de tareas encargado de dar el control del procesador a la tarea que debe ejecutarse en un momento dado; cuando finaliza la tarea o ya no necesita del procesador, el control es devuelto al planificador, el cual dará el control del procesador a la siguiente tarea.

El tiempo dedicado a cada una de las tareas es establecido por el programador, de tal manera que se puede garantizar que todas se realicen, y además permite la comunicación entre ellas. También se puede establecer la frecuencia con la que se ejecuta, así como desactivar o activar la tarea. Todo esto es muy útil para garantizar los tiempos de ejecución de cada una de las partes del programa, sin que haya interferencias entre sí.

En la interfaz diseñada se realizan cinco tareas: cuatro para la operación de cada uno de los motores y una para la detección de datos de entrada provenientes de la PC. La activación o desactivación de cada una de las tareas correspondientes a los motores dependerá de la petición de la PC. La tarea de detección de datos estará

todo el tiempo monitoreando si existe un cambio en el host, es decir, si hay un dato nuevo proveniente de la PC.

Cuando llega un nuevo dato desde la PC, se analiza a qué tarea corresponde ese paquete, con la finalidad de asignar los datos recibidos a las variables del programa que le correspondan y en su caso activar o desactivar la tarea. El dato recibido es un paquete de tres bytes; los cuatro bits menos significativos del primer byte definen la tarea a la cual pertenecen los datos. Si se trata de alguno de los motores: el bit 7 de dicho byte define el sentido de giro, el bit 6 el tipo de duración, el cual puede ser definida o indefinida, configuración que se puede apreciar en la Figura 16; el segundo byte recibido corresponde al valor de la duración en caso de ser ésta definida; el tercer byte define el valor de la velocidad del motor.

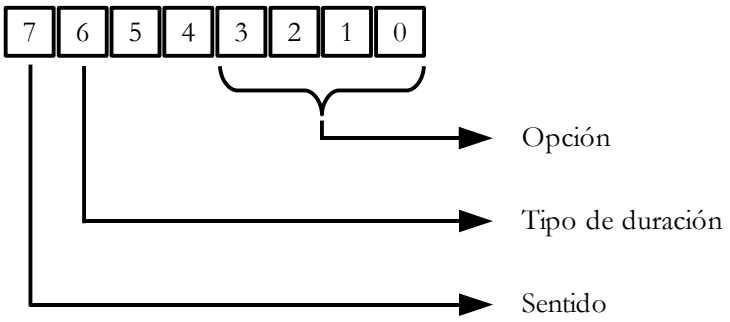

Figura 16 Configuración del primer byte recibido.

Con respecto a los tipos de duración, en la definida el motor funcionará el lapso de tiempo establecido o el número de revoluciones determinado, según sea el tipo de motor, y cuyo valor se define en el segundo byte del paquete recibido; en la indefinida, no se utiliza este byte, ya que el motor funcionará indeterminadamente hasta que reciba la orden de paro.

Para el bit correspondiente al tipo de duración se definió el cero lógico para cuando sea indefinida y uno lógico para la definida. En cuanto al sentido de giro, se

estableció que el cero lógico representa el sentido levógiro y el uno lógico para el dextrógiro.

Si se trabaja con la lectura de sensores, únicamente se utilizan los cuatro bits menos significativos del primer byte correspondientes a la selección de la opción de modo de trabajo o tarea. La tarea realiza dicha lectura y la envía de vuelta a la PC.

En la siguiente tabla se pueden observar los modos en los que trabaja el microcontrolador. Se destinaron cuatro bits para la selección del modo de trabajo con el fin de establecer hasta quince diferentes opciones, en el caso de esta interfaz únicamente son siete, pero se queda abierta la posibilidad de ampliarlo para futuras versiones.

0000 Parar	0100 Sensor Analógico 2
0001 Motor de pasos 1	0101 Sensores digitales
0010 Motor de corriente continua 1	0110 Motor de pasos 2
0011 Sensor Analógico 1	0111 Motor de corriente continua 2

Tabla 4.1 Opciones de los modos de trabajo del microcontrolador.

