UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

Facultad de Ingeniería

DISEÑO Y CONSTRUCCION DE ROBOTS MOVILES UTILIZANDO MICROCONTROLADORES

T E S I S

Que para obtener el título de:

INGENIERO MECANICO ELECTRICISTA
(AREA ELECTRICA ELECTRONICA)

p r e s e n t a

RUBEN ANAYA GARCIA

Director de Tesis:
Dr. Jesús Savage Carmona

México, D. F. **1998**

DEDICATORIAS

A mis padres: Herlinda y Lidio; por darme la vida y la oportunidad de comenzar mi preparación académica, además de permitirme desarrollarme como creyera conveniente.

A Verónica, por ser la mujer y parte importante de mi vida (aunque tu no lo creas).

A mis hijos: Rubén Alexis y Verónica Aleydis; por que desde que nacieron, se convirtieron en la razón principal para enfrentarme a las adversidades.

A mis hermanos: Francisco, Ismael, Oscar y Marisol, por el apoyo e interés que muestran en las actividades que emprendo, así como a sus esposas.

A Rosaura, Maribel , Nalleli, Francisco Javier y Paola, con la esperanza de ver que se realicen académica, personal y profesionalmente.

A mis compañeros de trabajo, amigos y alumnos que de una u otra forma me ayudaron y estuvieron pendientes en el desarrollo de mi tesis.

A mi director: Dr. Jesús Savage Carmona, por que sin su invaluable supervisión, no podría llevar a buen termino este trabajo.

Rubén Anaya García

AGRADECIMIENTO

A Dios

Por darme la oportunidad de contar
con mis seres queridos.

A mi madre Herlinda

Por que siempre estuviste al pendiente que nada nos faltara y me apoyaste sin esperar nada a cambio.

A mi padre Lidio

Por darme la base para enfrentarme a la vida y por que nunca escatimaste tu esfuerzo para darnos lo necesario para nuestra educación.

"Gracias por esperar tanto este momento"

Rubén Anaya García

AGRADECIMIENTO

A Verónica

Por que tu más que nadie sabes lo que me ha costado llegar a la conclusión de la carrera.

Por aceptar que las metas se tienen que alcanzar a base de sacrificios y trabajo diario.

A Rubén Alexis
Verónica Aleydis

Que han llenado de alegría y esperanza nuestro hogar.

Por que cada vez que los observo jugar y estudiar, me lleno de más motivación para continuar luchando por ustedes.

Y por que sin tener culpa alguna, han sacrificado algunos de sus juguetes para mi s experimentos.

Rubén Anaya García

AGRADECIMIENTO

A los integrantes del jurado:

Profesores distinguidos de la Facultad de Ingeniería, por tener la distinción para con mi persona y aceptar formar parte de mi jurado, además de brindarme su amistad:

M. I. Aurelio A. Millán Nájera

Ing. Alberto Templos Carbajal

Dr. Jesús Savage Carmona

Dr. Carlos Rivera Rivera

Ing. Marco A. Morales Aguirre

Rubén Anaya García

AGRADECIMIENTO

Al Dr. Jesús Savage Carmona

Por que siempre estuvo al pendiente del desarrollo de mi tesis, aportando sus gran experiencia para el buen inicio y culminación de este trabajo.

Además por brindarme la oportunidad de integrarme a esta área tan basta e interesante.

A la Universidad Nacional Autónoma de México, Facultad de Ingeniería y a los profesores que la integran

Por transmitirme sus conocimientos y experiencia para la obtención de mi formación profesional

Rubén Anaya García

Introducción

El trabajo que a continuación se presenta, contendrá la información necesaria para la construcción de un robot móvil, con múltiples aplicaciones a ejecutar, con conceptos generales sobre la robótica, los microprocesadores y los microcontroladores, así como generalidades de los sensores más empleados en el diseño de los robots móviles y de los diferentes arreglos controladores de motores de corriente directa; además se explica la manera de emplear y programar tanto en lenguaje ensamblador como en lenguaje C al microcontrolador MC68HC11, aunque es de mencionar, que por la versatilidad de los circuitos empleados es posible su adaptación con otro tipo de microcontrolador.

Así mismo presentará los circuitos de interfaz de los sensores con el microcontrolador MC68HC11 y algunas de las tareas y aplicaciones que se pueden ejecutar con un robot como el que se pretende sea construido.

También existe una sección de apéndices, en los cuales se abordarán con mayor detalle la información contenida dentro del trabajo general, de tal forma que se disponga dentro de este ejemplar de la mayoría de las herramientas expuestas.

El objetivo principal de esta tesis es explicar de una manera sencilla el procedimiento para que una persona con deseos e interesada en esta área, pueda construir un robot móvil.

INDICE

Apéndices

Bibliografía

CAPITULO No. 1
Objetivo y Generalidades

1.1 Objetivo

El objetivo de este trabajo es presentar los elementos necesarios para la construcción y programación de un robot móvil, abarcando conceptos importantes de la robótica, los microprocesadores, microcontroladores, la forma de operación de los sensores, los dispositivos electrónicos y su adaptación con el sistema, así como la programación del mismo. Para este caso en particular, se desarrolla en base al MC68HC11E9, porque contiene varios subsistemas internamente y es de fácil programación; como consecuencia ahorra espacio y la cantidad de dispositivos que se tendrían que agregar; por lo tanto reduce su costo sustancialmente (sus características se detallan en el apéndice A).

Se diseñarán interfaces de control y sensado, tanto de fácil construcción como de adaptación a otro tipo de microcontrolador. De esta manera se pretende sirva como guía para que personas interesadas en esta área, tengan la oportunidad de construir o diseñar su propio robot.

1.2 Generalidades

El diseño y la práctica con los robots móviles, tiene desde hace algunos años un gran auge y más adeptos en esta disciplina, logrado por la diversidad de modelos y actividades que pueden realizar. El campo de aplicación se ha extendido en casi todas las áreas tecnológicas; para la comunidad científica es un medio de gran ayuda en las investigaciones que realizan; en el aspecto académico se contemplan en planes de estudio del área físico matemáticas materias relacionadas con la robótica y en el área comercial es posible adquirir algunos modelos sencillos como entretenimiento.

Todo comenzó con la idea y sueño, de contar con máquinas capaces de reproducir los movimientos de los seres humanos y de los animales, además de servir como medio de enlace entre un medio hostil y el hombre. El desarrollo de esta actividad es de carácter multidisciplinario, compartiéndose en áreas tan diversas y afines, como la automotriz, la mecánica, la electrónica, y la computación. Como consecuencia; una investigación en esta disciplina, exige una estrecha colaboración entre ellas.

La llegada de la electrónica digital y de la microelectrónica, ha logrado la reducción en costos y tamaños de los robots, con una funcionalidad y aplicación, cuyo único limite es la imaginación.

La robótica puede considerarse que hace referencia a una automatización importante de numerosos sectores de la actividad humana, en los cuales se ha estimado imprescindible, hasta hace poco tiempo la presencia del hombre. Los progresos extraordinarios realizados, particularmente en informática, asociados a las necesidades económicas del día en los países industrializados, permiten por un lado, construir robots y por otro verlos implantar en numerosas ramas de actividades.

1.2.1 Los campos de la robótica

El campo de aplicación de la robótica influye profundamente la forma y las propiedades de las máquinas y robots; puede facilitarse la comprensión considerando tres grandes campos de aplicación:

I) El campo de la producción:

Concretamente aquí es donde los industriales han desarrollado el máximo esfuerzo, ya que en la utilización de los robots ven numerosas ventajas; entre ellas la disminución de la mano de obra.

La asociación entre los robots y las máquinas, aportan dos ventajas con relación a los modos de producción tradicionales:

a) La automatización casi integral de la producción, que puede acompañarse de una mejor calidad del producto terminado, de una mayor fiabilidad en el mantenimiento de la calidad, y de una mejor adaptación de la cantidad producida a la demanda.

b) La rapidez de reconfiguración de la unidad de producción cuando se pasa de la fabricación de un producto, a la de un producto similar.

II) El campo de la exploración:

Se trata de un campo diferente, se quieren ejecutar trabajos en un lugar al que el hombre no puede permanecer, porque el medio es de difícil acceso y peligroso; es el caso para:

a) El medio submarino;
b) El medio espacial;
c) El medio irradiado de las centrales nucleares, etc.

La robótica permite ensayar dos tipos de solución para estas intervenciones:

a.1) El robot autónomo:

Es el que se va a enviar al medio hostil, programándole su misión; por ejemplo: recoger ciertas piedras y estudios en el planeta Marte, o bien inspeccionar las soldaduras de una central nuclear, etc.

a.2) La teleoperación (llamada también telepresencia):

Consiste en enviar una maquinaria (un robot que se llama maquina esclava), al medio hostil y poder controlarla a distancia desde otro lugar llamado puesto maestro, al mando del cuál se encuentra un hombre, que es el operador.

Es pues el hombre quien efectúa todas las tareas de reflexión y de la activación de los movimientos de la maquinaria esclava, esto es posible realizarlo con un sistema de visión que por lo general son cámaras de vídeo.

III) El campo de la asistencia individual:

El campo donde se desarrolla, es principalmente en la robótica médica, que permite mejorar las condiciones de vida de los paralíticos (parapléjicos y tetrapléjicos) o de los amputados.

1.2.2 Origen de la palabra robot

El término robot parece que empezó a utilizarse hacia los años 1920-1930, a raíz de una obra de teatro del autor checo Karel Tschapek titulada R.U.R. (Rossum´s Universal Robot). Esta obra pone en juego pequeños seres artificiales antropomorfos, que responden perfectamente a las ordenes de su maestro. Estos seres llevan el nombre de robot, del checo *robota (siervo),* idéntico al término ruso y que significa *trabajo.*

Se consideran las siguientes definiciones para un robot:

a) La de Oxford English dictionary:

Un aparato mecánico que se parece y hace el trabajo de un ser humano.

b) La del Robot Institute of America:

Un manipulador reprogramable y multifuncional concebido para transportar materiales, piezas, herramientas o sistemas especializados, con movimientos variados y programados, con la finalidad de ejecutar tareas diversas.

1.2.3 Propiedades características del robot

Más que buscar una definición, se presentan dos propiedades que caracterizan a un robot:

a) La versatilidad; es la potencialidad (posibilidad) estructural (mecánica) de ejecutar tareas diversificadas y/o ejecutar una misma tarea diversificada.

b) La autoadaptabilidad al entorno; un robot debe, por si solo, alcanzar su objetivo (la ejecución de una tarea), a pesar de las perturbaciones imprevistas (pero limitadas) del entorno, a lo largo de la ejecución de la tarea.

Un robot operacional puede representarse por cuatro entidades unidas entre sí como se indica en la figura 1-1.

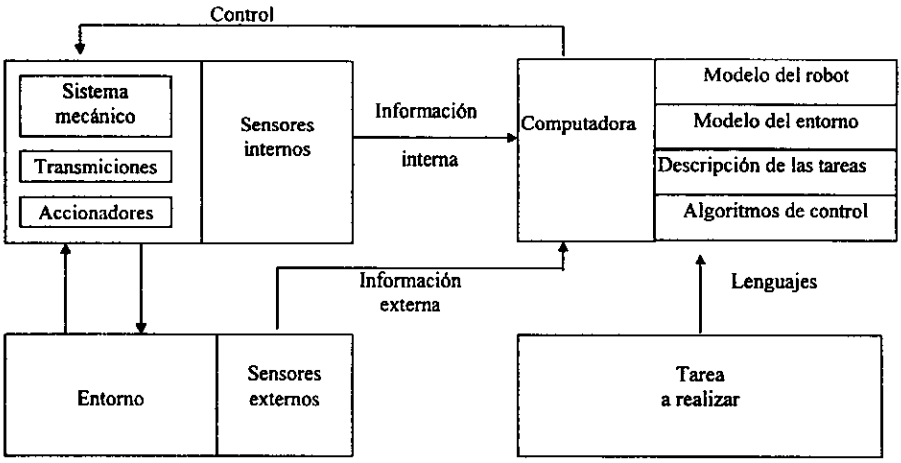


Figura 1-1. Descripción de un robot en funcionamiento.

1) *El sistema mecánico articulado*: Dotado de sus motores (que se llaman también accionadores o actuadores); estos pueden ser eléctricos, hidráulicos o neumáticos. Arrastran a las articulaciones del robot, mediante transmisiones, también de diversa naturaleza (cables, cintas, correas con muescas, engranes, tornillos sin fin, etc.). Para conocer en todo instante la posición de las articulaciones, se recurre a los captadores (potenciométricos, codificadores ópticos, etc.). Estos captadores se denominan propioceptivos, por analogía con el sistema humano.

2) *El entorno*: Es el universo en el que está sumergida la primera entidad. Para los robots con puesto fijo, se reduce a lo que se encuentra en el espacio alcanzable del robot, definido por el volumen barrido cuando éste pasa por todas las configuraciones posibles. En este entorno, el robot encontrará obstáculos que debe evitar y objetos de interés, es decir sobre los que debe actuar.

En consecuencia, existe una interacción entre la primera entidad (el robot físico) y el entorno; se toman informaciones sobre el estado de ese entorno (comparable al estado del robot), mediante los captadores que denominamos exteroceptivos (es decir, que permiten situar lo que hay en el exterior del robot físico con relación a el); encontrándose en esta rama, las cámaras, detectores de fuerza, captadores de proximidad, captadores táctiles, etc.

3) *Las tareas a realizar*: Es el trabajo que se desea que haga el robot . Es preciso por lo tanto poder describirlo; esto se hace mediante lenguajes que pueden ser por gestos (se enseña al robot lo que debe hacer), orales (se le habla) o por escrito (se le escribe en un lenguaje que comprenda).

4) *El cerebro del robot*: Es el órgano de tratamiento de la información, para los robots menos evolucionados casi siempre es un autómata programable. En el caso de los robot avanzados es una computadora, microprocesador o microcontrolador; el cerebro del robot cuenta con los siguientes elementos:

a) *Un modelo del robot físico*; es decir, las relaciones entre las señales de excitación de los accionadores y de los desplazamientos que son consecuencia de ellas.

b) *Un modelo del entorno*; es decir, una descripción de lo que se encuentra en el espacio que puede alcanzar, por ejemplo, las zonas que no debe atravesar ya que hay obstáculos.

c) *Programas*; los cuales le permiten comprender las tareas que se le pide que realice, además de controlar el robot físico, con el fin de que éste ejecute lo que debe; son algoritmos de control. El microprocesador en todo instante:

i) Percibe el estado del robot gracias a la información interna;

ii) Percibe el estado del entorno gracias a la información externa;

iii) Recurre a diversos modelos y programas registrados;

iv) Genera una orden (es decir, las señales de potencia de los accionadores), que hace progresar al robot físico hacia la ejecución correcta de la tarea que se le ha pedido.

1.3 Prototipo en diseño

La figura 1-2 muestra un robot móvil que tiene como base a un microcontrolador, quien contará con diversos sensores y actuadores; en los capítulos siguientes se analizarán los más adecuados para nuestro diseño y las posibles opciones con los que podemos contar.

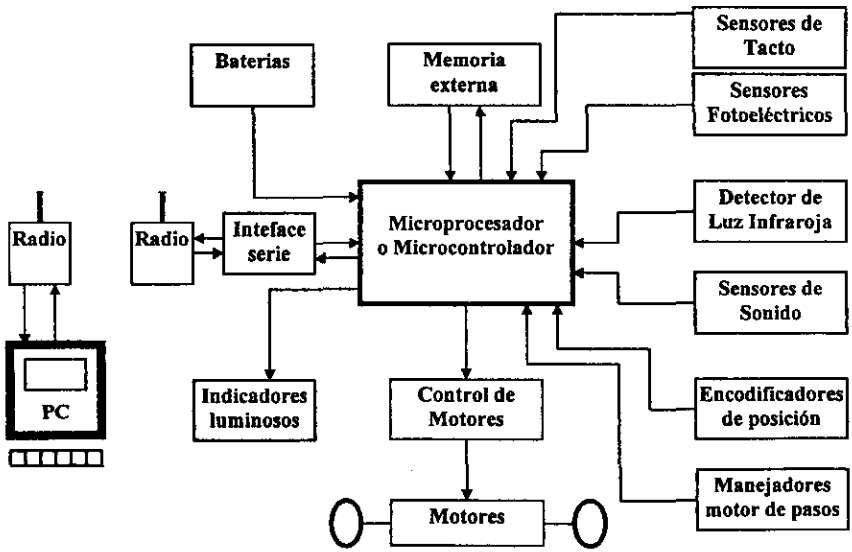


Figura 1-2. Diagrama general del robot móvil.

CAPITULO 2
Microprocesadores y Microcontroladores

Es difícil dar una definición exacta de lo que es un microprocesador, sobre todo porque en los últimos años el desarrollo de la electrónica en esta área ha dado lugar de un sin número de estos dispositivos, diseñados con enfoques hacia diversos campos de aplicación. Sin embargo todos los microprocesadores tienen ciertos atributos comunes que los distinguen de otros elementos electrónicos.

Un microcontrolador y un microprocesador son una combinación de dispositivos y circuitos digitales, que pueden realizar una secuencia programada de operaciones, a la cuál se le llama programa, con un mínimo de intervención humana, el programa lo ejecuta sobre datos que recibe, de los cuales obtiene resultados.

Las características principales de un microprocesador son su *universalidad* y su *programabilidad* que hacen de él un dispositivo tan versátil y poderoso, que puede utilizarse como elemento inteligente o *cerebro* en aplicaciones que van desde una computadora personal hasta un rastreador de satélites o un sistema de control de los robots móviles.

2.1 Conceptos Básicos de los Microprocesadores

Esencialmente, un microprocesador es un circuito de alta escala de integración (LSI), compuesto de muchos circuitos simples como son *flip-flop, contadores, registros,* decodificadores, comparadores, etc. Todos ellos dentro de la misma pastilla de silicio, de modo que el microprocesador puede ser considerado un dispositivo lógico de propósito general o universal.

2.1.1 Primeras definiciones

La programabilidad se refiere a la capacidad que tiene un microprocesador para que su función sea definida en un programa. El programa consta de una serie de instrucciones *relacionadas, ejecutadas secuencialmente* (una a la vez) por el microprocesador y que pueden implicar operaciones lógicas o aritméticas. Las instrucciones se especifican por medio de un código especial que constituye el lenguaje del microprocesador.

De lo anterior se puede deducir la existencia de dos géneros diferentes de elementos que se complementan y que juntos delimitan el concepto del microprocesador como un dispositivo útil. El primero de ellos lo forman el circuito mismo, sus componentes electrónicos, sus alambres de interconexión y en general todo aquello determinado por la configuración física del circuito integrado. A este género se le conoce como el **HARDWARE** o soporte físico del microprocesador, ya que su misma naturaleza no admite modificaciones. Por otro lado, se encuentran el código de instrucciones, los algoritmos y los programas que determinan el comportamiento del microprocesador, los cuales pueden ser alternados cuantas veces sea necesario, con el fin de adaptar y optimizar el desempeño del dispositivo a algún campo de aplicación en particular. Este segundo género recibe el nombre de **SOFTWARE** o soporte lógico.

2.2 Unidades funcionales básicas de un microprocesador

Aun cuando al microprocesador se considera un complejo sistema de microprocesamiento, constituido por una gran variedad de circuitos interrelacionados, es posible agrupar estos circuitos de acuerdo a la finalidad del trabajo específico que desempeñan dentro del sistema. Hay otras unidades funcionales que también pueden existir en un microprocesador; éstas son las Memoria de datos y los Puertos de entrada/salida.

Para que a un circuito se le pueda dar el nombre de microprocesador, debe contener en una sola pastilla, al menos los siguientes sistemas: la unidad de control, la unidad aritmético/lógica y registros internos.

Cuando un dispositivo posee únicamente las tres unidades funcionales básicas, se le conoce como UNIDAD DE PROCESAMIENTO CENTRAL (CPU) o simplemente MICROPROCESADOR (MP). Si un mismo circuito integrado tiene, además de las unidades básicas, otras tales como memoria (de programas y/o datos) y puertos, este circuito ya no se considera estrictamente un microprocesador, porque las unidades que contiene le dan la capacidad de una computadora. Por esta razón a estos microprocesadores se les llama también MICROCOMPUTADORAS EN UNA PASTILLA (Single Chip Microcomputers) o MICROCONTROLADOR.

En la figura 2-1 se muestra un diagrama de bloques de un microprocesador y de una microcomputadora en una pastilla.

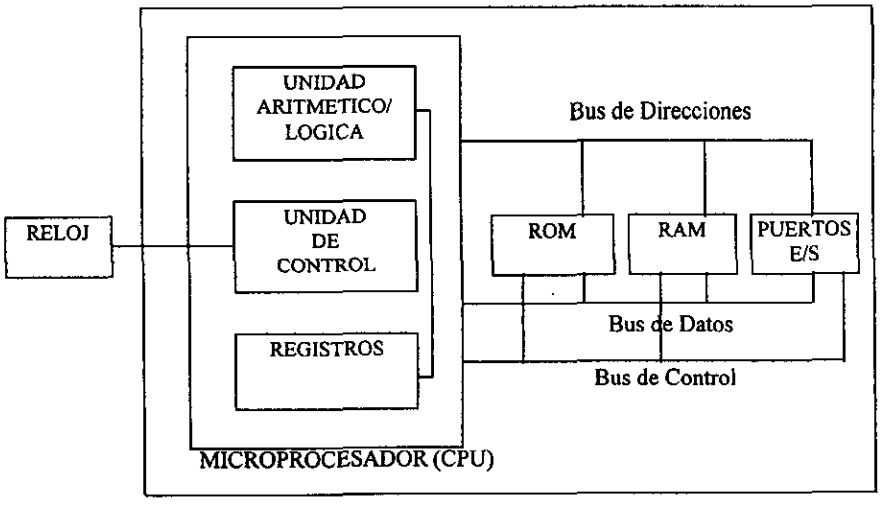


Figura 2-1. Diagrama de bloques de un microprocesador y de una microcomputadora en una sola pastilla.

A continuación se describirán las unidades funcionales que encuentran dentro de la CPU.

2.2.1 La unidad de control

La circuitería de control es la unidad funcional primaria dentro del microprocesador. Utilizando señales de reloj, la unidad de control mantiene la secuencias de eventos apropiada para llevar a cabo cualquier tarea de procesamiento; es decir, el microprocesador es un dispositivo síncrono.

La actividad fundamental de un microprocesador, regulada por la unidad de control, es cíclica, consiste en la búsqueda y obtención de datos e instrucciones, así como de la ejecución secuencial de estas últimas. Después de que una instrucción ha sido obtenida y decodificada, la circuitería de control envía las señales apropiadas a dispositivos internos como externos a la CPU.

Para iniciar la acción de procesamiento indicada por la instrucción. Frecuentemente la unidad de control es capaz de responder a señales externas que alteran el estado del microprocesador, ya sea interrumpiendo temporalmente su funcionamiento o provocando la ejecución de instrucciones especiales.

La parte más importante de la unidad de control lo constituye el GENERADOR DE CICLO DE MAQUINA (GCM), que se encarga de producir las señales de control, derivándolas de un reloj u oscilador maestro como referencia; la figura 2-2 presenta el diagrama de bloques de la unidad de control y su relación con otros elementos de la CPU.

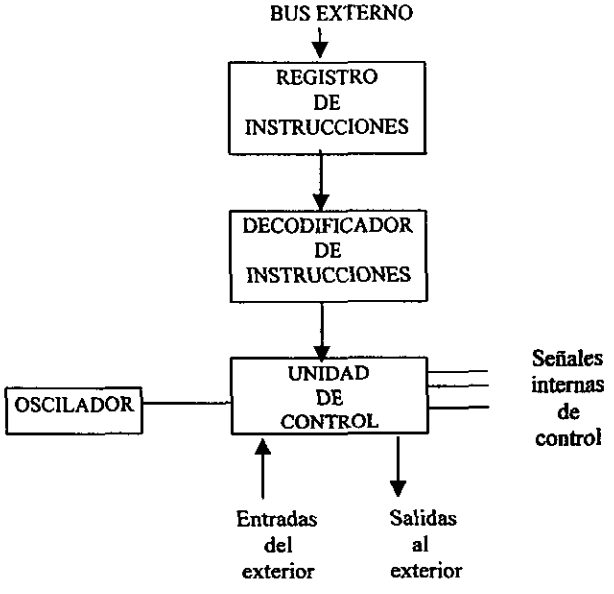


Figura 2-2. Unidad de Control.

2.2.2 Unidad aritmético/lógica

Todos los microprocesadores contienen una unidad aritmético/lógica que con frecuencia se conoce como ALU (Arithmetic/Logic Unit). La ALU como su nombre lo indica, es la parte del microprocesador que lleva a cabo las operaciones aritméticas y lógicas en los datos binarios (figura 2-3). Algunas de ellas se aplican sobre dos operandos, mientras otros requieren solo uno.

La ALU generalmente ejecutará las siguientes operaciones:

1.- Suma aritmética;
2.- Funciones lógicas AND, OR, EXOR;
3. Complemento;
4.- Rotación hacia la derecha o izquierda, etc.

Además, la ALU contiene un conjunto de flip-flops llamados BANDERAS (flags), los cuales guardan información relacionada con el resultado de una operación aritmética o lógica.

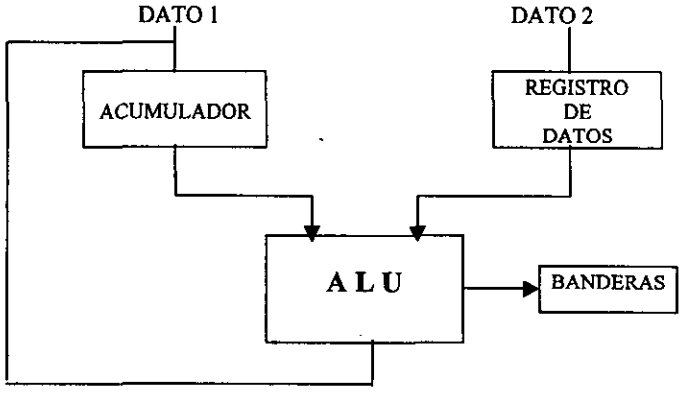


Figura 2-3. Unidad Aritmético/Lógica.

