Universidad Nacional Autónoma de México

Facultad de Ingeniería

DESAROLLO DE UNA INTERFAZ PARA EL CONTROL DE ROBOTS PEDAGÓGICOS PROGRAMABLES CON LENGUAJE NATURAL

Tesis para obtener el título de

INGENIERA ELÉCTRICA ELECTRÓNICA

Presenta

Adriana Yoloxóchil Jiménez Rodríguez

Director de tesis

Yukihiro Minami Koyama

Ciudad Universitaria, D. F., Agosto 2010

Para mis hijos Sara y Akori:
disfruten el producto de este trabajo.

Quiero expresar mi agradecimiento:

A mis papás. Por su constante apoyo y confianza, sepan que mis ideales, esfuerzos y logros han sido también suyos. Gracias Papá por heredarme el amor a la Ingeniería.

A mi esposo. Por el cariño, comprensión, paciencia y apoyo que siempre he recibido y con el cual he logrado culminar este trabajo.

A mis hijos Sara y Akori. Por ser mi inspiración y eje motor en el desarrollo de este proyecto.

A mis hermanas Citla y Dona. Por estar siempre allí alentándome para poder confiar que todo esto podría ser posible.

A Yukihiro. Por darme la oportunidad de trabajar con el, por compartir su conocimiento, sus consejos, su tiempo y sobre todo por tener mucha paciencia a lo largo de este tiempo y brindarme su amistad.

A Enrique Ruiz-Velasco: Promotor incansable de la robótica pedagógica, ya que de él surgió la idea para esta tesis.

A todos mis profesores que a lo largo de los años compartieron su conocimiento y contribuyeron a mi formación académica.

Y a todas aquellas personas que a lo largo del camino estuvieron presentes y me alegraron el camino para llegar hasta aquí.

ÍNDICE

RESUMEN

Para la comunicación con el mundo exterior, la computadora personal, o PC, actualmente emplea al puerto denominado USB, por las siglas en inglés de *Universal Serial Bus*, en lugar de los puertos serial o paralelo. Por ello, se abordó el diseño de una interfaz con el puerto USB para así poder controlar robots pedagógicos. La robótica pedagógica es un área de la pedagogía de reciente creación, pretende que los niños y los adolescentes aprendan ciencia y tecnología de forma lúdica, por medio del diseño y la construcción de un robot con dos o tres grados de libertad, en el que apliquen conocimientos elementales de matemática, física e informática. Con la interfaz electrónica que se diseño se pueden controlar cuatro motores, dos de corriente directa, o CD, y dos de pulsos, y permite la entrada de dos señales analógicas y cuatro señales digitales. Para la comunicación con la PC se emplea un microcontrolador PIC18F4550 como transmisor–receptor; en él se tiene grabado el firmware que le permite realizar dicha función.

Todo lo leído y escrito por el puerto USB se maneja como archivo, y por tanto se requiere guardarlo en una biblioteca dinámica (DLL, por las siglas de *Dynamic Linking Library*) para poder utilizarla en esta aplicación. Para que el sistema operativo Windows pueda reconocer a la interfaz, se requirió generar su propio controlador.

La aplicación se desarrolló en lenguaje C#, con objeto de que tuviera una presentación gráfica e interactiva con el usuario. Se realizaron dos versiones, una para el control de cada uno de los motores y la visualización del estado de los sensores, y otra en la que muestra la manera en que funcionan las estructuras básicas de programación interactuando con los motores y sensores.

La interfaz que se diseño puede mover los motores de pulsos en cualquier sentido y en múltiplos de cuarto de vuelta, y los motores de CD durante un tiempo con resolución de un segundo con el sentido deseado. Los sensores analógicos se conectan a las terminales del convertidor analógico–digital del microcontrolador, y las entradas digitales a terminales digitales del mismo microcontrolador.

CAPÍTULO 1
INTRODUCCIÓN

Sin duda alguna el diseñar y construir un robot trae consigo una serie de beneficios, ya que en el proceso se estimula la imaginación y la creatividad, se promueve el trabajo en equipo, se enfrenta al planteamiento y resolución de problemas utilizando principios de física, matemática e informática. Y al existir un interés natural de los niños por la tecnología, la robótica pedagógica abre un camino atractivo y gratificante hacia ella, así como para el aprendizaje de las ciencias.

Con el afán de proporcionar herramientas para esta nueva disciplina, se optó por el desarrollo de una interfaz que sea capaz de permitir a los niños controlar un robot por medio de la computadora. Para el diseño de dicha interfaz se debe considerar que ésta sea robusta, de fácil manipulación, que su manejo sea fácil de aprender y que sea atractiva para los niños. A lo largo del diseño, análisis y toma de

decisiones, no se debe perder de vista hacia quién va dirigido y que se trata de una herramienta para controlar dispositivos básicos que se usan en la robótica.

El diseño se divide en tres partes fundamentales: el hardware, en el cual están conectados los elementos de entrada y salida del robot; el firmware, que establece la comunicación entre los circuitos electrónicos y la computadora; el software, que incluye la aplicación de programación y control del robot.

Debido a que un objetivo de la robótica pedagógica es que el niño diseñe los mecanismos del robot, el hardware debe ser adaptable a diferentes diseños, con posibilidad de diferentes tipos y cantidades de entradas y salidas de control.

La comunicación debe ser en tiempo real, y la interfaz debe poder conectarse a cualquier computadora, por lo cual desde el planteamiento del proyecto se estableció que dicha conexión se realice a través del puerto USB.

El programa de aplicación de la PC debe tener un ambiente interactivo, gráfico, atractivo y de fácil manejo. Se requiere que se consideren los principios de la programación como los conceptos de paralelismo, recursividad, concurrencia[1], secuencia, y que los aborde con un lenguaje natural para el niño. En este trabajo no se considerará la resolución de este último problema, aunque se propondrá una alternativa. Todo esto debido a que esta tesis abarca la primera etapa del proyecto, que esta enfocada a la parte electrónica.

Con base en todo lo anteriormente mencionado, se estableció el objetivo de esta tesis, que es diseñar una interfaz electrónica para controlar robots pedagógicos que se conecte a una PC a través del puerto USB, y construir un prototipo que sea robusto, funcional y atractivo para los niños. Asimismo, se

1 Paralelismo: función que permite realizar varias tareas al mismo tiempo. Recursividad: capacidad que tiene un programa de llamarse a sí mismo. Concurrencia: Virtud que se tiene para que varias acciones se ejecuten simultáneamente.

incluye la creación de un programa gráfico de aplicación, para verificar el funcionamiento de dicha interfaz.

En el siguiente capítulo se analizan las partes del robot móvil: sensores, actuadores, sistema de control, así como los dispositivos de comunicación. Todo esto para tomarlo como guía en la elección de los elementos para el diseño de la interfaz electrónica a implementarse.

En cuanto al tercer capítulo, se presentan las posibles soluciones para el sistema de control, así como para las etapas de sensores y de potencia de la interfaz. Partiendo de las necesidades pedagógicas así como de las características propias de los lenguajes de programación, en este capítulo también se establece el modelo para la aplicación gráfica de la PC.

La descripción detallada de cada una de las partes de la interfaz electrónica se encuentra en el capítulo cuatro. En el apartado hardware se describe el sistema de control seleccionado y sus características; en la parte de firmware se aborda la programación del microcontrolador, incluyendo su comunicación por medio del puerto USB con la PC.

En lo que se refiere al programa de aplicación e interfaz gráfica, en el capítulo cinco se describen las dos variantes que se realizaron.

En el capítulo seis se exponen los resultados obtenidos y las modificaciones hechas para llegar a la versión final del prototipo, así como las conclusiones y las recomendaciones para mejorar la interfaz en futuras versiones.

Finalmente, en el apéndice se puede observar el diagrama de conexiones de la interfaz, el programa completo del firmware del microprocesador, y el programa en C# correspondiente a la interfaz gráfica.

CAPÍTULO 2
ELEMENTOS PARA EL DISEÑO DE LA INTERFAZ ELECTRÓNICA

2.1 ELEMENTOS DE UN ROBOT

Existen varias definiciones de lo que es un robot. Para este trabajo, la que mejor se adapta es la siguiente: "*Un robot es un dispositivo mecánico, equipado con actuadores y sensores bajo el control de un sistema computacional, el cual opera en un espacio de trabajo dentro del mundo real. El robot realiza tareas por medio de movimientos controlados en tal espacio de trabajo*" [1].

En lo que la mayoría de definiciones coinciden, es en los elementos que conforman a los robots: el sistema mecánico, los actuadores, los sensores y el sistema de control.

2.1.1 Sistema mecánico

El sistema mecánico se considera la estructura sobre la cual estarán colocados los demás elementos del robot, por lo que se requiere, por una parte, que proporcione rigidez para evitar inestabilidad y deformaciones, y por otra, que permita el libre movimiento cuando así se requiera; tal es el caso en las articulaciones.

Para el diseño del sistema mecánico se debe de considerar la selección de materiales; en general, se recomienda que sean ligeros y que posean propiedades mecánicas que minimicen su deformación.

2.1.2 Actuadores

Un robot utiliza dispositivos que generan fuerza, los cuales le sirven para la locomoción o la manipulación. La locomoción se refiere al desplazamiento de todo el robot; la manipulación es la acción que realiza el robot para interactuar sobre objetos del medio ambiente.

Tanto para la locomoción como para la manipulación, se necesita el empleo de *actuadores;* los hay neumáticos, hidráulicos y eléctricos; estos últimos son los que se emplean frecuentemente en los robots móviles.

2.1.3 Sensores

Los robots pueden tener dos clases de sensores, los internos y los externos. Los internos son aquéllos que le indican al robot en qué estado se encuentra, y son esenciales si se requiere un control de precisión del robot. Para obtener datos del entorno, se utilizan los sensores externos, y depende del diseñador elegir el tipo de sensores de acuerdo a la tarea que el robot realizará.

Existe una gran variedad de sensores, según la variable física que se requiera medir; los hay de posición, distancia, velocidad, aceleración, temperatura, presión, fuerza, flexibilidad en los materiales, luz, sonido, humedad, entre otros.

2.1.4 Sistema de control

El sistema de control es el que se encarga de regular los movimientos del robot, así como todo tipo de acciones como los cálculos y los procesos de la comunicación.

El sistema de control se puede diseñar con circuitos integrados sencillos o con dispositivos muy sofisticados. La selección depende de la complejidad del control, del número de elementos en la interfaz, las variables a controlar o el tipo de sensores, entre otras consideraciones.

2.2 ROBOTS PEDAGÓGICOS

Con base en la definición general de robot y sus partes constitutivas, es posible introducirse al caso particular del robot pedagógico.

“Un robot pedagógico es un dispositivo tecnológico construido exprofeso, que cumple con los principios de la robótica pedagógica y que tiene propiedades educativas que enriquecen los entornos de aprendizaje, favoreciendo los procesos cognoscitivos de los estudiantes de distintas edades y niveles educativos, durante su iniciación en el estudio de la ciencia y la tecnología” [2].

Un robot pedagógico consta de cuatro sistemas: mecánico, eléctrico, electrónico e informático.

El sistema mecánico es el soporte físico del robot, sobre el que van montados sus demás elementos. En su diseño se deben considerar las articulaciones que tendrá así como la colocación de motores y los sensores, de tal manera que permita su

libre movimiento y a la vez que sea robusto. En la robótica pedagógica esta parte es sumamente importante, ya que es en la que el niño participa de manera directa en el diseño y la construcción del robot. Los materiales ocupados para la elaboración del sistema mecánico pueden ser muy variados, desde materiales de recuperación como cajas, envases, envolturas, hasta estructuras mecánicas diseñadas para construcción y modelado de juguetes como Mecanos, piezas de ensamble tipo Lego o de Fischertechnik, entre otras marcas existentes en el mercado.

El sistema eléctrico al que hace referencia la robótica pedagógica lo constituyen motores, y en algunos casos también a sus controladores.

Con respecto al sistema electrónico de la robótica pedagógica, se consideran los sensores, los convertidores y las interfaces y sistemas de control externos a la PC.

La comunicación entre el niño y el robot es fundamental para el control de este último por parte del primero; se ha utilizado para ello el puerto paralelo, en la mayoría de los casos, así como diferentes lenguajes de programación, algunos muy específicos para niños como es el caso de LogoObjeto, el cual brinda características importantes desde el punto de vista didáctico-pedagógico, y otros más avanzados como Basic y C. Lo anterior corresponde a la parte del sistema informático de la robótica pedagógica.

2.3 MOTORES UTILIZADOS EN LA ROBÓTICA MÓVIL

De los motores eléctricos, los más utilizados en la robótica son el motor de CD, el motor de pulsos y el servomotor.

2.3.1 Motor de CD

El funcionamiento del motor de CD se basa en el principio de la ley de Lorentz: por un conductor por el que circula una intensidad de corriente y que está colocado dentro de un campo magnético, el conductor sufre una fuerza perpendicular al plano formado por el campo magnético y la intensidad de corriente.

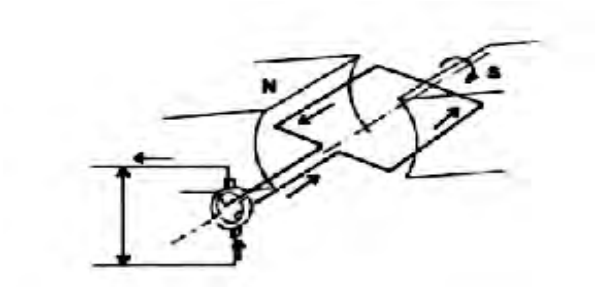


Figura 1 Principio de funcionamiento del motor de CD.

En la Figura 1 se muestra una espira de alambre inmersa en el campo magnético producido por un imán permanente fijo, y a la que se le aplica una intensidad de corriente lo que causa su giro.

En el motor de CD se aprovecha y multiplica este fenómeno, ya que son varios los alambres conductores que conforman al rotor, propiciando su giro continuo.

A este tipo de motores se les puede controlar la velocidad y el sentido de giro, en tanto que la posición angular de su eje de rotación no se puede controlar, al menos directamente.

La velocidad del motor depende de la tensión aplicada a sus terminales: para que comience a girar se requiere una tensión mínima, y al incrementarla, la velocidad también aumentará, siempre y cuando no exceda la tensión permitida. Existe otro modo de controlar la velocidad que es con la modulación por ancho de pulso, o PWM, por las siglas en inglés de *Pulse Width Modulation*; se envía a las terminales

del motor un tren de pulsos de frecuencia fija (del orden de 1 khz), cuyo ciclo de trabajo se controla, con objeto de producir un efecto similar a la variación del voltaje.

En cuanto al sentido de giro del motor, éste depende de la polaridad del voltaje que se aplica a sus terminales, por lo que para cambiar dicho sentido basta con intercambiar las terminales del motor, o bien, cambiar la polaridad de la alimentación. Sin duda la segunda opción es más fácil de realizar, y la manera usual de efectuarlo es con un arreglo de transistores conocido como puente H.

Existen varias maneras indirectas de conocer la posición de la flecha del motor: con potenciómetros acoplados al eje, detectores magnéticos, capacitivos u ópticos, entre otros. Los ópticos son de los más utilizados y consisten en un emisor y un detector de luz, entre los que gira un disco acoplado al eje del motor. El disco tiene zonas opacas y transparentes por las que bloquean o dejan pasar el haz de luz del emisor al detector, provocando pulsos eléctricos. Entre más de dichas zonas tenga, mayor es la resolución de posición angular que se obtiene. Los datos capturados por medio de los pulsos eléctricos sirven para conocer el sentido y el ángulo girado por el rotor; si se requiere que el sistema de control utilice esta información, se debe establecer su retroalimentación, obteniendo así un control en lazo cerrado.

2.3.2 Motor de pulsos

Un motor a pulsos consta de dos partes principales: el rotor y el estator (Figura 2). El rotor es la parte central del motor, conformada generalmente por un imán permanente, dentado, y que gira debido a que el estator tiene bobinas que cuando se excitan adecuadamente, generan un campo electromagnético que produce el movimiento del rotor en alguna dirección.

Lo anterior implica que para que el motor dé un paso, basta excitar a una combinación determinada de bobinas, enviando a cada una de ellas un pulso eléctrico [3].

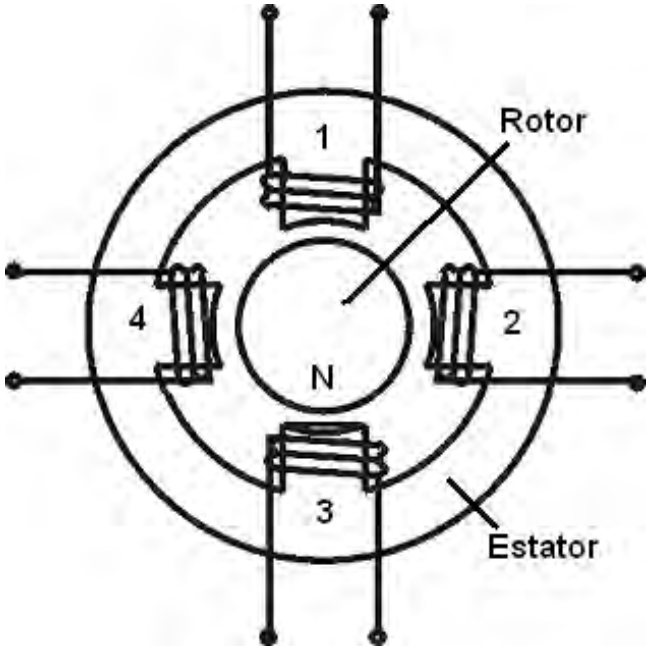


Figura 2 Motor de pulsos.