Como se explicó anteriormente, el motor de pulsos necesita una secuencia de pulsos determinada en sus terminales para poder girar; el periodo de los pulsos determina la velocidad de giro, mientras que el sentido de giro depende del orden de la secuencia de dichos pulsos.

La tarea correspondiente al giro del motor de pulsos, genera dicha secuencia a la velocidad solicitada, y realiza la medición del tiempo para desactivarse, cuando la duración es definida. En la Figura 17 se tiene el diagrama de flujo correspondiente a la tarea del motor de pulsos.

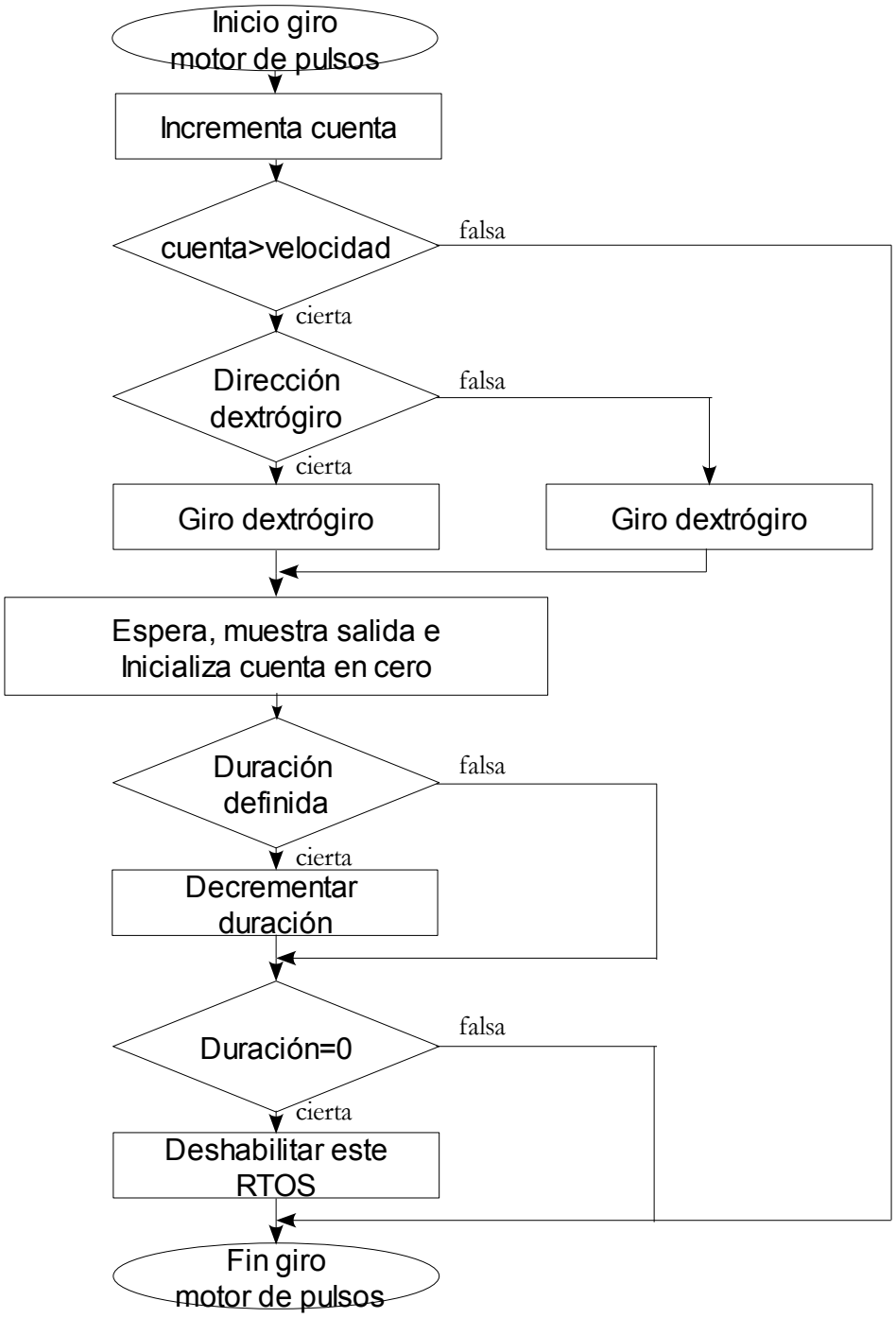

Figura 17 Diagrama de flujo de la operación del motor de pulsos.

Para los motores de CD su funcionamiento es diferente; si se requieren encender, basta con activar la función PWM del microcontrolador, utilizando únicamente el parámetro del ciclo de trabajo para controlar la velocidad. La tarea para la operación de un motor de CD únicamente se activa