2.2.3 Los registros internos

Los registros internos son unidades de almacenamiento temporal dentro de la CPU, algunos de ellos tienen usos específicos, mientras que otros son de propósito general.

Para llevar la cuenta de cuál instrucción es la que debe ejecutarse en seguida, la unidad de control mantiene un registro con la dirección de la siguiente instrucción en el programa. A este registro se le llama CONTADOR DE PROGRAMA (program Counter) o PC. El microprocesador actualiza el contenido del PC incrementándolo cada vez que obtiene una instrucción de la memoria.

Una de las entradas de control al microprocesador es la entrada de RESET (restablecer); cuando se restablece el microprocesador, la unidad de control carga el contador del programa con ceros. Este valor inicial establece la dirección de memoria de donde se va a obtener la primera instrucción.

Después que se ha obtenido una instrucción de la memoria, la CPU la almacena en un registro conocido como REGISTRO DE INSTRUCCIONES (Instruction Register) o IR. La instrucción almacenada en el IR es decodificada y usada para activar una de varias líneas. Cada línea representa un conjunto de actividades asociadas con la ejecución de una instrucción particular. El dispositivo que traduce la instrucción en acciones concretas es el DECODIFICADOR DE INSTRUCCIONES (Instruction Decoder).

La primera palabra de una instrucción es el código de operación para esa instrucción. El CODIGO DE OPERACION indica a la unidad de control las operaciones requeridas en la ejecución de la instrucción.

Las instrucciones de un microprocesador frecuentemente requieren más información de lo que puede contener una sola palabra de memoria. Por lo tanto, es común que las instrucciones consten de dos o tres palabras.

Las otras palabras son datos que representan ya sea una dirección o una constante. Después de que el código de operación se ha leído de la memoria y puesto en el registro de instrucciones, su decodificación indica si la instrucción requiere información adicional. Las partes de una instrucción de varias palabras están contenidas en localidades sucesivas de la memoria.

Existe un registro íntimamente relacionado con la ALU, denominado ACUMULADOR. Generalmente el acumulador contiene uno de los operandos que serán manipulados por la ALU y, también muy frecuentemente, el resultado de la operación se deposita en ese registro, reemplazando a uno de los operandos originales.

1.2.4 La memoria del programa

Aunque anteriormente se dijo que la memoria no era parte integrante del microprocesador considerándolo como CPU, es conveniente mencionar la memoria del programa, puesto que sín ella el microprocesador se convierte en un dispositivo inservible.

La MEMORIA DEL PROGRAMA es una memoria de lectura solamente. Como por ejemplo una ROM o una EPROM que contiene el programa, es decir, la secuencia de instrucciones que va a determinar las tareas que realice el microprocesador. En algunos casos la memoria del programa también almacena parámetros o tablas de datos que no sufren modificaciones. Usualmente, las EPROM se utilizan durante la etapa de desarrollo del prototipo del sistema, para así poder depurar y alterar el programa.

La ROM se prefiere cuando el sistema ya se encuentra en producción y se ha llegado a la versión final del programa.

2.3 Conceptos Adicionales de Microprocesadores

Con frecuencia un microprocesador contiene un número de registros adicionales que no tienen asignada ninguna función en particular, sino que más bien se usan el tareas generales como lugares de almacenamiento temporal para guardar operandos o resultados intermedios.

2.3.1 Los registros internos de propósito general

La disponibilidad de registros de propósito general elimina la necesidad de mover los datos de un lado a otro entre la memoria y el acumulador, mejorando así la velocidad de procesamiento así como la eficiencia.

2.3.2 La memoria de datos

La memoria de datos es una memoria de lectura escritura (RAM), cuyo objetivo es permitir el almacenamiento temporal de datos o *programas de aplicación. Por sus características de* lectura/escritura, la información que reside se puede alterar fácilmente; sin embargo se pierde cuando se apaga la fuente de alimentación.

La mayoría de datos se puede considerar como una extensión de los registros internos de propósito general, ya que cada una de sus localidades es precisamente un registro. La diferencia está en que los registros de la memoria de datos son externos al microprocesador, y por lo tanto la velocidad de la transferencia de datos es más lenta.

2.3.3 Los puertos de entrada/salida

Los puertos de entrada/salida son los circuitos que se encargan de establecer el contacto del microprocesador (CPU) con el mundo exterior.

Los puertos se conectan a dispositivos periféricos que generan información para ser procesada por la CPU (dispositivos de entrada) o que aceptan datos provenientes del microprocesador y los transforma en información significativo para el mundo exterior (dispositivos de salida). Estos periféricos incluyen una gran variedad de elementos electrónicos y electromecánicos, cuya complejidad va desde unos simples interruptores y lámpara indicadoras, hasta complejos sistemas de adquisición de datos, pantallas de vídeo, etc.

2.3.4 Los buses de interconexión

Un microprocesador se comunica con los otros circuitos del sistema a través de grupos de líneas o buses de interconexión. Con base en el tipo de información que conducen, las líneas de interconexión se pueden agrupar en tres buses: datos, direcciones y control.

El BUS DE DATOS es un conjunto de líneas bidireccionales, que transportan información del microprocesador hacia la memoria o puertos y de éstos hacia el microprocesador.

El bus BUS DE DIRECCIONES es unidireccional; es decir, por él solamente circula información proveniente del microprocesador. Comprende a las líneas que transmiten una dirección generada por el CPU, la cuál selecciona a un puerto o a una localidad de memoria. Esta dirección especifica el origen o destino de la información que transmitirá por el bus de datos.

La sincronización y el sentido de la transferencia de información es el bus de datos (hacia dentro o hacia afuera del microprocesador) y el tipo de transferencia se indica por medio de las señales de control originadas por la CPU. Todas ellas conforman el llamado BUS DE CONTROL. Cada una de las señales en el bus de control es unidireccional. Algunas de ellas son salidas del microprocesador, mientras otras son entradas a este.

2.4 Clasificación de los microprocesadores

Existen dos criterios principales para la clasificación de los microprocesadores, uno se basa en la longitud de palabra y el otro es la tecnología de fabricación.

La longitud de palabra se refiere al número de bits que puede procesar simultáneamente un microprocesador y está determinada por su arquitectura, es decir, por el tamaño de los registros, de la ALU y de sus buses internos. La longitud de palabra ha crecido a través de los años, desde 4 bits del primer microprocesador hasta 32 bit de los microprocesadores más recientes.

En lo que toca a la tecnología de fabricación, los primeros microprocesadores se implantaron con tecnología PMOS; sin embargo, actualmente la tecnología de fabricación más difundida es la NMOS. Ultimamente se ha desarrollado bastante la tecnología CMOS para dispositivos con bajo consumo de energía.

2.5 Características adicionales de los microcontroladores

De acuerdo a la tecnología y funcionalidad de un microcontrolador, agrega en la mayoría de los casos, los subsistemas del Temporizador, Convertidor A/D, Comunicación Serial Síncrona y Asíncrona, Sistemas de Puertos Paralelos y un manejo de instrucciones más extenso.

La característica principal de un microcontrolador, es la inclusión en un chip de la gran mayoría de recursos, necesarios para controlar el funcionamiento de dispositivos o instrumentos, lo que es permitido por los subsistemas que lo conforman y hace de fácil adaptación la expansión a un sistema con mayores recursos.

Un diagrama típico de un microcontrolador es el que se muestra a continuación, donde se trata del diagrama interno del microcontrolador MC68HC11E9, del cuál se proporcionará información detallada de su forma de operación en el apéndice A.

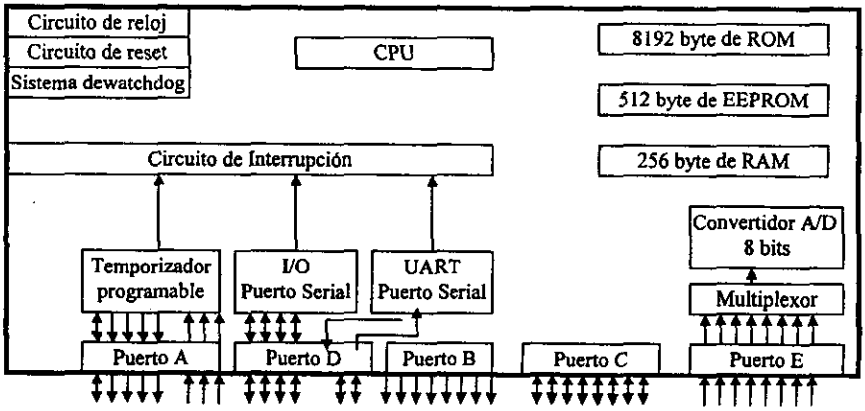


Figura 2-4. Diagrama interno del microcontrolador MC68HC11E9.

CAPITULO 3
Sensores

3.1. Transductores, sensores y actuadores

Se denomina transductor, a todo dispositivo que convierte una señal física en una señal correspondiente pero de una forma física distinta; es por lo tanto, un dispositivo que convierte un tipo de energía en otro.

La tendencia actual, particularmente en robótica, es emplear el término sensor para designar el transductor de entrada, y el término de actuador o accionamiento para designar el transductor de salida. Los primeros pretenden la obtención de información, mientras que los segundos buscan la conversión de la energía.

Un transductor es un dispositivo que convierte un tipo de variable física (fuerza, presión, temperatura, velocidad, intensidad luminosa, etc.) en otro tipo de variable; las más comunes voltaje o corriente eléctrica, ya que de esta manera es más conveniente su uso e implementación en cualquier sistema electrónico.

Un sensor es un transductor que es usado para hacer la medición de una variable física de interés. La mayoría de los sensores o transductores requieren de calibración, para que estos resulten lo más confiables al realizar la medición en turno. La calibración es el procedimiento por el cuál se fijan los parámetros es decir; tener un valor de referencia entre la variable medida y la señal de salida.

Dado que hay seis tipos de señales: mecánicas, térmicas, magnéticas, eléctricas, ópticas y moleculares (químicas), cualquier dispositivo que convierta una señal de un tipo en una señal de otro tipo debería considerarse un transductor, y la señal de salida podría ser de cualquier salida forma física útil. En la práctica, no obstante, se consideran transductores por antonomasia aquellos que ofrecen una señal de salida eléctrica. Esto se debe al interés de este tipo de señales en la mayoría de procesos de medida. Los sistemas de medida electrónicos ofrecen, entre otras las siguientes ventajas:

1. Debido a la estructura electrónica de la materia, cualquier variación de un parámetro no eléctrico de un material, viene acompañada por la variación de un parámetro eléctrico; eligiendo el material adecuado, permite realizar transductores con salida eléctrica para cualquier magnitud física no eléctrica.

2. Dado que en el proceso de medida no conviene extraer energía del sistema donde se mide, lo mejor es amplificar la señal de salida del tranductor.

3. Además de la amplificación, hay una gran variedad de recursos, en forma de circuitos integrados para acondicionar o modificar las señales eléctricas. Incluso hay transductores que incorporan físicamente en un mismo encapsulado como parte de estos recursos.

4. Existen también numerosos recursos para presentar o registrar información si se hace electrónicamente, pudiendo manejar no solo datos numéricos, sino también textos, gráficos y diagramas.

5.- La transmisión de señales eléctricas es más versátil que la de señales mecánicas, hidráulicas o neumáticas .

Los transductores analógicos proveen una señal de este tipo, ya sea voltaje o corriente, esta señal puede ser interpretada como el valor de la variable física; mientras los transductores digitales producen una señal de respuesta de tipo digital, que incluyen niveles lógicos. De hecho estos son los más utilizados en sistemas de automatización y procesos de control, ya que este tipo de señal es compatible con los microprocesadores .

3.1.2 Interfaces, dominios de datos y conversiones

Con el término interfaz se designa, al conjunto de elementos que modifican las señales, cambiando incluso el dominio de datos, pero sin cambiar su naturaleza, es decir; permaneciendo siempre en el dominio eléctrico.

Se denomina dominio de datos al nombre de una magnitud mediante la que se representa o transmite información.

En el dominio analógico, la información está en la amplitud de la señal, bien se trate de carga, corriente, tensión o potencia. En el dominio temporal, la información no está en las amplitudes de las señales, sino en las relaciones temporales: periodo o frecuencia, anchura de pulsos, fase, etc. En el dominio digital, las señales tiene sólo dos niveles lógicos, con valor de 0 o 1 . La información puede estar en el número de pulsos, o venir representada por palabras serie o paralelo codificadas.

3.2 Tipos de sensores

Según la señal de salida los sensores se clasifican en analógicos o digitales. En los analógicos la salida varía, a nivel macroscópico de forma continua. En los sensores digitales, la salida varía en forma de saltos o pasos discretos; no requiere conversión A/D y la transmisión de su salida es más fácil, tienen también mayor fidelidad, fiabilidad y muchas veces mayor exactitud.

Atendiendo al modo de funcionamiento, los sensores pueden ser de deflexión o de comparación. En los sensores que funcionan como deflexión, la magnitud medida produce algún efecto físico, que engendra algún efecto similar, pero opuesto, en alguna parte del instrumento, y que esta relacionado con alguna variable útil, por ejemplo un dinamómetro.

En los sensores que funcionan por comparación, se intenta mantener nula la deflexión mediante la aplicación de un efecto bien conocido, opuesto al generado por la magnitud a medir; hay un detector del desequilibrio y un medio para restablecerlo, por ejemplo una balanza manual.

Según el tipo de relación entrada-salida, los sensores pueden ser de orden cero, de primer orden, de segundo orden o de orden superior. El orden está relacionado con el número de elementos almacenadores de energía independientes que incluye el sensor, y repercute en su exactitud y velocidad de respuesta. Esta clasificación es de gran importancia cuando el sensor forma parte de un sistema de control de lazo cerrado.

Desde el punto de vista de ingeniería electrónica, es más atractiva la clasificación de los sensores de acuerdo con el parámetro variable: resistencia, capacidad, inductancia, añadiendo luego los sensores generadores de tensión, carga o corriente, etc.

3.3 Características estáticas de los sistemas de medida

Características deseables de los sensores:

1. Exactitud;
2. Precisión;
3. Rango de operación;
4. Velocidad de respuesta;
5. Calibración;
6. Fiabilidad;
7. Costo, tamaño reducido y comodidad de operación.

3.3.1 Exactitud, fidelidad, sensibilidad

La exactitud es la cualidad que caracteriza la capacidad de un instrumento de medida de dar indicaciones que se aproximen al verdadero valor de la magnitud medida. El valor exacto, verdadero o ideal, es el que se obtendría si la magnitud se midiera con un método ejemplar.

La exactitud de un sensor se determina mediante la denominada calibración estática. Consiste en mantener todas las entradas excepto a una en una a un valor constante. La entrada en estudio se varía entonces lentamente, tomando sucesivamente valores constantes dentro de un margen , y se van anotando los valores que toma la salida. La representación de estos valores en función de los de la entrada define la curva de calibración.

La discrepancia entre la indicación del instrumento y el verdadero valor de la magnitud medida se denomina error. La diferencia entre la indicación del instrumento y el verdadero valor se denomina error absoluto.

$$\text{error absoluto} = \text{resultado} - \text{valor verdadero}$$

Sin embargo, lo más común es especificar el error como cociente entre el error absoluto y el verdadero valor de la magnitud medida, cociente que se denomina error relativo.

$$\text{error relativo} = \frac{\text{error absoluto}}{\text{valor verdadero}}$$

La fidelidad (precisión) es la cualidad que caracteriza la capacidad de un instrumento de medida, de dar el mismo valor de la magnitud medida, al medir varias veces en unas mismas condiciones determinadas, prescindiendo de su concordancia o discrepancia con el valor real de dicha magnitud.

La sensibilidad o factor de escala es la pendiente de la curva de calibración, que puede ser o no constante a lo largo de la escala de medida. Para un sensor cuya salida esté relacionada con la entrada x mediante la ecuación:

$y = f(x)$, la sensibilidad en el punto x_a, $S(x_a)$, es

$$S(x_a) = \left.\frac{dy}{dx}\right|_{x=x_a}$$

En los sensores interesa tener una sensibilidad alta y, si es posible, constante. Para un sensor con respuesta:

$$y = Kx + b$$

la sensibilidad es $S = k$, para todo el margen de valores de x aplicables. Para uno cuya respuesta sea

$$y = kx^2 + b$$

la sensibilidad es $S = 2kx$, y varía a lo largo de todo el margen de medida.

En este capítulo se tratarán los tipos de sensores utilizados en el diseño del Robot Móvil y como se realizará la interfaces con el microcontrolador.

Se considera para la instrumentación; de la mayor parte de los sensores, la característica del convertidor Analógico/Digital del HC11, el cuál es sensible solamente en el rango de 0 a 5 Volts, con 8 bits de resolución, de tal forma que se podrán tener 256 posibles niveles discretos.

3.4 Sensores fotoeléctricos

En los últimos años, técnicas comunes a los campos de la óptica y la electrónica, han originado una nueva tecnología denominada *optoelectrónica*; este ha sido uno de los campos de crecimiento más rápido en la industria moderna, puesto en evidencia en las siguientes aplicaciones.

- Maquinas de fotocopia;
- Infrarrojos para el seguimiento de cohetes y para visión nocturna;
- Sistemas de reconocimiento desde aviones o cápsulas espaciales, mapas meteorológicos y detección de incendios forestales;

- Aplicación de láser, tales como curación de desprendimiento de retina, curación de cáncer, comunicaciones, identificación de objetivos y la holografía;
- Control fotoeléctrico industrial como contadores, sistemas de parada y arranque, control de nivel de proximidad, medida de grosores, procesos de alimentos y clasificación;
- Lectura en computadoras de cintas y tarjetas perforadas;
- Encendido automático del alumbrado público y sistemas de exposición fotográfica;
- Detección de incendio y robo;
- Espectrometría;
- Reconocimiento de caracteres;
- Posicionado y alineación de servosistemas;
- Lectura de códigos de barras, etc.

Los dispositivos fotoelectrónicos utilizan la luz como medio de transmisión entre un emisor y un receptor, es decir, una fuente luminosa y un elemento fotosensible. La luz es una radiación electromagnética comprendida en una banda de longitudes de onda de unos 100 a 8000 nm. La longitud de onda caracteriza el color de la luz. En una banda de 400 nm (violeta) a 700 nm (rojo), la luz es perceptible por el ojo humano (luz visible). Por encima de los 700 nm, la luz se denomina infrarroja (IR) y por debajo de los 400 nm, ultravioleta (UV).

3.4.1 Elementos fotosensibles.

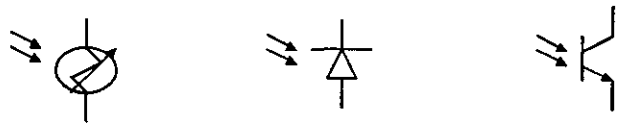
Los receptores de dispositivos fotoelectrónicos trabajan con elementos fotoeléctricos, que en la actualidad son casi exclusivamente semiconductores.

En la fotoresistencia se aprovecha la variación de la resistencia que se produce al incidir la luz y que es independiente del sentido de la corriente.

En el fotodiodo se aprovecha la variación de resistencia que se produce en la iluminación, posee polaridad determinada.

En el fototransistor se forman, al incidir la luz, pares de portadores de carga, que se refuerzan por la corriente engendrada por este.

Símbolos:



a) b) c)

Figura 3-1. Símbolos de los elementos fotosensibles: a) Fotoresistencia, b) Fotodiodo y c) Fototransistor.

3.4.1.1 Fotoresistencia

Son componentes cuya resistencia disminuye cuando son expuestas a la luz, de construcción sencilla, económicas, de gran sensibilidad, aunque de respuesta lenta. Su resistencia es muy elevada en la obscuridad.

La variación del valor de la resistencia , al ser iluminado este elemento, es independiente del sentido de la corriente, por cuya razón es posible su empleo en corriente alterna.

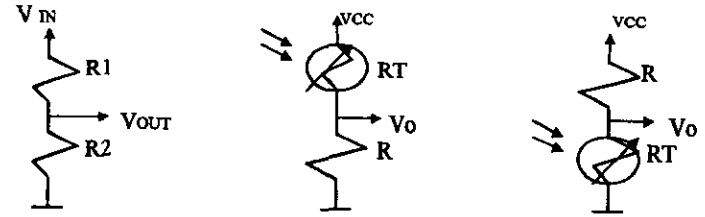


Figura (a) Figura (b) Figura (c)

Figura 3-2. Circuito de detección para una fotoresistencia.

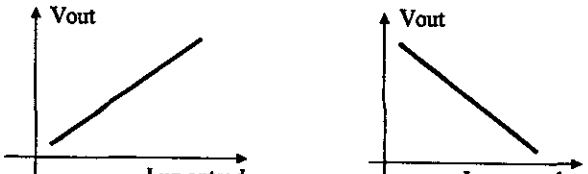
Analizando los circuitos de la figura 3-2a, 3-2b y 3-2c, se obtienen las siguientes ecuaciones:

$$VOUT = VIN\ (\ R2/R1+R2)\quad \text{ecuación 3.1}$$

$$Vo= VCC(\ R/R+RT\ )\quad \text{ecuación 3.2}$$

$$Vo= VCC(RT/RT+R)\quad \text{ecuación 3.3}$$

Encontrando las gráficas entre el voltaje resultado de la incidencia de luz que se presenta en la fotoresistencia, según se muestra en la figura 3-3a y 3-3b donde se observa para la primer gráfica; que a mayor luz incidente en el fotoelemento, se obtendrá a la salida un mayor valor de voltaje, y para la gráfica 3-2b; se encuentra que a mayor intensidad de luz, se obtendrá a la salida un valor de voltaje menor.



Circuito (a) Circuito (b)

Figura 3-3. Gráfica de Voltaje de salida Vs. Intensidad de luz.

3.4.1.2 Fotodiodos

Los fotodiodos se hacen trabajar con una tensión inicial en sentido de bloqueo, Sin dicha tensión trabajan, en sentido de paso, como fotoelementos. La corriente en la obscuridad es reducida. Con resistencias de trabajo de elevado valor óhmico pueden engendrarse variaciones de tensión que alcanzan la plena tensión de servicio.

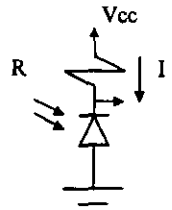


Figura 3-4. Circuito de detección de un fotodiodo.

3.4.1.3 Fototransistores

En un fototransistor actúa la luz sobre la zona de la base como una corriente de base, la que controla el espacio base-emisor y con ella la corriente del colector. Los fototransistores son más sensibles que los fotodiodos, aunque de respuesta más lenta.

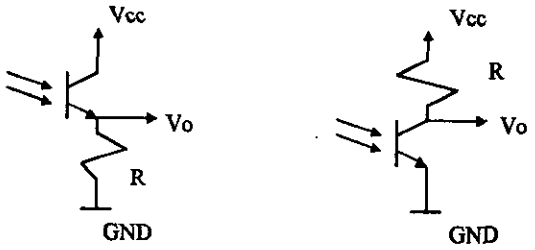


Figura 3-5. Circuito de detección de un fototransistor.

3.4.1.4 Detector de luz infrarrojo

Estos sensores detectan a una longitud de onda de 880 nm aproximadamente, donde el espectro no es perceptible al ojo humano, en conjunto para este circuito se requiere de un diodo emisor de luz infrarrojo y un detector (GP1U52X), donde este último contiene en el mismo empaque su amplificador, filtro y un limitador, esta unidad es distribuida por Radio Shack y Sterling Electronics.

Este detector responde a una señal modulada transmitida por el led infrarrojo, su frecuencia de operación es de 40 KHz. Esta frecuencia se puede realizar con varios arreglos de circuitos, en este caso, se implemento con el circuito integrado LM555; la figura 3-6 presenta el circuito utilizado.

El objetivo de este arreglo es detectar objetos que estén próximos al robot y que al encontrar dicho obstáculo, el robot pueda tomar la decisión adecuada para evitar la coalición.

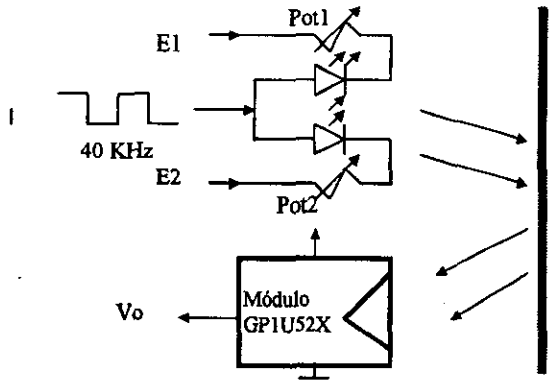


Figura 3-6. Transmisor y receptor infrarrojo.

Las señales ópticas de tipo infrarrojas, por su longitud de onda, pueden presentar una mayor inmunidad a interferencias de señales externas no deseadas por el sistema de detección de un objeto, mediante el sensado del rebote de un haz óptico. Además ofrece una mayor facilidad de manejo a un menor costo sobre otras señales ópticas que pueden ofrecer mayor inmunidad al ruido.

La interfaz infrarroja consta de las siguientes partes:

i) Etapa emisora.- En esta etapa se va a generar una señal de rayos infrarrojos codificada, la cuál se va a transmitir desde un punto en dirección al área donde se desea detectar la presencia de un objeto.

ii) Etapa receptora.- Su función será la de captar la señal infrarroja de rebote desde el área de detección , filtrar todas aquellas señales ópticas del medio que no sean deseables para una correcta detección, la de eliminar aquellas señales infrarrojas que no contengan el código de la señal emitida, así como de evaluar la intensidad del rebote del haz recibido.

3.4.1.5 Sensores Piroeléctricos

Uno de los sensores más útiles para dotar al robot, con un mecanismo para interacturar con humanos, es un sensor piroeléctrico. Este tipo de sensor es esencial en la detección de movimiento y útil en alarmas antirrobo.

La salida de un sensor piroeléctrico, cambia cuando pequeñas variaciones en la temperatura de este sensor ocurre en un intervalo de tiempo, lo cual es detectado por su elemento principal cristal de litio-tantalio; por lo tanto la energía es inducida cuando el cristal cambia de temperatura.

Los sensores piroeléctricos son económicos, diseñados para detectar radiación en el rango de 8-10 micrómetros (el rango de energía infrarroja emitida por los humanos), por ello que sea conveniente su uso en alarmas de seguridad.