Cada pulso provoca la rotación del rotor del motor en un incremento de ángulo preciso, denominado paso. Debido a ello se puede controlar en lazo abierto, siempre y cuando la carga no afecte su giro, ya que en todo momento se puede conocer su posición angular con base en el número de pulsos enviados al motor.

Los parámetros que se deben de considerar para el diseño de circuitos de control del motor de pulsos son: par dinámico de trabajo, ángulo de paso, el número de pasos por vuelta, frecuencia de pulsos máximo, par de mantenimiento, debido a que varían dependiendo del fabricante.

2.3.3 Servomotores de modelismo

El servomotor de modelismo es un motor de CD con reducción de velocidad y control incluido, el cual permite posicionar la flecha del motor en un ángulo no mayor a 180°, en la mayoría de los casos [3]. Además de las dos terminales del motor de CD, tiene una tercera terminal por la que se introduce una señal de PWM con la que se determina la posición de la flecha del motor, que depende del ancho de pulso de dicha señal. La relación del ancho de pulso con la posición de la flecha depende del fabricante. Dado que el circuito de control del motor está incluido en el servomotor, su costo es mayor que el de un motor de CD.

2.4 CARACTERÍSTICAS Y TIPOS DE SENSORES

Debido a que la interfaz, objeto de este trabajo, debe ser flexible para poder adaptarse a varias aplicaciones, es necesario analizar cuáles sensores tienen características de conexión similares, con objeto de que a una misma entrada se pueda conectar cualquiera de ellos.

El criterio para la selección de los sensores es considerar aquéllos que son más usados en los robots pedagógicos, que sean sencillos de utilizar, que sean para uso rudo, y conviene no perder de vista uno de los objetivos comunes a cualquier diseño que es la economía.

En primera instancia, se descartaron los que son sofisticados o nuevos en el mercado, o su costo es elevado, y después, aquéllos con aplicaciones muy específicas.

A continuación se presenta un resumen de los sensores que son más utilizados en la mini robótica, y los cuales se pueden tomar como referencia para diseñar la interfaz.

Los sensores de luz son muy empleados. El LDR (*Light Dependent Resistor*) es un resistor que varía su resistencia eléctrica en función de la luz que incide sobre su superficie: mientras mayor sea la intensidad de luz que incida en la superficie del LDR, menor será su resistencia, y a menor intensidad de luz incidente mayor será dicho valor.

Los módulos integrados de sensores fotorreflectivos y de ranura son frecuentemente utilizados en los robots móviles, y su principio de funcionamiento es muy similar: ambos tienen una fuente luminosa (led infrarrojo) y un receptor (fototransistor), y los cuales se muestran en la Figura 3.

En el sensor fotorreflectivo el led infrarrojo y el fototransistor se encuentran en el mismo plano, y para que este último se active con la luz emitida por el led infrarrojo, debe colocarse una superficie que refleje dicha luz. Este tipo de sensor puede servir para detectar objetos; la distancia de detección suele variar dependiendo del dispositivo: el CNY70 (de Telefunken) funciona para distancias de 0 a 5 mm aproximadamente; el GP2D02 (Sharp) es capaz de detectar objetos a 0.80 m de distancia.

Figura 3 Sensor fotorreflectivo y fotointerruptor.

En el caso del sensor de ranura, llamado también fotointerruptor, los elementos se encuentran alineados de tal manera que el haz del led infrarrojo siempre está activando al fototransistor, a menos que un objeto se interponga entre ellos.

Un elemento que puede servir para determinar la posición angular de un mecanismo es el potenciómetro rotacional, el cual se trata de un resistor variable cuyo valor depende de la posición en que se encuentre su cursor.

Un dispositivo que permite interactuar con el exterior es el interruptor, que también se puede considerar como sensor; existe una amplia gama de ellos y uno de los más utilizados es el interruptor de palanca, que se trata de un conmutador de dos posiciones con muelle de retorno a la posición de reposo y con una palanca de accionamiento más o menos larga, según el modelo elegido. Es empleado como sensor de detección de obstáculos, ya que a la hora de chocar con el obstáculo es cuando se activa, aunque este método es algo primitivo, considerando que existen sensores capaces de detectar objetos sin tener contacto físico con ellos.

Existen interruptores de efecto Hall, que aprovechan las propiedades magnéticas de los materiales y se activan con la presencia de un campo magnético. Si no hay campo magnético aplicado, la tensión en la salida es de la mitad del voltaje de alimentación; si se acerca al sensor el polo sur de un elemento magnetizado, la tensión de salida aumenta, en cambio, si se acerca el polo norte, el voltaje de salida disminuye.

Otra aplicación de los sensores de efecto Hall consiste en la detección de materiales ferromagnéticos, utilizándose como sensor de proximidad. Este tipo de sensores suelen constar de un elemento conductor o semiconductor y un imán.

Entre los sensores de temperatura, uno de los más comerciales es el circuito integrado LM35, que además de económico cuenta con una resolución de 1ºC y un rango de –55ºC a +150ºC, características aceptables para aplicaciones simples. La temperatura medida es linealmente proporcional a la tensión eléctrica en sus terminales de salida y con un factor de escala de 10mV/ºC.

2.5 CONVERTIDOR ANALÓGICO–DIGITAL

Algunos de los sensores obtienen lecturas analógicas y otros digitales. Se desean procesar señales analógicas con una PC, usualmente se requiere efectuar una conversión analógica–digital, para lo cual existen diferentes circuitos integrados, algunos dedicados únicamente a esta tarea, y otros con más funciones. Una de las características importantes que se requiere establecer para la conversión analógica–digital es el número de bits de resolución que se requiera obtener para el procesamiento posterior de los datos con la PC.

2.6 ALTERNATIVAS PARA EL SISTEMA DE CONTROL

El control puede ser tan sencillo o tan complejo como se desee. Puede realizarse con algún circuito lógico programable, o con algún microcontrolador.

2.6.1 Circuitos lógicos programables

Los dispositivos lógicos programables o PLD, por las siglas en inglés de *Programmable Logic Device*, favorecen la integración de varios diseños lógicos en un solo circuito integrado. Los PLD no tienen una función establecida de origen, sino que el usuario debe programarla mediante software especializado.

En la actualidad se pueden encontrar en el mercado diversos circuitos PLD, que varían en sus características, su arquitectura y su escala de integración.

Los PLA (*Programmable Logic Array*), los PAL (*Programmable Array Logic*) y los GAL (*Generic Array Logic*), están formados por arreglos o matrices de compuertas y pueden reemplazar a circuitos digitales de pequeña y mediana escala de integración.

Todos los PLDs integran en sus circuitos dos arreglos denominados AND y OR, con los cuales es posible el diseño de circuitos combinacionales y secuencias sencillos.

El PLA está formado por un arreglo AND y otro OR ambos programables. En la figura 4 se puede observar un esquema con tres entradas y dos salidas, cuatro compuertas AND cuyas entradas se pueden configurar y dos compuertas OR también con entradas configurables.

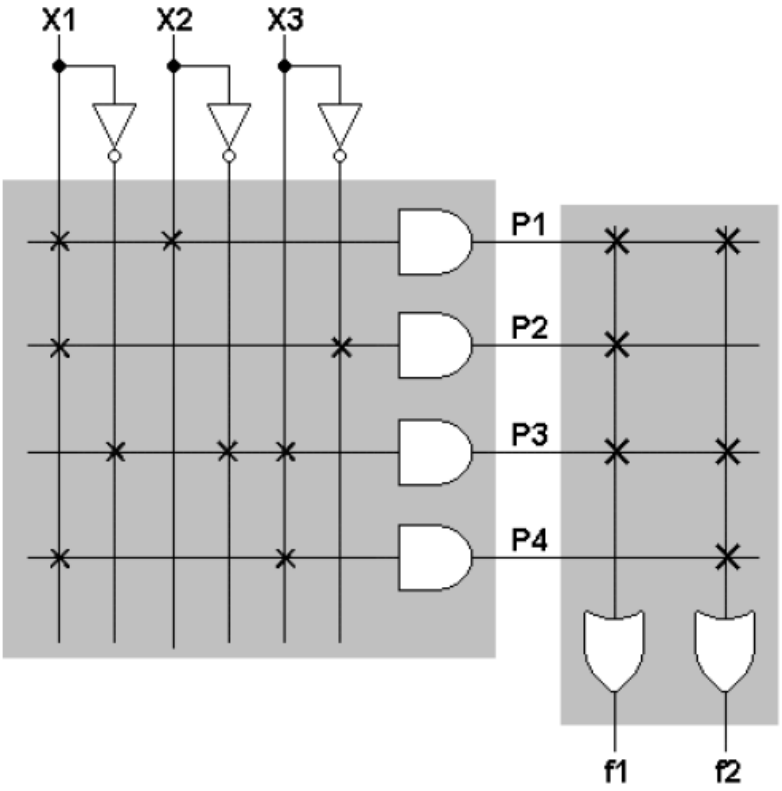


Figura 4 Compuertas AND Y OR con entradas configurables [8].

En cuanto al PAL y al GAL, el arreglo AND es programable y el OR es fijo, tal como se muestra en la Figura 5. Aunque el GAL y el PAL son similares, aquél es reprogramable debido al uso de la tecnología EECMOS, por las siglas en inglés de *Electrically Eraseable Complementary Metal Oxide Semiconductor*.

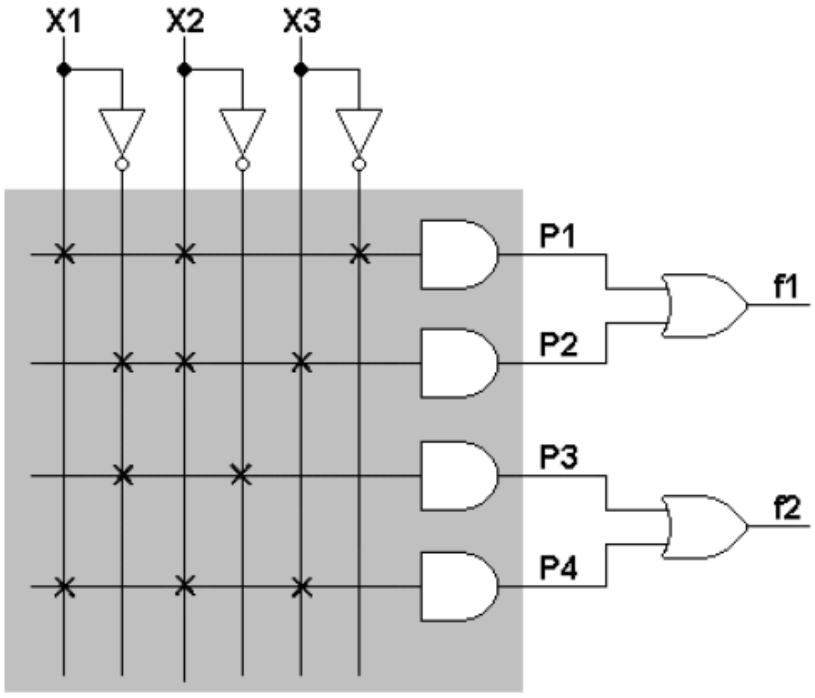


Figura 5 Compuertas AND configurable Y OR fija [8].

El CPLD, o dispositivo lógico programable complejo, por las siglas en inglés de *Complex Programmable Logic Device*, consiste en un arreglo de múltiples PLD agrupados en un solo circuito integrado. Se encuentra estructurado mediante bloques lógicos configurables, y tiene aplicaciones de mayor magnitud con respecto a las del GAL, ya que es equivalente a decenas de miles de compuertas lógicas.

El FPGA, por las siglas en inglés de *Field Programmable Gate Array,* o Arreglo de Compuertas Programables en Campo, es decir, que se programan en la aplicación y no de fábrica, contiene tres tipos de elementos programables: bloques de lógica configurable (CLB), bloque de entrada/salida (IOB) e Interconexiones programables. Los bloques de lógica constan de una sección lógica y registros de almacenamiento, pueden trabajar con multiplexores o con tablas que se almacenan en memoria, y en los que se puede implementar cualquier función booliana. En la

Figura 6 se observa la distribución que tienen estos dispositivos de la compañía Xilinx; para el caso de dispositivos de otras compañías, la distribución cambia aunque no sus elementos.

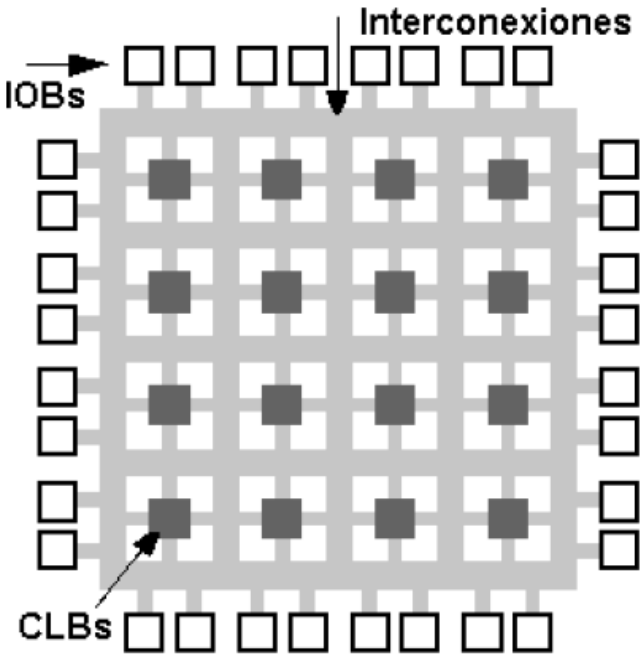


Figura 6 Diagrama de un FPGA [8].

La amplia posibilidad de interconexiones y los miles de bloques configurables, le confiere al FPGA una gran flexibilidad y complejidad en los diseños. Este circuito puede sustituir a cientos de miles, o hasta millones, de compuertas lógicas.

2.6.2 Microcontroladores

Un microcontrolador es un circuito integrado programable y con una estructura preestablecida, que cuenta con una unidad de control con sus diferentes elementos: registros de datos, contador de programa, decodificador de instrucciones y registro de instrucciones. Este dispositivo ejecuta un programa en código máquina propio de cada procesador, que se haya guardado en su memoria.

Además, puede contener componentes extra, llamadas también unidades funcionales, como un convertidor analógico–digital, comparadores, módulos de comunicación, moduladores PWM, que son de gran ayuda para realizar una tarea determinada. La utilización de un microcontrolador en un diseño reduce notablemente el número de componentes, y por tanto el número de fallas, así como el tamaño físico del circuito que se desea diseñar.

2.7 CARACTERÍTICAS DE LA COMUNICACIÓN POR MEDIO DEL USB

Además del diseño de la interfaz electrónica para el control de robots pedagógicos, se requiere que el usuario tenga una interfaz visual en la PC, por medio de la cual pueda enviar y recibir datos tales como parámetros de control y señales de los sensores, por lo que se debe de considerar la comunicación con la PC como una parte importante del sistema de control.

La comunicación entre el robot y la computadora puede ser alámbrica o inalámbrica, pero es recomendable que exista un vínculo físico entre ambos por razones pedagógicas, ya que esto permite al niño comprender la relación acción–objeto, y que no lo vea como algo mágico. En etapas posteriores de aprendizaje, sería muy útil el hecho que fuera inalámbrico, ya que tiene ventajas con respecto al alámbrico, sobre todo de movilidad.

Para la comunicación alámbrica la alternativa viable es por medio del puerto USB, ya que en la actualidad todas las computadoras cuentan con más de uno, y que los puertos paralelo y serial tienden a desaparecer.

Se describirán brevemente las características de la comunicación por USB, así como su funcionamiento, las cuales se tomarán en consideración para el diseño de la interfaz.