cuando la duración es definida, ya que se encarga de llevar a cabo la cuenta de dicha duración; una vez llegado el fin de la cuenta, apaga a la función PWM y se desactiva. La duración está definida en segundos. En la Figura 18 se muestra el diagrama de flujo correspondiente.

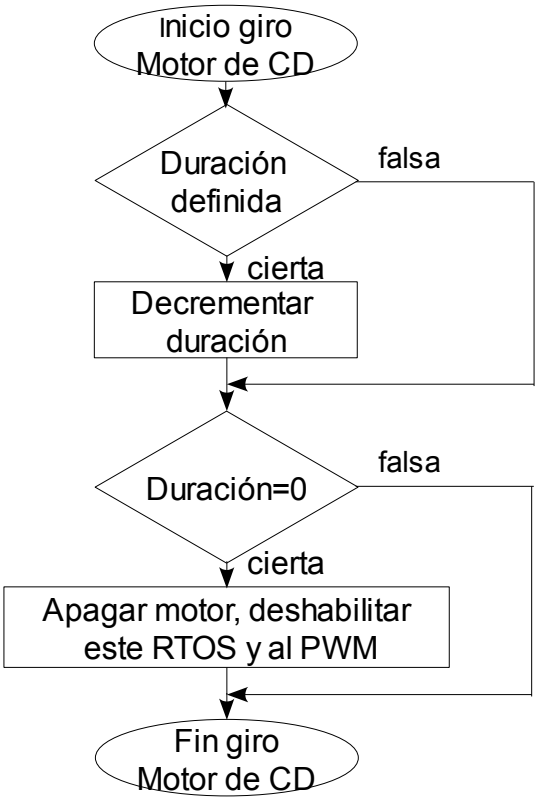

Figura 18 Diagrama de flujo de la operación del motor de CD.

El microcontrolador cuenta con el modulo de CCP (Comparación/Captura/PWM), el cual trabaja con un registro de 10 bits (CCPRH + 2bit) que es comparado con un contador de 10 bits (TMR2 + 2 bits). Cuando el contador llega al valor del registro la salida CCP se pone en cero; por otra parte, se compara el contenido del registro PR2 con el del contador TMR2, y cuando se hacen iguales, la salida CCP se pone en alto, el contador se reinicia y el valor del registro CCPRH se carga con el registro CCPRL. En la Figura 19 se muestra el diagrama de bloques de este proceso.