El sensor Eltec 442-3 incorpora dos cristales de litio-tantalio. El amplificador diferencial suministra los voltajes a través de los cristales a la salida del sensor. En el caso que ambos cristales estén a la misma temperatura, el sensor produce una señal de salida que permanece a un valor constante de 2.5 Volts (considerando una fuente de 5 Volts).

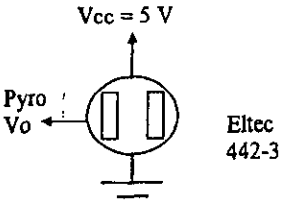


Figura 3-7. Sensor piroeléctrico y su acondicionamiento.

Si una persona camina en frente del sensor, moviéndose de izquierda a derecha, la señal levanta a un valor de 1 Volt y después bajará el mismo voltaje, para finalmente regresar al voltaje inicial; por otro lado cuando la persona se desplaza de derecha a izquierda, sucederá lo inverso, es decir la señal primero bajará, entonces subirá y después se establecerá, como se muestra en la figura 3-8.

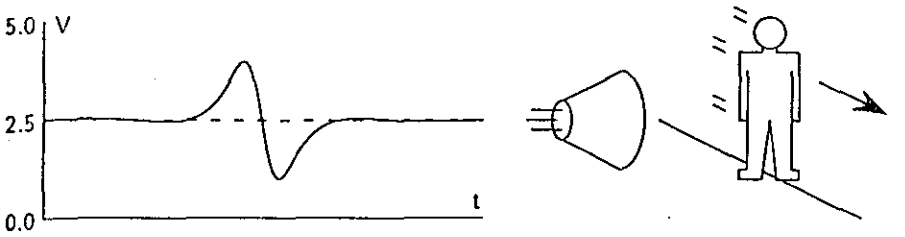


Figura 3-8. Señal entregada por un sensor piroeléctrico.

3.4.1.6 Cámaras

La tecnología actual de las cámaras de vídeo ha evolucionado rápidamente, por lo que hoy en día se cuentan con cámaras muy compactas y de costo no tan elevado, la figura 3-9 presenta un ejemplo de una carrera de este tipo. La mayor utilidad se encuentra en los circuitos cerrados de televisión, implantados como sistemas de seguridad.

La adaptación de las cámaras de vídeo para la aplicación de robots móviles, se realiza transmitiendo esta señal a una estación de trabajo, utilizando un transmisor y un receptor, conectados al robot y a la estación de trabajo respectivamente, para poder realizar el procesamiento de la imagen.

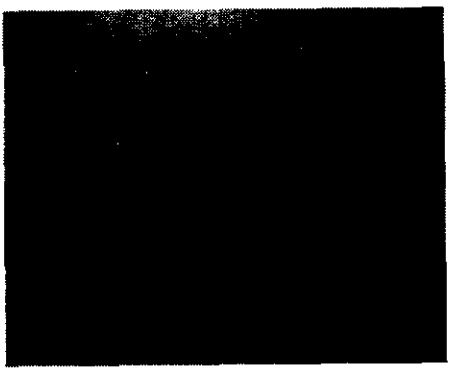


Figura 3-9. Cámara de vídeo.

3.5 Sensores de contacto

Entre los más sensores más conocidos de este tipo se encuentran los microswitches, los flexibles y los detectores de fuerza.

3.5.1 Microswitch

Los microswitches son pequeños dispositivos, empleados con la finalidad de determinar cuando el robot ha chocado directamente con un objeto; son de fácil adaptación electrónica, ya que al tener contacto con el obstáculo, es posible obtener una señal que puede ser capturada sin necesidad de agregar una interfaz compleja, para ser procesada la información y realizar la acción programada (figura 3-10).

Se pueden utilizar tantos microswitches como se desee, ya que se pueden conectar en todo el alrededor del robot para que este tenga una mayor gama de puntos de contacto, para tener un control y monitoreo más exacto del entorno que rodea al robot.

Se pueden realizar dos tipos de arreglos con los microswitches, uno tomando la salida de cada uno de estos switches, para que ingresen al microcontrolador de manera independiente cada uno, la otra es haciendo un arreglo con varios de ellos, de manera que al hacer contacto cualquiera de ellos, se tenga como resultado un divisor de voltaje, para que al ser capturado por el convertidor analógico/digital del microcontrolador, se pueda realizar una tarea específica.

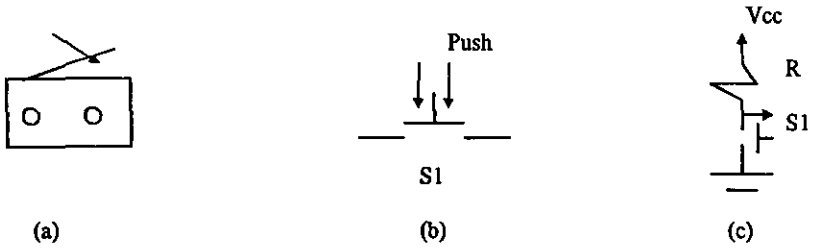


Figura 3-9. a) Esquema de un microswitch, b) Símbolo,
c) Circuito de acondicionamiento.

3.5.2 Sensores Flexibles

Otro sensor de gran ayuda para la detección de obstáculos, es el sensor de tipo flexible, un ejemplo se muestra a continuación. Este dispositivo utiliza pistas conductivas, conectadas entre dos electrodos, para entregar una resistencia variable, dependiendo del grado de flexibilidad. Esta resistencia se puede adaptar al microcontrolador, de la misma forma que una fotoresistencia , para tener un divisor de voltaje y conectarse a un canal del convertidor A/D. El inconveniente que se tiene es el rango reducido de variación entre cuando se encuentra sin flexionar y cuando se tiene la máxima inclinación (figura 3-11).

Figura 3-11. Sensor flexible.

3.5.3 Sensores detectores de fuerza

Interlink Electronics fabrica una línea de sensores de fuerza, que son similares a los sensores de inclinación, basados en tecnología de pistas conductivas. La resistencia varía en relación a la fuerza aplicada en su superficie; presentan un rango muy grande de valores, a su vez están disponibles en varios tamaños y formas, que van desde 0.2 pul hasta 25 pulgadas de longitud.

3.6 Posición y orientación

Es de gran ayuda el saber la posición de un robot dentro de un entorno, así como conocer la distancia que se ha recorrido, desde una posición inicial a una final; en esta sección se estudiará el tipo de transductores que se emplearon para este objetivo .

3.6.1 Shaft Encoders

Un shaft encoders es un sensor que mide la posición o la velocidad de rotación de un motor, comúnmente son montados a la salida de un motor, a señal derivada de este sensor se puede considerar como la orientación de una rueda, la señal que entrega se considera como un tren de pulsos.

3.6.1.1 Potenciómetro

Un potenciómetro puede ser utilizado como un encodificador de posición absoluta, cada posición de la rueda produce una resistencia única. Los encodificadores absolutos son comúnmente usados para determinar la posición en brazos de robot, la figura 3-12 muestra un dispositivo de este tipo.

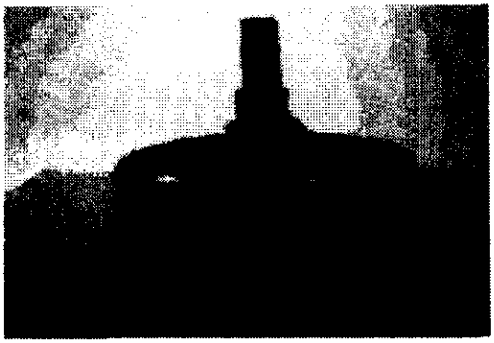


Figura 3-12. Potenciómetro.

3.6.1.2 Fotointerruptor

Una forma para lograr sincronizar la velocidad de giro de las ruedas del robot, se conecta un shaft encoder a cada motor. Se componen de dos dispositivos integrados en un mismo empaque, por un lado un led emisor de luz infrarroja y un fototransistor por el otro lado.

Se coloca un disco con ranuras, tantas como sea posible, para que se pueda obtener una mejor resolución para tener una aproximación más exacta de la velocidad, distancia o ángulo de giro de nuestro robot, de manera que cuando existe transmisión entre el infrarrojo y el fototransistor, se tenga una señal 0 o 1 lógico, dependiendo del arreglo implementado para este sensor y se pueda procesar por el microcontrolador; a este tipo de dispositivo se le conoce como fotointerruptor.

3.6.1.3 Fotoreflector

Otra implementación de un shaft encoder es con el dispositivo conocido como un fotoreflector, el cuál refleja la luz que emite en LED infrarrojo sobre un disco rayado, la cual es reflejada hacia un fototransistor. Un circulo marcado de líneas blancas y negras alternadas , será el mecanismo que realice la tarea de reflexión, según el color que esté pasando en ese momento por el sensor.

Un fotoreflector comercial que se puede utilizar, es el fabricado por Hamamatsu P306201, contiene internamente el circuito detector, el comparador y el amplificador, con lo que con dos resistencias externas es posible su implementación, la figura 3-13 presenta un ejemplo de modelos típicos, tanto de un foto interruptor como del fotoreflector, ya que es posible encontrarlos en diferentes modelos.

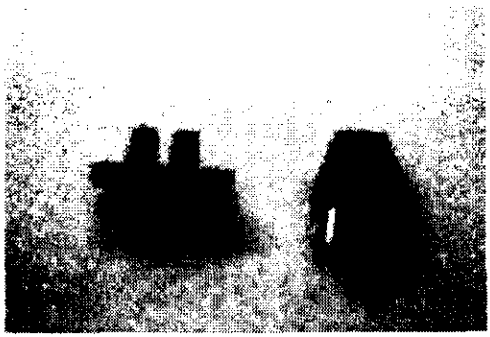


Figura 3-13. A la izquierda el foto interruptor; a la derecha el fotoreflector.

3.6.2 Gyros

Otro sensor que es de gran ayuda para determinar la velocidad de rotación de robot son los giroscopios. Son dispositivos mecánicos que emplean el principio de conservación de momento angular para almacenar uno o mas puntos de cortes en la misma dirección como el exterior del giroscopio.

3.6.3 Brújulas

Una brújula proporciona la manera de obtener información real de la orientación del robot, los cuales operan bajo el principio de atracción magnética; este tipo de sensores es de gran ayuda cuando se escriben algoritmos de navegación.

En áreas al aire libre, las brújulas son muy confiables, se calibran con respecto el campo magnético del polo norte, pero si el robot es operado dentro de cuartos, el uso de la brújula llaga a ser más problemático, esto se debe principalmente a los campos magnéticos producidos por los cables eléctricos, estructuras de acero de las construcciones y los componentes de metal del mismo robot, pueden todos ellos producir grandes errores en la lectura de la brújula, en el apéndice F se presenta el circuito de interfaz de un dispositivo de este tipo.

3.7 Sensores de autoevaluación

Un sensor de autoevaluación es cualquier sensor utilizado para medir el estado interno del robot; este tipo se sensores monitorean, de tal manera que pueden indicarnos cuando es tiempo de reemplazar la batería, cuando un motor se esta sobrecalentando o cuando un componente no está funcionando correctamente.

3.7.1 Sensado del nivel de una batería

Para sensar el nivel de voltaje de una batería, el robot indicará cuando debe recargarse, una forma sencilla de obtenerlo, es colocando un indicador de nivel de voltaje, colocando un arreglo conectado antes de ingresar el voltaje al regulador de voltaje, o después del mismo, como se presenta en la figura 3-14.

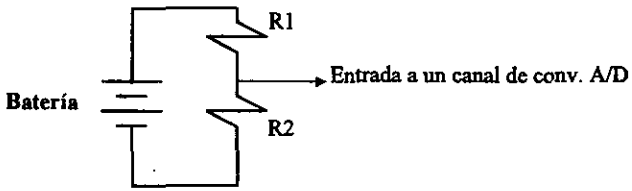


Figura 3-14. Circuito para detectar la carga de una batería.

3.7.2 Temperatura

Este sensor es importante para medir la temperatura interna del robot, ya que de esta manera se evitarían los sobrecalentamientos de los circuitos que lo conforman, por ejemplo, para altas temperaturas, la vida útil se reduce para el microcontrolador, los motores y las baterías; los dispositivos mas comúnmente utilizados pueden ser los termistores, quienes cambian su valor de resistencia dependiendo del la intensidad de calor existente; un sistema de ventilación correcto se requerirá en este caso.

CAPITULO 4
Motores y manejadores de potencia

4.1 Elementos motrices de los robots

Los dispositivos que en general producen el movimiento en los robots, pueden clasificarse en tres grandes grupos, según la energía que consumen:

a) Hidráulicos;
b) Neumáticos;
c) Eléctricos.

Los actuadores neumáticos e hidráulicos hacen uso de aire comprimido y de un flujo de presión, respectivamente. Dada su importancia y la creciente implantación en todas las áreas de la robótica, los elementos más utilizado son motores eléctricos.

Dentro de la gran variedad de tipos de motores eléctricos, los más adecuados para el movimiento son los de corriente directa y los motores de paso a paso.

Un motor eléctrico convierte energía eléctrica a energía mecánica; es posible encontrarlos de diversas formas y tamaños, existen motores de corriente directa CD y motores de corriente alterna CA, así como una gran variedad de modelos en cada uno de ellos.

Los motores de CA son típicamente utilizados para equipo industrial y son energizádos con la toma de la línea de corriente, ocupando para su funcionamiento, una , dos o tres fases, estos no son regularmente empleados en los robots móviles, ya que lo que se intenta es que estos sean lo más autónomos posibles, consiguiendo lo último suministrando la potencia con baterías de corriente continua.

Los motores de CD son comúnmente usados para trabajos relativamente pequeños, aunque sin dejar de ser importantes, ya que la mayor parte de los aparatos electrónicos y juguetes cuentan con cuando menos un motor de este tipo; así mismo, es posible encontrarlos fácilmente en el mercado, de varios tamaños, formas y en una gran variedad de voltajes de operación.

Existe un tipo de motores de CD con más de dos terminales, llamados servomotores, son los más utilizados en el diseño y construcción de robots; en nuestra aplicación se emplearon los motores de tres terminales, que son usados en el control de los aeroplanos de juguete, en carros de radio control y en el movimiento de los brazos manipuladores.

En el ensamble de estos motores, incorpora internamente un motor de CD, un sistema de engranes, un límite de giro, un potenciómetro de posición (feedback) y un circuito integrado para el control general; las terminales involucradas son polarización, tierra y el control de entrada, donde la señal de ancho de pulso indicara la velocidad del motor.

4.2 Interfaz con los motores

Un microprocesador no puede manejar directamente un motor, puesto que no puede suministrar la corriente requerida, para su correcto funcionamiento. Para incluir este control, es necesario adaptar algunos circuitos de interfaz, con el objetivo que el microcontrolador tenga el control de los motores. Esta interfaz de circuitos puede ser implementada con una gran variedad de tecnologías, como relevadores, transistores, MOSFETS y circuitos manejadores de potencia; en todas las tecnologías, la topología básica de circuitos es el mismo.

4.2.1 Empleo de transistores de potencia

Existen dos configuraciones básicas:

a) Configuración tipo T; y

b) Configuración tipo H.

4.2.2 Configuración tipo T

Se muestra en la figura 4-1, precisa de dos transistores de potencia y dos fuentes de alimentación.

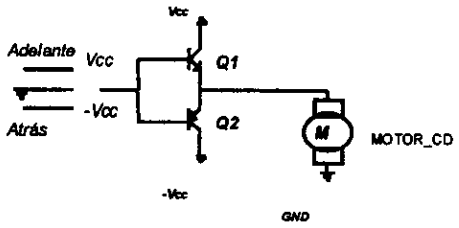


Figura 4-1. Configuración de tipo T, con dos transistores.

4.2.3 Configuración tipo H

Está compuesto de cuatro elementos, como se ejemplifica en la figura 4-2, si el switch S1 y S4 en la figura son cerrados mientras los switches S2 y S3 están abiertos, fluye la corriente de izquierda a derecha, en el caso contrario, si los switches S2 y S3 son cerrados y los switches S1 y S4 están abiertos, la corriente circulará de derecha a izquierda, funcionando el motor en sentido inverso.

La velocidad de conmutación se realiza controlando el switch S5.

En el puente H, los switches son abiertos y cerrados de una forma, que al polarizar un par de ellos, la corriente circulará en un sentido y en el otro caso al polarizar el otro par, circulara la corriente en sentido opuesto.

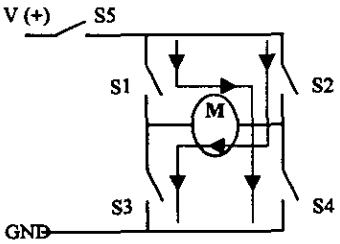


Figura 4-2. Configuración H con switches.

Emplea cuatro transistores de potencia y una sola fuente de alimentación, como se aprecia en la figura 4-3.

El funcionamiento de los transistores de la configuración H es tal, que siempre están conduciendo o bloqueados por parejas. Por ejemplo, si T1 y T4 conducen, se requiere que T2 y T3 no conduzcan y viceversa. Regulando el tiempo de conducción se varía la velocidad del motor.

En el puente H, los switches son abiertos y cerrados de una forma, que al polarizar un par de ellos, la corriente circulará en un sentido y en el otro caso al polarizar el otro par, circulara la corriente en sentido opuesto.

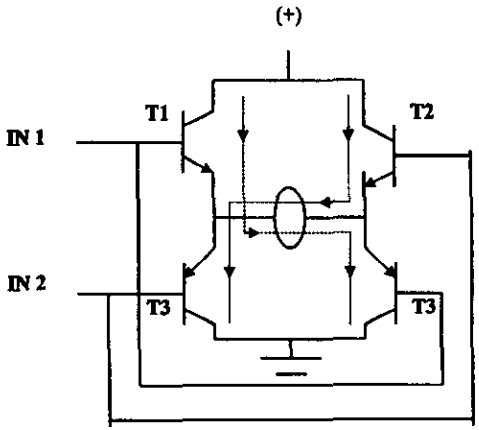


Figura 4-3. Configuración tipo H con cuatro transistores.

4.3 Métodos de control de potencia

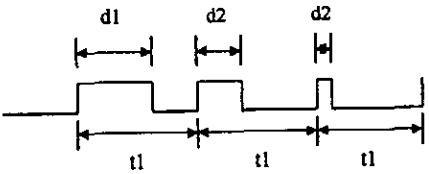
Se pueden implementar dos métodos para controlar la potencia:

1.- Modulación por ancho de pulso, PWM (Pulse Width Modulation);

2.- Modulación de la frecuencia de pulso, PFM (Pulse Frecuency Modulation).

4.3.1 Sistema PWM

Este sistema emplea, normalmente, una sola fuente de alimentación, para conseguir variaciones de frecuencia, el amplificador de conmutación se diseña para una frecuencia constante y sólo se cambia el ancho del pulso, como se aprecia en la figura 4-4.



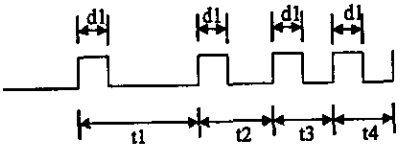
t1: constante
d1, d2, d3: diferente

Figura 4-4. Variación de ancho del pulso
Velocidad de ancho de pulso = ton/tperiodo

El amplificador conecta a la fuente de alimentación con una frecuencia fija, pero con un ángulo variable, de esta manera se consigue la tensión media más apropiada en la carga.

4.3.2 Sistema PFM

Emplea una ancho del pulso constante, pero varía la frecuencia. Los resultados obtenidos, empleando este método, son muy parecidos a los del PWM, según se muestra en la figura 4-5.



d1 : constante
t1, t2, t3, t4: diferente

Figura 4-5. Variación de la frecuencia, con ancho de pulso constante.

4.4 Circuitos integrados manejadores de potencia

Estos manejadores hacen muy practico la interfaz de los motores con el microcontrolador, ya que reducen sustancialmente el consumo de energía y el espacio que ocupan, además cuenta con circuitos limitadores de corriente y protección contra sobrevoltajes.

Existe un Chip fabricado por Motorola, el MPC1710A que se muestra en la figura 4-6, el cuál presenta la capacidad de poder controlar un motor de corriente directa, operando el último en ambos sentidos.

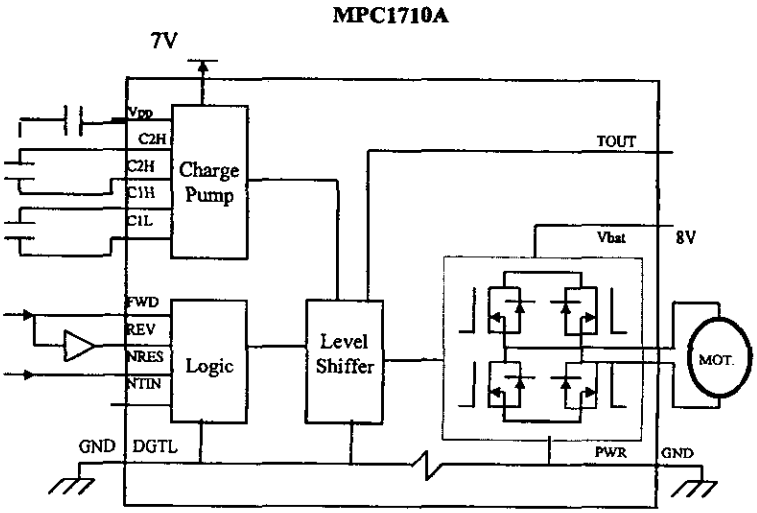


Figura 4-6. Circuito MPC1710

Por otro lado, existe el circuito, conocido como L293, que es un circuito integrado de 16 pines, fabricado por National, internamente cuenta con dos arreglos tipo H, se ilustra en la figura 4-7 el diagrama a bloques de una parte de este circuito.

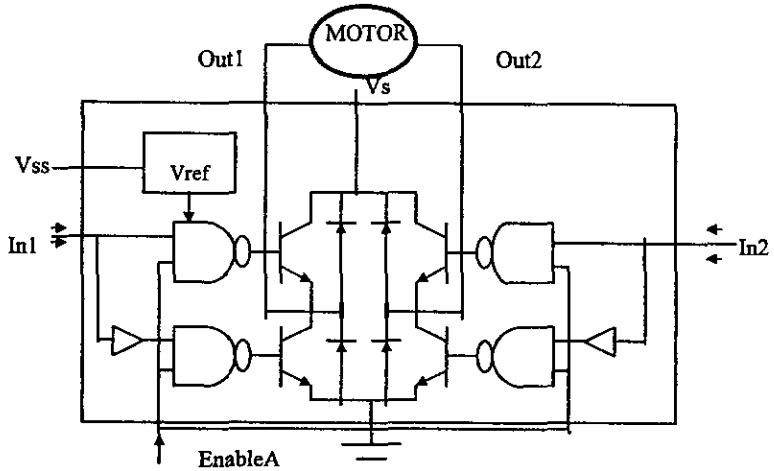


Figura 4-7. Circuito manejador de motor de CD, L293.

La ventaja de este circuito radica en que se pueden controlar por su arquitectura interna, dos motores al mismo tiempo girando en sentido horario y antihorario, o como se presenta en el apéndice B, se pueden controlar hasta cuatro motores al mismo tiempo, pero únicamente en un solo sentido; además este circuito opera con un mínimo de componentes externos, en este caso solo se conectan dos circuitos inversores; para asegurar el estado de corte y conducción de las parejas de transistores.

4.5 Motores de paso a paso

Los motores paso a paso o simplemente PAP, son un tipo especial de motores que permiten el avance de su eje en ángulos muy precisos y por pasos en las dos posibles direcciones de movimiento, izquierda y derecha. Aplicando a ellos una determinada secuencia de señales digitales, avanzan por pasos hacia un lado u otro y se detienen exactamente en una determinada posición.

Cada paso tiene un ángulo muy preciso, determinado por la construcción del motor, lo que permite realizar movimientos exactos sin necesidad de un sistema de control de lazo cerrado.

Los motores paso a paso presentan grandes ventajas con respecto a la utilización de los servomotores debido a que se pueden manejar digitalmente sin realimentación, su velocidad se puede controlar fácilmente, tienen una larga vida, son pequeños, robustos y poseen un elevado torque en bajas revoluciones, lo que permite un bajo consumo tanto en vacío como en plena carga, su mantenimiento es mínimo debido a que no tiene escobillas.

Debido a esta ventaja, tienen una gran variedad de aplicaciones dentro de las cuales se podrían mencionar las siguientes: máquinas herramientas, robots, impresoras para computadoras, graficadoras, máquinas de coser, unidades de disco, etc.

4.5.1 Funcionamiento

El funcionamiento de los motores paso a paso se basa en el simple principio de atracción y repulsión que ocurre entre los polos magnéticos. Como se sabe, un imán tiene dos polos llamados Norte y Sur. El principio básico del magnetismo establece que los polos iguales se repelen y los polos diferentes se atraen.

4.5.2 Tipos de motores paso a paso

Según su construcción, existen tres tipos de motores PAP:

a) De imán permanente

En este tipo de motor, su rotor es un imán permanente que posee una ranura en toda su longitud y el estator está formado por una serie de bobinas enrolladas alrededor de un núcleo o polo. Su funcionamiento se basa en el principio explicado anteriormente de atracción y repulsión de los polos magnéticos.

b) De reluctancia variable

En estos motores el rotor está fabricado por un cilindro de hierro dentado y el estator está formado por bobinas que crean los polos magnéticos. Como este tipo de motores no tiene un imán permanente, su rotor gira libremente cuando las bobinas no tienen corriente, lo que puede ser inconveniente en un momento dado si han una carga que presione el eje; estos motores puede trabajar a mayor velocidad que los anteriores.

c) Híbridos

Son motores de pasos, que incluyen algún mecanismo para amplificar el par o la potencia. *Algunos emplean engranes para lograr pares, aunque reducen la amplitud de los pasos.*

Los motores paso a paso, tanto unipolares como bipolares, pueden trabajar en dos modos de operación, de paso completo y de medio paso. En el primer caso, con cada secuencia el rotor gira un determinado ángulo que depende de la fabricación del motor. En el modo de medio paso, cada secuencia produce un giro en grados correspondiente a la mitad de su paso normal.