Características del puerto USB

- Permite a los dispositivos conectarse y desconectarse sin necesidad de reiniciar el equipo (Plug&Play).
- La velocidad puede ser 1.5 Mbps (*low speed*), 12 Mbps (*full speed*), ó 480 Mbps (*high speed*).
- Puede suministrar hasta 5 V con una intensidad de corriente de 500 mA a los dispositivos que se le conecten.
- Consiste en un par de cables (D+ y D–) por los que se transmiten los datos, además de los cables de alimentación de 5 V y "tierra", que garantizan una transmisión con cables cuya longitud sea de hasta de 5 metros.
- Para que Windows reconozca y pueda establecer la comunicación con el dispositivo USB, se requiere un programa controlador, conocido como *drive*r, desarrollado exprofeso para dicho dispositivo. Existen dispositivos con características comunes que utilizan la misma forma de comunicarse con el entorno, a la agrupación de estos dispositivos se le denomina clase. Si un dispositivo USB es parte de alguna clase predefinida, el sistema operativo tendrá listo un controlador para la interfaz, y no será necesario desarrollar ningún controlador para este dispositivo. Entre las clases más utilizadas están: HID (*Human Interfaz Device*) utilizada en mouses, teclados, joystick, por mencionar algunos; MSD (*Mass Storage Device*) utilizadas en el almacenamiento de la información; CDC (*Comunication Device Class*) utilizada en dispositivos empleados para establecer comunicación como modems.
- EL sistema USB permite una estructura de conexión tipo árbol, donde existe un solo *Host*. El *Host* es un sistema completo que incluye tanto software como hardware, encargado de d*etectar la inserción o desconexión de*

dispositivos USB, gestionar el flujo de control y de datos entre él y los dispositivos, coleccionar estadísticas de actividad y estado, y proveer una cantidad limitada de energía a los dispositivos conectados [4]. Al *Host* se le pueden conectar uno o más *Hubs*, los cuales son utilizados para conectar más dispositivos y así tener la posibilidad de crecer la estructura, como la que se muestra en la Figura 7. Se pueden conectar hasta un total de 127 dispositivos USB, cada uno con las mismas prestaciones,

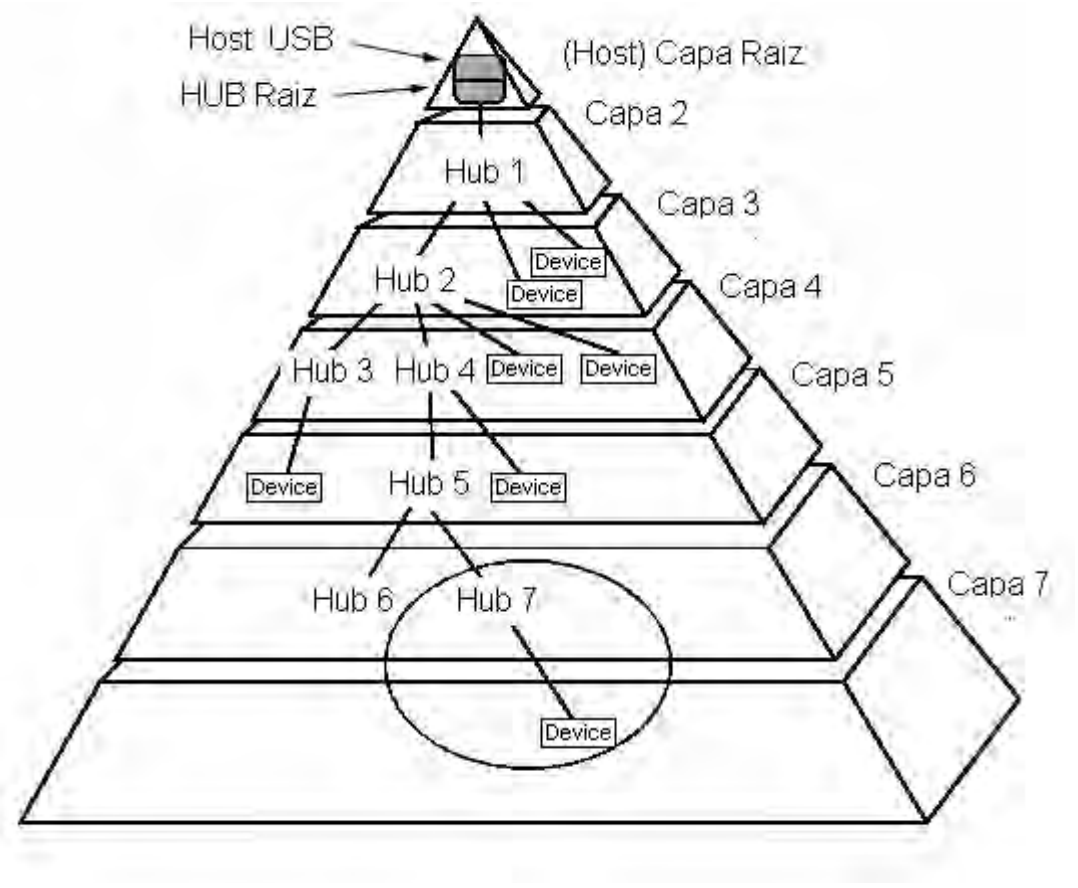


Figura 7 Estructura del USB.

- El USB es un bus punto a punto, con inicio en el Host y fin en un dispositivo o *Hub*. El dispositivo no puede iniciar ninguna transmisión si no es requerida por el *Host.*

- Los dispositivos USB, deben contener los descriptores que permitan al Host identificarlos. La información contenida en dichos descriptores determina el tipo de transmisión que se permite, la máxima velocidad de transmisión, el número de terminales (*endpoints*), entre otros datos. Un e*ndpoint* es un espacio de memoria dentro del dispositivo donde se almacena información [4]. Un e*ndpoint IN* es un generador de datos, mientras que un e*ndpoint OUT* es un consumidor de datos. Un dispositivo común puede requerir varios *endpoints* para poder crear un esquema eficiente de transferencia de datos.
- Cuando se conecta un dispositivo al *Host* se inicia una enumeración, en la cual se interroga al dispositivo sobre sus características principales a través de los descriptores del dispositivo, después le asigna una dirección, carga el controlador y permite la transmisión de datos. Este proceso de enumeración es una actividad continua y es desarrollada por el programa controlador en el Host.
- Una información fundamental es el *Vendor IDentificator* (VID) que consiste en un número para identificar al fabricante, y el *Product IDdentificator* (PID) número para identificar al tipo de dispositivo, los cuales son necesarios para que el controlador del sistema operativo lo reconozca.

CAPÍTULO 3
ANÁLISIS DE LAS POSIBLES SOLUCIONES

Dado que en este proyecto se pretende crear una herramienta para la elaboración de robots pedagógicos, es necesario acotar el trabajo a realizarse. Tal como se mencionó en el Capítulo 1, el sistema mecánico del robot dependerá del diseñador, así como el resultado final de la aplicación que se realice, y por consiguiente, la tarea que pretende realizarse es el diseño de la interfaz electrónica, y el desarrollo de la comunicación que requiere dicho robot con la PC para su control.

Este proyecto consta de tres etapas (hardware, firmware, software); para poder diseñar el prototipo se deben de tomar en cuenta las opciones que se tienen en la implementación de ellas. En este capítulo se plantean las consideraciones y posibilidades en cada una de estas tres etapas de conformación del prototipo.

3.1 INTERFAZ ELECTRÓNICA

3.1.1 Motores

Para implementar la interfaz electrónica se debe de establecer el tipo y la cantidad de cada uno de los elementos que se pretende utilizar. Con respecto a los motores, su número debe ser tal que permita diseños variados, pero evitando, en la medida de lo posible, la dificultad por el número de conexiones.

Se considera que con cuatro motores es posible crear diseños de robots con varias aplicaciones interesantes. Cabe destacar que podría presentarse el caso que se requiera el funcionamiento simultáneo de todos ellos, así como otros casos en los que tan solo se requiera uno de ellos, y también se debe de pensar en la manera que dichos elementos se puedan conectar y desconectar fácilmente.

De los tres motores descritos en el capítulo anterior, el uso de los servomotores se descartó por su alto costo y debido a que las aplicaciones de la robótica pedagógica no requieren la precision en la posicion que te ofrecen los servomotores.

Se decidió usar motores de CD, tanto por su versatilidad, como por su mayor contenido pedagógico, ya que tiene la forma de controlar físicamente tanto su sentido de giro como su velocidad, y por lo tanto es más ilustrativo para el niño.

Los motores de CD son muy económicos, se pueden encontrar en la mayoría de juguetes que se mueven, aunque tienen el inconveniente de que su velocidad de giro es muy rápida, y por lo cual muchas veces se requiere utilizar reductores de velocidad externos; este hecho añade un magnífico pretexto para el aprendizaje de los quebrados y de las razones y proporciones por parte de los niños.

Se optó también por la utilización del motor de pulsos, para contrastar su funcionamiento con el del motor de CD, y de este modo hacer énfasis en las cualidades de cada uno, así como su utilización en la vida cotidiana.

Para el control de los motores de pulsos es importante establecer una secuencia, lo cual se puede conseguir por medio de software, de un dispositivo programable, o bien, de un circuito integrado diseñado exprofeso para generar dicha secuencia, como es el caso del circuito L297, que es un controlador de motores de pulsos bipolares y unipolares. Lo único que requiere para su funcionamiento es la señal de reloj, una señal de sentido de giro y una señal de habilitación, entre otras. Este circuito no es más que un circuito secuencial de cuatro bits de salida.

Ya que se estableció el tipo de motores que se van a utilizar, ahora se requiere definir la etapa de potencia. Para su diseño se requiere garantizar tanto la intensidad de corriente como el voltaje de operación de los motores, sea cual sea el sistema de control de la interfaz que se considere.

Existe la configuración denominada puente H, la cual sirve para controlar el sentido de giro de un motor de CD además garantiza el suministro de la intensidad de corriente que requieren los motores tanto de DC como de pulsos.

El puente H consiste en cuatro transistores, ya sea TBJ o MOSFET, que trabajan en conmutación, y que se comportan como interruptores controlados por las señales que les llegan a sus bases o sus compuertas, tal como se muestra en la Figura 8.

Cuando se introduce a la entrada I1 una señal alta y a la entrada I2 un voltaje bajo, los transistores Q1 y Q4 (NPN y PNP) entran en saturación simultáneamente, mientras que Q2 y Q3 están en corte por ser de polaridad contraria (NPN y PNP, respectivamente). En estas condiciones el motor gira en un sentido.

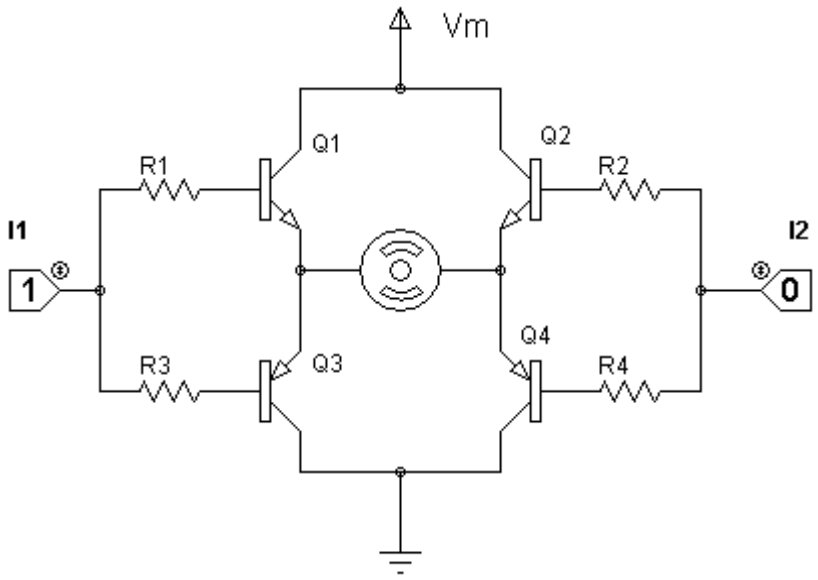


Figura 8 Puente H.

Cuando se invierten las señales de entrada, es decir a I1 se introduce una señal baja y a I2 un voltaje alto, los transistores que se saturan son ahora Q2 y Q3, mientras que los que entran en estado de corte son Q1 y Q4. Esto hace que el motor gire en sentido contrario [3].

En el mercado existen circuitos integrados con dicha configuración. Entre los más conocidos están el L298 para voltajes menores de 48 V y hasta 2 A, el LMD18200 con capacidad de corriente de 3 A y voltajes hasta de 55 V; MC33887 de Motorola, 5 a 40 V TTL/CMOS. Además el L293D trabaja como tal, para voltajes hasta de 36 V con intensidades de corriente de 600 mA,

Lo que se necesita conocer, para definir qué puente H se debe elegir, es la intensidad de corriente de consumo, así como el voltaje de operación de los motores. Para circuitos de bajo consumo de intensidad de corriente (hasta 3 A), lo más conveniente es el empleo de circuitos integrados comerciales; de lo contrario, se requiere construirlo con elementos discretos tales como transistores de potencia.

3.1.2 Sensores digitales y analógicos

Para que un robot sea más o menos autónomo, éste requiere un mínimo conocimiento de su entorno, y es por medio de diversos sensores es que lo logra, por tal motivo es importante elegir el tipo adecuado de sensores con los que se diseñará.

Para un buen control del robot, en la mayoría de los casos, debe ser capaz de obtener las lecturas de sensores en tiempo real, para que éste pueda tomar decisiones con respecto a su movimiento de inmediato.

Varios sensores analógicos tienen tres terminales de salida correspondientes a la polarización, la tierra y la señal de salida. Aprovechando esta característica, fue posible diseñar la interfaz de manera que fuera adaptable a varios tipos diferentes de sensores analógicos, por medio de la creación de entradas genéricas.

Para el caso de dichos sensores analógicos, se puede trabajar con rangos, con umbrales, o bien, realizar su conversión analógica digital, en cuyo caso se requiere un circuito convertidor. En caso de trabajar con umbrales es necesario un comparador, en el que se define un voltaje de referencia, de manera que la entrada es comparada con dicho voltaje, teniendo como salida un estado lógico bajo si la entrada se encuentra debajo de la referencia, o un estado alto si está arriba de ella. Es posible invertir dichas salidas, intercambiando las terminales de la señal de entrada y del voltaje de referencia.

Si con los sensores analógicos se pretende detectar ciertos rangos de valores, se requiere implementar más comparadores, uno por cada valor límite diferente, y la salida general se puede obtener por medio de un circuito combinacional que tome en cuenta las salidas de todos los comparadores.

Asimismo, algunos sensores digitales, tales como el fotointerruptor, también cuentan con terminales similares a los sensores analógicos. Sin embargo, existen

otros, como los interruptores de botón o los de palanca, que sólo requieren dos terminales para su correcto funcionamiento. Por consiguiente, para la construcción de la interfaz se consideraron entradas de dos terminales para este tipo de sensores.

3.1.3 Sistema de control

Para definir cómo se implementará el sistema de control, se debe de tomar en cuenta una de las características establecidas para la interfaz: la comunicación con la computadora a través del puerto USB. Para ello es necesario un transceptor, que se encargue de realizar funciones tanto de transmisión como de recepción.

Tomando en cuenta esta consideración, se tienen al menos dos opciones: utilizar la computadora como interfaz gráfica y también como sistema de control, que envíe las instrucciones a cada uno de los elementos del hardware; o bien, utilizar la PC únicamente como interfaz gráfica y el sistema de control se encuentre en la propia interfaz.

Si se utiliza la computadora como sistema de control, se aprovecharía su procesador, aunque tendría varios inconvenientes, como podrían ser algunos problemas en el flujo de la comunicación, las interrupciones por procesos propios de la máquina, además de que su funcionamiento podría variar según las características de cada equipo.

La utilización de la PC únicamente como interfaz gráfica que provea las órdenes para que el dispositivo trabaje, y por consiguiente, implementar el sistema de control en dicha interfaz, utilizaría la comunicación con la PC únicamente cuando el usuario necesite modificar el comportamiento del robot que aquélla tenga conectada. Para realizar esta opción, se podrían emplear dispositivos lógicos programables, FPGA o microcontroladores, entre otros dispositivos.

Los dispositivos lógicos programables son de gran utilidad para desarrollar funciones secuenciales, y los hay con diferentes características. En caso de que se utilicen para el sistema de control, es posible establecer las secuencias de los motores de pulsos, cambiar su sentido de giro y su velocidad, aunque ésta estaría limitada a dos o tres diferentes. También se puede controlar el sentido y la velocidad de motores de CD, por medio del diseño de un circuito generador de PWM, y de una señal de sentido de giro. Asimismo, es factible la adquisición de los datos de los diversos sensores conectados a la interfaz, y únicamente haría falta un transceptor que permita la comunicación con la PC.

El FPGA cuenta con un sistema de desarrollo con conexión USB, lo cual tendría la ventaja de que el transceptor y el sistema de control se podrían encontrar juntos en el mismo dispositivo. Dado que es un circuito con mayor nivel de integración que los PLD, se podrían realizar todas las tareas de la interfaz utilizando únicamente un solo dispositivo.

Es frecuente el uso del microcontrolador para el diseño de sistemas de control. Existe una amplia gama de integrados, por lo que se considerarán aquéllos que ya incluyan la comunicación por medio del puerto USB, que sean fáciles de conseguir, tanto el circuito como las herramientas de desarrollo, y que sus periféricos, y el tamaño de su memoria de programa sean los adecuados.

A continuación se mencionan los integrados de algunas compañías que se consideraron, ya que sus circuitos cumplen con varias de las características deseables.

Silicon Lab, cuenta con una gama de 16 microcontroladores con comunicación USB de las familias C8051F320/21 C8051F326/27, C8051F34x. En la Figura 9 se puede observar un diagrama de bloques donde se muestra su configuración. Son de bajo costo.