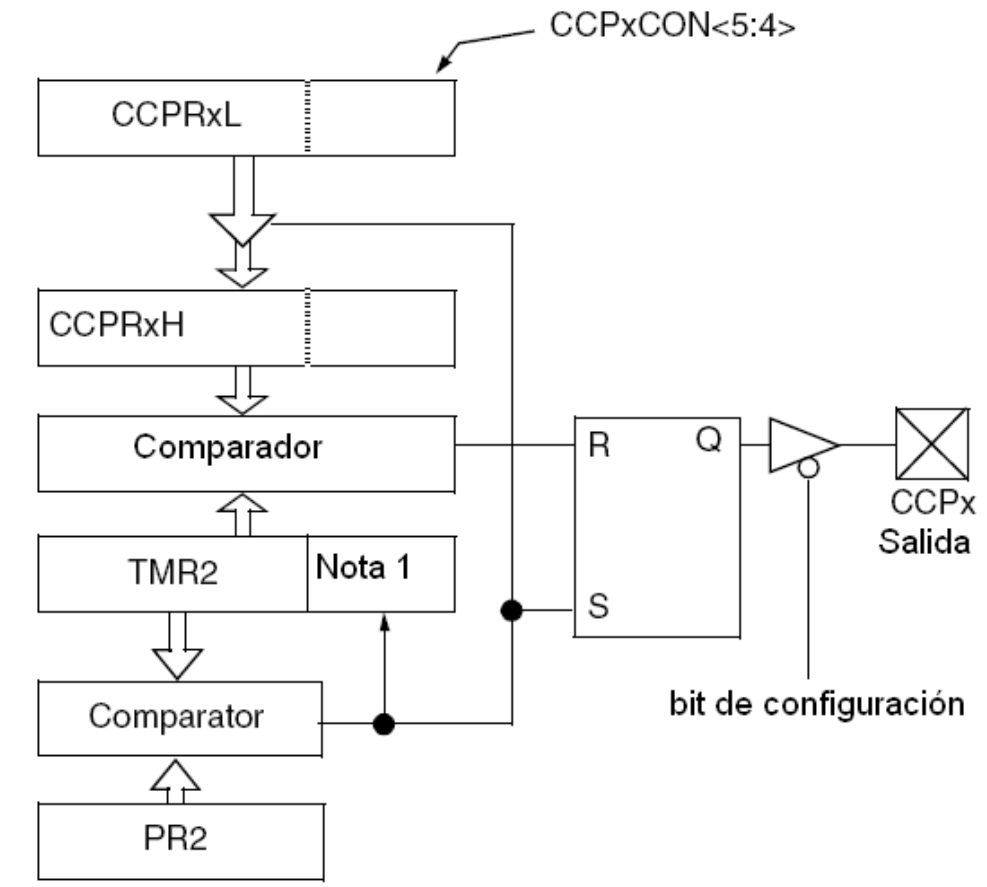

Figura 19 Diagrama de bloques del funcionamiento del módulo PWM

De esta forma el periodo de la señal PWM dependerá del valor que contenga el registro PR2 y el registro CCPRL determinara el ciclo de trabajo de la señal PWM, quedando en función de las siguientes expresiones.

$$T_{PWM} = \frac{(PR2+1)\,x4xPreescaler\,delTMR2}{F_{osc}}$$

$$CT = \frac{(CCPRxL + CCPxCON(5:4))\,xPreescaler\,del\,TMR2}{F_{osc}}$$

El rango de frecuencias para la señal PWM que se puede establecer es entre 1.2 khz y 5 Mhz, se trabajo con la frecuencia de 2.44 khz debido a que con este valor los registros PR2 y CCPRL contienen valores enteros y la resolución de la señal es optima.

En el apéndice se puede observar el diagrama de flujo completo del firmware del PIC18F4550 empleado.

4.4 COMUNICACIÓN USB

Como se mencionó en el Capítulo 2, el dispositivo USB debe contener los descriptores necesarios, lo que se realizó en la programación del PIC. Existen descriptores de configuración, interfaz, endpoint, dispositivo y cadena, que nos proporcionan las características de comunicación del dispositivo. A continuación se explican cada uno de dichos descriptores.

De configuración: provee la información de la alimentación requerida por el dispositivo, así como las características de la configuración. En este caso se estableció que la intensidad de corriente de alimentación máxima fuera de 100 mA, la cual es la que se da por defecto. El número de interfaces que soporta el PIC es de sólo una.