En la práctica, generalmente los motores PAP poseen cinco terminales, debido a que existe una terminal común, que es el punto medio de los devanados de las bobinas.

4.6 Sistemas de control para motores de pasos

Una buena regulación del movimiento de un motor de pasos exige un sistema de control apropiado, estando relacionados estos dos elementos; en el sistema de control se requiere de las señales que controle la velocidad, sentido de giro y la generación de la secuencia de polarización, como se muestra en la figura 4-8.

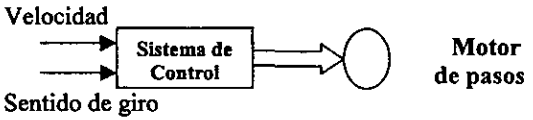


Figura 4-8. Esquema de sistema del control de un motor de pasos

CAPITULO 5
Lenguajes de programación

5.1 Lenguajes de programación de los microcontroladores

Para escribir un programa que va a ser ejecutado por un microprocesador existen básicamente tres alternativas: lenguaje de maquina, lenguaje ensamblador y lenguaje de alto nivel. Solo los programas escritos en lenguaje de maquina pueden ser ejecutados directamente por el microprocesador.

Los programas escritos en lenguaje de alto nivel tienen que ser traducidos primero a lenguaje de maquina a fin de que puedan ser ejecutados.

5.1.1 Lenguaje de máquina

Un programa en lenguaje de maquina es simplemente una secuencia de instrucciones y datos en código binario (unos y ceros), estos programas son por su naturaleza dificiles de leer, escribir, entender y corregir. Las computadoras únicamente trabajan en lenguaje de maquina, como cada instrucción está compuesta de un valor numérico, que componen el código de operación de la instrucción a realizar. Escribir directamente en lenguaje de maquina es un tanto dificil, ya que se substituyen los mnemonicos por sus valores numéricos.

5.1. 2 Lenguaje ensamblador

El lenguaje ensamblador utiliza mnemónicos para representar los códigos de operación de las instrucciones y símbolos alfanuméricos para representar las direcciones y los datos del programa.

La ventaja de utilizar nombres simbólicos para las direcciones, en lugar de un valor numérico, es que no solo facilita la escritura del programa, sino también simplifica la inserción o eliminación de instrucciones. Cuando el programa está escrito en lenguaje ensamblador usando direcciones simbólicas, el añadir o quitar una instrucción no causa ningún problema por que al traducir el programa a lenguaje de maquina, automáticamente se hacen los ajustes necesarios.

La desventaja de los programas escritos en lenguaje ensamblador son que requieren un programa especial llamado ensamblador, que se encarga de traducir lo mnemónicos y los símbolos alfanuméricos a lenguaje de maquina; además sigue siendo necesario un conocimiento detallado de la arquitectura del microprocesador.

5.1.3 Lenguaje de alto nivel

Los lenguajes de alto nivel permiten la escritura del programa en una forma más relacionada con el problema a resolver que con las características del microprocesador; generalmente cada instrucción en lenguaje de alto nivel es equivalente a varias instrucciones en lenguaje ensamblador.

Sin duda, con los lenguajes de alto nivel el tiempo invertido en el diseño y codificación de un programa es mucho menor. Sin embargo, para que los programas puedan ser ejecutados se requiere un traductor que convierta las instrucciones de alto nivel en lenguaje de maquina. A este programa traductor se le conoce como COMPILADOR.

Otra ventaja de los programas escritos en lenguajes de alto nivel es su independencia respecto a un procesador en particular, lo único que se necesita es el sistema en el que se desea ejecutar el programa y se posea un compilador para el lenguaje en el que el programa está escrito.

Conociendo todas las ventajas de los lenguajes de alto nivel entonces se plantea la pregunta: ¿Por qué complicar las cosas utilizando ensamblador?. Existen varias razones para esto, una de ellas es que los compiladores son más complejos y por lo tanto requieren mayor cantidad de memoria y son más lentos que los ensambladores. Otra es que, dado el gran número de posibilidades que deben tomar en cuenta los compiladores, el código en lenguaje de maquina que genera tiende a ser ineficiente.

Los programas recientes, regularmente se escriben utilizando, el programa Edit de Msdos, o el editor de los lenguajes de programación, donde se crea el programa fuente, conteniendo instrucciones en Lenguaje Ensamblador y con saltos relativos; para que con este programa fuente, con ayuda de un ensamblador, se trasformen las instrucciones a lenguaje de maquina y genere los programas de tipo S19 y LST.

5.2 Formato de los programas S19 y LST

5.2.1 Archivos con formato S19

El formato S19 fue creado por motorola con el propósito de codificar programas o archivos de datos en un formato sencillo para la transferencia de datos entre sistemas de computadoras. El proceso de transportación puede ser entonces monitoreado visiblemente y los archivos en formato S19 formados por registros S pueden ser editados fácilmente. Los registros S son esencialmente cadenas constituidas por diferentes campos que permiten identificar el tipo de registro, la dirección de memoria, los códigos o datos y el checksum. Cada byte de datos binarios está codificado por dos caracteres que representan un número hexadecimal .

Los cinco campos que forman un registro S son:

TIPO	LONGITUD	DIRECCION	CODIGO/DATOS	CHECKSUN

Los campos están compuestos de la siguiente manera:

CAMPOS	NUMERO DE CARACTERES	DESCRIPCION
TIPO	2	Tipo de registro S, S0, S1, S9, etc.
LONGITUD	2	Indica el número de pares de caracteres en el registro, excluyendo el tipo y longitud.
DIRECCION	4, 6 u 8	Los 2, 3, o 4 bytes de dirección que indican la dirección de memoria en donde se cargará en campos de datos.
CODIGO/DATOS	0 - 2_N	De 0 a n bytes de código ejecutable, datos cargables en memoria, o información descriptiva.
CHECKSUM	2	El byte menos significativo del complemento a uno de la suma de los valores representados por los pares de caracteres, tomando en cuenta los campos de longitud del registro, el de dirección y el de código/datos.

Ejemplo de un archivo S19:

```
S11301008E01FFCE10001C39901C3030011F30804E
S111301110FCB61032B110332B12B61033B110322B9F
S113012014B61034B110322B1620E1B61032B110CF
S1130130342B202A14B61033B110342B1C2A0AB6DF
S11301401034B110332B1820C3B61034B110322B35
S1130150EE20B9C601E70420B3C602E70420ADC609
S1008016004E70420A7E0
S9030000FC
```

5.2.2 Archivos con formato LST

El archivo de tipo lista (LST), es generado cuando el programa fuente se ha ensamblado; este archivo muestra los mnemónicos y códigos de maquina de cada instrucción.

Una parte de un archivo de tipo lista LST, es el siguiente:

```
0180                          *INICIO DEL PROGRAMA*
0181
0182
0183  2000          Inicio    org $2000          Dirección de inicio
0184  2000                    sect 0
0185
0186
0187  2000  8e7ff0            LDS #STACK         Inicializa el SP.
0188
0189  2003  ce10 00       LDX #REGBAS      x <- 1000
0190  2006  8625               LDAA  #$25
0191  2008  a73c               STAA HPRIO,X     Selecciona modo expandido
0192
0193  200a  0f                 SEI              Deshabilita interrupciones
```

```
0194
0195                          * inicia el programa principal
0196   200b   bd24c5                jsr _main
0197
0198   200e   7ec00a wait            jmp $c00a          Salta al origen de buffalo
```

A continuación se explican cada una de sus partes:

a) Numero de Línea (Columna 1)

Es el número progresivo de las líneas de programación, el numero de línea esta dado en un valor decimal, y el numero 65536 es el mayor posible de cualquier programa, es de gran ayuda ya que en caso de error, el ensamblador indica el numero de línea en el que se encuentra este.

b) Dirección (Columna 2)

La dirección corresponde a la localidad de memoria destinada a cada instrucción, este valor está en notación hexadecimal.

c) Código Objeto (Columna 3)

Cada instrucción es transformada a lenguaje de maquina y los códigos se muestran en su valor hexadecimal, estas instrucciones pueden ser de 1, 2, o 3 Bytes de longitud dependiendo de la instrucción y modo de direccionamiento involucrado.

d) Etiquetas (Columna 4)

Las etiquetas son usadas en el programa fuente, para indicar la dirección de inicio de una subrutina, de un salto relativo o la localización de una constante.

e) Operadores (Columna 5)

Se indica las instrucciones en lenguaje ensamblador.

f) Operandos (Columna 6)

La columna de operandos puede contener un valor, una dirección o una etiqueta.

g) Comentarios (Columna 7)

Los comentarios ayudan para explicar brevemente que es lo que realiza cada línea de un programa, estos comentarios no generan códigos de maquina, son copiados directamente del programa fuente.

5.3 Procedimiento para ensamblar un programa

Existen varios programas ensambladores, como por ejemplo el AS11, IASM11, etc.

Para el primer caso, es necesario editar los programas utilizando lo mnemónicos, para generar un programa que se tenga la extensión ASM o ASC para que desde el sistema operativo se indica lo siguiente.

AS11 Nombre_programa.extensión

Entonces se genera el programa con el mismo nombre que el fuente, pero con formato S19.

Por otro lado, al emplear el ensamblador IASM11, tiene la ventaja de contar con un editor de texto, integrado con un sistema de comandos para configuración. La manera de obtener un archivo S19 se numera a continuación.

1.- Se teclea IASM11 Nom_prog;
2.- Editar el programa;
3.- Presionar la tecla F10 para pasar a otro sistema de comandos;
4.- Configurar la salida del archivo, seleccionando en ASSAMBLE el formato deseado;
5.- Para ensamblar, con la tecla de función F4;
6.- El ensamblador creará los formatos seleccionados, S19, LST, HEX, etc.

5.4 Compiladores

Los compiladores son una herramienta para programar una aplicación en un microcontrolador. Un Compilador es un programa similar a un ensamblador, que transfiere instrucciones en lenguaje de alto nivel a códigos en lenguaje de maquina; existen compiladores en Basic, Fortran, Pascal, y otros que por mucho tiempo han sido utilizadas en computadoras. En años recientes, el Lenguaje C se ha convertido en uno de los más populares. Los compiladores son escritos para un lenguaje de programación especifico y para una unidad en particular. Los compiladores de C son escritos para que puedan operar con IBM-PCs, Apple, Macintoshes y muchos otras computadoras que usan sistema UNIX. La ventaja de los compiladores es su facilidad de poder hacer modificaciones al programa fuente, de una manera rápida y fácil.

Tal vez el inconveniente principal del compilador de C, es que genera programas más grandes que con el ensamblador, y en algunas ocasiones no produce un código eficiente como con el lenguaje ensamblador, pero en la mayoría de las ocasiones el C puede ser utilizado sin ningún problema.

La programación en C se ha convertido indiscutiblemente en una poderosa y popular herramienta de desarrollo para el ingeniero diseñador. Por esta razón, como parte de la extensa gama de herramientas de desarrollo con microprocesadores, cotidianamente se lanzan al mercado ***compiladores cruzados en C*** que permiten la programación en lenguaje de alto nivel.

5.4.1 Compilador cruzado para el HC11

El compilador cruzado para el MC68HC11 permite traducir el programa de aplicación de lenguaje C, en código de ensamblador para los microcontroladores MC68HC11. Este compilador cruzado consiste de lo programas siguientes: un programa manejador (ICC11.EXE), un preprocesador de C (ICPP.exe) y un compilador (ICCOM.EXE); como se muestra en el diagrama de la figura 5-1.

El compilador cruzado es un sistema compilador tradicional que acepta y se ajusta al lenguaje ANSII. El compilador cruzado invoca al programa manejador (ICC11.exe), a las funciones necesarias y al archivo que contiene las rutinas básicas de ensamblador (BASICS.S) para poder procesar el archivo en C (ARCHIVO.C).

El archivo BASICS.S contiene también el inicio del programa en donde se inicializa el apuntador de pila y se hace el llamado del *main()*, que es la rutina que se ejecuta al principio de un programa en C. Si el compilador cruzado no detecta ningún error, produce un archivo objeto, que contiene el código en ensamblador (ARCHIVO.S). Si existió un error, se indica donde se encuentra y de que tipo se trata.

El programa manejador (ICC11.EXE) direcciona al preprocesador (ICPP.EXE) para que busque en los directorios especificados los encabezados de los archivos que contienen las funciones que se han desarrollado en C y que son necesarios para generar el archivo con el código en ensamblador.

El preprocesador (ICPP.EXE) agrega al archivo en C las funciones necesarias (headers files), generando un archivo en lenguaje de alto nivel C (ARCHIVO.I). Finalmente el compilador (ICCOM.EXE) traduce el código del archivo en lenguaje de alto nivel en un código fuente para el ensamblador.

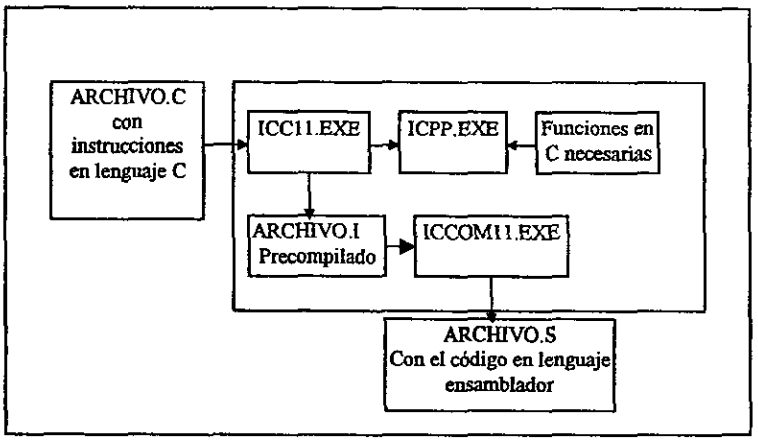


Figura 5-1. Compilador cruzado para el MC68HC11.

5.5 Ejecución de un programa

La manera de programar, generar y correr una aplicación en el microcontrolador 68HC11 es la siguiente:

1.- Se crea el programa fuente en C, utilizando el editor de textos de su preferencia.

2.- Se guarda el programa utilizando la extensión C.

3.- Empleando el programa CC6811.bat, que realiza la liga entre el compilador, ensamblador y programas con rutinas y declaraciones en lenguaje ensamblador, se generan los programas con formato s, i,S19 y LST.

Se ejecuta de la siguiente manera:

CC6811 Nombre_programa

El archivo CC6811.bat contiene lo siguiente:

```
Icpp.exe -DHC11 -D__LCC__ -I/icc11/include %1.c %1.i
Iccom11.exe -v %1.i %1.s
Ias11 -o %1 -l basics.s %1.s
```

4.- Una vez teniendo el programa con el formato S19, está listo para ejecutarse.

5.- Con el PCBUG11, que es un programa que permite tener el control de nuestra unidad, se puede cargar, correr, monitorear el estado de los registros, manipulación de datos en memoria, etc.

6.- Se cuenta con otro archivo, EPROM.bat que realiza la tarea de configuración y selección del puerto de comunicación serial donde se conectó el microcontrolador, así mismo de cargar un macro que configura el modo de operación, para trabajar en expandido; el contenido del archivo EPROM es el siguiente:

PCBUG11 -E PORT= (1/2) MACRO=EPROM

7.- Una vez en el sistema de PCBUG11, se carga el programa con la siguiente instrucción:

LOADS Nombre_programa

8.- Se manda a ejecutar el programa:

G Dirección_inicio

9.- El sistema con la aplicación seleccionada comenzará a ejecutarse. En el apéndice B se proporcionará mayor información.

Capitulo 6
Construcción del Robot Móvil

Este capítulo presenta los circuitos, que se obtuvieron como resultado de las pruebas y experimentos , con el objetivo de llegar al diseño final del robot móvil, para cubrir los requerimientos del diseño planteados al principio.

6.1 Fuente de alimentación

La alimentación de todos los circuitos se realizan con dos baterías de NiCd, que suministran 9.6 Volts a 500mAh, empleados regularmente en carros de radio control. Se reduce su voltaje nominal con circuitos reguladores de la familia 7805 y 7806, con la finalidad de obtener el voltaje de operación óptimo para cada sección del robot.

En la figura 6-1 se presenta el diseño simple de una fuente de alimentación de 5 volts, compuesto de dos capacitores que eliminan el rizo existente en la fuente principal; mismo que va conectada al circuito LM7805, que regula el voltaje de entrada a 5 V, cuya corriente entregada puede variar desde 300 mA hasta 10 Amperes, dependiendo de la versión de este circuito.

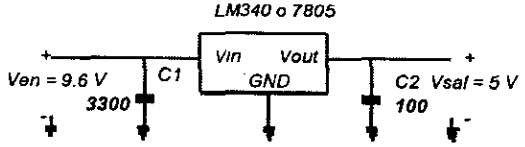


Figura 6-1. Fuente de 5 Volts para la alimentación del microcontrolador.

Para la figura 6-2 se muestra el diseño de la fuente que proporciona el voltaje tanto de los motores, como de los circuitos requeridos para esta etapa. La finalidad de emplear fuentes de alimentación distintas, es tener aisladas la parte de control que envía el HC11 y los circuitos propios del control de los motores.

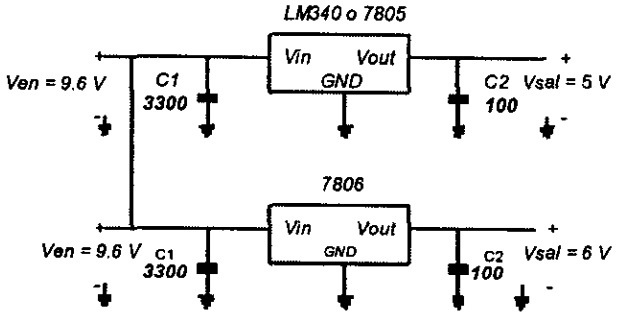


Figura 6-2. Fuente de 5 y 6 Volts para alimentación de los motores.

6.2 Adaptación del robot móvil

Las dos posibles opciones que se consideraron para la estructura del robot móvil, en la distribución de los motores y llantas son:

a) **Un motor en la parte tracera**; que controla el desplazamiento del robot, en cuyo eje se encontrarán las dos llantas y un motor en la parte delantera, conectado a una rueda giratoria, de tal forma que controle la dirección de giro del robot, como se muestra en la figura 6-3; estos motores pueden ser motores de corriente directa o motores de paso, para nuestro caso se emplearon motores de CD.

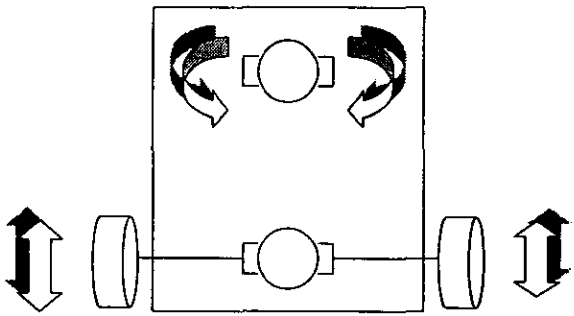


Figura 6-3. Esquema de un motor atrás - un motor adelante.

b) **Dos motores en la parte tracera**; conectando una llanta en cada uno de los motores y una rueda de movimiento libre en la parte delantera; el control de desplazamiento y de dirección lo realizan los dos motores; de manera que para hacer avanzar al robot hacia adelante y hacia atrás se hacen girar ambos motores en el mismo sentido, y en el caso del giro a la derecha e izquierda, se realiza haciendo funcionar los motores en sentido opuesto, donde el motor que gire hacia atrás será el que mandará en el movimiento del robot, los posibles movimientos se presentan en los esquemas contenidos en la fig. 6-4.

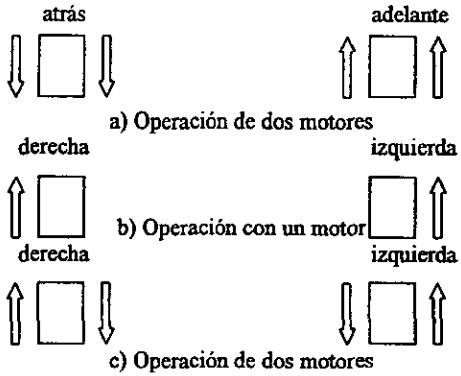


Figura 6-4 a, b, c. Posibles movimientos del robot.

La figura 6-5 presenta la implantación para este tipo de arreglo.

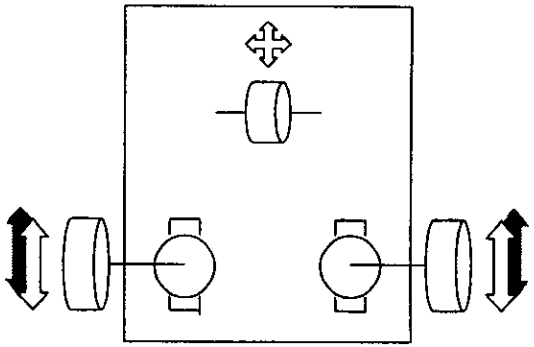


Figura 6-5. Esquema con dos motores en la parte tracera.

6.3 Control de los motores de corriente directa

Para controlar los motores se utilizó el circuito integrado L293D, que como se analizó en el capítulo cuatro, nos presenta la capacidad de controlar los dos motores, lo que nos ahorra el agregar una cantidad mayor de dispositivos electrónicos y como consecuencia reduce mucho el espacio ocupado para esta sección.

Como se muestra en la figura 6-6, y de acuerdo a las características de este circuito, se requieren de cuando menos cuatro señales de control, las cuales serán otorgadas por el microcontrolador. Para controlar un motor se requiere de una señal que entregará el comando de dirección del motor, es decir; hacia donde deseamos que gire (derecha o izquierda), y otra señal que nos proporcionará la velocidad de giro del motor. Para el control del otro motor, se necesita de la misma cantidad de señales.

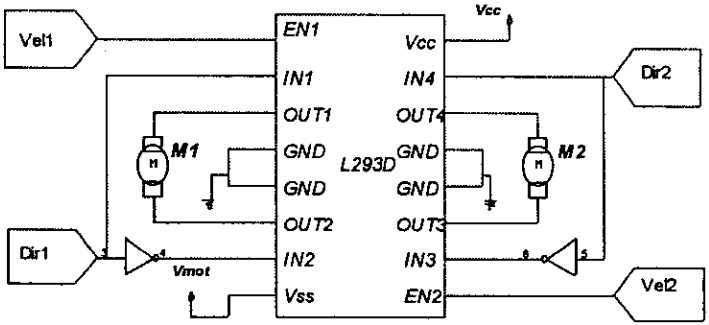


Figura 6-6. Circuito de control de los motores de CD.

El diseño más apropiado que se determinó emplear para controlar los motores, es el que se presenta en la figura 6-7, muestra el agregado de una etapa de acoplamiento y aislamiento, entre las señales que provienen del microcontrolador y la etapa de control de los motores; para tener este tipo de control se requieren de seis señales, cuatro para la dirección y dos para la velocidad de giro de los motores.

Así mismo, otra ventaja que presenta este circuito es contar con un pin asignado exclusivamente para suministrar el voltaje de polarización de los motores, el cuál puede tener un valor entre 0.2 hasta 32 volts, como se indica en la hoja de especificaciones del L293D en el apéndice C.

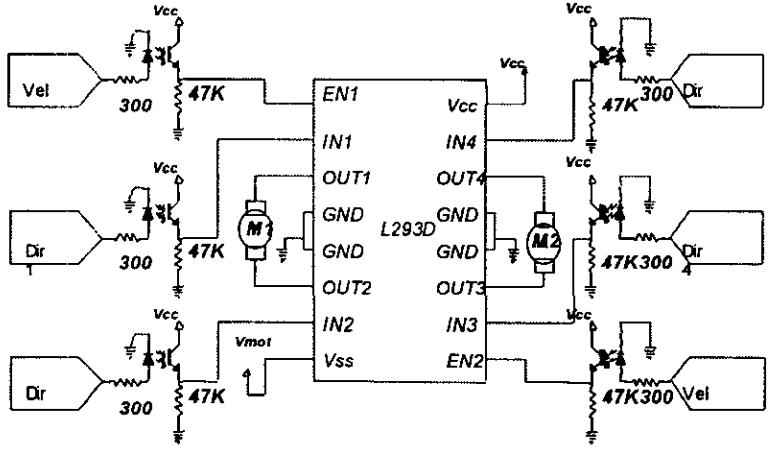


Figura 6-7. Circuito de control de los motores empleando optoacopladores.

6.4 Adaptación de sensores fotoeléctricos

La función que se busca con este tipo de sensores es interactuar con el medio ambiente, de acuerdo a la intensidad de luz que incide en ellos, el objetivo que se persigue, es que el robot realice acciones programadas, de acuerdo a la lectura que se encuentren en ellos.

a) Fotoresistencias

Las fotoresistencias son los sensores fotoeléctricos mas sencillos que existen, empleandos con una resistencia en serie se obtendrá un divisor de voltaje, cuyo valor será interpretado por el microcontrolador.

En el caso del HC11, se conectará al convertidor analógico digital; con este arreglo se obtendrán 256 posibles valores, mismos que nos tendrán informados del comportamiento del robot, con respecto a las fuentes luminosas que se encuentran en su entorno. Existen dos maneras de procesar esta información, una de ellas es obteniendo directamente el valor de la conversión digital (figura 6-8), que nos ayudaría si se desea programar con lógica difusa, y la otra manera será empleando a la salida del divisor de voltaje como una de las entradas de un comparador de voltaje, para obtener un nivel lógico (figura 6-9).

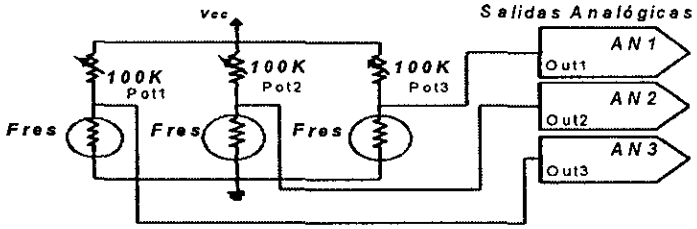


Figura 6-8. Empleo de fotoresistencias, obteniendo a su salida un valor analógico.