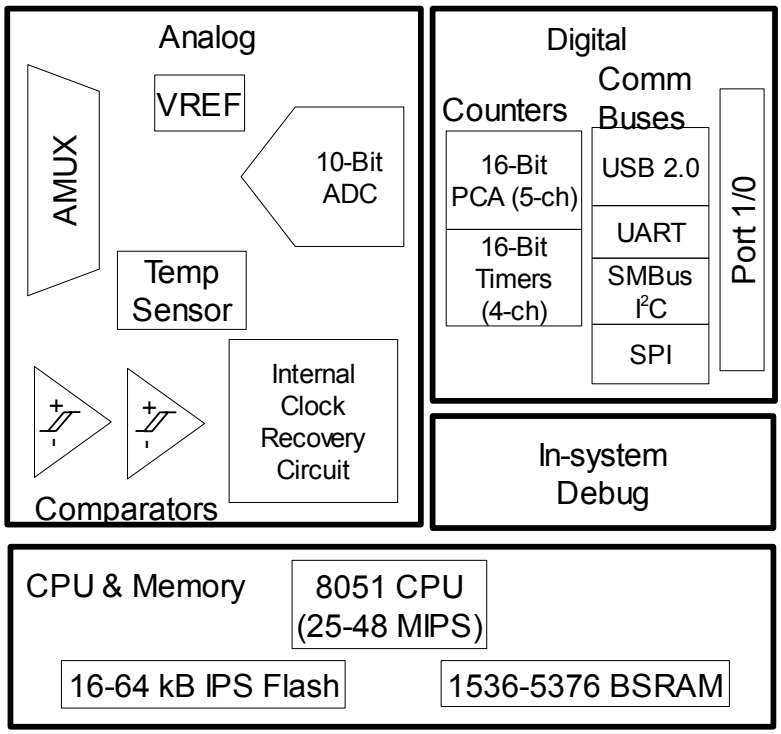


Figura 9 Diagrama de bloques de un microcontrolador de Silicon Lab.

ATMEL cuenta con una familia reducida de microcontroladores con comunicación USB, la familia AT90USBXXX, la cual puede ser una buena solución para este proyecto, ya que son accesibles, de bajo costo, y la compañía proporciona tutoriales.

Las especificaciones del circuito que se consideró de interés son las siguientes: memoria flash de programación de 8 kb, 512 b de SRAM, 512 b de memoria EEPROM, con la posibilidad de 16 millones de instrucciones por segundo (MIPS)

con un reloj de 16 Mhz, voltaje de operación de 2.7 a 5.5 V, aunque sólo es disponible en empaquetados TQFP32 y VQFN.

De la compañía Microchip existen más de 40 microcontroladores con opción de comunicación USB, con diversas características de memoria de programa, número de temporizadores, velocidad permitida en la transmisión, periféricos, número de instrucciones por segundo que pueden ejecutar, tipo de empaquetado y tipo de arquitectura.

Microchip proporciona documentos técnicos de sus microcontroladores en un lenguaje comprensible, el software para programarlo es gratuito, son compatibles con otros chips de la misma compañía, tienen soporte técnico, al ser ampliamente conocidos y usados, existen foros en Internet para poder compartir experiencias con otros programadores, además de su bajo costo.

En el mercado se pueden encontrar tarjetas de desarrollo, que permiten el control vía puerto USB de cierto número de entradas o salidas digitales. Pero el inconveniente de estas tarjetas es que recurrentemente no se llegan a ocupar todos los elementos que proporciona, por lo que dicha tarjeta es subutilizada, y su costo es relativamente alto para lo que se obtiene de ellos, por lo que una opción sería diseñar una tarjeta según las necesidades específicas de este trabajo.

Algo que se debe considerar en el diseño es minimizar tanto las fuentes de error como el número de circuitos integrados a utilizar, por lo cual no es conveniente la utilización del GAL, ya que se necesitarían varios de estos dispositivos para la implementación de la interfaz.

El empleo de un microcontrolador tendría la ventaja de disminuir el número de elementos así como las posibilidades de error, por lo podría ser una buena opción, y sólo se necesitaría elegir el circuito que mejor se adapte a las necesidades de este proyecto.

3.2 ALTERNATIVAS PARA LA COMUNICACIÓN MEDIANTE EL USB

Sin duda la utilización del puerto USB no es sencilla, en comparación con el puerto paralelo o el serial. Una posibilidad sería la utilización del cable serial–USB, de tal forma que físicamente se utilizaría el puerto USB de la PC, pero con los protocolos de comunicación serial que son más sencillos. Comercialmente es fácil encontrar este tipo de convertidores, que poseen un conector DB–9 y trabaja con el estándar de comunicación RS–232. Cada una de las nueve terminales tiene una función específica: unas transmiten datos, otras los reciben y otras controlan la comunicación. En su versión más simple, se utilizan únicamente tres de estas terminales: *Received Data* RXD, *Transmitted Data* TXD, *Signal Ground* SG. Esta versión sirve para comunicarse con los microcontroladores o dispositivos que permitan sustituir el trabajo de las otras terminales de control de la comunicación por medio de software, el cual se encarga de controlar la transmisión enviando un bit de inicio y otro de paro.

Debido a que el protocolo de comunicación RS–232, no es compatible con entradas y salidas TTL, se requiere un circuito integrado capaz de acoplar las señales.

Para el empleo del cable convertidor USB–serial, se deben de tomar en cuenta las siguientes consideraciones:

- La velocidad de transmisión es menor en el puerto serial que en el USB
- No se aprovechan las cualidades del puerto USB
- La creación de una conexión extra aumenta consigo posibilidades de error
- La gran ventaja que se tendría es que no se requiere un controlador de Windows para la interfaz electrónica
- El control por medio del puerto serial es muy conocido, y se puede realizar directamente sin la necesidad de un programa especializado. Se puede

utilizar la *Hyperterminal* de Windows, lo que serviría como sistema de prueba de la interfaz electrónica; además, existen bibliotecas en la mayoría de los lenguajes de programación para este puerto, lo que la hace una opción viable.

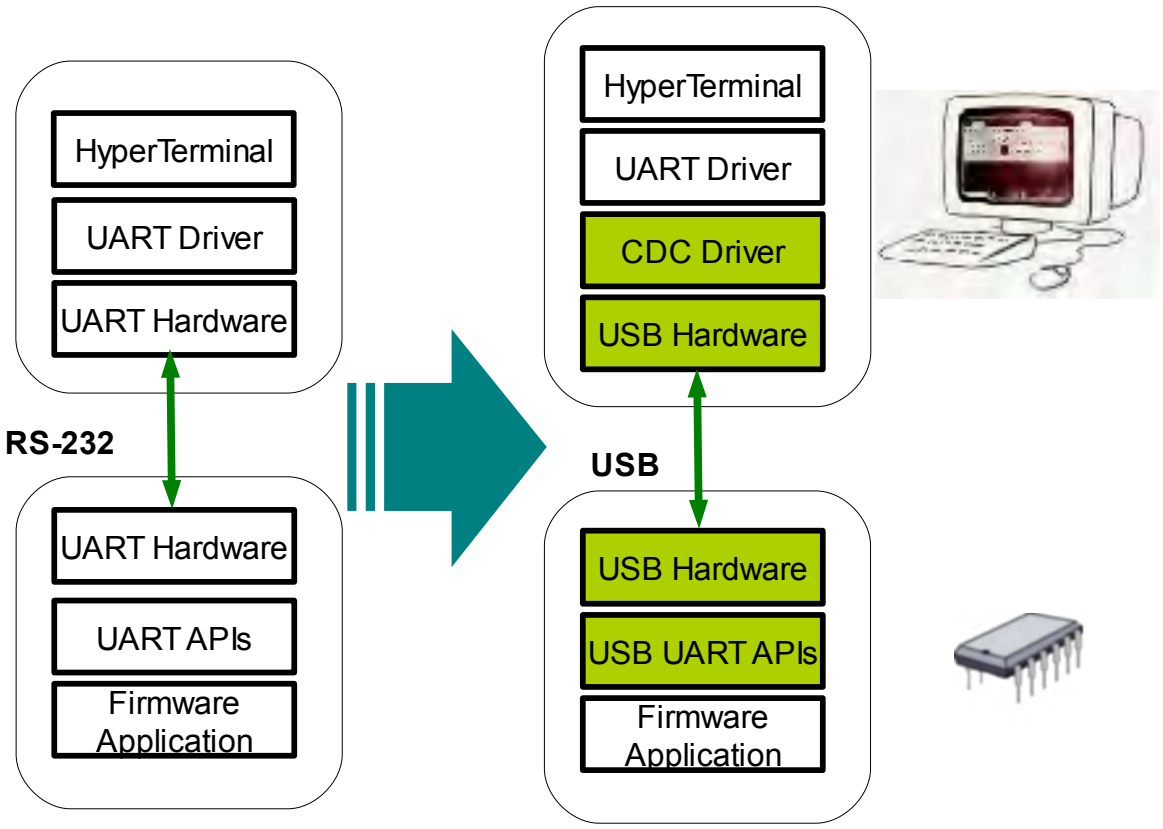


Figura 10 Diagrama comparativo del protocolo RS–232 con la comunicación USB.

En el diagrama de la Figura 10, se puede observar una comparación entre el protocolo RS–232 y la comunicación por el puerto USB.

La otra alternativa para establecer la comunicación es abordar la configuración del puerto USB y aprovechar sus cualidades, para lo que se debe realizar una búsqueda de los transceptores existentes en el mercado.

3.3 PROGRAMA DE APLICACIÓN

Para el desarrollo del programa de aplicación se tomaron en cuenta las consideraciones que se mencionan a continuación.

Partiendo del concepto de que la programación es la actividad de representar ideas y procesos en algún lenguaje formal, la idea original del proyecto es la creación de un lenguaje de programación con el que se puedan dar instrucciones al robot diseñado por el niño, de manera directa, con lenguaje natural, que sea interactivo, intuitivo, gráfico y sencillo.

Con el fin de no retrasar el desarrollo del proyecto, se optó por la realización de un programa gráfico de aplicación, que sirva para verificar el funcionamiento de la interfaz electrónica.

Debido a que se trata de una interfaz con la posibilidad de interactuar con varias aplicaciones, el lenguaje debe ser de uso generalizado, preferentemente haciendo referencia a sus partes físicas, para lo cual es conveniente basarse en imágenes, dado que en la mayoría de casos se trata de un lenguaje nuevo para el niño, y con la finalidad de facilitar la comprensión de la interrelación de las interfaces física y gráfica.

El usuario, a través de esta aplicación, podrá controlar los parámetros de cada uno de los elementos de la interfaz, que para el caso de los motores serían su sentido y velocidad de giro, así como su estado de encendido–apagado, y para los sensores sería su lectura. Se pretende que se puedan ejecutar procesos en paralelo y de forma independiente.

C. Luden en su libro “Lenguajes de programación: principios y práctica" [5] , señala una serie de características deseables en los lenguajes de programación, hace una reseña de los principales lenguajes y señala cómo cumplen o violan estas

recomendaciones, por lo que recalca que son características deseables y no forzosas, y por tanto, se deben de considerar a la hora de tomar las decisiones. Dichas características son las siguientes: eficiencia, expresividad, capacidad de mantenimiento, legibilidad, confiabilidad, seguridad, simplicidad, y capacidad de escritura. Todas ellas están de una u otra manera relacionadas con la eficiencia y se refiere a ella en cada una de las tareas del lenguaje de programación: en la ejecución del programa, en el código, en la traducción y en la programación.

La interfaz, al ser una herramienta para la robótica pedagógica, será considerada como material didáctico, y como tal debe de cumplir ciertas características para su mejor aplicación. Es por esta razón que para el diseño del programa se tomaron en cuenta algunas consideraciones de un buen material didáctico:

- Cree situaciones atractivas de aprendizaje
- Facilite la apreciación del significado de sus acciones
- Prepare el camino para nociones valiosas
- Dependa de la percepción de imágenes visuales
- Sea polivalente.

Tomando en cuenta tanto las necesidades del proyecto como las características deseables de un lenguaje de programación y las consideraciones para un buen material didáctico, se procedió al diseño de la aplicación, que se presenta con todo detalle en el Capítulo 5.

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.

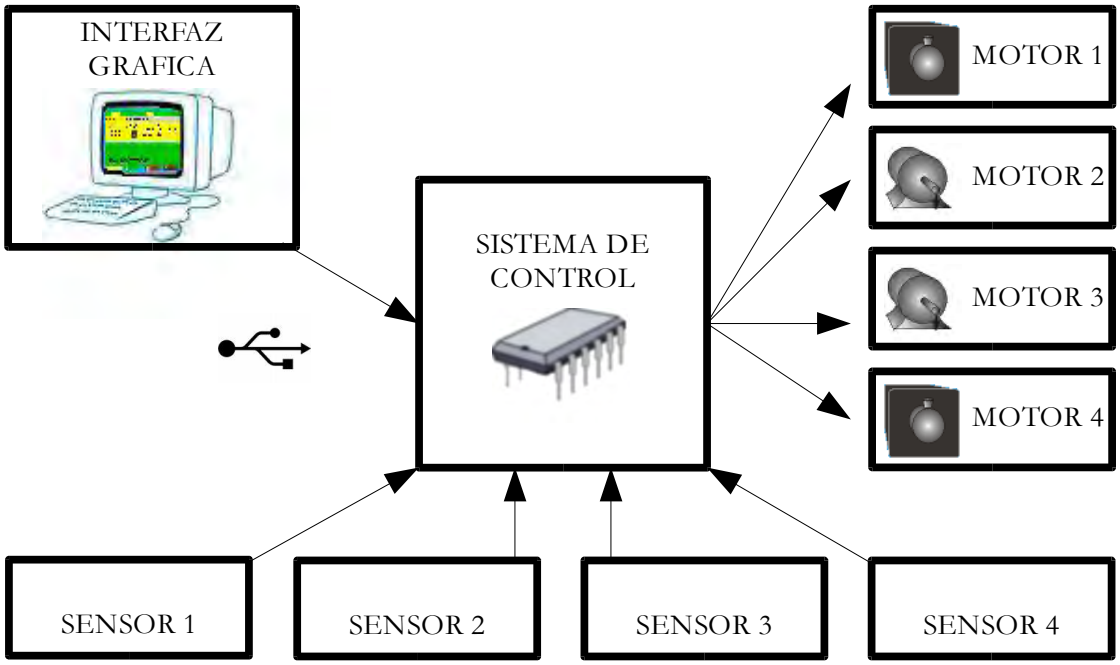


Figura 11 *Diagrama de bloques de la interfaz electrónica.*

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.

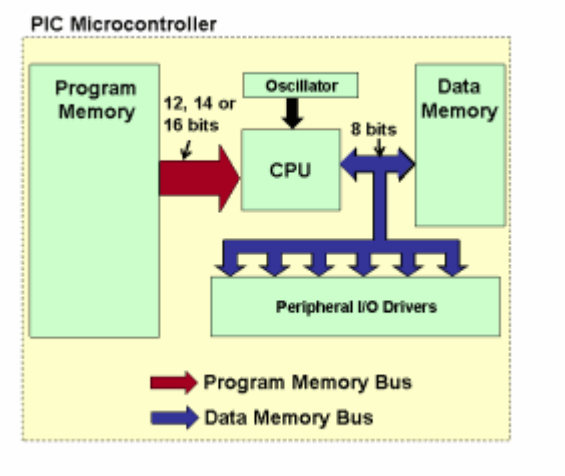


Figura 12 Arquitectura del PIC18F4550.

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.

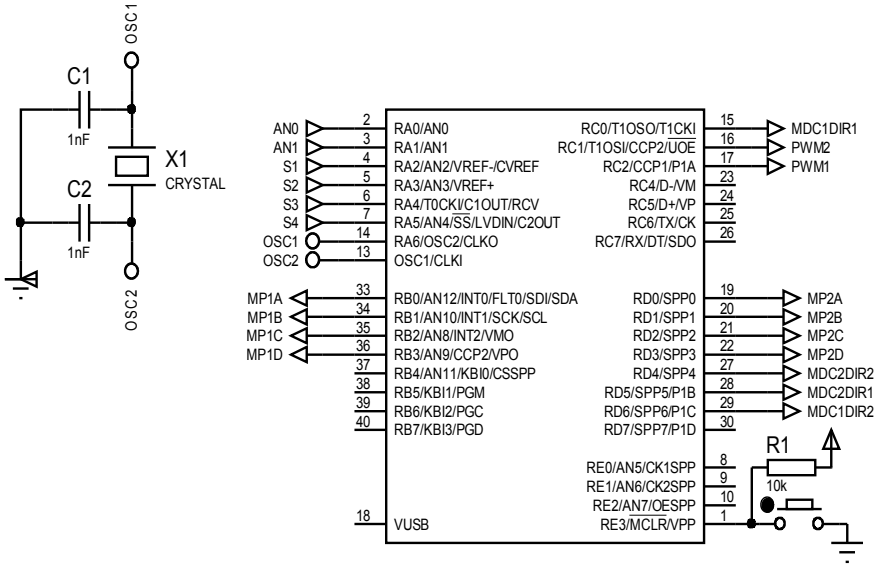


Figura 13 Configuración mínima para la operación del PIC 18F4550.