Interfaz: establece el número de endpoints usados por ella misma, y la clase a la que pertenece. Se utilizan dos *endpoints,* uno de entrada de datos y otro de salida, aparte del *endpoint* cero, el cual siempre está presente, porque con él se inicia la comunicación. Debido a que el dispositivo no pertenece a ninguna clase establecida, en su código se indica que el diseñador es el que proveerá el controlador para que Windows reconozca al dispositivo cuando éste se conecte.

Endpoint: identifica el tipo de transferencia, además cada e*npoint* es identificado mediante un número, una dirección y su orientación, lo que permite hacer referencia a cada uno de forma inequívoca. El tipo de transferencia a utilizar es la tipo bulk, la cual garantiza calidad en la entrega. También en este descriptor se establece el tamaño máximo del paquete de entrada y salida, el cuál para este caso se estableció de 64 bytes.

Dispositivo: provee información del fabricante, número de producto, número de serie, la clase de dispositivo y el número de configuraciones. El VID debe ser 04D8h, que identifica al fabricante *Microchip;* el PID será 000Bh [6]. En este descriptor se indica el tamaño máximo del *endpoint0;* gracias a él se puede enumerar al dispositivo, después se configura la comunicación con los demás *endpoints.* El tipo de transferencia que utiliza este *endpoint 0* es el de control, que garantiza tiempo y calidad de entrega, es decir, que tiene prioridad sobre otras transferencias y no hay pérdida de datos

Cadena: provee información que se muestra en la barra de tareas, a la hora de establecer comunicación con el Host y frecuentemente describe al dispositivo. Este descriptor es opcional.

Por parte de la PC se requiere del controlador de Windows para que reconozca al dispositivo. El controlador empleado se tomó de la tesis “Desarrollo de una interfaz

USB para el control remoto de sistemas electromecánicos" [7] , en la que la parte fundamental del trabajo aborda la comunicación vía puerto USB.

El controlador fue desarrollado bajo el modelo *Windows Driver Foundation* (WDF), el cual es orientado a objetos y controlado por eventos. Trabaja en *Kernel Mode Driver Framework* (KMDF), que provee las funciones y macros dedicadas exclusivamente al control y administración de dispositivos, interfaces, endpoints, y las transferencias de todos los tipos. Además, implementa las características fundamentales para los controladores como soporte de Plug and Play, administración de energía, colas de E/S, transferencias tanto síncronas como asíncronas.

Una vez que se tiene el controlador por parte de la PC y los descriptores en el dispositivo, se puede establecer la comunicación USB. Ahora se puede acceder a la comunicación, que se realiza por medio de una biblioteca dinámica, o DLL, por las siglas en inglés de *Dynamic–Link Library*.

La biblioteca utilizada es mpusbapi.dll proporcionada por Microchip; con ella se tiene acceso a siete funciones, del las cuales para esta aplicación sólo se ocupan cuatro: _MPUSBOpen, _MPUSBRead, _MPUSBWrite y _MPUSBClose.

Para enviar y/o recibir datos, se ocuparán tres de estas funciones, como se muestra en la Figura 20.

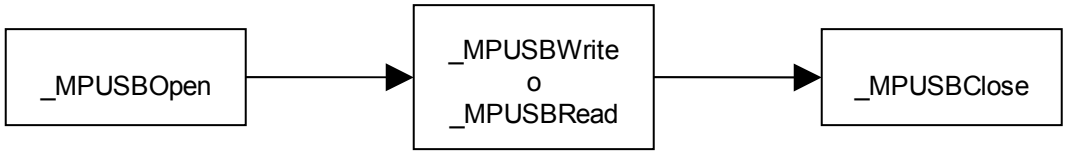

Figura 20 Estructura del envío/recepción de datos.

La función _MPUSBOpen se encarga de definir la ruta por donde se deben de dirigir los datos. Tanto la función _MPUSBRead como_MPUSBWrite, son para leer y escribir, respectivamente, y utilizan los mismos argumentos, pero con algunas variaciones. Por último, la función _MPUSBClose cierra el acceso al endpoint.