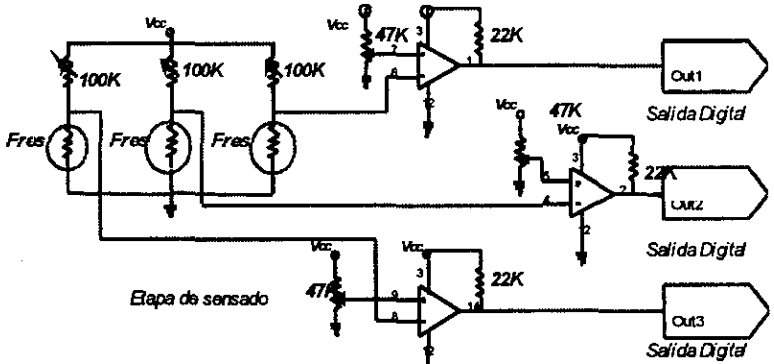


Figura 6-9.Circuito de interfaz con fotoresistencias, obteniendo una salida digital.

b) Fotoreflectores

Empleando los sensores de efecto fotoreflectivo, se aprovechan dos de las regiones de operación de los transistores, corte y saturación como se muestra en la figura 6-10.

Este tipo de sensores están compuestos de un led infrarrojo y un fototransistor, la interfaz con el microcontrolador se realiza en forma similar que las fotoresistencias.

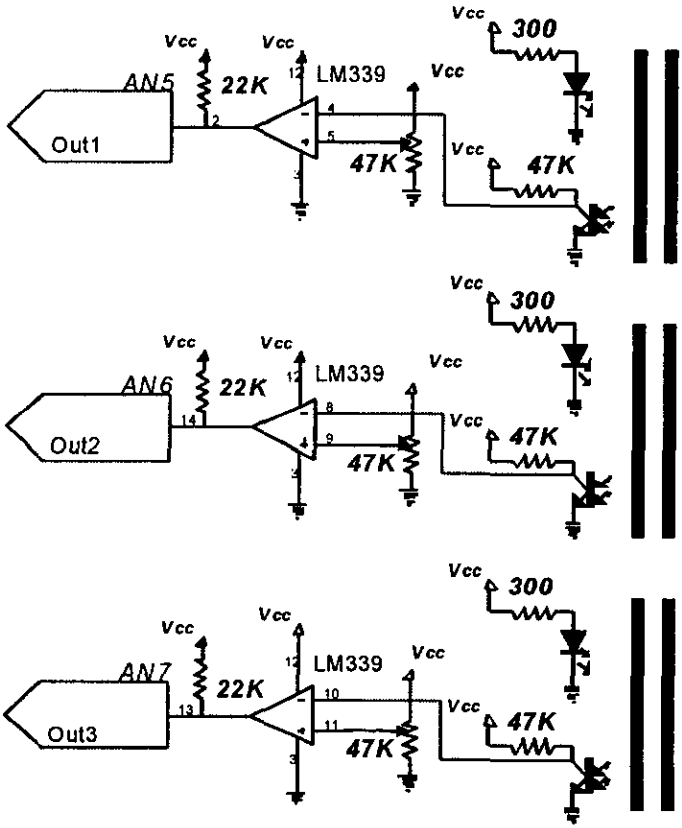


Figura 6-10. Interfaz con sensores de efecto fotoreflectivo.

6.5 Detección de obstáculos

Para la detección de obstáculos se emplean dos tipos de transductores, el primero de ellos utilizando el circuito detector de luz infrarroja GP1U52X, cuyo objetivo es detectar a distancia la presencia de un objeto, por lo tanto no tendrá contacto físico con el. En el segundo caso, cuando el robot tiene contacto físico con el obstáculo se emplearán microswitches.

a) Interfaz empleando microswitches

El uso de los microswitches facilita su adaptación para cualquier sistema de detección, en el caso del contacto directo, la interfaz se realiza agregando una resistencia en serie con este sensor, con la finalidad de abrir o cerrar el circuito, con lo cuál permitirá la circulación de corriente; según la figura 6-11, el voltaje de salida siempre estará entre los niveles de polarización y tierra, es decir 0 y 5 Volts.

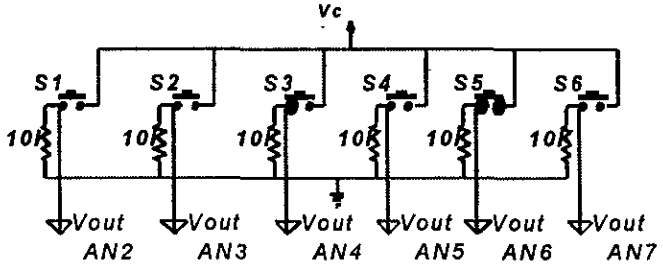


Figura 6-11. Detección de obstáculos utilizando microswitches.

Se pueden conectar tantos microswitches como se deseen, para obtener la información más precisa del entorno que rodea al robot, en nuestro caso se emplearon un total de seis sensores, como se muestra en la figura 6-12.

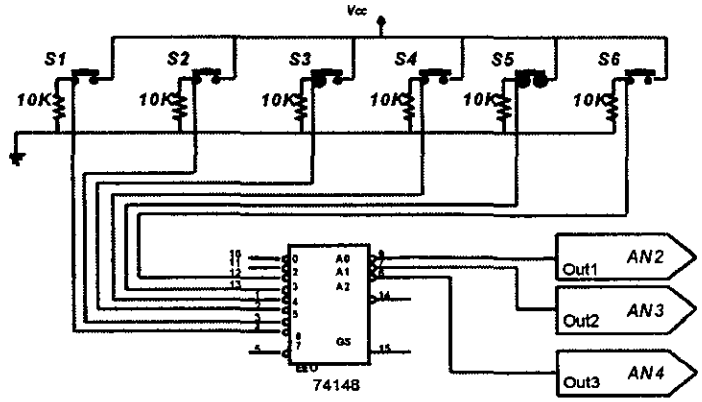


Figura 6-12. Detección de obstáculos empleando microswitches.

b) Interfaz infrarroja

Como se explica en el capítulo 3 y en el apéndice D esta interfaz se realiza generando una señal que oscile en el rango de los 40 Khz, que es la frecuencia de detección del módulo empleado para este fin (GP1U52X).

La frecuencia de oscilación se genera con el circuito integrado temporizador NE555, en modo de operación astable como se muestra en la figura 6-13; en el apéndice E se dará una breve descripción de este circuito integrado y de sus modos de operación; la distancia mínima de detección está controlada por el potenciómetro que se incluye en el circuito final.

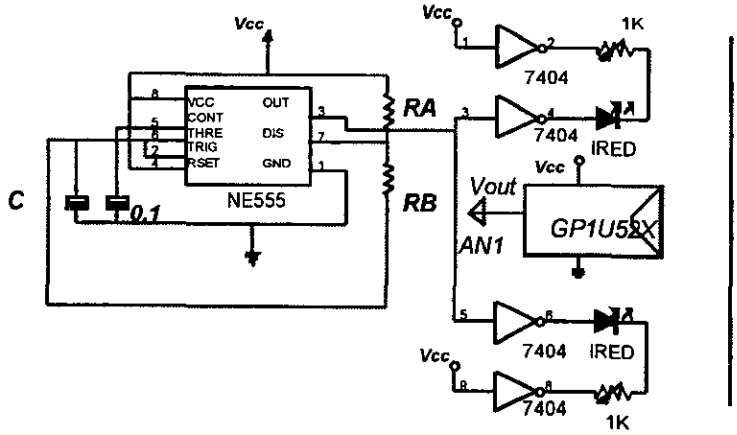


Figura 6-13. Interfaz infrarroja.

Para obtener los valores de las resistencias se empleo la siguiente ecuación, misma que se detalla en el apéndice E:

$$f = 0.695(RA+2RB)C1$$

Si se fijan los valores de C = 4.7pF y de RA = 4.7 KOhms; se despeja la ecuación, para obtener la siguiente:

$$RB = (T/0.695C - RA) ½$$

Sustituyendo valores, se obtiene el valor de la otra resistencia:

$$RB = 1476.72 \text{ ohms.}$$

6.6 Control de posición y orientación

Se describió en el capitulo de sensores, la forma de operación de los distintos dispositivos empleados para controlar el desplazamiento del robot. Se adaptaron por su facilidad de uso los fotoreflectores, pero se podrían emplear indistintamente los interruptores ópticos o los magnéticos. Se presentan en la figura 6-14 a, b y c los circuitos adaptados para tener el control de la distancia recorrida a sí como de la velocidad del robot móvil.

a) Interruptor óptico

Un interruptor óptico esta compuesto por un led emisor de luz infrarrojo y un fototransistor; dispositivos externos únicamente requiere de dos resistencias, una conectado al infrarrojo para limitar la corriente y otra resistencia de pull up conectada en el colector del fototransistor.

Existe una pequeña ranura que separa a los dos dispositivos, donde se introduce un disco ranurado o perforado, con la intención de cortar o permitir que el haz infrarrojo incida en el fototransistor.

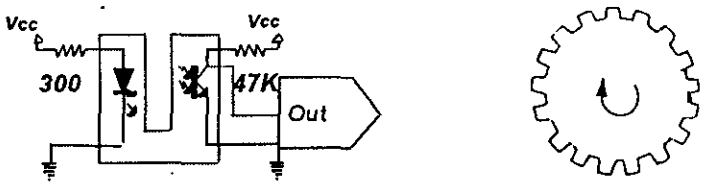


Figura 6-a. Control con un foto interruptor.

b) Fotoreflector

Se compone de los mismos elementos que el interruptor óptico, tanto en lo interno, como en lo externo, pero colocados de distinta forma; se emplea el efecto de reflexión del haz infrarrojo; mismo que será capturado por el fototransistor, cortando o saturando a este último, dependiendo si existe reflexión.

En este caso se conecta frente a este sensor un disco con superficie blanca y rayas negras, con tantas como se desee; ya que entre mayor sea la cantidad de rayas se podrá tener un mejor control del movimiento y comportamiento del robot.

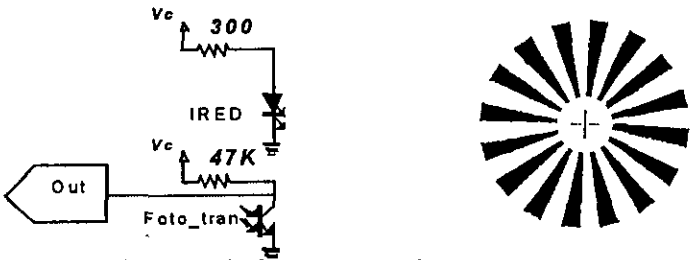


Figura 6-14b. Control con un fotoreflector.

c) Switch magnético

Esta interfaz está compuésto de un imán y un switch que se cierra al pasar el primero por el segundo; se requiere de una resistencia en serie con el switch. La única desventaja que presenta es que se deben de colocar una cantidad grande de pequeños imanes alrededor de diámetro de la llanta, para tener un control adecuado de los movimientos del robot.

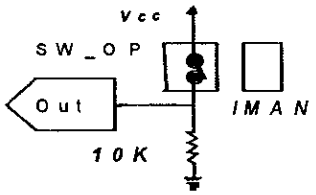


Figura 6-14c. Control con un interruptor magnético.

6.7 Diseño general del robot móvil

La siguiente figura muestra el diagrama a bloques de los componentes del robot móvil, en el cuál se adaptarán las interfaces explicadas anteriormente.

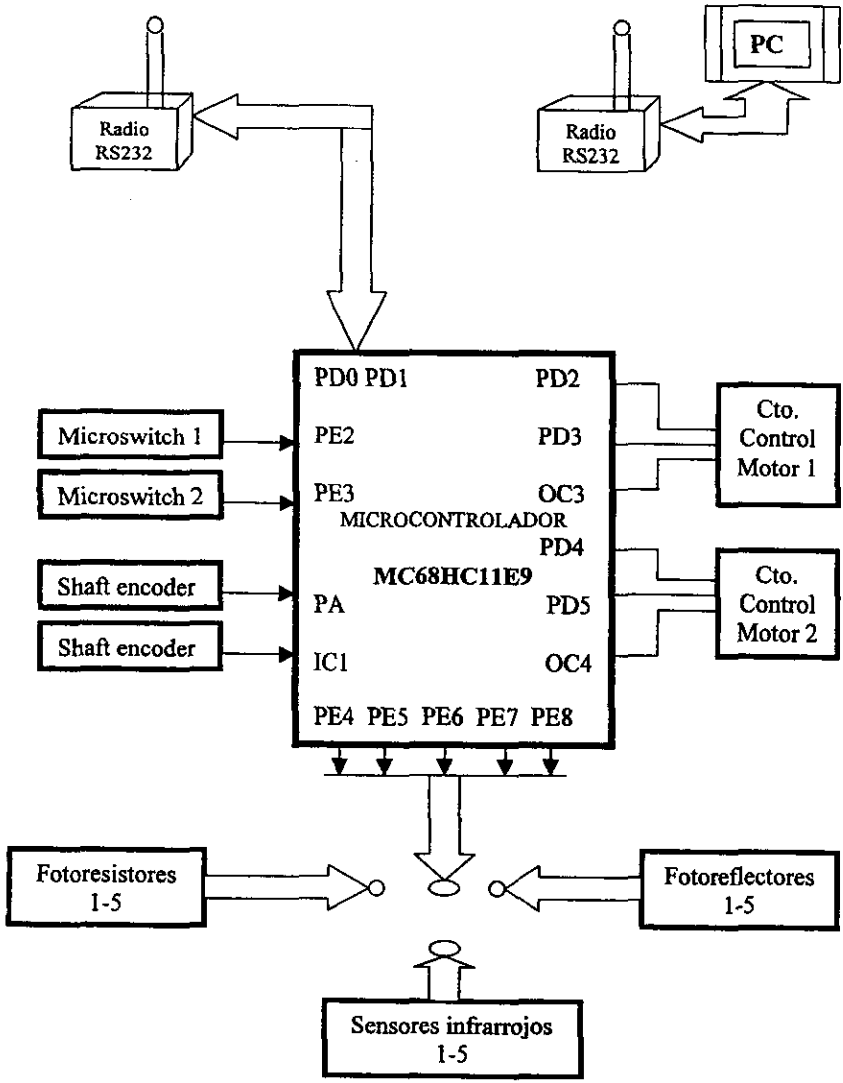


Figura 6-15. Diagrama a bloques del robot móvil

CAPITULO 7
Control y aplicaciones del Robot Móvil

Existe un sin fin de aplicaciones que se puede ejecutar con el Robot Móvil desarrollado en el presente trabajo, ya que tiene la facilidad de cambiar de una tarea a otra sin mucho problema y realizando pocas modificaciones. El uso del robot es de relativa sencillez, solo es necesario cargar el programa que se desea; el procedimiento para ejecutar la aplicación, es el mismo para todos los casos; se encuentra descrito en el capítulo 5.

Cuando se desarrollan las etapas de prueba, los programas son almacenados en la memoria RAM externa del microcontrolador, por lo que es posible hacer las modificaciones y compilar el programa tantas veces como sea necesario, hasta que se tenga una versión confiable y a la entera satisfacción del operador, para posteriormente ser programado en una memoria de tipo EPROM.

Las tareas y aplicaciones pueden ser tan sencillas o complicadas como se deseen; en este capítulo se presentan algunas aplicaciones probadas con el robot móvil; en el caso de requerir una tarea diferente a las mostradas, el robot estaría en condiciones de ejecutarla; únicamente se colocan las interfaces necesarias, así como realizar los programas para esa aplicación, en el apéndice G se presenta el código en lenguaje C, para algunas de las siguientes aplicaciones.

7.1 Control desde una computadora

La manipulación general del robot se realiza desde una computadora, controlando los movimientos y acciones deseados; de esta manera se podrá desplazar al robot, hacer que gire determinado ángulo, tomar la lectura de uno o varios canales del convertidor analógico digital del microcontrolador HC11.

Para esta tarea, además del programa previamente compilado y ensamblado, se requiere de otro programa escrito en lenguaje de alto nivel, que sea capaz de transmitir y recibir información de una terminal a otra, en nuestro caso se emplea el programa ejecutable de nombre RS232.

Existen dos maneras de realizar el control; una de ellas es empleando el cable de interface RS232 directa entre la PC y robot, la segunda acción es ocupando un par de radios RS232, transmisor y receptor, para evitar el uso del cable y contar con transmisión de radio frecuencia. El programa del robot contiene los siguientes comandos:

Acción	Comando
Avanzar hacia delante	**ad**
Girar hacia la derecha	**gi**
Lectura de los ocho canales del con A/D	**an**

7.2 *Sigue un camino marcado*

La tarea programada a realizar por el Robot Móvil, es seguir una trayectoria marcada sobre el piso, preferentemente pista blanca y fondo negro, o viceversa; aunque es posible realizar esta tarea con características distintas a las anteriores, solo es necesario calibrar y programar los valores que arrojan los sensores empleados en esta aplicación (figura 7-1).

El caminos a seguir puede tener diferentes características, como son caminar sobre línea continua, avanzar sobre una línea discontinua, contar con puntos de cruce o con bifurcaciones; avanzar sobre las diferentes condiciones requiere en su mayoría de un algoritmo propio, tomando como base la lectura de los sensores y las acciones a realizar por el robot.

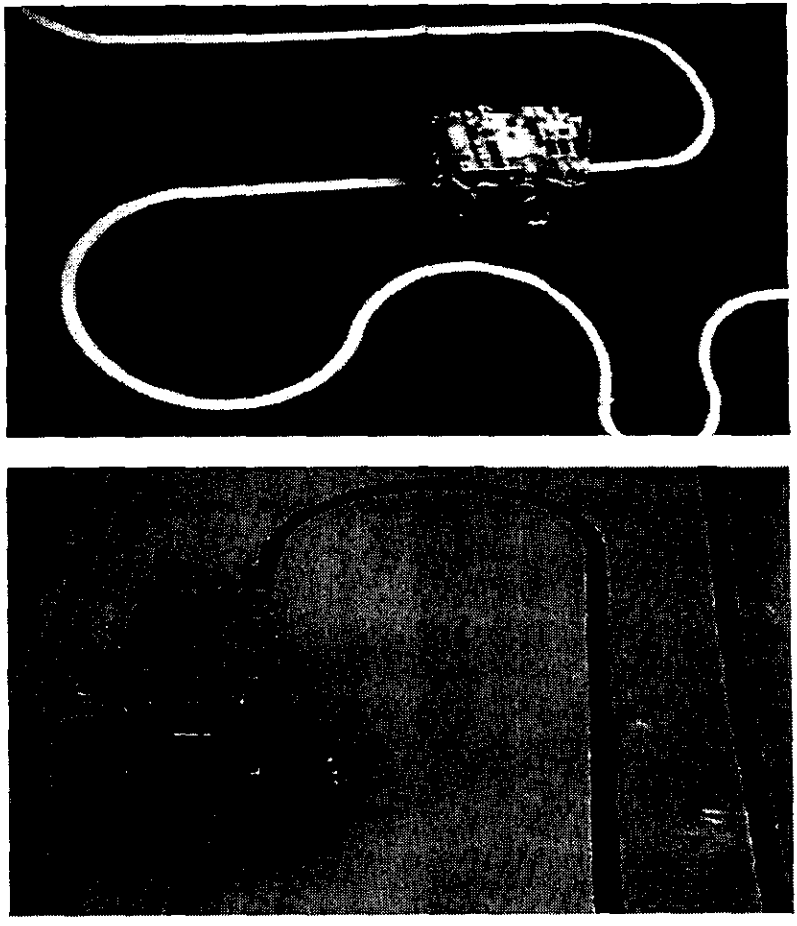


Figura 7-1. El robot sigue la trayectoria marcada en superficies de color distinto.

7.3 Desplazamiento del robot evitando obstáculos

Es posible realizar esta tarea de dos maneras distintas:

a) Detección de obstáculos empleando sensores de contacto;

b) Detección de obstáculos empleando interface infrarroja.

Para el primer caso, lo que se pretende es enviar al robot hacia un punto destino, haciendo el desplazamiento en el entorno de robot, teniendo contacto directo con los objetos, al producirse este, el robot realizará una acción previamente programada, como puede ser retroceder o girar determinado ángulo, para después continuar avanzando. En el segundo caso, la detección se realiza empleando los sensores infrarrojos, con los cuales es posible evitar los obstáculos y no tener contacto directo con ellos (figura 7-2).

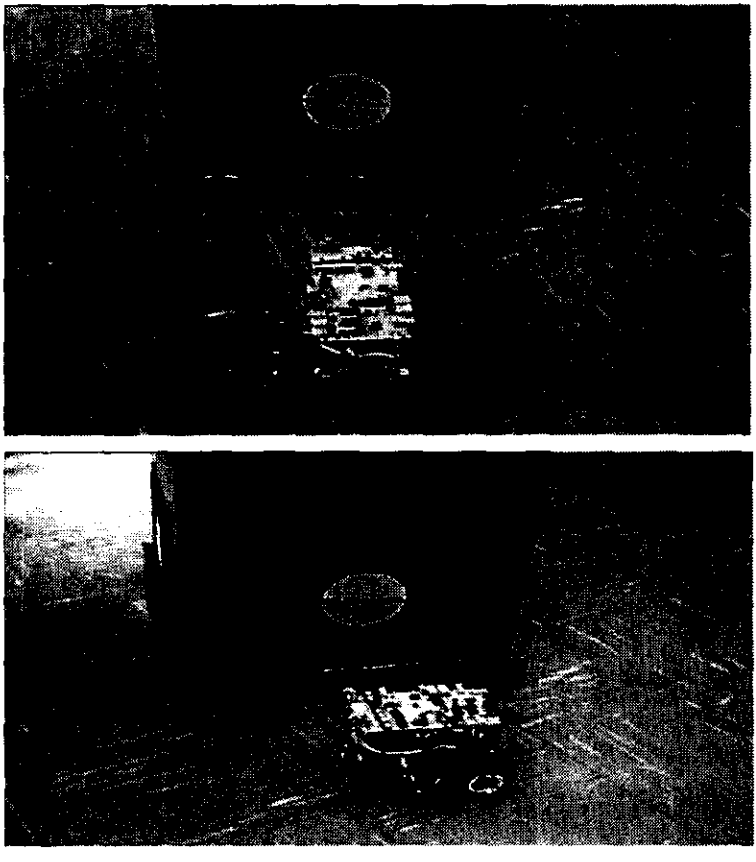


Figura 7-2. El robot detecta el objeto (figura superior), entonces realiza la acción de evitar el contacto (figura de abajo).

7.4 Movimiento del robot para realizar mapas

Se emplean las mismas funciones que el apartado anterior, se deja navegando al robot, de manera que obtenga la ubicación de los distintos objetos y elementos dentro de una oficina o habitación; se tendrá registrado la distancia, así como el ángulo de giro con el que se desplaza. La obtención de esta información se realiza transmitiendo los datos desde el robot hacia la PC, misma que se encontrará ejecutando un programa capaz de capturar y almacenar la información.

Una aplicación similar, es colocar al robot en un laberinto, de tal forma que encuentre la salida; el primer recorrido servirá de reconocimiento para encontrar la trayectoria correcta; en la segunda ocasión deberá realizar el recorrido en menos tiempo que el de reconocimiento y con el menor número de fallas.

7.5 Desplazamiento en áreas peligrosas o inaccesibles

Se controla el movimiento del robot desde una terminal, empleando una pequeña cámara de vídeo, un transmisor y un receptor de vídeo, conectado al equipo que monitorea, procesa y controla los desplazamientos del robot. Para realizar esta tarea, también se requiere contar con los transmisores de radio frecuencia, para prescindir de los cables.

7.6 Actividades de recreación

Se pueden incluir varias actividades curiosas o de entretenimiento para ser realizadas por el Robot Móvil, como se ha venido mencionando, depende de la inventiva e imaginación que se tenga, entre las que se pueden mencionar las siguientes:

a) Robot que siga a una persona utilizando una lámpara de luz como guía;

b) Robot que persiga a otro robot a donde el último se dirija;

c) Robot que permanezca desplazándose dentro de un escritorio o una mesa, de tal forma que no se caiga de ella; etc.

7.7 Aplicaciones avanzadas

Es posible la manipulación y utilización del Robot Móvil, en aplicaciones más avanzadas, donde se requerirán conocimientos más amplios en otras áreas de investigación, llamece procesamiento de voz, imágenes, telecomunicaciones y afines, como consecuencia de interfaces más complejas.

A continuación se presentan ejemplos de este tipo, que en alguna ocasión se ha logrado ejecutar por estudiantes con conocimientos en esas área:

a) Comando del robot empleando procesamiento de voz, es decir; se controlan todas las acciones que deseamos se ejecuten indicando con voz natural todos los movimientos y operaciones;

b) Planeación de rutas empleando sistemas expertos; el robot se desplazará con ayuda de los datos obtenidos del sistema experto, quien planeará el camino más adecuado y óptimo a seguir por el robot, para que se desplace de un punto denominado origen a otro punto llamado destino;

c) Aplicación y uso de redes neuronales, para que el Robot Móvil navegue en un entorno desconocido y se eviten las coaliciones; etc.

Capitulo 8
Resultados y conclusiones

8.1 Resultados

El resultado de la construcción de este robot móvil, es producto de mucho tiempo de investigación y trabajo, para llegar a las interfaces finales que se adaptaron al robot, sin dejar de ser las definitivas; ya que el desarrollo en esta área implica la constante actualización. Se comenzó a trabajar con modelos basados en carros de radio control, funcionando satisfactoriamente con algunas de las aplicaciones presentadas en el interior de este trabajo, pero al tratar de colocar otras interfaces que implicaban mayor carga para el robot y como consecuencia una mayor demanda de corriente de los motores de CD, ya no funcionaba como se esperaba.

Por lo tanto se optó por construir en su totalidad la parte mecánica del robot; se probó con varios tipos de motores, hasta encontrar los que actualmente se emplean; los cuales son un par de servomotores de tamaño pequeño, que tienen la capacidad de soportar una mayor cantidad de peso que los anteriores, ya que por el sistema de transmisión interno y la potencia con la que gira, es posible tener un alto torque; el único inconveniente es que su sistema de transmisión reduce la velocidad de giro de eje, donde van conectadas las llantas del robot, aún con esto, es la mejor opción para nuestro diseño.