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.

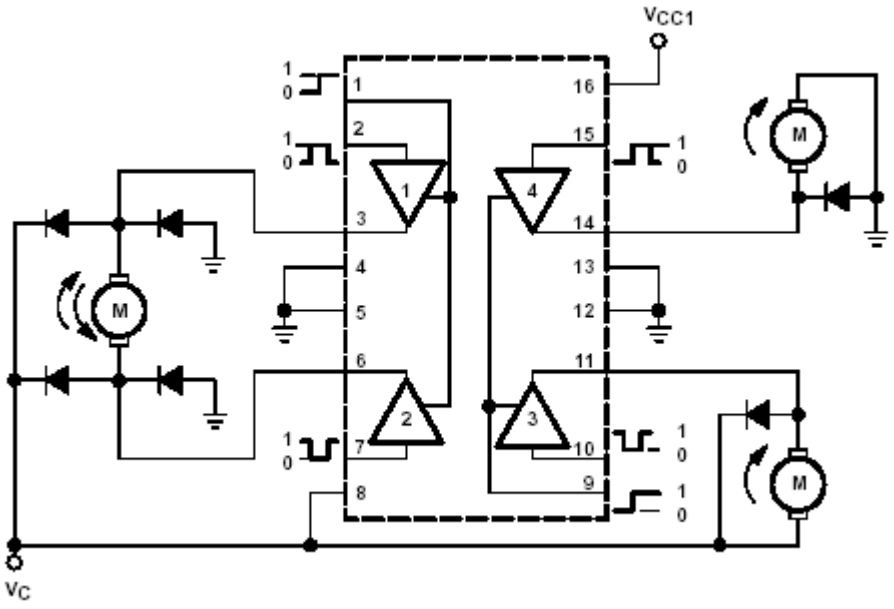


Figura 14 Diagrama del circuito integrado L293D.

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.

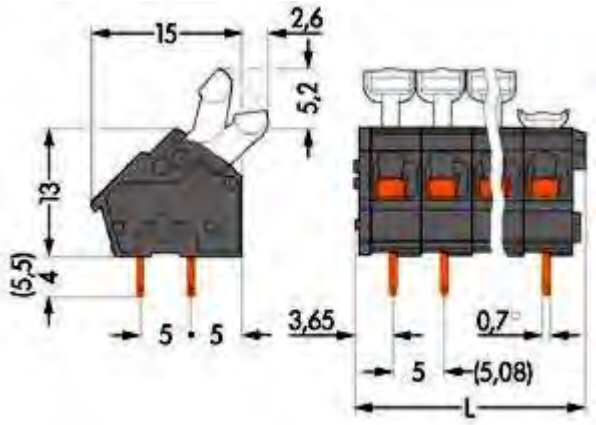


Figura 15 Terminales utilizadas para la conexión de los cables.

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.

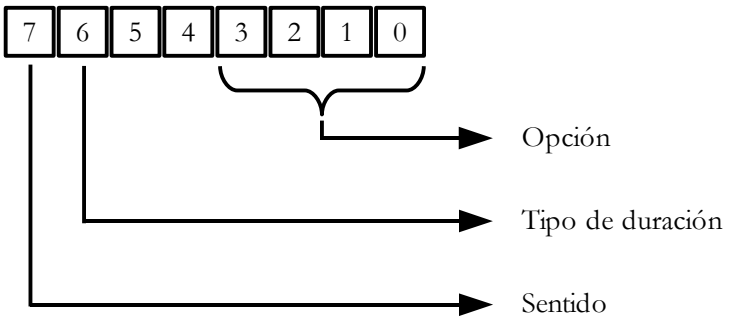


Figura 16 Configuración del primer byte recibido.

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.

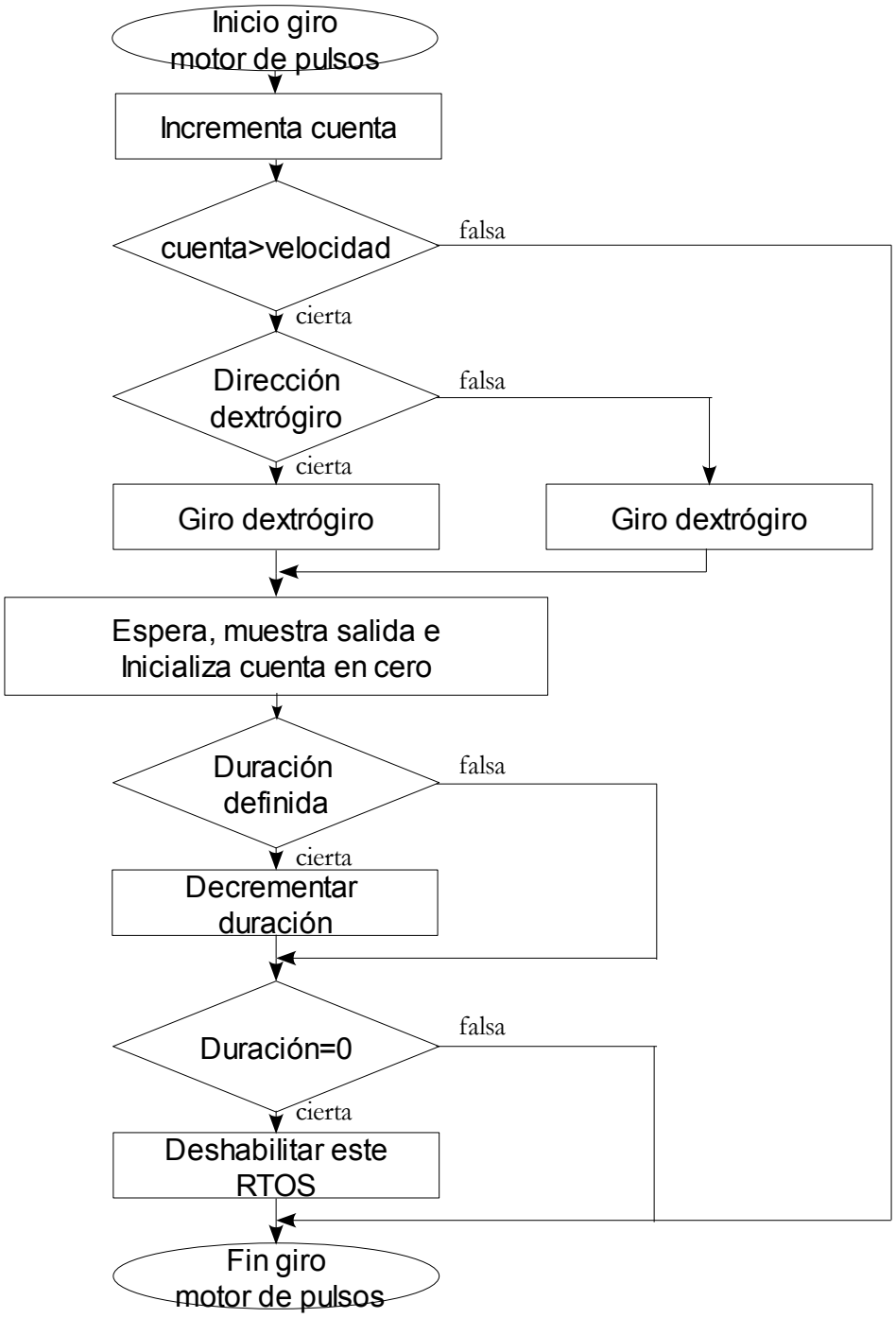


Figura 17 Diagrama de flujo de la operación del motor de pulsos.

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.

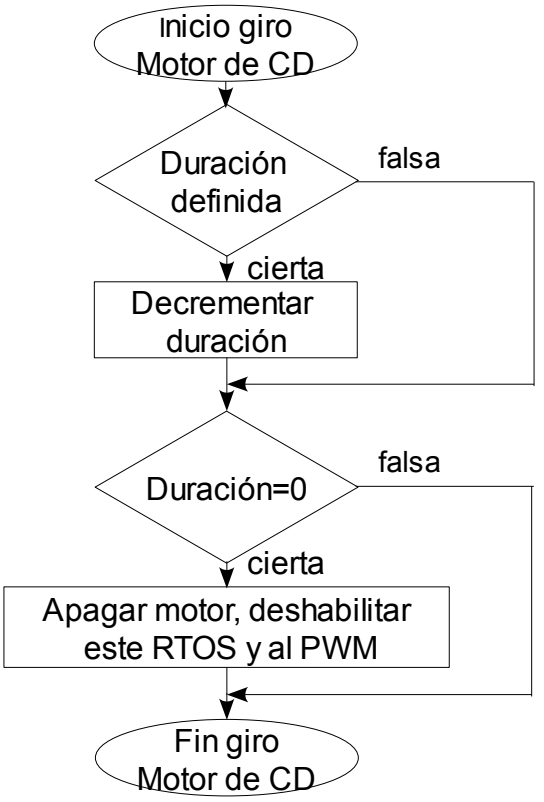
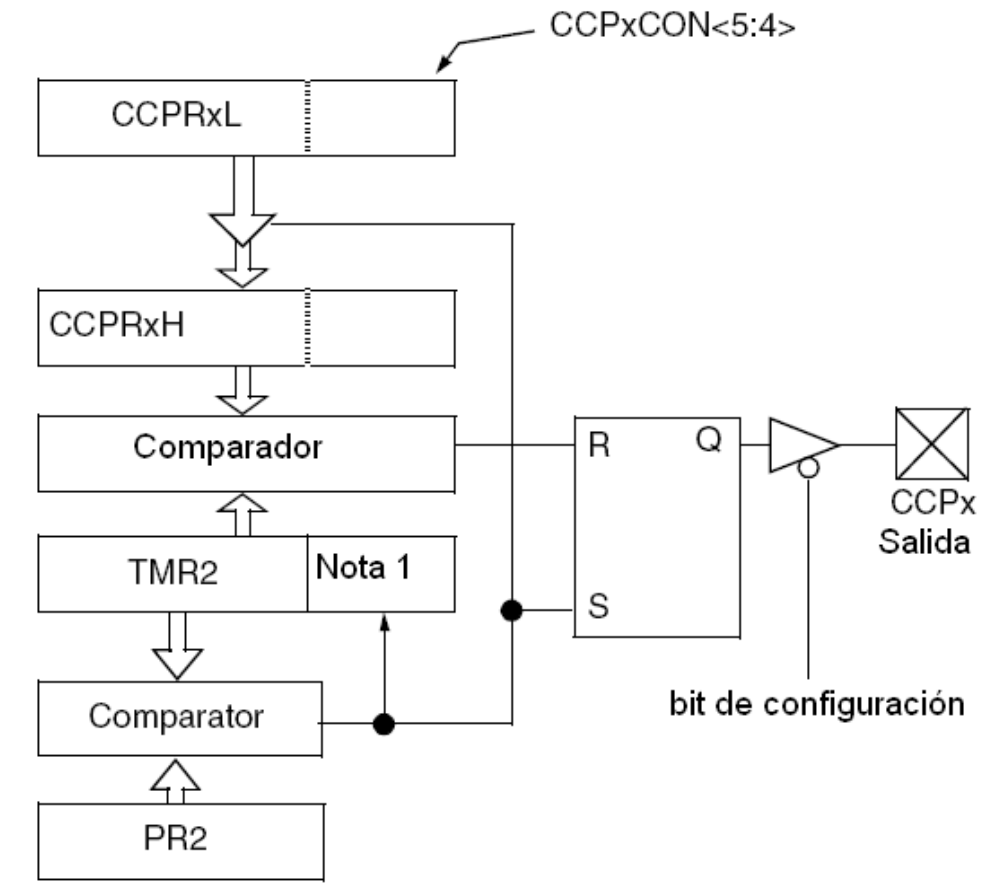


Figura 18 Diagrama de flujo de la operación del motor de CD.

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.



Nota 1: El valor del registro TMR2 de 8 bits es concatenado con dos bits del reloj interno o dos bits de el prescalador para crear una base de tiempo de 10 bits.

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.

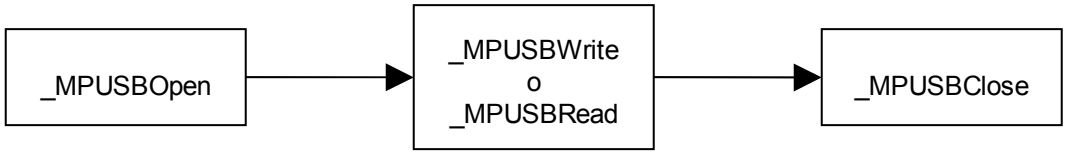


Figura 20 Estructura del envío/recepción de datos.

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.

CAPÍTULO 5
DISEÑO DE LA INTERFAZ GRÁFICA

Actualmente se puede encontrar una gran variedad de programas de cómputo orientados a los niños, con diferentes enfoques y de diferentes disciplinas. En lo que coincide la mayoría es que son muy gráficos; esto es fácil de lograr gracias a los avances en los lenguajes de programación, sobre todo aquéllos orientados a objetos.

Se decidió utilizar el lenguaje de programación C#, debido a que tiene características favorables, como es el hecho de realizar aplicaciones gráficas tipo Windows con relativa facilidad, permite la utilización de bibliotecas externas, y puede aplicarse en programación para páginas Web, lo cual puede ser una ventaja a futuro. Para la ejecución de la aplicación se requiere únicamente el .NET Framework, el cual está incluido en Windows Vista, y para los demás sistemas este software es gratuito, y por consiguiente se puede descargar sin ningún problema.

Se desarrollaron dos aplicaciones gráficas llamadas “control independiente” y “principios de programación”; la primera trabaja con cada elemento de manera independiente, mientras que la segunda integra a los elementos de la interfaz para su funcionamiento.

5.1 INTERFAZ GRÁFICA “CONTROL INDEPENDIENTE”

Esta interfaz gráfica presenta un menú de inicio como el mostrado en la Figura 21, Por medio de este menú se tiene acceso a las ventanas de control de los componentes de la interfaz. La distribución de los botones de acceso es similar a la posición que ocupan sus elementos en la interfaz física, todo esto para facilitar su manipulación.

Figura 21 Ventana del menú de opciones.

La interfaz gráfica permite la apertura de todas las ventanas simultáneamente, lo cual sirve para aprovechar el procesamiento en paralelo que el microcontrolador puede realizar.

Para manipular cada uno de los motores de manera independiente y simultanea, basta con abrir la ventana correspondiente, donde se encuentran los controles gráficos para cada uno de los parámetros. En el esquema de la Figura 22 se pueden observar los elementos que es posible variar en los motores de pulsos.

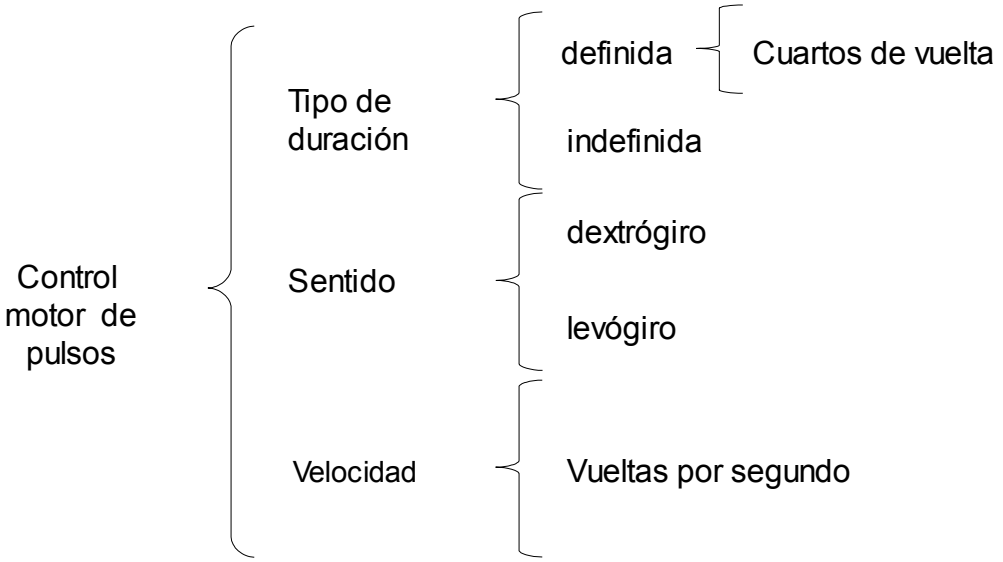


Figura 22 Parámetros a variar en el motor de pulsos.

Cuando se trabaja con duración indefinida, se debe *desmarcar* el cuadro *"Duración en cuartos de vuelta"* que se encuentra en la parte superior de la ventana, ver Figura 23. De este modo, la manipulación del motor se puede trabajar en tiempo real, permitiendo variar su velocidad por medio del *scrollbar,* y su sentido de giro con los botones.