8.2 Conclusiones

De acuerdo a los objetivos y alcances propuestos en un inicio, se puede considerar que cumplió con las espectativas; se trató de ser lo más claro posible al abordar cada uno de los temas que se incluyen en este trabajo.

El funcionamiento es confiable para cada una de las aplicaciones presentadas, aunque como se mencionó, si se desean otras actividades a realizar por el robot, solo se tiene que realizar el programa y agregar las interfaces correspondientes.

Se mantiene una estrecha relación con las materias de microprocesadores y microcomputadoras, que se imparten dentro de los planes de estudio en la facultad de ingeniería; ya que se emplearon en su totalidad los diferentes subsistemas del MC68HC11E9, y se presenta la información para programar tanto en lenguaje ensamblador como en lenguaje C a este microcontrolador; además de facilitar la información para que las interfaces aquí descritas puedan ser empleadas en otro tipo de sistemas, para que apliquen de manera practica los conocimientos teóricos adquiridos en las aulas de clase y construyan físicamente un sistema con funcionalidad y perspectivas ilimitadas.

Así mismo con la construcción de este tipo de robots, es posible el intercambio académico con estudiantes de otros centros educativos del país, que nos mantendrá a la vanguardia de lo que se realiza en esta área, por lo menos a nivel nacional.

8.3 Futuras adaptaciones

En lo que se refiere a la parte mecánica, es susceptible de muchas mejoras, ya que tan solo se adaptaron a los motores las llantas del robot móvil, con lo cuál no se tiene un sistema de transmisión que nos proporcione la estabilidad de los desplazamientos del robot.

Por otro lado, como el circuito de expansión de memoria ocupa los puertos B y C del microcontrolador MC68HC11, tan solo se tendrán disponibles los puertos A, D y E, para las funciones del temporizador, la comunicación serial y del convertidor analógico digital respectivamente, por lo tanto se reduce la cantidad de señales capturadas por el microcontrolador al mismo tiempo; la solución sería colocar circuitos de expansión de puertos para obtener mayor una capacidad de captura; existe los circuitos MC68HC24 y el 82C55 que nos aportarían la capacidad descrita, y así tener todas las funciones incluidas en el mismo programa.

8.4 Perspectivas de los robots móviles

Si bien es cierto la construcción de robots móviles se ha manifestado con mayor frecuencia en la comunidad científica, al grado de contar con robots capaces de ser enviados a lugares don de le es imposible por el momento al ser humano asistir, ya sean centrales nucleares, detección de la presencia de bombas, hasta el logro más reciente que fue la construcción de un robot por parte de la NASA, que fue enviado al planeta Marte con gran éxito, mismo que cumplió satisfactoriamente sus tareas, para lo cuál entraron en relación varias áreas de investigación.

Por lo tanto al ver que existe un futuro promisorio para las investigadores en esta área, guardando las proporciones reales, es posible la construcción de robots móviles con aplicaciones más practicas y útiles a la sociedad; ya que por el momento nos encontramos en la etapa de experimentación, tratando de motivar a más personas para integrarse a esta área de investigación, y colaborar con su desarrollo académico y explotar las potencialidades humanas con las que se cuentan.

APENDICE A
MICROCONTROLADOR MC68HC11E9

El MC68HC11E9 es un semiconductor de alta densidad de óxido de metal complementario (HCMOS), unidad microcontroladora (MCU). Esta MCU tiene un voltaje de operación bajo, de alta velocidad, con un bus multiplexado de una velocidad nominal de 2MHZ.

Las características especiales de este microcontrolador son:

- Cinco puertos paralelos, con doble función cada uno de ellos
- Expansión de sistemas de tiempo, con cuatro etapas preescalables
- Se resalta el no retorno a cero (NRZ)
- Interface de Comunicación Serial (SCI)
- Convertidor Analógico Digital de ocho canales, con ocho bits de resolución
- Bloque protector de mecanismo para EEPROM y CONFIG
- Bus multiplexado
- Empaquetado de 58 pines
- Tope para el ahorro de energía y modos de espera
- 64 Kbytes de memoria direccionable
- Interface de Comunicación con Periféricos (SPI)
- 512 Bytes de EEPROM
- 512 Bytes de RAM
- 12 Kbytes de ROM, con el Buffalo programado
- Circuito de Interrupción real
- Acumulador de pulsos
- Captura de entrada
- Comparación de salida

Modos de operación de HC11

Emplea dos pines, para seleccionar el modo de operación, el MODA y el MODB, básicamente existen dos modos de operación normales, que son el single chip y el expandido, y dos modos de operación especiales, el bootstrap y el test. La selección del modo de operación es de acuerdo a los valores codificados en estos pines.

Single Chip

Solo se dispone de la memoria interna del microcontrolador. Los puertos B y C, así como las terminales STRA y STRB están disponibles como entrada y salida paralelas de propósito general; todo el software necesario para controlar el MCU está contenido en los recursos internos.

Expandido

En el modo de operación expandido, permite accesar a la memoria externa dentro de la totalidad de los 64 Kbytes de espacio direccionable, el espacio incluye al espacio de memoria interna del modo single chip, los puertos B y C se emplean para realizar la expansión de memoria, el puerto B formará la parte alta del bus de direcciones y el puerto C tendrá la función multiplexada para la parte baja del bus de direcciones y el bus de datos, además incluye las señales AS, E y R/W.

Test

Permite el acceso privilegiado a recursos internos del MCU, este es una variación del modo expandido y es normalmente empleado para pruebas internas de producción por el fabricante.

Bootstrap

Es una variación especial del modo single chip, este permite dar entrada a programas en la RAM interna. Al seleccionarse este modo de operación, durante el restablecimiento una pequeña ROM de boostrap se hace presente en el mapa de memoria. Esta contiene un pequeño programa el cuál inicializa la interface serial asíncrona de comunicación (SCI), lo cuál permite al usuario cargar programas dentro de la RAM interna, finalizando esto, el control lo tendrá el programa recién cargado. Este es el modo de operación en el cual se trabaja con mayor frecuencia.

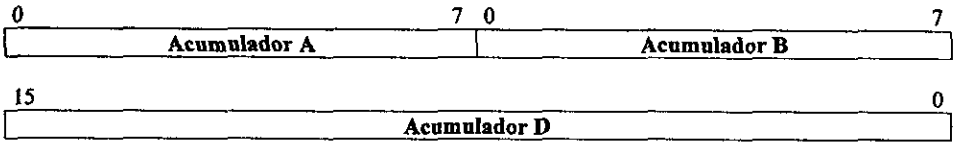
Modelo del Programador

La familia del MC68HC11 tiene ocho registros de unidades centrales de proceso, disponibles para el programador. Cada registro es explicado en los siguientes párrafos.

Acumuladores (A, B y D)

Los acumuladores A y B son registros de 8 bits, de propósito general, utilizados para almacenar los operandos y los resultados de las operaciones lógicas o aritméticas o para la manipulación de información.

Estos dos acumuladores concatenados forman el doble acumulador que será de 16 bits de longitud, donde el acumulador B es la parte baja y el acumulador A es la parte alta de este doble acumulador.



Registro de índice (IX)

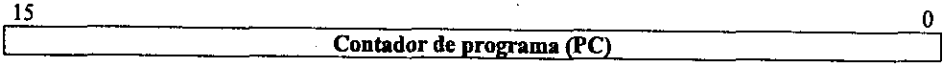
El registro IX es un registro de 16 bits, donde su utilidad se encuentra principalmente para el modo de direccionamiento indexado. Provee unos 16 valores que pueden ser sumados a un offset de ocho bits para accesar una dirección efectiva. Este registro puede ser también utilizado para realizar tiempos de retardo o como un área de almacenamiento temporal.

Registro de índice Y (IY)

Es un registro de 16 bits, utilizado para el modo de direccionamiento indexado, similar al del registro IX. Sin embargo la mayoría de las instrucciones donde se emplea el registro IY, tiene dos bytes y requiere un byte extra de código de máquina y un ciclo extra al momento de su ejecución.

Contador de programa (PC)

El contador de programa PC es un registro de 16 bits, contiene la dirección de la siguiente instrucción que será ejecutada.



Apuntador a la pila (SP)

El stack pointer SP es un registro de 16 bits que contiene la dirección de la siguiente ubicación libre en el stack. El stack es configurado como una secuencia de últimos en entrar, primeros en salir (LIFO), lee registros de escritura que permiten datos importantes para ser almacenados durante interrupciones y llamados a subrutinas.

Registro de banderas (CCR)

El registro de banderas es un registro de 8 bits, en el cuál cinco bits son utilizados para indicar los resultados de la instrucción recién ejecutada, dos bits mascarables de interrupciones y un bit de paro. Estas banderas pueden ser checadas individualmente por el programa y realizar una tarea específica como resultado de su estado; cada bit se explica a continuación.

7	6	5	4	3	2	1	0
S	X	H	I	N	Z	V	C

(C) Acarreo (Carry/Borrow)

El bit C indica si en la última operación realizada, ya sea lógica o aritmética existió un acarreo, esta bandera actúa también como una bandera de error para las operaciones de multiplicación y división, así mismo se afecta durante las operaciones de rotación de los registros.

(V) Desbordamiento (Overflow)

El bit V indica si ocurrió un sobreflujo como resultado de una operación.

(Z) Cero (Zero)

Se fija cuando el resultado de la última operación lógica, aritmética o manipulación de datos es cero.

(N) Negativo (Negative)

El bit N indica si el resultado de la última operación aritmética, lógica o manipulación de datos es negativo, es decir, refleja el estado del bit más significativo (MSB) de un resultado. Un número es positivo cuando el MSB es cero y es negativo cuando MSB es uno.

(H) Medio Acarreo (Halft Carry)

Indica el acarreo existente en una operación aritmética entre los bits tercero y cuarto; es de utilidad cuando se desea hacer el ajuste a decimal del resultado de una suma.

(I) Interrupción de Máscara

Este bit puede ser activado por software y hardware para habilitar todas las fuentes de interrupción de máscara.

(X) Interrupción de Máscara

El bit X se activa solamente por hardware (RESET o XIRQ) y limpiado solamente por el programa de instrucciones de transferencia de A hasta CCR o retornando de interrupciones (RTI).

(S) Inhabilitador de Paro

Este bit permite habilitar o deshabilitar la instrucción de stop.

Modos de Direccionamiento

El MCU cuenta con seis modos de direccionamiento, inmediato, directo, extendido, indexado, inherente y relativo, que son utilizados para obtener o almacenar un dato, de o hacia alguna localidad de memoria.

Modo de Direccionamiento Inmediato (INM)

En el modo de direccionamiento inmediato, el argumento actual está contenido en el byte(s) que siguen a la instrucción.

Formato: ***Instrucción #Dato***

Ejemplo:

```
LDAA #$A0          ; Carga el dato $A0 en hexadecimal al acumulador A
LDAB #%01100011    ;Carga el dato binario indicado al acumulador B
```

Modo de Direccionamiento Directo (DIR)

En el modo de direccionamiento directo, el byte menos significativo de la dirección efectiva de la instrucción aparecerá en el byte siguiente al opcode. El byte más significativo de la dirección efectiva es tomado como $00. Este modo de direccionamiento hace referencia al espacio de memoria comprendido entre $0000 y $00FF, es conocido como direccionamiento página cero.

Formato: ***Instrucción $00Dirección***

Ejemplo:

```
LDAA #$A0   ;Carga el dato indicado al acumulador A
STAA $50    ;Envía el contenido de A a la dirección $50
```

Modo de Direccionamiento Extendido (EXT)

La dirección efectiva de la instrucción aparece explícitamente en los dos bytes siguientes al opcode. Por lo tanto se puede hacer referencia al área total del microcontrolador es decir de la dirección $0000 a la $FFFF.

Formato: ***Instrucción $Dirección***

Ejemplo:

```
LDAA #$01FF ;Carga el dato indicado al acumulador A
STAA $1004  ;Envía el contenido del acumulador A en la dirección $1004
```

Modo de Direccionamiento Indexado (INDX, INDY)

Los registro de índice X o Y son empleados para hacer referencia a la dirección efectiva donde se obtendrá o almacenará el dato. La dirección efectiva es variable y depende del contenido actuál del registro de índice a emplear y de un OFFSET sin signo, contenido en la instrucción.

Formato: ***Instrucción*** ***$Offset,Reg_índice***

Ejemplo:

```
LDX #$1000   ;Inicializa el registro de índice X
LDAB $31,X   ;Carga el contenido de la dirección $1031 en B
```

Modo de Direccionamiento Inherente (INH)

En este modo de direccionamiento, toda la información necesaria para ejecutar una instrucción, está contenida en el código de operación.

Formato: ***Instrucción***

Ejemplo:

```
INX          ;Incrementa el contenido del registro X
ABA          ;Suma el contenido de los acumuladores A y B
```

Modo de Direccionamiento Relativo (REL)

Su utilidad se encuentra en los saltos o brincos relativos, comúnmente ejecutados después de una comparación o lectura del CCR; para lo cuál es posible la toma de decisiones de bifurcación. Las instrucciones de bifurcación generan dos bytes, uno para el opcode y el otro para el offset relativo. Si la condición de bifurcación es verdadera, el contenido de los 8 bits del byte siguiente al opcode (offset) son sumados al contenido del contador de programa para formar la dirección efectiva de la bifurcación, de otra manera, el control pasa a la instrucción siguiente a esta.

Formato: ***Instrucción Etiqueta***

Ejemplo:

```
LDAA $100A   ;Carga en A el contenido de la dirección $100A
CMPA #$F0    ;Compara A con el dato #$F0
BEQ igual    ;Si Z=1 brinca a igual
BNE diferente ;Si no es igual brinca a diferente
```

Puertos Paralelos

En la serie E del microcontrolador MC68HC11, se disponen de 5 puertos paralelos, y un total de 38 pines de entrada y salida; a continuación se presentas estos:

Pines de entrada	Pines Bidireccionales
3	2
-	-
-	8
-	6
8	-

Puerto A

Al puerto A no le afecta el modo de operación, viene configurado para esta versión del microcontrolador de la siguiente manera, tres pines de salida, tres de entrada y dos que se pueden seleccionar como entrada o salida.

PORTA Datos de Puerto A $1000

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAI	OC2	OC3	OC4	IC4/0C5	IC1	IC2	IC3

Puerto B

En single chip y bootstrap todos los pines del puerto B son salidas de propósito general; en el modo expandido y test, forman la parte alta del bus de direcciones.

PORTB Datos del Puerto B $1004

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
ADR15	ADR14	ADR13	ADR12	ADR11	ADR10	ADR9	ADR8

Puerto C

En single chip y bootstrap, el puerto C se puede configurar para que trabajen sus pines como salida o entrada, dependiendo de la selección realizada en el registro DDRC. En el modo expandido y test, el puerto C tiene la función multiplexada para la parte baja del bus de direcciones y el bus de datos.

DDRC Registro de Dirección de Datos del Puerto C $1007

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0

DDC(7:0) Dirección de datos para el puerto C

0 = Bit seleccionado como entrada
1= Bit seleccionado como salida

PORTC Datos del puerto C $1003

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
ADR7 DATA7	ADR6 DATA6	ADR5 DATA5	ADR4 DATA4	ADR3 DATA3	ADR2 DATA2	ADR1 DATA1	ADR0 DATA0

Puerto D

Al puerto D o le afecta el modo de operación, pero únicamente se dispone de 6 bits de propósito general de entrada o salida, seleccionados por la configuración del registro DDRD; tiene su doble función con los subsistemas SCI y SPI.

DDRD Registro de Dirección de Datos del puerto D $1009

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0

PORTD Datos del puerto D $1008

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		PD5	PD4	PD3	PD2	PD1	PD0
		PD5	PD4	PD3	PD2	PD1	PD0
		SS	SCK	MOSI	MISO	TxD	RxD

Puerto E

El puerto E es de 8 bits de entrada general, no le afecta el modo de operación, comparte su función con el convertidor analógico digital.

PORTE Datos del Puerto E $100A

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0

Convertidor Analógico Digital (A/D)

El convertidor analógico digital utilizado en el HC11 es un convertidor de aproximaciones sucesivas, usa la técnica de restribución de carga capacitiva para convertir señales analógicas a valores digitales. El sistema A/D es un convertidor de ocho canales, 8 bits y entradas multiplexadas está compuesto por cuatro bloques funcionales multiplexor, convertidor analógico, control digital y almacenamiento de resultados.

Multiplexor. El multiplexor selecciona una de las entradas para conversión. La selección de entrada es controlada por el valor de los bits CD, CC, CB y CA en el registro ADCTL.

Convertidor Analógico. La conversión de una entrada analógica seleccionada por el multiplexor ocurre en este bloque. Contiene el arreglo de un Capacitor Digital Analógico (DAC), un comparador y un registro de aproximaciones sucesivas (SAR). Cada conversión es una secuencia de 8 operaciones de comparación, comenzando con el bit menos significativo (MSB). Cada comparación determina el valor de un bit en el registro de aproximaciones sucesivas.

Control Digital. Todas las operaciones del convertidor A/D son controladas por bits en el registro ADCTL. Los bits de este registro indican el estado de la conversión, el control de la conversión, el control de las conversiones simples o continuas, y si las conversiones se harán sobre canales simples o múltiples.

Registro de resultados. Existen cuatro registros de ocho bits cada uno para el almacenamiento de resultados (ADR1 - ADR4). Cada uno de estos registros debe ser accesado por el procesador de la CPU. La bandera de conversión completa (CCF) indica cuando un dato válido está presente en el registro de resultados.

Secuencia de Conversión

Las operaciones del convertidor A/D son ejecutadas en secuencias de cuatro conversiones cada una; una secuencia de conversión puede ser repetida continuamente o parar después de una iteracción. La bandera de CCF cambia después de las cuatro conversiones en una secuencia que muestra la disponibilidad de datos en el registro de resultados.

Modos de Operación

Existen dos modos de operación del convertidor Analógico Digital, el de canal sencillo y el de canales múltiples.

Operación sobre un canal simple.

Es la selección y operación de un solo canal del convertidor A/D, pudiéndose seleccionar cualesquiera de los ocho canales disponibles, esto es de acuerdo a la combinación de bits en el registro ADCTL, existen dos tipos de operaciones sobre un canal simple. Cuando SCAN=0, se realiza una conversión en 4 tiempos consecutivos. El primer resultado se almacena en el registro ADR1 y el cuarto resultado es almacenado en ADR4; después que las cuatro conversiones son completadas, el sistema se detiene hasta que un nuevo comando de conversión es escrito en el registro ADCTL. En el segundo tipo, cuando SCAN=1, las conversiones son efectuadas continuamente almacenándose en los registros ADR1 - ADR4 y sobre escribiéndose en los cuatro tiempos.

Operación sobre canales múltiples.

Es posible la selección y operación de cuatro canales del convertidor A/D al mismo tipo, configurando el primer bloque o el segundo bloque de cuatro entradas. Existen dos tipos de operaciones sobre canales múltiples. Cuando SCAN=0 , un grupo de cuatro canales son convertidos uno a la vez. El resultado es almacenado en ADR1 y el cuarto resultado es almacenado en ADR4 . Después de 4 conversiones, el sistema se detiene hasta que un nuevo comando de conversión es escrito en el registro ADCTL. En el segundo tipo, cuando SCAN=1 , las conversiones son efectuadas continuamente sobre el grupo de canales seleccionados, almacenándose en los registros ADR1-ADR4 y sobre escribiéndose cada tiempo.

Registros involucrados

OPTION **$1039**

Bit 7							Bit 0
ADPU	CSEL		DLY				

ADPU: Habilita el Convertidor Analógico Digital
0 = Deshabilitado
1 = Habilitado

CSEL: Selección del Reloj
0= El A/D y la EEPROM usan el sistema de reloj E.
1= El A/D y la EEPROM usan el reloj interno RC.

DLY: Habilita el oscilador iniciando retardos.

Registro de Estado/Control del A/D

ADCTL Control del Analógico Digital $1030

Todos los bits de este registro pueden ser leídos o escritos, excepto CCF (bit 7), el cuál es solamente de lectura como indicador de estado.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit1	Bit 0
CCF	–	SCAN	MULT	CD	CC	CB	CA

CCC Bandera de conversión completa.
Este bit cambia cuando se tiene una conversión válida.
1 = Conversión completa

SCAN Control de rastreo continuo.
0 = SCAN sencillo
1 = SCAN continuo

MULT Control de canal simple/múltiple canal.
0 = canal simple
1 = múltiple canal

CD - CA Selección de los canales del convertidor A/D.

Si MULT= 0 (Selección de canal simple)

CD	CC	CB	CA	Canal seleccionado
0	0	0	0	AN0
0	0	0	1	AN1
0	0	1	0	AN2
0	0	1	1	AN3
0	1	0	0	AN4
0	1	0	1	AN5
0	1	1	0	AN6
0	1	1	1	AN7

Si MULT=1 (Selección de un bloque se cuatro canales)

CD	CC	Resultado
0	0	AN0 resultado en ADR1
		AN1 resultado en ADR2
		AN2 resultado en ADR3
		AN3 resultado en ADR4
0	1	AN4 resultado en ADR1
		AN5 resultado en ADR2
		AN6 resultado en ADR3
		AN7 resultado en ADR4

Registros de resultados (ADRx)

Estos registros son solamente de lectura que mantienen el resultado de la conversión en 8 bits. Los datos en los registros de resultados del convertidor A/D son válidos cuando ,la bandera de CCF se activa.

ADR1 Registro de Resultados 1 $1031

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

ADR2 Registro de Resultados 2 $1032

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

ADR3 Registro de Resultados 3 $1033

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

ADR4 Registro de Resultados 4 $1034

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Interface de Comunicación Serial (SCI)

El SCI permite al MCU ser eficientemente una interface con los dispositivos periféricos que permiten un formato de datos seriales en forma asíncrona. El SCI usa un estandar de no retorno a cero (NRZ) con una variedad de rangos de baudios derivados del circuito de reloj de cristal. Se utilizan los pines PD0 para recibir datos y PD1 para trasmitir; el circuito de generación de velocidades en bauds contiene un reloj programable, preescalador y divisor.

Formato de datos

El formato de los datos seriales requiere lo siguiente:

- Una línea desocupada en un estado de prioridad alta para la transmisión recepción de un mensaje.
- Un bit de comienzo (cero lógico) que es transmitido-recibido, indicando el comienzo de cada caracter.

- Datos que son transmitidos y recibidos primero con el bit menos significativo.
- Un bit de parada (uno lógico) usado para indicar el final de un frame) Un frame consiste de un bit de inicio, un caracter de 8 o 9 bits de datos y un bit de parada).
- Un break, el cual está definido como la transmisión o recepción de un cero lógico para algunos múltiples números de frames.

La selección de la longitud de la palabra es controlada por el bit M del registro SCCR1.

Operación de transmisión

El transmisor incluye un registro de transmisión de datos paralelos, llamados al SCDR y transmitidos a un buffer que pone los datos en forma serial. Los contenidos del registro serial pueden ser solo escritos a través del SCDR. Este doble sistema de buffer permite que un caracter sea cambiado serialmente mientras otro caracter está esperando en el SCDR para ser transferido. La salida del registro es aplicado al PD1 tan largo como la transmisión en progreso o el bit de habilitación de transmisión del registro de control de comunicaciones SCCR2 es almacenado.

Operación de recepción

En operaciones de recepción, la secuencia de transmisión es invertida. El dato es recibido en el registro serial y es transferido al registro receptor de datos paralelos SCDR como una palabra completa. Este sistema de doble buffer permite a un caracter ser enviado serialmente mientras otro caracter es preparado en el SCDR.

Características de Wakeup

Las características del wakeup reduce los servicios SCI superiores en sistemas múltiples de recepción. El sofrware para cada receptor evalúa el primer caracter de cada mensaje. Si el mensaje es interpretado por un receptor deferente, el SCI puede colocarse en modo de pausa, parando el resto del mensaje para generar requerimientos para el servicio. En cualquier lugar que comience un nuevo mensaje, las causas lógicas de la pausa de los receptores es para reiniciar y evaluar el caracter inicial del nuevo mensaje.

Dos métodos de wakeup son aceptables, linea lenta de wakeup o dirección de marca; en la línea lenta de wakeup, un receptor en pausa se reanuda así es como pronto la línea R/D llega a ser lenta. En la dirección de marca del wakeup, uno de los MSB de un caracter es usado para indicar que el mensaje es una dirección que reanuda todos los receptores en pausa.

Registro Involucrados en el SCI

SCDR Registro de Comunicación de Datos Serial $102F

Es el registro donde se almacena el dato, con la doble función, ya sea como transmisor o receptor de información.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
R7/T7	R6/T6	R5/T5	R4/T4	R3/T3	R2/T2	R1/T1	R0/T0

SCCR1 Registro de Control 1 de Comunicación Serial $102C

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
R8	T8		M	WAKE			

R8: Bit 9 de recepción

T8: Bit 9 de transmisión

M: Longitud del carácter SCI
0: Un bit de inicio, ocho bits de datos y un bit de paro
1: Un bit de inicio, nueve bits de datos y un bit de paro

WAKE: Selector de método de wakeup

1: Marca de dirección
0: Línea lenta

SCCR2 Registro de Control 2 de Comunicación Serial $102D

El registro SCCR2 provee los bits de control que habilita o deshabilita las funciones de interrupción.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

TIE Habilita la interrupción de transmisión
0 = Deshabilita interrupciones TDRE
1 = Solicitud de interrupciones de SCI cuando el estado de la bandera TDRE cambia

TCIE Habilita interrupciones para completar transmisión
0 = Deshabilita interrupciones TC
1 = Solicitud de interrupción de SCI cuando el estado de bandera TC cambia

RIE Habilita interrupciones de recepción
0 = Deshabilita interrupciones RDRF y OR
1 = Solicitud de interrupción para RDRF u OR

ILIE Habilita interrupciones de línea desocupada
0 = Deshabilita interrupciones IDLI
1 = Solicitud de interrupción de SCI cuando el estado de bandera IDLE cambia

TE Habilita transmisor
0 = Deshabilita transmisor
1 = Habilita transmisión

RE Habilita receptor
0 = Deshabilita recepción
1 = Habilita recepción

SCSR Registro de Estado de Comunicación Serial $102E

El SCSR indica el estado del puerto serial.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

TDRE Registro de transmisión de datos desocupados
0 = SCDR activo
1 = SCDR desocupado

TC Bandera de transmisión completa
0 = Transmisor activo
1 = Transmisor desocupado

RDRF Registro de recepción de datos llenos
0 = Registro SCDR vacio
1 = Registro SCDR lleno, indica que se puede obtener el dato

BAUD Registro de Velocidad $102B

Este registro es empleado para seleccionar diferentes rangos de velocidad, que pueden ser usados como control para la recepción o la transmisión.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TCLR	0	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0

Interface de comunicación serial con Periféricos

El SPI es un sistema serial de entradas y salidas de alta velocidad. El SPI (Interface Serial Periférica) puede ser usado para una expansión simple de entrada/salida o para permitir que muchos MCUs sean interconectados en una configuración multimaestra. La fase de reloj, polaridad es programable por software para permitir una directa compatibilidad con un gran número de dispositivos periféricos.

Cuatro líneas de señales básicas están asociadas con el sistema SPI: El Maestro Salida – Esclavo Entrada (MOSI), el Maestro Entrada – Esclavo Salida (MISO), el reloj serial (SCK) y la Selección de Esclavo (SS). Las señales del SPI son asignadas a los bits de los puertos D (5-2).

El SPI es un doble buffer para lectura, pero no para escritura. Si una escritura es intentada mientras un dato es transmitido, la transmisión es interrumpida. Esta condición causará en el bit de colisión de escritura del SPSR al ser almacenado.

En el modo maestro, el pin SCK es una salida. Dependiendo del bit CPOL del registro SCP, el SCK es lento o bajo, hasta que el dato es escrito al registro de cambio. Una vez escrito el dato, ocho pulsos son generados para cambiar los ocho bits del dato.

En el modo esclavo, los receptores lógicos comienzan una lógica delta en el pin SS y en la entrada del reloj. Así, el esclavo es sincronizado con el maestro. Los datos del maestro son recibidos serialmente en la línea MOSI de esclavo y carga los ocho bits del cambio del registro.

Registros involucrados

Registro de Datos de Entrada/Salida Periférica SPDR $102A

Registro de Control Serial Periférica SPCR $1028

Registro de Estado Serial Periférica SPSR $1029

Sistema Temporizador (TIMER)

El temporizador incluye los siguientes subsistemas:

- Temporizador, preescalador y contador
- Función de comparación de salida
- Función de captura de entrada
- Acumulador de pulsos

Temporizador, prescalador y contador

Es la base de todas las funciones de tiempo, así como controla la velocidad de operación del microcontrolador. Contiene dentro de los registros asociados a esta función, un preescador divisible entre 1, 4, 8 y 16 el tiempo base.

Cuando el contador TCNT cambia del valor $FFFF a $0000, entonces el temporizador activa la bandera de sobreflujo, con la cuál puede habilitar una interrupción.
Registros asociados:

TCNT Cuenta de temporizador $100E, $100F

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit2	Bit 1	Bit 0

TFLG2 Bandera de Interrupción del temporizador 2 $1025

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TOF							

TOF Bandera de sobreflujo del temporizador
Se activa cuando TCNT cambia de $FFFF a $0000

TMSK2 Máscara de Interrupciones del Temporizador 2 $1024

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TOI						PR1	PR0

TOI Habilitación de Interrupción cuando ocurra un sobreflujo

PR1:PR2 Selección del preescalador

[illegible]	PR0	[illegible]
	0	
	1	
	0	
	1	

Función de Salidas de Comparación

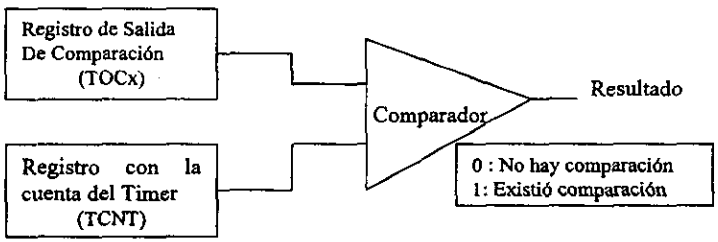
Se utilizan como indicadores de tiempo y para controlar la generación de ondas.

Durante cada ciclo de reloj, las salidas de los registros de comparación son comparados con el contador que corre libremente, si el valor de los registros es igual al del contador, entonces:

1.- El bit de bandera de interrupciones es prendido, además puede ocurrir lo siguiente:

a) El estado del pin de salida asociado puede cambiar de estado
b) Una interrupción puede ocurrir

Función de la salida de comparación



Registros asociados

Registros de salida de comparación	TOC1	$1016, $1017
	TOC2	$1018, $1019
	TOC3	$101A, $101B
	TOC4	$101C, $101d

Registro donde se almacena el valor con el cuál se compara el contador, para que al ser iguales, se genere la acción seleccionada.

TMSK1 Bandera de Interrupción del Temporizador 1 $1022

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OC1F	OC2F	0C3F	OC4F	I4/OC5F			

OC1F – OC5F Se activa la bandera correspondiente cuando existió comparación

TMSK1 Máscara de Interrupciones del Temporizado $1022

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OC1I	OC2I	OC3I	OC4I	I4/OC5I			

OC1I – OC4I Habilitación de interrupciones cuando exista comparación el pin seleccionado

TCTL1 Control del Temporizador $1020

OLx
0
1
0
1

Función de Captura de las Entradas

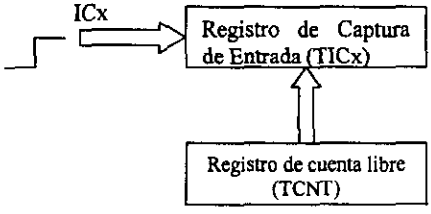
Se captura el valor del timer de 16 bits cuando una acción ocurre en el pin de entrada IC1 – Ic3.

Se utiliza para:

1.- Medir tiempos entre intervalos (ocurridos en hardware)
2.- Relacionar a eventos en tiempo real

Cuando una transición ocurre en el pin seleccionado, entonces se pueden dar los siguientes casos:

a) El valor del contador libre va al registro de captura de entrada, entonces una bandera de estado es prendida.
b) Una interrupción puede ocurrir.



Registros asociados:

Registros de Captura de entrada del Timer	TIC1	$1010, $1011
	TIC2	$1012, $1013
	TIC3	$1014, $1015

TFLG1 Registro de Bandera de Interrupción del Timer $1023

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
					IC1F	IC2F	IC3F

IC1F – IC3F Bandera del registro de la entrada de captura seleccionada

TMSK1 Registro de Mascara de Interrupción del Timer $1022

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
					IC1I	IC2I	IC3I

IC1I – IC3I Habilitación de las interrupciones para la entrada de captura seleccionada

TCTL2 Registro de Control del Timer 2 $1021

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A

Registro de control de la captura de entrada, según la siguiente configuración:

EDGxA
0
1
0
1

Acumulador de pulsos

Es útil como sistema contador de eventos y para acumulación de tiempos.

Responde a transiciones en el pin PAI, incrementando el contador/acumulador de pulsos de 8 bits; en el modo gated el contador de 8 bits es incrementado por E/64, después de ocurrir una transición en PAI.

Si una transición ocurre, entonces:

1.- La bandera de acumulación de pulsos es prendida
2.- El contador puede ser incrementado
3.- El bit de sobreflujo del acumulador de pulsos puede ser prendida
4.- Una interrupción puede ocurrir.

Registros asociados:

PACNT Cuenta del Acumulador de Pulsos $1027

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Contiene el valor del acumulador de pulsos.

TFLG2 Registro de bandera de interrupción del Timer 2$1025

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		PAOVF	PAIF				

PAOVF Bandea de sobreflujo del acumulador de pulsos; se activa cuando en el registro PACNT cambia la cuenta de $FF a $00

PAIF Bandera de Interrupción del acumulador de pulsos

Registro de Mascara de Interrupciones del Timer TMSK2 $1024

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		PAOVI	PAII				

PAOVI Habilita interrupciones de sobreflujo del acumulador de pulsos
0 = Deshabilitado
1 = Se requiere cuando PAIF del registro TFLG2 es activado

PAII Habilita interrupciones del acumulador de pulsos

0 = Interrupciones del acumulador de pulsos deshabilitado
1 = Interrupción requerida cuando el bit PSIF de TFLG" es activado.

PACTL Control del Acumulador de Pulsos $1026

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DDRA7	PAEN	PAMOD	PEDGE				

DDRA7 Dirección de datos del pin PA7
0 = Entrada
1 = Salida

PAEN Habilitación del acumulador de pulsos
0 = Acumulador de pulsos deshabilitado
1 = Acumulador de pulsos habilitado

PAMOD Modo de operación del acumulador de pulsos
0 = Operación de cuenta de pulsos
1 = Acumulación de tiempos

PEDGE Control del flanco del acumulador de pulsos
0 = Incrementa el contador con flanco de bajada
1 = Incrementa el contador con flanco de subida

Interrupciones en Tiempo Real

Se utiliza para procesar a intervalos periódicos a intervalos especificados por el usuario, usando el contador de cuenta libre, pudiendo ocurrir lo siguiente:

1.- La bandera de interrupción de tiempo real es prendida, además una interrupción puede ocurrir.

Registros asociados:

TCNT Cuenta de temporizador $100E, $100f

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit2	Bit 1	Bit 0

TFLG2 Registro de bandera de interrupción del Timer 2$1025

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	RTIF						

RTIF Bandera de Interrupción de tiempo real

Registro de Mascara de Interrupciones del Timer TMSK2 $1024

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	RTII						

RTII Habilitación de interrupciones en tiempo real

PACTL Control del Acumulador de Pulsos PACTL $1026

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
						RTR1	RTR0

RTR1:RTR0 Selección de la razón de las interrupciones

	RTR0	
	0	
	1	
	0	
	1	

APENDICE B
Combinación de Lenguaje Ensamblador y Lenguaje C

La programación del robot móvil se realizó en lenguaje C, combinándose con rutinas escritas en lenguaje ensamblador, las cuales están incluidas en el archivo BASICS.S. Algunas de las principales funciones y rutinas que se incluyen en este archivo se muestran a continuación:

B.1 Lectura de una localidad de memoria de ocho bits (peek)

peek *Permite leer la localidad de memoria indicada y lo coloca en una variable entera.*

Formato: **int=peek(dirección);**

Rutina escrita en ensamblador:

```
_peek  EQU *
       TSX          Transfiere el valor del SP a IX
       LDX 2,X      Obtiene la dirección
       LDAB 0,X     Lee el dato de memoria (8 bits)
       CLRA         Limpia el acumulador A
       RTS
```

Ejemplo de aplicación.

```
main();
        {
                int ch1;
                ch1=peek(ADR1);
        }
```

B.2 Lectura de un dato o palabra de 16 bits (peekword)

peekword *Leer una palabra, constituida por dos localidades de memoria adjuntas.*

Formato: **int=peekword(dirección);**

Rutina escrita en ensamblador.

```
_peekword   EQU *
            TSX          Transfiere el contenido de SP a IX
            LDX 2,X      Obtiene la dirección
            LDD 0,X      Lee el dato de memoria (16 bits)
            RTS
```

Ejemplo de aplicación:

```
main()
        {
        int ch2;
        ch2= peekword(ADR2);
        }
```

B.3 Almacenar un dato de 8 bits (poke)

poke *Cargar un dato de ocho bits en la dirección indicada.*

Formato: **poke (direccion, dato);**

Rutina escrita en ensamblador:

```
_poke   EQU *
        TSX          Transfiere el contenido del SP a IX
        LDAB 5,X     Obtiene el dato de la memoria (8 bits)
        LDX 2,X      Obtiene la dirección
        STAB 0,X     Envía el dato a la dirección
        CLRA         Limpia al acumulador A
        RTS
```

Ejemplo de aplicación:

```
main()
      {
      poke(ADCTL,0x30);
      }
```

B.4 Almacenar una palabra (poke_word)

poke_word *Almacenar un dato de 16 bits en dos localidades de memoria.*

Formato: poke_word(dirección, palabra);

Rutina escrita en ensamblador:

```
      poke_word EQU *
            TSX          Transfiere el contenido del SP a IX
            LDD 4,X      Obtiene de memoria el dato (16 bits)
            LDX 2,X      Obtiene la dirección
            STD 0,X      Envía la palabra a la dirección
            RTS
```

Ejemplo de aplicación:

```
Main()
{
poke_word(REG_BAS,0x1000);
}
```

B.5 Activación de bits (bit_set)

bit_set *Activar los bits que indica una máscara.*

Formato: **bit_set(Dirección, Máscara);**

Rutina escrita en ensamblador:

```
_bit_set    EQU *
            TSX          Transfiere el contenido del SP a IX
            LDAB 5,X     Obtiene el dato especificado por la máscara
            LDX 2,X      Obtiene la dirección
            ORAB 0,X     Activa los bits especificados en la máscara
            STAB 0,X     Envía el resultado
            RTS
```

Ejemplo de aplicación:

```
Main()
      {
      bit_set(ADCTL,0x30);
      }
```

B.6 Poner en cero bits (bit_clr)

bit_clr *Apaga los bits que se indican en la máscara*

Formato: **bit_clr(Dirección, Máscara) ;**

Rutina en ensamblador:

```
_bit_clr    EQU *
            TSX          Transfiere el contenido del SP a IX
            LDAB 5,X     Obtiene el dato especificado por la máscara
            LDX 2,X      Obtiene la dirección
            NEGB         Complemento a dos del contenido de B
            DECB         Decrementa en uno al contenido del acumulador B
            ANDB 0,X     Limpia los bits especificados por la máscara
            STAB 0,X     Envía el resultado
            RTS
```

Ejemplo de aplicación:

```
main()
      {
      bit_clr(PORTA,0x30);
      }
```

B.7 Activación del canal AN0 del convertidor A/D

Asignación de direcciones a las siguientes variables

```
OPTION    EQU $1039
REGBAS    EQU $1000
ADCTL     EQU $1030
ADR1      EQU $1031
```

Origen del programa

```
          ORG $100                  Origen del programa
          LDX   #REGBAS
          LDAA #$80                 Habilita el convertidor A/D
          STAA OPTION               ADPU=1
          LDAA #$10                 Configura conversión continua, canal AN0
          STAA ADCTL
INI       NOP
ESPERA    BRCRL $30,X,$80,ESPERA    Espera a CCF=1; conversión válida
          LDAA $31,X                Lee el dato de la conversión
          STAA DATO1                Almacena el resultado en una dirección
          BRA INI
```

B.8 Lectura de los ocho canales del convertidor A/D

Asignación de direcciones a las siguientes variables

```
OPTION    EQU $1039
REGBAS    EQU $1000
ADCTL     EQU $1030
ADR1      EQU $1031
ADR2      EQU $1032
ADR3      EQU $1033
ADR4      EQU $1034

          ORG $100
          LDAA #$90                 Habilita al convertidor A/D
          STAA OPTION               ADPU=1
          LDAA #$30                 Configura conversión múltiple
          STAA ADCTL                Primer bloque de cuatro canales
INICIO    NOP
ESPERA    BRCLR $30,X,$80,ESPERA    Espera conversión válida, ¿CCF=1?
          LDAA ADR1                 Lee la conversión de AN0
          STAA CANAL0
          LDAA ADR2                 Lee la conversión de AN1
          STAA CANAL1
          LDAA ADR3                 Lee la conversión de AN2
          STAA CANAL2
          LDAA ADR4                 Lee la conversión de AN3
          STAA CANAL3
          LDAA #$34                 Configura conversión múltiple
```

```
            STAA ADCTL              Segundo bloque de cuatro canales
ESPERA2     BRCLR $30,X,80,ESPERA2  Espera conversión válida, ¿CCF=1?
            LDAA ADR1               Lee la conversión de AN4
            STAA CANAL4
            LDAA ADR2               Lee la conversión de AN5
            STAA CANAL5
            LDAA ADR3               Lee la conversión de AN6
            STAA CANAL6
            LDAA ADR4               Lee la conversión de AN7
            STAA CANAL7
            BRA INICIO
```

B.9 Activación del puerto serial

```
REGBAS      EQU $1000
BAUD        EQU $102B
SCCR1       EQU $102C
SCCR2       EQU $102D
SCSR        EQU $102E
SCDR        EQU $102F
            ORG $100
            LDX #REGBAS
            CLR SCCR1,X         1 bit de inicio, 8 de datos y uno de paro
            LDD #$302C
            STAA BAUD,X         Velocidad de transmisión a 9600 bauds
            STAB SCCR2,X        Habilita transmisor y receptor
            LDAA #$40           Deshabilita XIRQ
            TAP
            LDX #SER_INT        Dirección de interrupción
            STX JSCI+1
            BRA *
```

Funciones escritas en lenguaje C, incluidas en la librería **stdio.h**, que es posible agregar en un programa.

B.10 Escribir datos en una terminal utilizando el puerto serial

```
/*Rutina para enviar datos por el puerto serial  */
puts_string(s)
char s[ ];
{
 int i;
 for(i=0;s[i]!=0;i++)    {
                          putchar(s[i]);
                          }
}
```

B.11 Enviar una cadena y valores en hexadecimal al puerto serial

```
/* Esta función envía una cadena y datos en hexadecimal al puerto serial */

puts_string_hex(s,value)
char s[ ];
int value;
        {
         puts_string(s);
        puthex(value);
        }
```

B.12 Envía una cadena al puerto serial

```
/* Esta función envía una cadena al puerto serial, it puts a new line at the end. */

puts(s)
char s[ ];
{
 int i;

 for(i=0;s[i]!=0;i++)     {
                          putchar(s[i]);
                           }
 putchar('\n');

}
```

B.13 Obtiene un dato desde el puerto serial

```
/* It gets a string from the serial port. */
gets(s)
char s[];
{
 int i;
 for(i=0;(s[i]=getchar()) != 0xd;++i) if(s[i]==0x8)i=i-2;

 s[i]=0;
 putchar('\n');
 i--;
 return(i);
}
```

APENDICE C
LM18293/L293B MANEJADOR DE POTENCIA

Descripción general

El LM18293 es un circuito integrado diseñado para manejar motores hasta de 1 A. Entre las aplicaciones típicas, incluye manejo de cargas inductivas como solenoides, relevadores, motores de corriente directa y motores de pasos; emplea internamente los transistores de potencia y utiliza un buffer para señales de nivel bajo.

En la figura C-1 se presenta el patigrama de este circuito integrado; contiene cuatro entradas para ingresar las señales de control de los motores, aceptan niveles estándares de lógica TTL y DTL para realizar su interface; dos señales de habilitación para controlar la velocidad que también acepta la misma lógica. Cada habilitador controla dos canales; cuando el pin de habilitación está deshabilitado (cero lógico), las salidas correspondientes se encontraran con lógica de tres estados; si el pin de habilitación no está conectado (flotando) el circuito funcionará como si estuviera habilitado.

Se cuenta con dos pines para proporcionar el voltaje; el pin 8 proporciona la potencia del motor y el pin 16 para suministrar el voltaje independiente al anterior, que polariza los circuitos internos.

El chip está incluido en un diseño DIP de 16 pines, el dispositivo es capaz de operar con voltajes máximos de 36 volts.

La figura C-2 presenta un ejemplo de aplicación con este circuito; en el cuál se conectaron tres motores al mismo tiempo, uno de ellos operando con giros en ambos sentidos y los dos motores restantes, únicamente pueden girar en un solo sentido.

Características

- Salida por canal de 1 Ampere
- Reemplazo directo por el circuito integrado L293B y L293D
- Empaquetado DIP de 16 pines
- Protección térmica contra sobrecargas
- Cero lógico hasta 1.5 volts
- Alta inmunidad al ruido

Máximos rangos de voltaje

- Voltaje para las cargas (Vs) 36 volts
- Voltaje de la fuente lógica(Vss) 36 volts
- Voltaje de entrada (Vi) 7 volts
- Habilitación de voltaje (Ve) 7 volts
- Corriente de salida 2 amperes

Características eléctricas

Vs = 24V, Vss = 5V, T=25oC, L = 0.4V, H = 3.5 V.

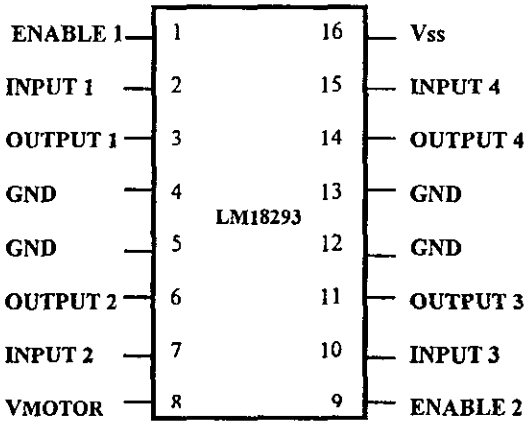


Figura C-1. Asignación de pines.

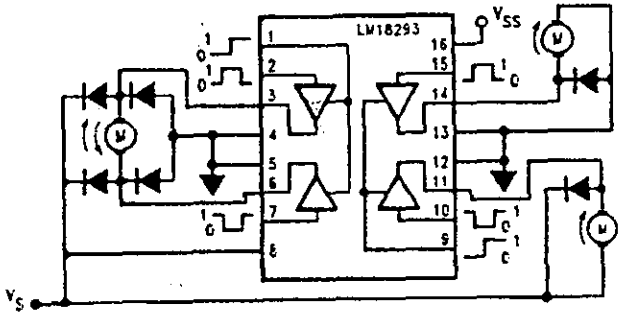


Figura C-2. Aplicación LM18293

APENDICE D

CIRCUITO INTEGRADO GP1U52X
Receptor Demodulador Infrarrojo

Descripción general

El GP1U52X es un circuito integrado híbrido diseñado para sistemas de detección infrarroja de alta confiabilidad en televisores, videocaseteras, componentes de audio y otras.

La figura D-1 muestra el diagrama de bloques del CI GP1U52X que utiliza un pequeño fotodiodo que tiene su sensibilidad pico en el rango de rayos infrarrojos de menor valor en frecuencia. Su filtro óptico interno reduce o elimina la falsa operación que pudiera causarse por otras fuentes lumínicas. La salida del fotodiodo se alimenta a un preamplificador limitador para proporcionar una señal limpia al resto del circuito. El filtro pasa banda elimina todas las señales fuera de la banda de paso (36 KHz – 44 KHz). La señal restante es alimentada a un demodulador, un integrador y un circuito de generación de onda, para que a la salida se tenga una señal de onda limpia, sin la portadora.

La figura D-2 indica el patigrama de este circuito, mismo que incluye las entradas de polarización, 0 y 5 volts; y el pin asignado para la salida.

Especificaciones

- Máximo voltaje de aplicación 6.3 V
- Voltaje de operación recomendable 4.7 – 5.3 V
- Disipación de corriente 5.0 mA
- Frecuencia central de banda de paso 40 KHz
- Ancho de banda (-3 dB de 40 KHz) 4 KHz
- Banda de paso infrarroja 980 +- 100 nm

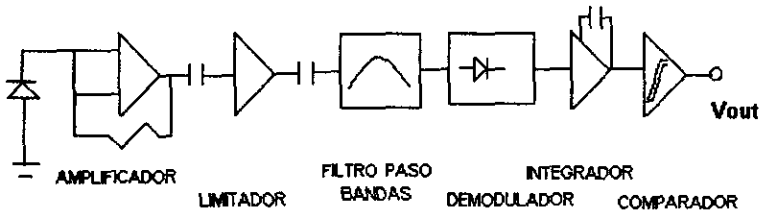


Figura D-1. Diagrama de bloques del GP1U52X.

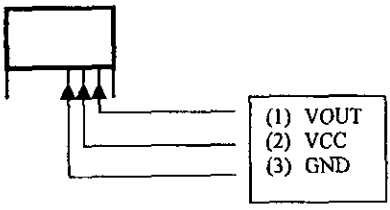


Figura D-2. Asignación de pines del GP1U52X

Ejemplo de aplicación

Los circuitos de la figura D-3, presenta un ejemplo de aplicación con un transmisor y el receptor incorporando el GP1U52X. El transmisor usa dos temporizadores integrados en un solo chip (LM556), para manejar el LED infrarrojo y transmitir un tono. El receptor emplea un decodificador de tono (LM557), para indicar el estado ON/OFF, cuando ha sido activado por el transmisor.

Se fija el temporizador para oscilar entre 100 a 1000 Hz. La salida activa al segundo temporizador, que está oscilará a 40 KHz. La salida del segundo temporizador manejará al LED infrarrojo.

A la salida del GP1U52X se coloca un transistor, quien se acopla a la entrada del decodificador de tono; cuando el 557 detecta la frecuencia dentro de su banda de paso, coloca la salida en alto.

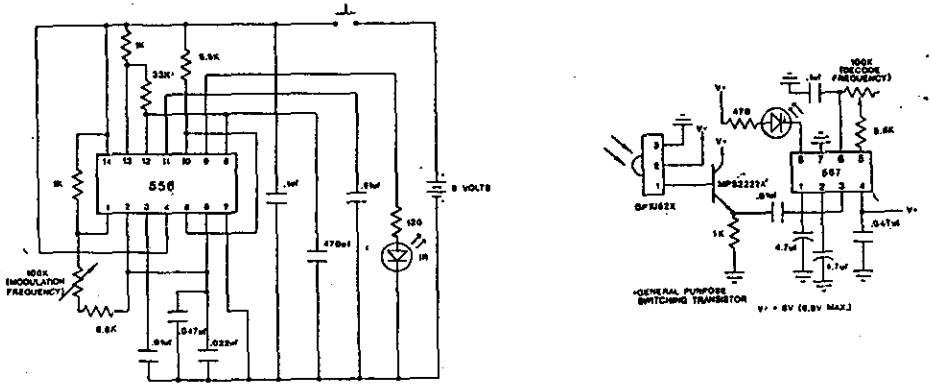


Figura D-3. Ejemplo de aplicación.