En caso que se trate de duración definida, el motor sólo funcionará durante el número de vueltas establecido, con la velocidad y sentidos definidos al inicio.

La ventana de control de los dos tipos de motores es muy similar; lo que varía son los parámetros de duración definida y velocidad.

Figura 23 Ventana para el control de un motor de pulsos.

Para los motores de pulsos, la duración se da en cuartos de vuelta y la velocidad en vueltas por segundo, mientras que en los motores de CD la duración está dada en segundos y la velocidad en porcentaje de la velocidad máxima. Dado que las terminales de alimentación del motor de CD se conectan dentro de la interfaz al regulador de 9 V, la velocidad real de dicho motor dependerá de su voltaje nominal, así como de sus propias características.

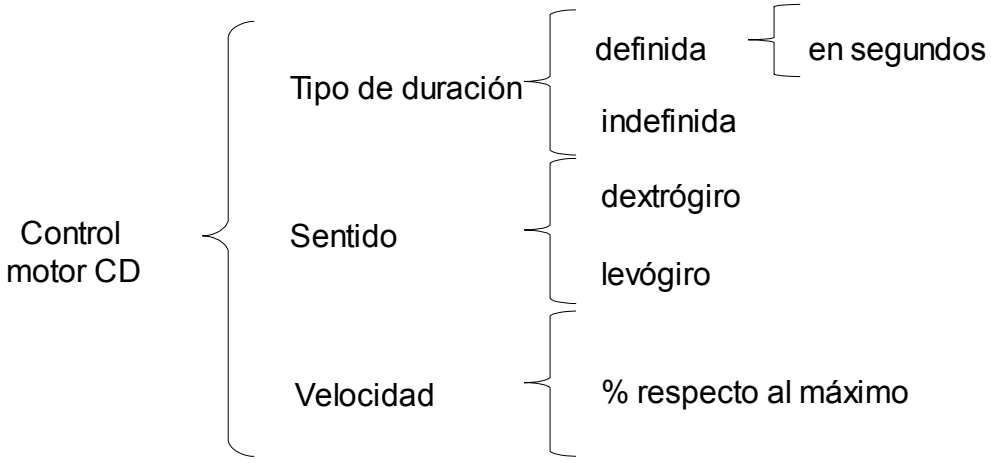


Figura 24 Parámetros a variar en el motor CD.

En la Figura 24 se muestran los elementos a variar en el motor de CD, y en la Figura 25 la ventana de control.

Figura 25 Ventana para el control del motor de CD.

Con respecto a las señales de entrada, los cuatro sensores digitales se pueden visualizar en una sola ventana. Cada uno está representado por un foco que se muestra prendido o apagado, según sea el caso, lo cual se efectúa en tiempo real, y que se puede visualizar en una ventana como la mostrada en la Figura 26. En otra ventana se puede observar la lectura de los sensores analógicos, como se muestra en la Figura 27; dichos sensores devuelven un valor de 0 a 255 correspondiente al valor leído.

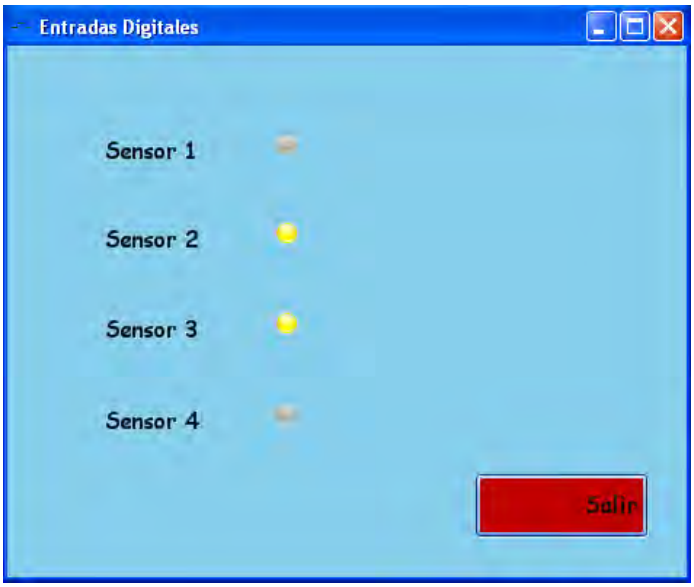


Figura 26 Ventana para la lectura de las entradas digitales.

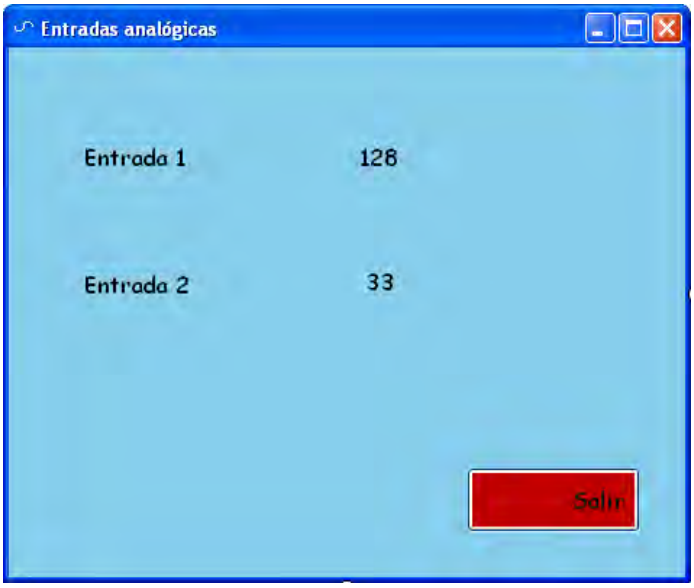


Figura 27 Ventana para la lectura de sensores analógicos.

5.2 INTERFAZ GRÁFICA "PRINCIPIOS DE PROGRAMACIÓN"

Como el titulo lo indica, esta interfaz pretende acercar al niño a los principios de programación de manera interactiva, y que pueda observar las consecuencias de sus decisiones.

Son tres sentencias básicas de la programación las que se abordaron en la interfaz: *if, for, while*; cabe resaltar que la idea de cada una de ellas es la que se pretende plasmar.

Esta aplicación inicia con un menú de opciones, que se puede apreciar en la ventana de la Figura 28. En ella se pueden observar cuatro botones: tres de ellos corresponden a las sentencias mencionadas: si (*if*), para (*for*) y mientras (*while*); además, se cuenta con uno de salida. Cada uno de los botones abre una ventana diferente.

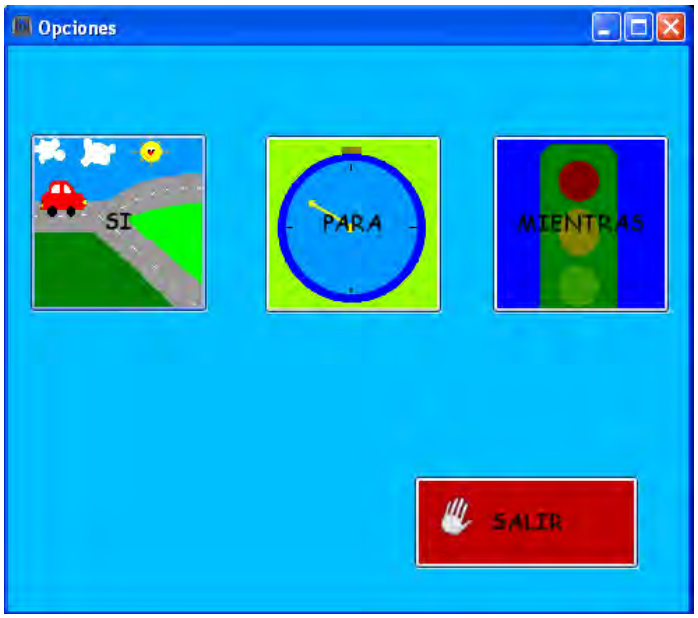


Figura 28 Ventana del menú de opciones.

La opción *SI* proviene de la sentencia de programación *if*, y se trata de una toma de decisión: si una condición se cumple, entonces se ejecuta una acción.

La sentencia *PARA* corresponde a la sentencia *for* y consiste en que se repite una acción un determinado número de veces.

Para que una acción se ejecute continuamente, siempre y cuando se cumpla una condición, se utiliza la opción *MIENTRAS*, siendo análogo a la sentencia *while*, con la variante que permanecerá en la posibilidad de regresar a ejecutarse la opción, siempre y cuando no se salga de la ventana.

Las ventanas de cada una de las opciones están divididas en cinco secciones, tal y como se pueden apreciar en la Figura 29: 1 Estructura de la sentencia, 2 Condición, 3 Acción, 4 Pseudocódigo, 5 Botones.

Figura 29 Formato de las ventanas.

Estructura de la sentencia. Se enuncia la sentencia, remarcando en recuadros amarillos y letras rojas, la condición, la acción y aquellos parámetros que el usuario debe definir.

Condición. Muestra las entradas con las que se puede trabajar, y el usuario debe de establecer el estado de la entrada que será tomada como condición para realizar o no la operación establecida.

Acción. Muestra cuatro pestañas correspondientes a cada motor, donde se encuentran los parámetros que se pueden controlar de cada uno. Sólo se puede visualizar la ficha de un motor a la vez. Si se quiere establecer el funcionamiento de otro motor, sólo basta con seleccionar la pestaña del motor deseado con el botón izquierdo del *mouse*.

El usuario debe definir el estado de cada uno de los motores, que por omisión se encuentran apagados; para encender uno de ellos, se debe seleccionar con el

botón izquierdo del *mouse* el recuadro que dice Motor x encendido; también se puede cambiar el sentido de giro del motor y la velocidad a la que trabajará, siempre y cuando se cumpla con la condición.

El sentido de giro puede ser levógiro u horario y dextrógiro o antihorario; en la ventana aparece con los letreros izquierda y derecha, por considerarse más ilustrativo para los niños, y que se refieren a la regla de la mano izquierda o derecha, según sea el caso.

Los motores 1 y 4, son de pulsos y su velocidad varía de: 0.3 a 3 vueltas por segundo. Cada marca de la barra de duración está dada en cuartos de vuelta, y por consiguiente lo más que se puede girar es dos vueltas y media.

En cuanto a los motores 2 y 3, se trata de los de corriente directa. La duración de su operación está dada en segundos.

Pseudocódigo. Una vez establecidas las condiciones y las acciones se procede a oprimir el botón seleccionar, que despliega en esta sección el pseudocódigo de lo que se estableció, es decir, describe lo que va a realizarse.

Botones. Las ventanas de cada una de las sentencias cuenta con cuatro botones: Seleccionar, Ejecutar, Parar y Salir.

Seleccionar. Al oprimirlo se desplegará en la sección de pseudocódigo una descripción de cómo va a trabajar, sin ejecutar ninguna acción. Se pueden modificar las condiciones y las acciones sin problema alguno, siempre y cuando cada que se cambie algo, se oprima este botón para mantener actualizada la sección del pseudocódigo.

Ejecutar. Una vez que el usuario esté de acuerdo con lo que marca el pseudocódigo, puede oprimir este botón, con lo cual se iniciará la lectura del sensor

o los sensores, y si cumple con la condición establecida, entonces ejecutará la acción como lo indica el pseudocódigo mencionado.

Parar. En cualquier momento de la ejecución se puede oprimir este botón, el cual interrumpirá toda acción, regresando a la interfaz a su estado inicial.

Salir. Cierra la ventana que se esté mostrando.

5.2.1 Interfaz para la sentencia *if*

Los pasos a seguir para su funcionamiento son cuatro:

1 Establecer la condición, activando o desactivando los cuadros correspondientes a los sensores.

2 Establecer los parámetros deseados de cada motor, en caso de que se cumpla la condición anterior.

3 Oprimir el botón Seleccionar.

4 En caso de estar de acuerdo con el pseudocódigo, oprimir botón Ejecutar, en caso contrario, regresar al paso 1.

Cuando se ejecuta el programa:

Se lee el estado de los cuatro sensores digitales y los despliega en la parte derecha de la sección de pseudocódigo; los compara con la condición establecida por el usuario; si se cumple, entonces se ejecuta la acción establecida por el usuario; en caso contrario, no hace nada. En la Figura 30 se muestra la ventana correspondiente a la sentencia SI y en la Figura 31 el diagrama de flujo que describe su funcionamiento.

Figura 30 Ventana de la sentencia If.

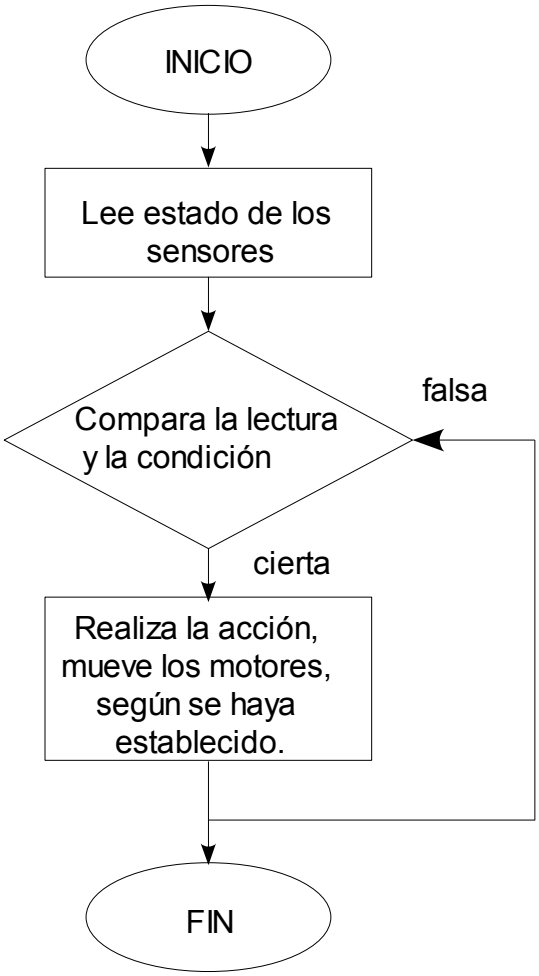


Figura 31 Diagrama de flujo de la sentencia if.

5.2.2 Interfaz para la sentencia *for*

Para la ejecución de esta sentencia se deben de seguir los siguientes pasos:

1 En la sección de condición, establecer los valores de inicio y fin de la cuenta con números de dos dígitos, cuidando que el valor final sea mayor que el valor inicial.

2 Establecer los parámetros para cada motor y en caso que se cumpla la condición del punto anterior, se pondrán en marcha.

3 Oprimir el botón Seleccionar.

4 En caso de estar de acuerdo con el pseudocódigo, oprimir botón Ejecutar; en caso contrario, regresar al paso 1.

Cuando se ejecuta el programa, en la esquina superior derecha de la sección del pseudocódigo, aparece un contador que inicia en cero, y cuando llega al número establecido en el campo *Inicia*, se activan los motores según esté registrado en el pseudocódigo, y seguirán girando hasta que la cuenta llegue al valor introducido en el campo *Finalizar*. En la Figura 32 se puede apreciar la ventana correspondiente, y en la Figura 33 su diagrama de flujo.

Figura 32 Ventana para la sentencia for.

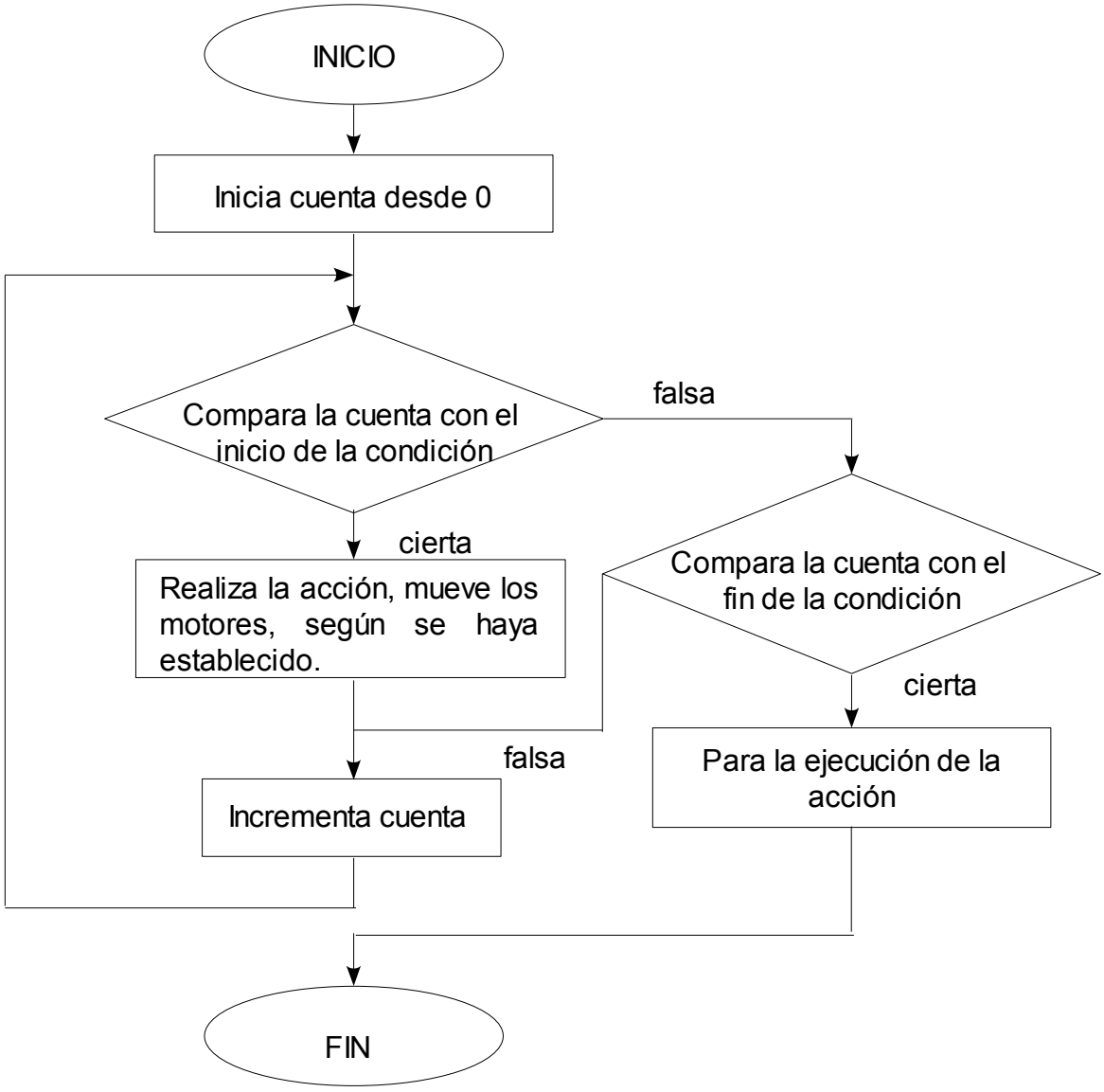


Figura 33 Diagrama de Flujo de la sentencia for.