APENDICE E
TEMPORIZADOR LM555/LM555C

El Temporizador LM555 es uno de los más populares y versátiles circuitos integrados que se hayan construido. Este incluye 23 transistores, dos diodos y 16 resistores en una oblea de silicio dispuesta en una pastilla de 8 patas tipo DIP (mini dual in line package). Este circuito es un dispositivo sumamente estable para generar tiempos precisos de oscilación o retraso. Se suministran terminales adicionales si se desea disparar o reiniciar. En el modo de operación de tiempo de retraso, el tiempo se controla con un condensador y un resistor externos. Para operación astable como oscilador, la frecuencia de oscilación y el ciclo de trabajo se controlan con dos resistores y un condensador.

Características

- Reemplazo directo para el SE555/NE555
- Tiempos de microsegundos hasta horas
- Dos modos de operación, astable y monoestable
- Ciclo de trabajo ajustable
- La salida puede suministrar o consumir 200 mA
- Salida y alimentación compatible con TTL
- Estabilidad en temperatura mejor que 0.005% por °C
- Salida normalmente encendida y normalmente apagada.

Aplicaciones

- Temporización precisa
- Generación de pulsos
- Temporización secuencial
- Generación de tiempo de retraso
- Modulación por ancho de pulso
- Modulación por posición de pulso
- Generador de rampa lineal

Rangos absolutos máximos

♦ Voltaje de alimentación	18V
♦ Disipación de potencia	600 mW
♦ Rangos de temperatura de operación	0°C a +70°C
♦ Rango de temperatura de almacenamiento	-65°C a +150°C
♦ Temperatura de terminal	300°C

Características eléctricas (Ta= 25°C, Vcc = +5V a + 15V)

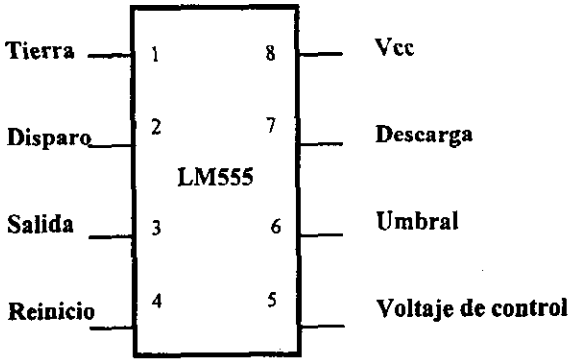


Figura E-1. Vista superior del circuito LM555

Operación monoestable

En este modo de operación, el temporizador funciona como un multivibrador monoestable (figura D-2). Inicialmente el condensador se mantiene descargado por un transistor interno del temporizador. Cuando se aplica un pulso de disparo negativo menor a 1/3 Vcc a la terminal 2, el multivibrador se ajusta con lo cual libera el cortocircuito a través del condensador y lleva la salida a alto.

Entonces el voltaje a través del condensador se incrementa exponencialmente por un periodo:

$$T = 1.1\ R_A C \qquad \text{(E-1)}$$

Al final del cual el voltaje es igual a 2/3 Vcc. Entonces el comparador reinicia al multivibrador el cual a su vez descarga al condensador y lleva a su estado bajo de salida. Cuando se tiene una señal alta, el circuito puede reiniciarse durante este tiempo, con la aplicación de un pulso negativo a la terminal de reinicio (4); entonces la salida permanecerá en el estado bajo hasta que se aplique nuevamente un pulso de disparo.

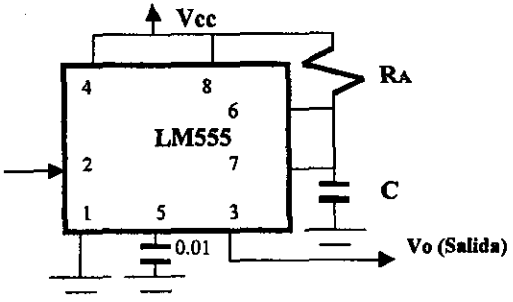


Figura E-2. Operación monoestable

Operación astable

Si el circuito se conecta como se muestra en la figura D-3 (terminales 2 y 6 conectadas) se disparará a sí mismo y oscilará, entonces el circuito temporizador estará trabajando en modo astable..

Como un multivibrador, el condensador externo se descarga a través de $R_A + R_B$. Por lo tanto, el ciclo de trabajo puede ajustarse exactamente por la relación de estos resistores.

En este modo de operación, el condensador se carga y descarga entre 1/3 Vcc y 2/3 Vcc; es decir, los tiempos de carga y descarga y por lo tanto la frecuencia son independientes del voltaje de alimentación.

El tiempo de carga (salida alta) está dado por:

$$ta = 0.693\ (R_A + R_B)\ C \qquad \text{(E-2)}$$

Y el tiempo de descarga (salida baja) por:

$$tb = 0.693\ (R_B)\ C \qquad \text{(E-3)}$$

Por lo tanto, el período total es:

$$T = ta + tb = 0.693\ (R_A + 2R_B)C \qquad \text{(E-4)}$$

La frecuencia de oscilación es:

$$F = 1/T = 1.44\ /\ (R_A + 2R_B)\ C \qquad \text{(E-5)}$$

El ciclo de trabajo es:

$$D = R_B\ /\ R_A + 2R_B \qquad \text{(E-6)}$$

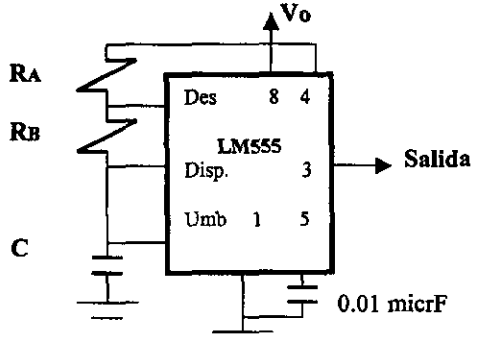


Figura E-3. LM555 operación astable

APENDICE F
Sensor analógico de efecto hall No. 1525
(Brújula)

El sensor analógico de efecto hall, es un pequeño dispositivo de tan solo 0.05 pulgadas de diámetro y 1.10 pulgadas de largo, como se muestra en la figura F-1. El sensor requiere de un voltaje de entrada regulado de 5 volts de CD, para entregar un par de señales seno y coseno, que representa la orientación con respecto al campo magnético de la tierra, consume aproximadamente 18 miliamperes con una fuente de 5 volts.

La salida entregada son dos señales, una señal senoidal y otra señal senoidal defasada 90 grados de la primera (señal cosenoidal) con valores picos de 2.88 volts y 2.12 volts máximos y mínimos respectivamente, cada salida entrega 0.4 mA.

La figura F-1 presenta el esquema y patigrama de este sensor, los pines asignados de tierra y Vcc se pueden conectar al mismo punto. Se construye el sensor para operar en posición vertical (los pines estarán viendo hacia abajo). En la misma figura se puede ver el tipo de señal entregada por la brújula, donde analizando ambas señales, es posible determinar la posición que se tiené con respecto al campo magnético de polo norte de la tierra.

En la figura F-2 se muestra el circuito empleado para adaptar la interfaz del sensor con un microcontrolador de ocho bits , se puede notar que se requerirán de dos canales para procesar esta información. La programación para obtener la orientación del sensor es sencilla, ya que tan solo es necesario fijar la tabla de los valores más representativos para realizar el proceso de comparación.

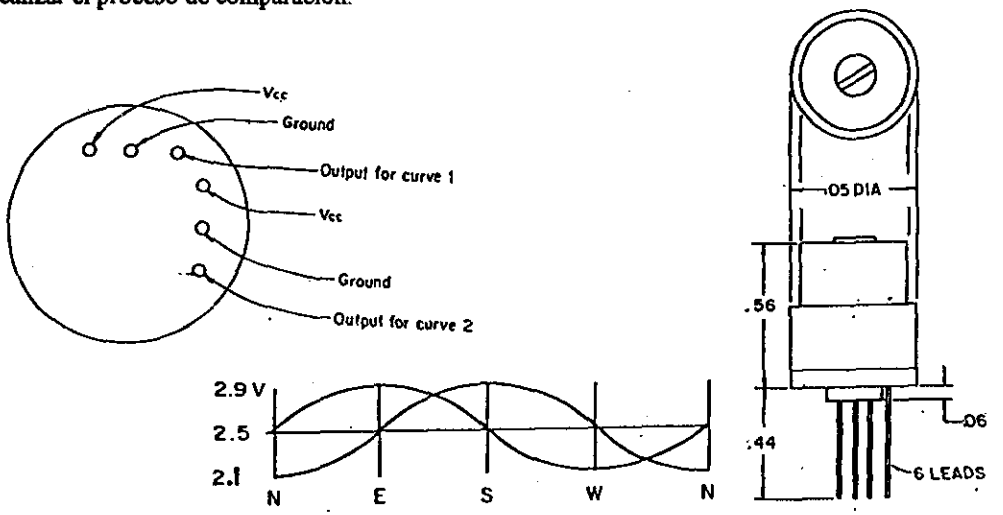


Figura F-1. Patigrama del sensor de efecto hall y la gráfica de salida entregada.

Temperatura de operación óptima: -20 °C a 85 °C
Peso aproximado: 2.25 gramos
Voltaje de operación: 5 volts

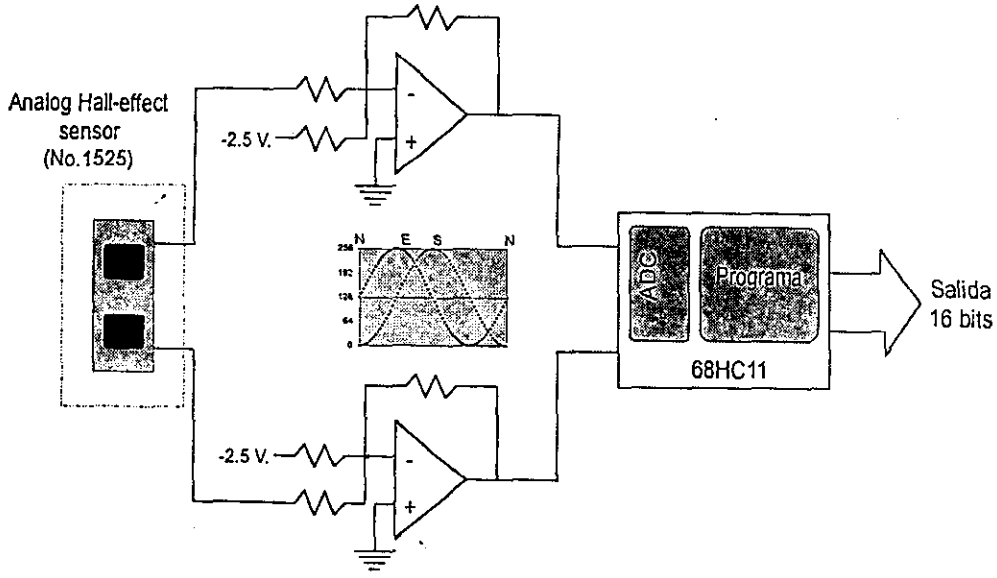


Figura F-2. Circuito de interfaz con el microcontrolador MC68HC11.

APENDICE G
PROGRAMAS DE APLICACION

Programa No.1

El siguiente programa controla las funciones del robot, configura los pines D2, D3, D4 y D5 y las señales OC3 y OC4, para controlar la velocidad y los motores de CD, así como activar el puerto serial y el convertidor analógico digital.

**** CONTROL DEL ROBOT MOVIL****

Comando	ad	Adelante
	aa	Atrás
	gd	Gira Derecha
	gi	Gira izquierda
	an	Lee Conv. A/D

```
#include "fuzzy.h"
#include "stdio.h"
#include "rabochik.h"
#define PORTD     0x1008
#define OC1       0x100D

#define ADELANTE                0x18   /* 000d4d3000 */
#define VEL_ADEL                0x30   /* 00c6c50000 */
#define ATRÁS                   0x24   /* 00d500d200 */
#define VEL_ATRAS               0x30
#define DERECHA                 0x14   /* 000d40d200 */
#define VEL_DER                 0x30
#define IZQUIERDA               0x28   /* 00d50d3000 */
#define VEL_IZQ                 0x30
#define STOP                    0x00
#define TIEMPO_ADEL             0x1fff
#define TIEMPO_ATRÁS            0xff
#define VEL_ON_IZQ              0X3fff

main(void)

{
   char s[300];
   long tiempo, tiempo_atras;
   char direccion,direccion_atras;

   /* Inicializa algunas funciones */
   inicializa();
   init_communications();
   init_ad();

   puts("\n\n           Rabochnik Version 0.30");
```

```
while(1){

   puts_string("\n-> ");
   gets(s);

    direccion = 's';
    direccion_atras = 's';
    tiempo=0;
    tiempo_atras=0;

          ** Espera los comandos para ejecutar la tarea **

       if((s[0]=='a') && (s[1]=='d'))    Movimiento hacia ADELANTE
                                        {
                          puts_string("adelante \n");
        direccion = 'a';
        direccion_atras = 't';
        tiempo = TIEMPO_ADEL;
        tiempo_atras = TIEMPO_ATRAS;
        }

     else if((s[0]=='a') && (s[1]=='a'))Movimiento hacia ADELANTE
        {
            puts_string("atras \n");
        direccion = 't';
        tiempo = TIEMPO_ADEL;
        direccion_atras = 'a';
        tiempo_atras = TIEMPO_ATRAS;
        }

     else if((s[0]=='g') && (s[1]=='d')) Giro hacia la DERECHA
        {
        puts_string("gira derecha \n");
        direccion = 'd';
        tiempo = TIEMPO_ON_DER;
        direccion_atras = 'i';
        tiempo_atras = TIEMPO_ATRAS;
        }

     else if((s[0]=='g') && (s[1]=='i')) Giro hacia la IZQUIERDA
        {
        puts_string("gira izquierda \n");
        direcion = 'i';
        tiempo =TIEMPO_ADEL;
        direcion_atras = 'd';
        tiempo_atras = TIEMPO_ATRAS;
        }

     else if((s[0]=='a') && (s[1]=='n')) Lectura del Conv. A/D
        {
        a_d();
        }
```

```
        else if((s[0]=='d') && (s[1]=='m')) Función de refresco
            {
             puts_string("dummy \n");
             dummy();
             }
       mueve(direccion,tiempo);
       direccion = 's';
       tiempo=0;
       mueve(direccion,tiempo);
       mueve(direccion_atras,tiempo_atras);
       mueve(direccion,tiempo);
 }
}

inicializa()
{
       poke(0x1020, 0x28); OC3 y OC4 se limpian en comparación exitosa
       poke(0x100C, 0x30); OC1 afectará al OC3 y OC4
       poke(0x100D, 0);    OC3 y OC4 adquirirán el valor de cero en cuenta
                           exitosa
       pokeword(0x101C, vel_curvas);
       pokeword(0x101A, vel_rectas);
       poke(0x1039, 0x90); enciende el A/D con retardo
       bit_clr(0x1028, 0x20); desactiva el open drain del puerto D
       poke(0x1009, 0x3C);    activa como salidas D2,D3,D4 y D5
}

             **  Función para que el Robot Móvil realice los **
             **             distintos movimientos             **

mueve(dir, tiem)
char dir;
long tiem;
{  long i,j;

   if (dir=='a')
   {  poke(PORTD, ADELANTE);
      poke(OC1, VEL_ADEL);
      puts_string("ADELANTE \n");
   }
   if (dir=='t')
   {  poke(PORTD,ATRAS);
      poke(OC1, VEL_ATRAS);
      puts_string("ATRAS \n");
   }
   else if (dir=='d')
   {  poke(PORTD,DERECHA);
      poke(OC1,VEL_DER);
      puts_string("DERCHA \n");
   }
   else if (dir=='i')
   {  poke(PORTD,IZQUIERDA);
      poke(OC1,VEL_IZQ);
      puts_string("IZQUIERDA \n");
   }
```

```
   else if (dir=='s')
   {
      poke(PORTD,STOP);
      poke(OC1,STOP);
   }
   for (j=0;j<10;j++)
     for (i=tiem;i>=0; i--);
}

    ** Función para enviar la lectura de los canales del conv. A/D **
a_d()
{
 int num,channel,j;
 int ch1,ch2,ch3,ch4;
 int ch5,ch6,ch7,ch8;
 ch1=ch2=ch3=ch4=ch5=ch6=ch7=ch8=0;
 num=30;
 for(j=1;j<=num;j++){
      read_ad_8(&ch1,&ch2,&ch3,&ch4,&ch5,&ch6,&ch7,&ch8);
      puts_string_hex("j ",j);
        puts_string_hex("ch1 ",ch1);
        puts_string_hex("ch2 ",ch2);
        puts_string_hex("ch3 ",ch3);
        puts_string_hex("ch4 ",ch4);
        puts_string_hex("ch5 ",ch5);
        puts_string_hex("ch6 ",ch6);
        puts_string_hex("ch7 ",ch7);
        puts_string_hex("ch8 ",ch8);
     channel=get_int();
 }
 return(0);
}
             ** Lectura de los ocho canales del covertidor **
             **             Analógico Digital              **

read_ad_8(ch1,ch2,ch3,ch4,ch5,ch6,ch7,ch8)
int *ch1,*ch2,*ch3,*ch4;
int *ch5,*ch6,*ch7,*ch8;
{
 int status;

  /* Espera mientras la conversión sea completa, CCF=1*/
 while( ( (status=peek(ADCTL)) & 0x80)==0);

 /* Toma la lectura del primer bloque de cuatro canales */
 *ch1=peek(ADR1);
 *ch2=peek(ADR2);
 *ch3=peek(ADR3);
 *ch4=peek(ADR4);

 /* Configura para leer el segundo bloque de canales*/
  poke(ADCTL, 0x34);

 /* Espera mientras la conversión sea completa, CCF=1*/
 while( ((status=peek(ADCTL)) & 0x80)==0);
```

```
/* Lee el segundo bloque de cuatro canales de conversión */
 *ch5=peek(ADR1);
 *ch6=peek(ADR2);
 *ch7=peek(ADR3);
 *ch8=peek(ADR4);

  /* Modifica para activar el primer bloque de cuatro canales*/

 poke(ADCTL, 0x30);
}

dummy(){
int i;
i=1;
 asm("_END_PRG  equ *");
}
```

Ejemplo No. 2

Programa para seguir la línea, empleando los sensores fotoreflectivos, configura los pines D2, D3, D4 y D5, para controlar la etapa de potencia de los motores de CD, configura el puerto serial, para obtener los caracteres y tener monitoreado el estado de los sensores, así como activar y configurar al convertidor analógico digital.

** Programa básico para que el robot siga la línea **

```
#include "stdio.h"      /* Definición de variables */
#define PORTD 0x1008
#define DDRD 0x1009
#define ADCTL 0x1030
#define ADR1 0x1031
#define ADR2 0x1032
#define ADR3 0x1033
#define ADR4 0x1034
#define vel_m1 0xfeee
#define vel_m2 0xfeee
#define t_mova 200
#define veces 20
#define blanco 0xa0

void  continua(char anterior);
void fija_derecha(void);
void fija_izquierda(void);
void adelante(void);
void derecha(void);
void izquierda(void);
void espera(void);

                        ** Programa principal **
main()
{
int n,i,c,d;

init_communications();
init_ad();
```

```
inicializa();
while(1)
      {
      putchar('*');
      read_ad(&n,&i,&c,&d);
      if ((d<blanco && c<blanco && i<blanco) || (d<blanco && c>blanco &&
      i<blanco))  /*todos blanco *****/
      {
          adelante();
          putchar('b');
      }

      else if(d>blanco && c>blanco && i>blanco) /*todos negros*/
      {
      adelante();
        discon();
        putchar('n');
      }

      else if(d>blanco && c<blanco && i>blanco)
      {
      adelante();
      putchar('a');
      }

else if((d<blanco && c>blanco && i>blanco)|| (d<blanco && c<blanco &&
i>blanco))
      {
      derecha();
      putchar('d');
      }

else if((d>blanco && c>blanco && i<blanco)||(d>blanco && c<blanco &&
i<blanco))
      {
      izquierda();
      putchar('i');
      }
      else
      putchar('x');
    }
}

/* Función para configurar como salida los pines PD2, PD3, PD4 y Pd5, y
las funciones de velocidad OC3 y OC4 */

inicializa()
      {
       poke(0x1020, 0x28);
       poke(0x100c,0x30);
       poke(0x100d,0);
       pokeword(0x101c,vel_m1);
       pokeword(0x101a,vel_m2);
       bit_clr(0x1028,0x20);
       poke(0x1009,0x3f);
          }    /***** fin de inicializa*****/
```

```
void adelante(void)
        {
          poke(0x100d,0x30);
          poke(0x1008,0x18);
        }     /***** fin de adelante*****/
void izquierda(void)
        {
          poke(0x100d,0x30);
          poke(0x1008,0x28);
        }     /***** fin de izquierda*****/
void derecha(void)
       {
         poke(0x100d,0x30);
         poke(0x1008,0x14);
         }     /***** fin derecha *****/
void espera(void)
     { poke(0x1008,0x00);
       poke(0x100d,0x00); }

lim_acu()
       {
        poke(0x1027,0x00);
        poke(0x1026,0x50);
       }

retardo(long tiempo)
       {
       long i,j;
       for (j=0;j<veces;j++)
       for(i=tiempo;i>=0;i--);
       }

            *** Lectura del convertidor analógico/ Digital ***

read_ad(ch1,ch2,ch3,ch4)
int *ch1,*ch2,*ch3,*ch4;
    {
      int status;
      poke(ADCTL,0x34);
       while(((status=peek(ADCTL)) & 0x80)==0);
            *ch1=peek(ADR1);
            *ch2=peek(ADR2);
            *ch3=peek(ADR3);
            *ch4=peek(ADR4);
    }
```

Ejemplo No.3

Programa para evitar obstáculos, configura los pines D2, D3, D4 y D5 del puerto D y los pines OC3 y OC4 para generar la velocidad y control de los motores de CD, así mismo activa el convertidor analógico/Digital y el puerto serial para tener monitoreado las acciones que realice el robot, como consecuencia de la lectura en los sensores de tacto e infrarrojo.

** Programa que evita obstáculos empleando los sensores **
** Infrarrojos y Microswitches **

```
#define vel_rectas 0xeeff     /* Definición de variables */
#define vel_curvas 0xefff
#define t_izq   400
#define t_der   400
#define t_ade   2000
#define t_atr   7000
#define LIMITE_PARED 50
#define ADR1 0x1031
#define ADR2 0x1032
#define ADR3 0x1033
#define ADR4 0x1034
#define t_mov 100
#define veces 100
                        ** Programa principal **
main()
{
int ch5,ch6,ch7,ch8;
inicializa();
init_communications();
while (1)
      {
      putchar('*');
      if (pared())
            {
            fija_derecha();
            retardo(t_izq);
            putchar('c');
            }
            else {
                  read_ad(&ch5,&ch6,&ch7,&ch8);
                  if( ch6>0xee )
                  {
                  fija_derecha();
                  retardo(t_mov);
                  putchar('i');
                  }
                  else if( ch8>0xee )
                  {
                  fija_izquierda();
                  retardo(t_mov);
                  putchar('d');
                  }
```

```
                else
                {
                fija_adelante();
                putchar('a');
                }
                }
            putchar('*');
      }
}

int pared(void)
{
int frente=0, par=0, i;
for (i=0; i<1000; i++)
      {
      lee_ad5(&frente);
      if (frente>LIMITE_PARED)
      par = 1;
      }
      if (par==1)
      par = 0;
      else
      par = 1;
      return(par);
}

inicializa()
{
poke(0x1020, 0x28); OC3 y OC4 se limpian en comparación exitosa
poke(0x100C, 0x30); OC1 afectará a OC3 y OC4
poke(0x100D, 0);    OC3 y OC4 adquirirán el valor de cero en cuenta
                     exitosa
pokeword(0x101C, vel_curvas); Velocidad del motor derecho
pokeword(0x101A, vel_rectas); Velocidad del motor izquierdo
poke(0x1039, 0x90);           enciende el A/D con retardo
bit_clr(0x1028, 0x20);        desactiva el open drain del puerto
poke(0x1009, 0x3C);           activa como salidas D2 y D3
fija_adelante();
}

fija_adelante()
      {
      poke(0x100D, 0x30);
      poke(0x1008, 0x18);
      }

fija_atras()
      {
      poke(0x100D, 0x30);
      poke(0x1008, 0x24);
      }

fija_izquierda()
      {
      poke(0x100D, 0x30);
      poke(0x1008, 0x28);
      }
```

```
fija_derecha()
      {
      poke(0x100D, 0x30);
      poke(0X1008, 0x14);
      }

retardo(long tiempo)
      {
      long i,j;
      for (j=0;j<veces;j++)
      for (i=tiempo;i>=0; i--);
      }

lee_ad5(l)
int *l;

      {
      int status;
      poke(0x1030, 0x14);
      while (((status=peek(0x1030))& 0x80)==0);
      *l=peek(0x1031);
      }

read_ad(ch1,ch2,ch3,ch4)
int *ch1,*ch2,*ch3,*ch4;
      {
      int status;
      poke(0x1030,0x34);
      while(((status=peek(0x1030)) & 0x80)==0);
      *ch1=peek(ADR1);
      *ch2=peek(ADR2);
      *ch3=peek(ADR3);
      *ch4=peek(ADR4);
      }
```

1. Using Microprocessors and Microcomputers; The motorola Family
Wray Greenfield Ed.Prentice Hall Career and technology, Third edition, 1994.

2. Microprocesadores, Programación e Interconexión
José María Urunuela, Ed. Mc Graw Hill, 2ª Edición, 1989.

3. Sensores y Acondicionadores de Señal
Ramón Pallas Areny, Ed. Marcombo, 2ª Edición, 1994.

4. Fotoelectrónica
Wilhelm Henning, Ed. Marcombo, 1ª Edición, 1979.

5. Desing With Microcontrollers
John B. Peatman, Ed. Mc Graw Hill, 1988.

6. Industrial Robots; Technology, Programming and Applications
Groover, Weiss, Nagel and Odrey; Ed. Mc Graw Hill, 1986.

7. Reference Manual, M68HC11
Motorola

8 M68HC11 E Series; Technical Data
Motorola

9. M68HC11EVBU; Universal Evaluation Board User's Manual
Motorola, 1997.

10. Mobile Robots; Inspiration to Implementation
Joseph L. Jones and Anita M. Flinn; Ed. AK Peters, 1993.