5.2.3 Interfaz para la sentencia *while*

A continuación se describe el funcionamiento de la ventana de esta sentencia:

1 En la sección de condición, establecer: con cuál sensor analógico se quiere trabajar (A o B), la relación (mayor o menor) y el valor con el que se debe comparar,

como se puede observar en la sección condición de la ventana para la estructura while de la Figura 34.

Figura 34 Ventana para la estructura while.

2 Establecer los parámetros deseados para cada motor, y los ejecute en caso de que se cumpla la condición establecida en el paso anterior.

3 Oprimir el botón Seleccionar.

4 En caso de estar de acuerdo con el pseudocódigo, oprimir el botón Ejecutar, en caso contrario regresar al paso 1.

Cuando se ejecuta el programa, en la esquina superior derecha de la sección de seudocódigo se muestra la lectura del sensor analógico elegido como condición; si cumple con el pseudocódigo, entonces se ejecuta la acción. La lectura del sensor se realiza permanentemente, por lo que cuando deje de cumplirse la condición, la acción se interrumpe. Se tiene la posibilidad de volver a activar la mencionada

acción si se vuelve a cumplir la condición. La única manera de terminar este proceso es con el botón Parar. En la figura 35 se muestra su diagrama de flujo.

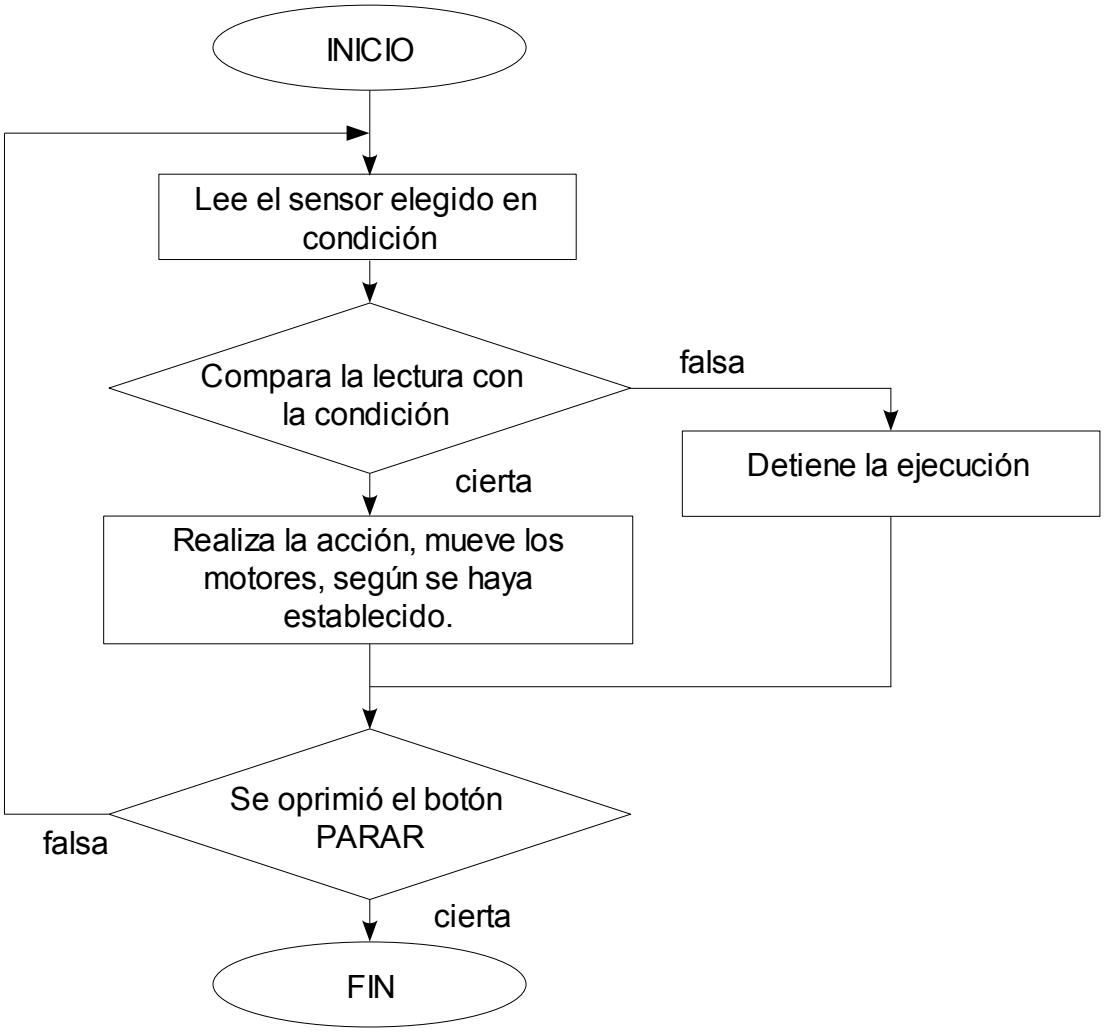


Figura 35 Diagrama de flujo del programa para la sentencia while.

CAPÍTULO 6
RESULTADOS, RECOMENDACIONES Y CONCLUSIONES

6.1 RESULTADOS

Este proyecto se realizó con base en una metodología de trabajo por etapas; una vez elegido el microprocesador, se procedió a establecer la comunicación con la *PC* vía *USB* en una aplicación de *C#*, que controlará el encendido de un led para comprobar el envío de información, así como la lectura del estado de un interruptor para verificar la recepción.

Una vez establecida la comunicación, se continuó el trabajo con cada uno de los actuadores, de manera independiente, tanto en el *firmware* como en el *software*. Se establecieron los parámetros a controlar así como sus rangos de operación. Posteriormente, se realizaron pruebas para verificar su funcionamiento, y ajustarlos a las necesidades del proyecto.

Se seleccionó el tipo de sensores con los que se trabajó, y para los cuales se definió cómo se conectarían con el microcontrolador, así como la forma como se mostrarían sus resultados.

Finalmente, se conjuntaron y se depuraron los diferentes programas en el *firmware*, de acuerdo a los puertos disponibles y la configuración de los mismos. El problema que se tuvo que resolver fue la continuidad de las instrucciones, para lo cual se requirió realizar en paralelo el funcionamiento de los motores y la lectura de los sensores, garantizando que los cambios desde la computadora se efectuarán en tiempo real. Se intentó utilizar las interrupciones del microcontrolador, para actualizar los datos recibidos, pero esto generó el problema de que se interrumpía toda la actividad del *PIC*, y en el caso del motor de pulsos se detenía su giro, debido a que la secuencia de pulsos se veía interrumpida. Además, era necesario declarar un nuevo *endpoint* para lograr la interrupción del *PIC*, cada vez que se recibiera un dato de la *PC*, lo cual implicaba modificar el descriptor del *firmware*.

La programación estructurada en lenguaje C hizo un tanto difícil el control en tiempo real del proceso, debido a que se requiere un monitoreo constante de los sensores sin interrumpir el funcionamiento correcto de los motores, los cuales no siempre estarán funcionando simultáneamente, además de que, en general, son diferentes los parámetros para cada uno de ellos.

Este problema se resolvió gracias a que el compilador del microcontrolador cuenta con el *RTOS* (*Real Time Operating System*), que se utiliza para tener aplicaciones multitarea, y de este modo permite optimizar el programa, se simplifica el desarrollo de la aplicación, y mediante el uso de tareas del *RTOS* se reducen los errores de programación.

En las primeras pruebas que se realizaron, se aprovechó la alimentación de 5 V que se dispone del puerto *USB*, lo cual presentaba grandes ventajas al no necesitar una

fuente de poder adicional. Conforme se fueron integrando los elementos de la interfaz, se requirió una intensidad de corriente mayor del que el puerto *USB* puede proporcionar, por lo que se decidió emplear una alimentación externa.

El prototipo final se probó en diferentes *PCs* con sistema operativo *Windows XP*, en las que se instaló el *software* correspondiente, y en todas ellas su funcionamiento fue el esperado de acuerdo con las especificaciones de diseño.

6.2 RECOMENDACIONES

Una vez que el proyecto en su conjunto trabajó bien durante varias pruebas de laboratorio, se decidió probarlo en la aplicación para el que fue diseñado, con niños de Primaria.

Primero, los niños diseñaron diferentes juguetes con movimiento: cada uno construyó la maqueta de un juego mecánico de una feria de diversiones, entre las que hubo unas sillas voladoras, una rueda de la fortuna, un carrusel y una canoa.

Posteriormente, se adaptó a cada una de dichas maquetas alguno de los motores de la interfaz, para que el niño lo pudiera controlar con el empleo de los sensores a través de la interfaz gráfica diseñada en este trabajo.

Con respecto a cuestiones técnicas, durante esta actividad se pudieron observar varias características que se sugiere modificar para futuras versiones de la interfaz, como las que se describen a continuación.

Los parámetros del motor de pulsos podrían modificarse con objeto de tener un control más preciso, además de que los tiempos de duración definidos arbitrariamente podrían ser mayores. Sería deseable conjuntar las estructuras de las sentencias lógicas para que los niños puedan realizar programas con varias secuencias.

La interfaz gráfica despertó un gran interés en los niños, quienes fácilmente la pudieron manejar, pero aún así, queda pendiente el desarrollo de un lenguaje de programación con lenguaje natural que sea útil para los niños, y que sea más formal en cuanto a su estructura.

Este prototipo se realizó de manera artesanal, debido a la necesidad de realizar modificaciones según el resultado de las pruebas de funcionamiento. A futuro se sugiere el diseño de la tarjeta impresa de este circuito, lo cual le daría una mejor presentación, facilitaría su producción en masa, reduciría costos de fabricación a largo plazo, y fortalecería su robustez.

6.3 CONCLUSIONES

En el objetivo se estableció que la interfaz se conectara a cualquier computadora; se realizaron pruebas en varias PC con sistema operativo Windows XP, y en las cuales esta interfaz funcionó sin ningún contratiempo. En caso de que se quisiera emplear bajo Windows Vista, basta con deshabilitar la firma digital para los controladores, en el momento en que el sistema se carga en la PC.

La interfaz fue atractiva para los niños en todos los sentidos, cuando se les hizo la invitación a asistir a la prueba se entusiasmaron, durante la sesión el interés no disminuyó pues quisieron probar sus diseños con la interfaz impacientemente, y al emplear el programa de cómputo se emocionaron controlando sus creaciones, por lo que en cuestión lúdica fue todo un éxito.

Cuando los niños trabajaron con la interfaz, no faltaron accidentes como jalones a los cables o dejar caer la interfaz completa. Sin embargo, el prototipo no sufrió daños, e incluso siguió funcionando bien en todos los casos.

De las experiencias obtenidas durante la puesta en marcha de la interfaz, se concluyó que el prototipo realizado cumplió con los objetivos establecidos, de ser robusto, funcional y atractivo para los niños.

El crear una herramienta para la robótica pedagógica trae consigo una gran satisfacción desde el punto de vista profesional y personal.

En la construcción de este prototipo, se tuvo la oportunidad de aplicar y relacionar los conocimientos adquiridos en diferentes materias de la carrera, así como conjuntar la electrónica con la computación, corroborando que en ingeniería un proyecto por lo regular es multidisciplinario y por consiguiente, no es conveniente limitarse a una sola especialidad, sino estar siempre con la disposición de incursionar en otras áreas y adquirir de esta manera nuevas habilidades y conocimientos, propiciando el aprendizaje continuo.

BIBLIOGRAFÍA

[1] Peñuelas, R.U. Presentación *“Robots móviles”.* Consultada en EDUCAFI, Octubre 2009.

[2] Ruiz–Velasco, S. E. *Educatrónica: innovación en el aprendizaje de las ciencias y la tecnología.* Ediciones Diaz Santos. México 2007.

[3] Palacios, M.E. Microcontrolador PIC16F84 “Desarrollo de proyectos”. Ed Alfaomega.México 2004

[4] García, B. E. *Compilador C CCS y simulador PROTEUS para Microcontroladores PIC.* Alfaomega. México. 2008.

[5] C. Luden , “Lenguajes de programación: principios y práctica" . Thomson Learning. México 2004.

[6] PIC18F2455/2550/4455/4550 Data Sheet, Microchip, 2006.

[7] Monrroy C., A. I., “Desarrollo de una interfaz USB para el control remoto de sistemas electromecánicos”, Tesis profesional, México 2007.

[8] Buj, G.R. Procedimiento de diseño de circuitos digitales mediante FPGAS. Universidad de lleida. España. 2007. Recuperado noviembre 2009. Disponible en: http://www.recercat.net/bitstream/2072/3834/1/Buj.pdf

[9] Brochn, T. E. El bus USB (Universal Serial Bus). Trabajo realizado para la asignatura PEI (Periféricos e Interfaces) en la universidad de Valencia España. Recuperado marzo 2008. Disponible en:

http://usuarios.lycos.es/kurganz/datos_tecnicos/flujo.html

[10] Carletti,J. E. Sensores: Conceptos generales descripción y funcionamiento. Argentina. 2007. Recuperado enero 2009. Disponible en:

http://robots-argentina.com.ar/Sendores_general.htm

APENDICE

FIRMWARE

```
#include <18F4550.h>
#device ADC=8
#fuses
HS,NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,PLL5,CPUDIV1,VREGEN//HSPLL,L
VP
#use delay(clock=20000000)
#use rs232(baud=19200, xmit=PIN_C6, rcv=PIN_C7)

#define USB_HID_DEVICE     FALSE
#define USB_EP1_TX_ENABLE  USB_ENABLE_BULK
#define USB_EP1_RX_ENABLE  USB_ENABLE_BULK
#define USB_EP1_TX_SIZE    64
#define USB_EP1_RX_SIZE    8

#include <pic18_usb.h>
#include <PicUSB.h>
#include <usb.c>
#define dato2      recibe[2]
#define dato1      recibe[1]
#define modo       recibe[0]
```

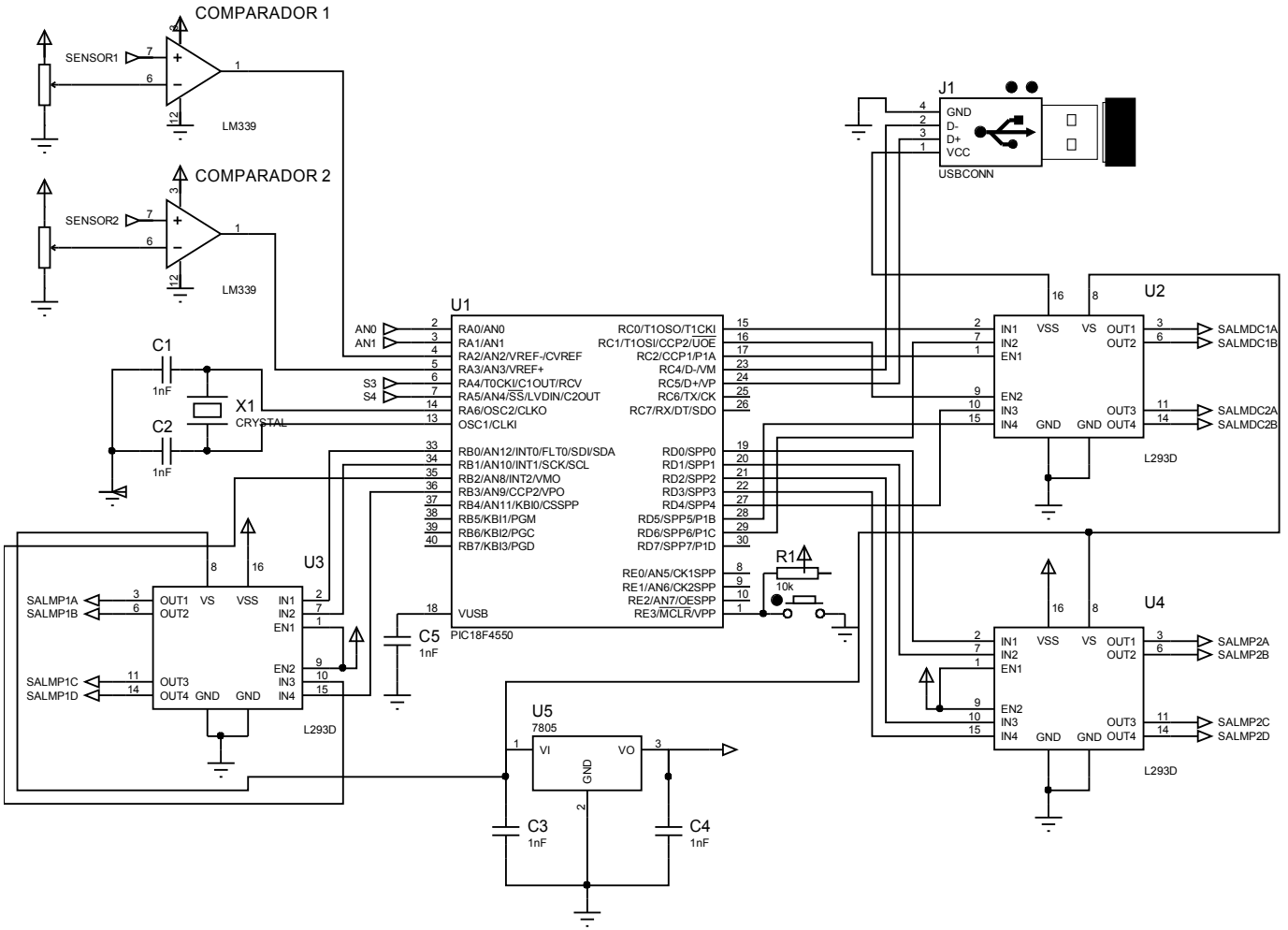
```
#BYTE TRISD=0XF95
#BYTE PORTD=0XF83
#use rtos(timer=1,minor_cycle=500us)

struct PORTD_BITS
  {
  BYTE MP: 4;
  BYTE MDC2 :   2;
  BYTE MDC1: 1;
  BYTE NO:    1;
  };
struct PORTD_BITS portd_bits;

int8 recibe[3];
int8 salida=0x33;
int8 valorADC[1], sensores[1];
int8 opcion, cuenta=0,cuenta2=0;
int1 dirMP, ddMP, dirMDC, ddMDC, MPon=0, MDCon=0,aux=0;
int1 dirMP2, ddMP2, dirMDC2, ddMDC2, MPon2=0, MDCon2=0;
int8  velMP=0,  velMDC;
int32 durMP, durMDC;
int8  velMP2=0,  velMDC2;
int32 durMP2, durMDC2;
int edomotor=1;

BYTE const posicion[4]={0b1100, 0b0110,0b0011,0b1001};
#task (rate=500us,max=500us)
void recibep();
//////////////       motor de pulsos  ///////////////
#task (rate=500us,max=500us)
void giroMP();
//////////////       motor de pulsos 2  //////////////
#task (rate=500us,max=500us)
void giroMP2();
//////////////       motor DC  ///////////////////
#task (rate=1ms,max=500us)
void giroMDC2();
//////////////       motor DC  ///////////////////
#task (rate=1ms,max=500us)
void giroMDC();

//////////////  Lectura entrada analógica  /////////
void senAn(int canal)
{
    set_adc_channel(canal);
```

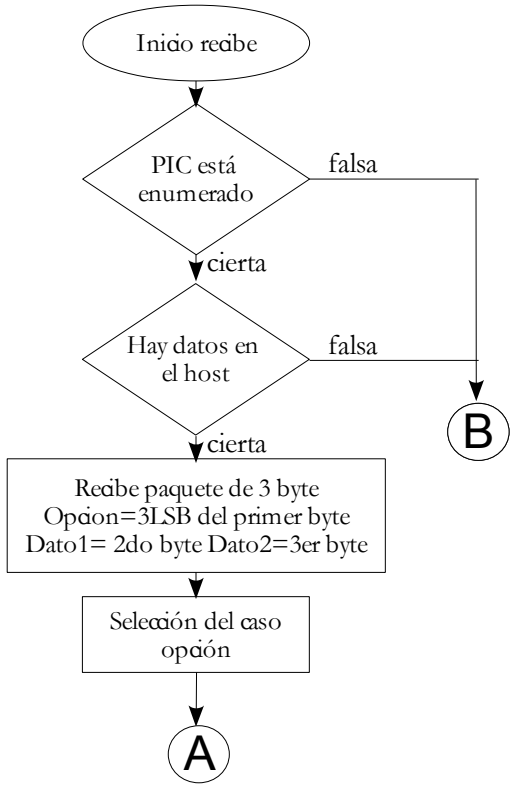
```
    delay_us(1000);
    valorADC[0]=read_adc();
    usb_put_packet(1,valorADC,1,USB_DTS_TOGGLE);
}

///////////////   lectura de los sensores digitales   //
void senDig(void)
{
    sensores[0]=0;
    if (input(PIN_A2)) bit_set(sensores[0],0);
    if (input(PIN_A3)) bit_set(sensores[0],1);
    if (input(PIN_A4)) bit_set(sensores[0],2);
    if (input(PIN_A5)) bit_set(sensores[0],3);

    usb_put_packet(1,sensores,1,USB_DTS_TOGGLE);
}


void main(void)
{
    setup_adc(ADC_CLOCK_INTERNAL);

setup_adc_ports(AN0_TO_AN1_ANALOG);
    TRISD=0X00;
    portd_bits=0x00;
    PORTD = portd_bits;
    output_C(0);
    output_low(PIN_B6);
    output_high(PIN_B7);
    printf("Inicializando\n");
    usb_init();
    printf("Inicializacion terminada\n");
    usb_task();
    printf("Esperando enumeracion\n");
    usb_wait_for_enumeration();
    output_low(PIN_C0);
    output_low(PIN_B7);
    output_high(PIN_B6);
    setup_timer_2(T2_DIV_BY_16, 127, 1);
    rtos_disable(giroMP);
    rtos_disable(giroMP2);
    rtos_disable(giroMDC);
    rtos_disable(giroMDC2);
    rtos_run();
}

void recibep()
{
    rtos_await(usb_enumerated());
        if (usb_kbhit(1))
    {
        usb_get_packet(1,recibe, 3);
        opcion=modo&15;

        switch (opcion)
        case 1:
            if (dato1!=0)
            {
                durMP=(dato1*25);
                velMP=dato2;
                dirMP=bit_test(modo,7);
                ddMP= bit_test(modo,6);
                MPon=1;
                rtos_enable(giroMP);
            }
            else
            {
                MPon=0;
                output_b(0);
                rtos_disable(giroMP);
            }
        break;
        case 2:
            if (dato1!=0)
```

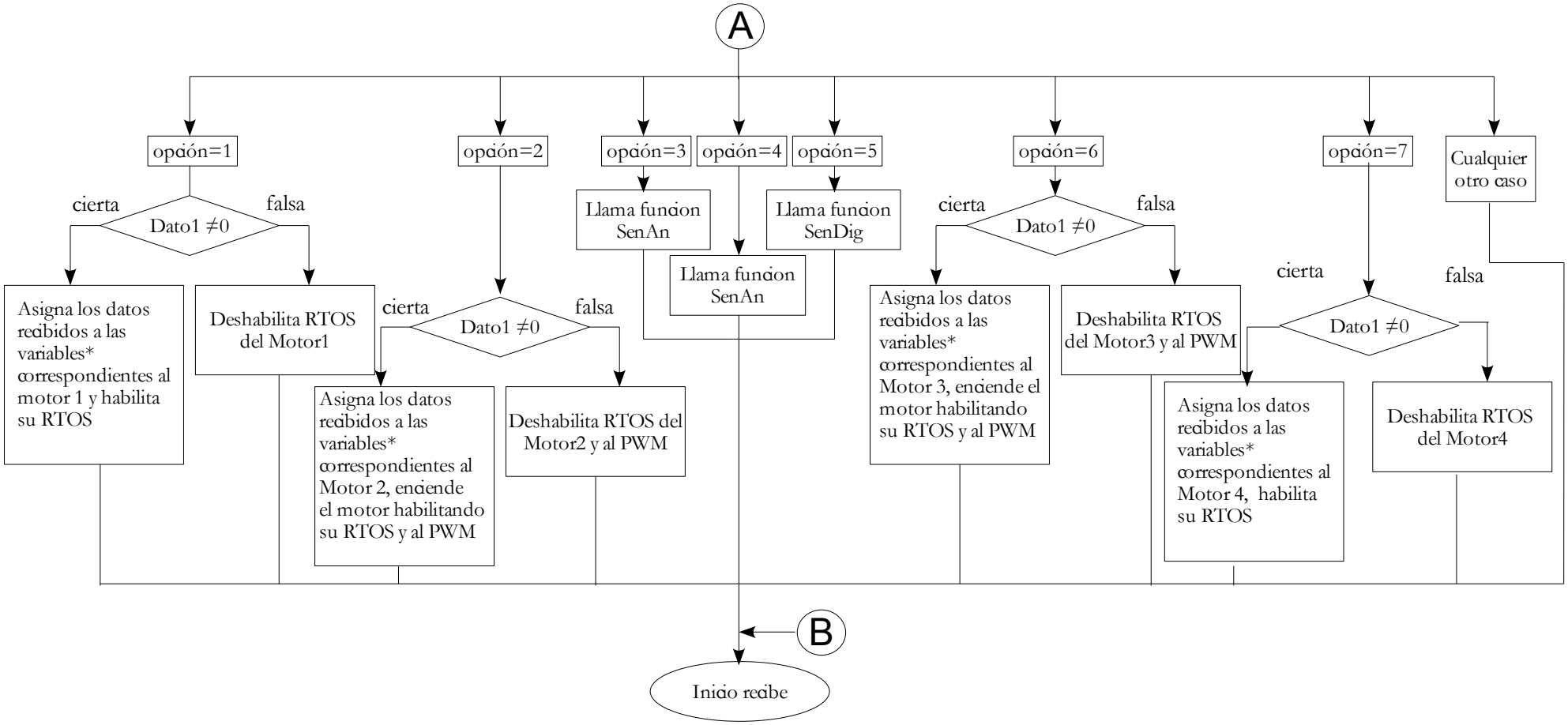
```
    {
        durMDC=dato1*1000;
        velMDC=dato2;
        ddMDC=bit_test(modo,6);
        rtos_enable(giroMDC);
        if (bit_test(modo,7))
         {
            portd_bits.MDC1=0;
            PORTD=portd_bits;
            output_high(PIN_C0);
        }
        else if (!bit_test(modo,7))
        {
            portd_bits.MDC1=1;
            PORTD=portd_bits;
            output_low(PIN_C0);
        }
        setup_ccp1(CCP_PWM);
        set_pwm1_duty(velMDC);
    }
    else
    {
        portd_bits.MDC1=0;
        PORTD=portd_bits;
        output_low(PIN_C0);
        setup_ccp1(CCP_OFF);
        rtos_disable(giroMDC);
    }
break;
case 3:
    senAn(0);
break;
case 4:
    senAn(1);
break;
case 5:
    senDig();
break;
case 6:
    if (dato1!=0)
    {
        durMP2=(dato1*25);
        velMP2=dato2;
        dirMP2=bit_test(modo,7);
        ddMP2= bit_test(modo,6);
        MPon2=1;
        rtos_enable(giroMP2);
    }
    else
    {
        MPon2=0;
        rtos_disable(giroMP2);
    }
break;
case 7:
    if (dato1!=0)
    {
        durMDC2=dato1*1000;
        velMDC2=dato2;
        dirMDC2=bit_test(modo,7);
        ddMDC2=bit_test(modo,6);
        rtos_enable(giroMDC2);
        if (dirMDC2==1)
        {
            portd_bits.MDC2=0x01;
            PORTD=portd_bits;
        }
        else if (dirMDC2==0)
        {
            portd_bits.MDC2=0x02;
            PORTD=portd_bits;
        }
        setup_ccp2(CCP_PWM);
        set_pwm2_duty(velMDC2);
    }
    else
    {
        portd_bits.MDC2=0;
        PORTD=portd_bits;
        setup_ccp2(CCP_OFF);
        MDCon2=0;
        rtos_disable(giroMDC2);
    }
break;
default:
break;
}
rtos_yield();
}
void giroMP()
```

```
{
    rtos_await( Mpon==1);
    ++cuenta;
    if (cuenta>velMP)
    {
        if (dirMP==1)  rotate_left(&salida,1);
        else   rotate_right(&salida,1);
        delay_us(800);
        output_b(salida);
        cuenta=0;
        if (ddMP==1)  --durMP;
        if (durMP==0) rtos_disable(giroMP);
    }
   rtos_yield();
}
void giroMP2()
{
    rtos_await( MPon2==1);
    ++cuenta2;
    if (cuenta2>velMP2)
    {
        if (dirMP2==1)
edomotor=((edomotor+1)&3);
        else edomotor=((edomotor-1)&3);
        delay_us(800);
        portd_bits.MP = posicion[edomotor];
        PORTD=portd_bits;
        cuenta2=0;
        if (ddMP2==1)  --durMP2;
        if (durMP2==0) rtos_disable(giroMP2);
    }
     rtos_yield();
}
void giroMDC2(void)
{
    rtos_await(ddMDC2==1);
    --durMDC2;
    if (durMDC2==0)
    {
        portd_bits.MDC2=0;
        PORTD=portd_bits;
        setup_ccp2(CCP_OFF);
        rtos_disable(giroMDC2);
    }
    rtos_yield();
}
 void giroMDC(void)
{
    rtos_await(ddMDC==1);
    --durMDC;
    if (durMDC==0)
    {
        portd_bits.MDC1=0;
        PORTD=portd_bits;
        output_low(PIN_C0);
        setup_ccp1(CCP_OFF);
        rtos_disable(giroMDC);
    }
    rtos_yield();
}
```

DIAGRAMA DE CONEXIONES




Inicio recibe
PIC está enumerado
falsa
cierta
Hay datos en el host
falsa
B
cierta
Recibe paquete de 3 byte
Opcion=3LSB del primer byte
Dato1= 2do byte Dato2=3er byte
Selección del caso opción
A


A
opción=1
opción=2
opción=3
opción=4
opción=5
opción=6
opción=7
Cualquier otro caso
cierta
falsa
Dato1 ≠0
Asigna los datos recibidos a las variables* correspondientes al motor 1 y habilita su RTOS
Deshabilita RTOS del Motor1
Asigna los datos recibidos a las variables* correspondientes al Motor 2, enciende el motor habilitando su RTOS y al PWM
Deshabilita RTOS del Motor2 y al PWM
Llama funcion SenAn
Llama funcion SenDig
Llama funcion SenAn
Asigna los datos recibidos a las variables* correspondientes al Motor 3, enciende el motor habilitando su RTOS y al PWM
Deshabilita RTOS del Motor3 y al PWM
Asigna los datos recibidos a las variables* correspondientes al Motor 4, habilita su RTOS
Deshabilita RTOS del Motor4
B
Inicio recibe

PRUEBA DE LA INTERFAZ

La prueba de la interfaz se realizó con niños de edad primaria de 6 a 8 años. La actividad estuvo dividida en tres etapas: diseño, armado y operación mediante la interfaz.

La temática propuesta a los niños fue la creación de una feria de diversiones y la aceptaron con agrado. En la etapa de diseño, se les pidió que pensaran en juegos mecánicos que tienen movimiento rotatorio, después que eligieran alguno y lo dibujara en papel.

Fotografía 1. Niños plasmando sus ideas en papel

La siguiente etapa fue construir el juego mediante material de reuso como botellas de plástico, tapas, envases, cajas, estambre, etc.. Se presentó un poco de dificultad sobre todo en los niños más chicos al tener que pasar su diseño de papel a maqueta. Sin embargo fue una gran oportunidad para el desarrollo de la creatividad, ya que ellos eligieron los materiales a utilizar.

Fotografía 2. Construyendo las maquetas

Por último se adaptó cada una de las maquetas a los motores de la interfaz. Se probó con el programa “Control independiente” para conocer como se constituía la interfaz electrónica, después se dejo que los niños trabajaran libremente con el programa “Principios de programación”

Fotografía 3. Probando su maqueta con la interfaz

Gracias a los niños Andrés Daniel, Dulce, Isabel, Jesús, Kevin, Natalia, Sara y Xantil que participaron en la puesta en marcha de la interfaz.